universität wien

MAGISTERARBEIT

Titel der Magisterarbeit

„Development of a prediction model for the decline in kidney function based on a retrospective analysis of 700,000 electronic health records“

Verfasst von

Christine Wallisch, BSc

angestrebter akademischer Grad

Magistra der Sozial- und Wirtschaftswissenschaften (Mag. rer. soc. oec.)

Wien, 2014

Studienkennzahl lt. Studienblatt:	A 066 951
Studienrichtung lt. Studienblatt:	Magisterstudium Statistik
Betreut von:	Ao. Univ.-Prof. Dr. Georg Heinze

Contents

List of Figures

List of Tables

Acknowledgement

I would like to express my appreciation and gratitude to my advisor Professor Dr. Georg Heinze for his support, patience and useful comments throughout the writing of my master thesis.
Furthermore, I would like to thank Dr. Milan Hronsky and Dr. Alexander Kainz who structured a part of the data and Professor Dr. Rainer Oberbauer and Professor Dr. Gert Mayer for providing access to the data.
I would like to thank my good friend Anna Dunietz who carefully proofread my master thesis.
Last but not least, I wish to thank my family and my friends for supporting me throughout my studies.

1 Introduction

This section introduces necessary medical information, describes the project SysKID that this master thesis details, and discusses the tasks and aims of this thesis.

1.1 Medical background

In order to understand the SysKID project, it is important to be familiar with medical definitions on chronic kidney disease (CKD).

What is CKD?

According to McClellan et al. (2006, p. 13) "chronic kidney disease (CKD) describes any degree of kidney injury or impaired kidney function that persists for $\geq$ 3 months." CKD is a frequent disease, and the number of people with the disorder is increasing (James et al., 2010, p. 1296).

The glomerular filtration rate (GFR) is the measurement unit for overall kidney health in clinical practice (Lewis, 2012, p. 32). It is a continuous marker and for practical purposes, five stages of CKD are based on GFR:

Table 1: **CKD stages** (reproduced from The Renal Association, 2013)

Stage	GFR*	Description
1	90+	Normal kidney function but urine findings or structural abnormalities or genetic trait point to kidney disease
2	60-89	Mildly reduced kidney function, and other findings (as for stage 1) point to kidney disease
3A	45-59	Moderately reduced kidney function
3B	30-44	Moderately reduced kidney function
4	15-29	Severely reduced kidney function
5	<15 or on dialysis	Very severe, or endstage kidney failure (sometimes called established renal failure)

*All GFR values are normalized to an average surface area (size) of 1.73 m^2; unit of GFR in $mL/min/1.73\,m^2$

How is GFR defined?

Lerma & Nissenson (2011, p. 26) offer the following explanation: "The GFR is the amount of plasma filtered through glomeruli per unit of time". Daugirdas (2011, p. 4) describes the GFR to be the "... volume of blood the kidneys 'clear' in a given unit of time. Because

this clearance occurs in the glomeruli, it is called the glomerular filtration rate or GFR."

This rate depends on age, sex, body size etc. of a person, so it needs to be standardized. This is done by adjusting GFR for body surface area. As the mean body surface area is $1.73\,m^2$, GFR is measured in $mL/min/1.73\,m^2$. A normal value for a young person would be 120 to 130 $mL/min/1.73\,m^2$. (Gilbert & Weiner, 2013, p. 26)

Because GFR is hard to determine, its value is estimated (eGFR) from serum creatinine, which is an endogenous filtration marker (Gilbert & Weiner, 2013, p. 26). Some formulas were developed for eGFR which are good approximations below $60\,mL/min/1.73\,m^2$, but whose precision is worse above this value. The latest is the CKD-EPI equation (Chronic Kidney Disease Epidemiology Collaboration), and it is accurate ever at normal or near-normal eGFR (Skali et al., 2001, p. 548).

The CKD-EPI equation is defined as follows (Skali et al., 2001, p. 548):

$$\begin{aligned} eGFR_{CKD-EPI} = 141 * min(Scr/k, 1)^a * max(Scr/k, 1)^{-1.209} * \\ 0.993^{age} * (1.018\ if females) * (1.157\ if blacks) \end{aligned} \tag{1}$$

where age is in years, Scr is serum creatinine in milligrams per deciliter, k is 0.7 for females and 0.9 for males, a is -0.329 for females and -0.411 for males. (Skali et al., 2001, p. 548)

1.2 Project SysKID

SysKID is the abbreviation for Systems Biology towards Novel Chronic Kidney Disease Diagnosis and Treatment. It is a large-scale health project supported by a European Union grant. The project's start was January 1^{st}, 2010, and it was planned to run for 5 years. (SysKID consortium, 2014a)

As chronic kidney disease affects many people in industrialized countries (10 % of European inhabitans suffer from it), SysKID focuses on this disorder and the decline in kidney function. If CKD is recognized in an early stage, the progression may be slowed down. CKD has been found to prompt cardiovascular complications and bone metabolism disorders. (SysKID consortium, 2014b)

Because CKD is often triggered by hypertension and diabetes SysKID concentrates on CKD in connection with these diagnoses. (SysKID consortium, 2014c)

Its aims are:

- "Aim 1: Identify persons at risk of developing chronic kidney disease utilizing epidemiology as well as molecular tools.
- Aim 2: Understand the molecular processes triggering early stage chronic kidney disease and identify associated biomarkers.
- Aim 3: Develop novel diagnostic and therapeutic strategies to control progression of chronic kidney disease.
- Aim 4: Perform pre-clinical verification of novel therapy approaches and perform clinical testing of novel diagnostics." (SysKID consortium, 2014c)

Clinicians, statisticians, epidemiologists, molecular researchers and bioinformaticians from universities and small and medium-sized enterprises and industrial partners all over Europe are paticipating in this project. (SysKID consortium, 2014b)

1.3 Tasks and aims

SysKID consists of several different tasks, one of them being to find out whether a patient receives adequate renal care. Another aim is to predict decline in kidney function. Two prediction models were developed in this thesis in order to address these objectives.

No statistical papers were found detailing methods to adequately monitor the presence of a beginning kidney problem. Many scientists, however, concentrate on the decline in kidney function. Halbesma et al. (2011) developed a prediction model for progessive CKD with a prospective population-based cohort study. The study included three screenings and was designed for the analysis of the coherence between albumin excretion and renal and cardiovascular outcome in the general population. A backward elimination and logistic regression were used to arrive at a prediction model for progressive CKD. The researchers calculated the c-index and an optimism corrected value of the c-index using bootstrapping from the same data to validate results. Tangri et al. (2011) modeled the time to kidney failure with a Cox proportional hazard regression. The development cohort consisted of electronic nephrology clinic health records, and an external data set was used for validation. The validation measures and methods included the c-index, integrated discrimination improvement, a calibration plot, the Akaike Information Criterion, and net reclassification improvement. In the study, 37 % of the development cohort and 38 % of the validation cohort were diabetics. Hunsicker et al. (1997) dealt with the progression of renal disease when diet was modified. The researchers analyzed a randomized prospective trial with two groups on different diets. They modeled the GFR slope with a mixed effects model and used backward elimination to select the variables. Another model was created by Fox et al. (2010) with a multi-marker panel of the sixth Framingham Offspring Study. The outcomes were microalbuminuria and incident CKD modeled with a multivariable logistic regression. 8 % of all participants were diabetics in this study. The validation was done by comparing a model with biomarkers to a model without. The researchers calculated the incremental c-index, the integrated discrimination index, calibration indices and performed risk reclassification. Dunkler et al. (2014) developed two multinomial prediction models for diabetics with different covariates using a cohort designed for another purpose. The outcomes were no CKD incidence or progression, CKD incidences, progression or death. Variable selection was carried out using a multivariable fractional polynomial algorithm. Model validation was done by analyzing explained variation, c-statistic, calibration-in-the-large, and calibration

slope. These measures were optimism corrected and were calculated for an external cohort.

In contrast to most of the previously mentioned papers, this thesis encompasses research on "real-life data" from sixty primary care physicians' offices in Austria. 700,000 electronic health records were available, and this paper's scope covers the approximately 31,000 diabetics found in these health records. In order to increase comfort and reduce costs, early CKD indications should be identified using only parameters collected in normal clinical routines and should not require any other extensive screening. To find out whether a patient was treated adequately in the sense of satisfactory monitoring of a potential kidney disease, a model was created to predict whether the eGFR value (estimated glomerular filtration rate) was determined for any given patient. A logistic LASSO regression that determines the influential factors - e.g. the diagnosis hypertension - was used and this model is referred to as the "care model" in the remainder of this paper. Another model - the "progression model" - was developed to predict a decline in kidney function. As our data was longitudinal, dynamic modeling was an obvious option of analysis. C-statistics, optimism corrected c-statistics, explained variation (R^2) and calibration plots with confidence intervals were used for model validation.

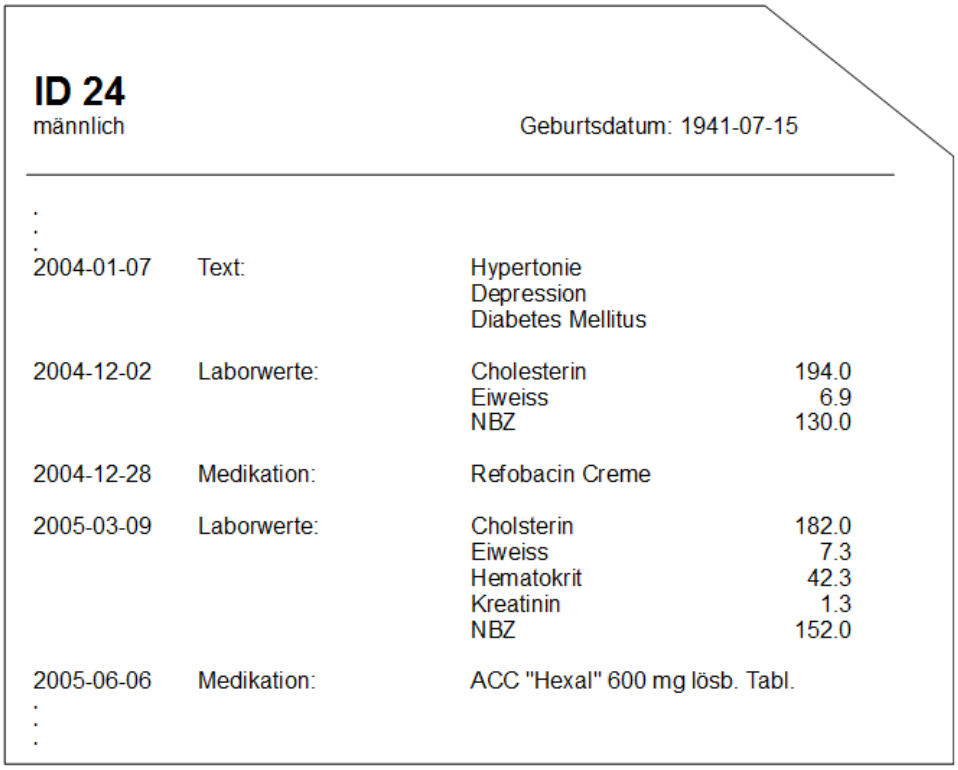
Following this introductory chapter, Chapter 2 provides a data description. Chapter 3 deals with the theoretical background and methods sourrounding the model development. Chapter 4 offers an overview of the model validation applied. Results are presented in Chapter 5, while Chapter 6 provides a discussion.

2 Data description

This chapter contains a description of the data. The data base used consists of more than 700,000 patients and has been recorded by sixty primary health care physicians in Austria.

A typical Austrian general practitioner's office houses a large collection of patient record cards. A record card typical for those available in the database may look as follows:

Figure 1: **Typical patient's record card**



ID 24
männlich
Geburtsdatum: 1941-07-15

2004-01-07	Text:	Hypertonie	
		Depression	
		Diabetes Mellitus	
2004-12-02	Laborwerte:	Cholesterin	194.0
		Eiweiss	6.9
		NBZ	130.0
2004-12-28	Medikation:	Refobacin Creme	
2005-03-09	Laborwerte:	Cholsterin	182.0
		Eiweiss	7.3
		Hematokrit	42.3
		Kreatinin	1.3
		NBZ	152.0
2005-06-06	Medikation:	ACC "Hexal" 600 mg lösb. Tabl.	

A record card documents all relevant information about a patient. Each person has his/her own ID, and birthdate and sex are noted. A date and further details are recorded for each event. Three categories of information per event are documented: diagnoses, laboratory measurements (type of blood parameter and measured value) and medication (ATC code that corresponds to a certain drug - this code is detailed later in this chapter). After preprocessing, the data base was organized in four tables, extracts of which are shown in Figure 2 including the patient no. 24 of Figure 1.

Figure 2: **Data base**

Table 1

	PatID	OrdID	birthdate	zip-code	sex	height	weight	province
1	4	1	10121920	1030	M	0	0	1
2	7	1	11011945	1020	M	0	0	1
3	11	1	31121915	1030	W	0	0	1
4	19	1	26071915	1030	W	0	0	1
5	24	1	15071941	1030	M	0	0	1
6	33	1	19031932	1030	M	0	0	1

Table 2

	LabKey	PatID	SystemDate	val	label
1	2	24	2004-12-02	194.0	cholesterol
2	3	24	2004-12-02	6.9	protein
3	7	24	2004-12-02	130.0	FBSL
4	2	24	2005-03-09	182.0	cholesterol
5	3	24	2005-03-09	7.3	protein
6	4	24	2005-03-09	42.3	hematocrit

Table 3

	PatID	SystemDate	ATCCodeGroup
1	24	2004-12-28	D06
2	24	2005-06-06	R05
3	24	2005-09-15	A10
4	24	2005-10-13	C01
5	24	2006-01-10	A02
6	24	2006-10-24	J01

Table 4

	PatID	SystemDate	Text
1	24	2004-01-07	Depression
2	24	2004-01-07	Hypertonie
3	24	2004-01-07	Diabetes mellitus
4	24	2005-10-13	Depression
5	24	2005-10-13	Metabolisches syndrom
6	24	2005-11-08	Depression

Table 1 is the reference table and contains patient ID, physicians' office ID, birthdate, zipcode, sex, province, height and weight of a patient. Height and weight are often missing (about 89 %).

Table 2 comprises laboratory measurements. The table consists of patient ID, the date on which the value was measured, a key which refers to the type of the laboratory value and the value itself. Laboratory values comprise calcium, cholesterol, protein, hematocrit, haemoglobin, creatinine, fasting blood sugar level (FBSL), phosphate, and triglycerides.

Table 3 refers to a patient's treatment with medicines. It contains patient ID, the date on which a medication was prescribed and the name of the medication. These names were assigned to ATC codes by Alexander Kainz. ATC stands for Anatomical Therapeutic Chemical and ATC codes are five-layered classifiers for drugs regarding the organ or system on which they act and their therapeutic, pharmacological, and chemical properties. Fourteen anatomical main groups form the first level. The second level is made up of therapeutical subgroups, and the third level consists of pharmagological subgroups. Chemical subgroups are distinguished at the fourth level, and the fifth is broken down into chemical substance groups. Table 2 shows a complete example representing the different levels of the metformin drug (WHO Collaborating Centre for Drug Statistics Methodology, 2011).

Table 2: **ATC codes** (reproduced from WHO Collaborating Centre for Drug Statistics Methodology, 2011)

Code	Level	Description
A	1	Alimentary tract and metabolism
A10	2	Drugs used in diabetes
A10B	3	Blood glucose lowering drugs, excl. insulin
A10BA	4	Biguanides
A10BA02	5	Metformin

The last table in the data base contains patient diagnoses. It contains patient ID, the date on which the diagnosis was made and a note made by the general practitioner. These

notes were written in free text and could only be used after proper classification into diagnoses. Diabetes mellitus, kidney disease and cardiovascular disease diagnoses trigger CKD. These diagnoses were identified in the free text and assigned by Milan Hronsky to their corresponding ICD10 codes. ICD is an abbreviation for International Classification of Diseases, and 10 stands for the 10^{th} version. It classifies diseases, disorders, injuries and further health conditions (World Health Organization, 2014).

In summary, these two important preprocessing steps allowed to make the data useable for the purpose on this thesis.

Only data on patients aged between 20 and 85 years who were diagnosed with diabetes mellitus were used in the two models - care and progression. Thus, we extracted all records or patients up from the date at which a diabetes diagnosis was present in the data base.

3 Model development

This section deals with the logistic and LASSO regression methods which are used for care and progression models. Data preparation for dynamic modeling and other variables are also discussed, as they play a vital role prior to the actual modeling.

3.1 Logistic regression

This first subsection details logistic regression and is based on an internal project report by Heinze (2012) entitled "Application of the R package glmnet for development of prediction models and for identification of biomarkers for CKD incidence or progression".

An n x p matrix with p independent variables, called X, is considered, where n denotes the number of observations. The vector y is a dichotomous response variable. The logistic regression model is then defined as:

$$P(Y = 1) = \pi = \frac{1}{1 + e^{-\beta_0 - \beta_1 X_1 - ... - \beta_p X_p}} \tag{2}$$

This equation gives the probability of $Y = 1$ as a function of the variables $X_1, ...X_p$ and the coefficients $\beta_0, ..., \beta_p$. β_0 is a constant and determines the model such that $P(Y = 1|X = 0) = \frac{1}{1 + e^{-\beta_0}}$. That means, β_0 are the log-odds for $Y = 1$.

The likelihood function needs to be maximized to find the parameters $\beta_0, ..., \beta_p$:

$$L(\beta|X, Y) = \prod_{i=1}^{n} \pi_i^{y_i} (1 - \pi_i)^{1-y_i} \tag{3}$$

Alternatively, the log likelihood function can also be maximized:

$$l(\beta|X, Y) = \sum_{i=1}^{n} y_i \log(\pi_i) + (1 - y_i) log(1 - \pi_i) \tag{4}$$

whereby $\pi_i = \frac{1}{1 + e^{-\beta_0 - \beta_1 x_{i1} - ... - \beta_p x_{ip}}}$.

The first derivatives of (4) can be set to zero to maximize the likelihood function and to

obtain the estimators for β:

$$\begin{aligned} \frac{\partial l(\beta|X,Y)}{\partial \beta_j} &= \sum_{i=1}^{n}(y_i - \pi_i)x_{ij} = 0 \\ \frac{\partial l(\beta|X,Y)}{\partial \beta_0} &= \sum_{i=1}^{n}(y_i - \pi_i) = 0 \end{aligned} \tag{5}$$

3.2 LASSO regression

LASSO is a relatively new method for variable selection and model development. LASSO is an abbreviation for "least absolute shrinkage and selection operator". As the name suggests, it shrinks coefficients and may set some of them to 0. Thus, it combines subset selection and ridge regression. (Tibshirani, 1996, p. 267)

For logistic LASSO regression, the same considerations and assumptions from the previous chapter on logistic regression are used. For practical purposes, however, all variables (X_i's) need to be standardized (Heinze, 2012). The following penalized log likelihood is then maximized:

$$l_{LASSO}(\beta|X,Y) = \sum_{i=1}^{n} y_i \log(\pi_i) + (1 - y_i)log(1 - \pi_i) - \lambda \sum_{j=1}^{p} |\beta| \tag{6}$$

The last term is a regularization term or penalty and penalizes large coefficients such that they are shrunk towards 0 or are set exactly to 0. β_0 is not included in this term. Even situations where $p > n$ can be handled with this approach. (Heinze, 2012)

In Equation 6, λ is a tuning parameter which is usually estimated using cross-validation techniques, and controls the relative weight of the penalty.

Large values of λ penalize the log likelihood heavily, and will lead to many coefficients set to zero. By contrast, if $\lambda = 0$ no penalization is applied and this is equivalent to maximum likelihood estimation.

Various methods are proposed to find an optimal value for λ.

For example, k-fold cross-validation is often used to optimize some criterion of predictive

accuracy. In the R package "glmnet" the deviance, the misclassification error or the c-index can be optimized (Friedman et al., 2014). Two standard possibilities to choose the optimal lambda are available in glmnet - "lambda.min" and "lambda.1se". Lambda.min entitles the minimal λ and lambda.1se stands for the largest value of λ which leads to a criterion that is within one standard error from the optimal value. The latter tries to avoid overfitting and selects fewer variables than lambda.min.

It is also possible to minimize the Akaike Information Criterion (AIC) (Akaike, 1973) or Bayesian Information Criterion (BIC) (Schwarz, 1978) without cross-validation and select the model with the smallest value of the criterion. In general, smaller models are obtained using the BIC criterion than using AIC (Zou et al., 2007, p. 2182). Multiple records on a patient, however, cannot be considered using the AIC and BIC criterion - a disadvantage.

Another alternative is to use stability selection. A set of regularization parameters is chosen and many samples of the data are bootstrapped. A LASSO regression is then run with the same series of λ for all bootstrap samples, and the results are the selection sets. The variables with a high selection probability are kept and those that have a low selection probability are rejected. The cutoff value at which to choose variables is a tuning parameter that must be determined. (Meinshausen & Bühlmann, 2010, pp. 421-423)

The method described first - a 20-fold cross-validation to optimize λ by the deviance - is used for this project. The misclassification error and the c-index were also considered as criteria. The function cv.glmnet() from the R package "glmnet" was used to carry out the cross-validation process.

3.3 Dynamic modeling

The raw data set described in the previous chapter required a restructuring prior to being ready to use for modeling.

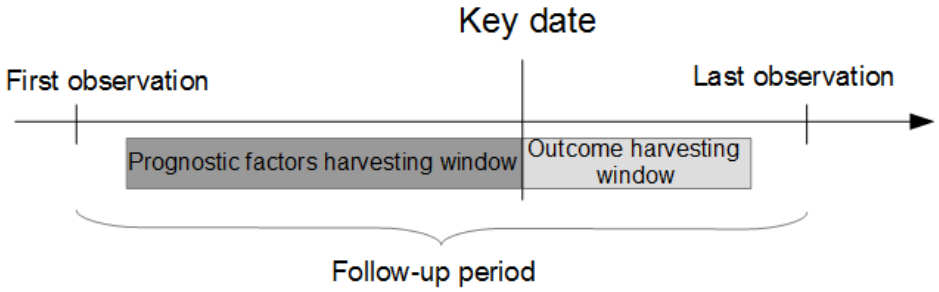
Usually, several prognostic factors are available at a common baseline date which is clearly defined for all subjects. A binary outcome variable can then be modeled using these baseline prognostic factors. In our study, a common baseline was not available, and thus, a dynamic modeling approach was chosen. This approach uses a floating baseline

("key") date, and can make use of multiple observations on the same patient.

3.3.1 Treatment of time

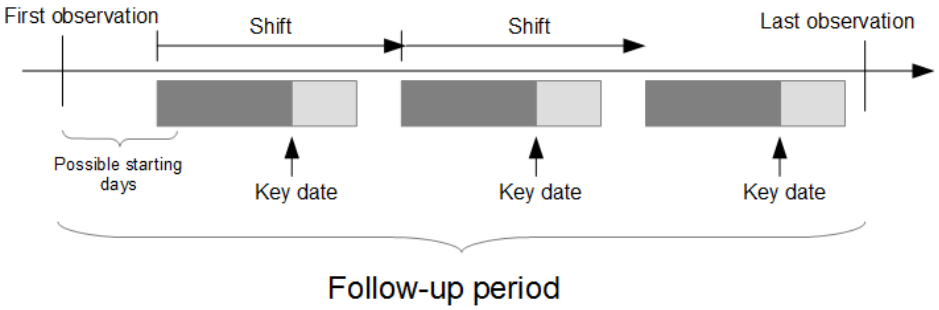
The previously mentioned floating baseline date or key date divides the time axis into two parts, one to collect prognostic factor information on a patient, and one to observe the outcome. More specifically, this is done by defining fixed lengths for the "prognostic factors harvesting window" and "outcome harvesting window", respectively, as illustrated in Figure 3.

Figure 3: **Time periods of a patient**



Patients often have a long observation period, which means that they went to the same doctor for a longer term, and a given patient may thus be considered several times. Prognostic factors and outcome harvesting windows within the follow-up period were therefore selected and shifted as often as possible. That is, the entire follow-up period of each patient was divided by a fixed time lag, and the number of possible shifts was determined. The modulo of this division is the days which were used to randomize the start of the prognostic factors harvesting window at the beginning of the follow-up period. Figure 4 shows this approach.

Figure 4: **Division of a patient's follow-up period into several records**



Thus, each patient may be considered several times if he/she visits his/her doctor over a longer period of time, and a wealth of information can be collected from all those visits.

3.3.2 Harvesting prognostic factors

All variables measured during the prognostic factors harvesting window are integrated such that a single static observation on these variables can be used for modeling. Specifically, this integration was done as follows:

- Sex was taken from the reference table.
- Age was defined as the difference between the key date and the birthday as recorded in the reference table.
- Medicines were categorized into level-2 ATC codes (drug classes). For each such ATC code, a value of 1 was assigned if any medicine with this code was at least once prescribed in the prognostic factors harvesting window, and 0 otherwise.
- Diagnoses which were considered here were type of diabetes mellitus, kidney diseases and circulatory system diseases, all represented by ICD10 codes (see Table 8). The value of the corresponding binary variables was set to 1 if a diagnosis was at least once recorded, 0 otherwise. Chronic diseases required special consideration, which is detailed later.
- A number of options exist as to how the blood parameters may be used in the model. The laboratory measurements were considered as binary and numeric variables. The binary variables show whether a laboratory value was measured in a certain period, and the numeric variables give the corresponding value. If a laboratory

value was measured more than once in a window, the most recent value was taken. The numeric variable was set to missing, if no measurement was available.

Another possibility is to handle numeric longitudinal data using a mixed model. This approach allows the progress of laboratory values to be modeled. As there were few blood parameter measurements in this project, this method was not deemed suitable. Another alternative considering blood parameters, medical information or diagnoses is to measure the time elapsed since last observation (TEL), as the time since the last measurement was observed is different for each patient. This approach may be applied by including two further variables, and could also be used for medication and diagnoses. The first covariate is an interaction term between the last recorded value and a function of TEL and illustrates the decreasing effect of older observations. The second variable is a function of TEL and shows that patients without complications may not be observed as often as patients with complications. (de Bruijne et al., 2003, pp. 449-450) This method is more complicated and leads to doubling of variables, so the first approach was applied.

3.3.3 Outcome variables

Care and *Progression* are the two outcome variables analyzed. *Care* is a binary variable and states whether creatinine was measured in the outcome harvesting window (1) or not (0). *Progression* is a binary variable and defines whether there was a decline in kidney function or not. The variable was determined by comparing the clinical stages of eGFR within the prognostic factors harvesting window and the outcome harvesting window. Only observations of patients with measurements of creatinine in both windows were considered in the progression model.

3.3.4 Considerations for modeling

It is necessary to take multiple observations of the same patient into account. In Chapter 3.2 on LASSO regression a 20-fold cross-validation to determine the optimal variables was chosen for the modeling process. Since some patients may have appeared multiple times in the data set, the patients - not the records - were randomized to 20 groups - the "folds". These "folds" were used in the cross-validation process, and the assigned fold number were passed to the foldid parameter of the cv.glmnet() function (Friedman et al., 2014).

An alternative modeling approach could be a Cox proportional hazard model, which models the time until an event occurs - e.g. a decline in kidney function. This approach was not used because it was not deemed very informative in this case. It was assumed that people tend to visit their doctor when they feel unwell or on a regular basis.

3.4 Representation of prognostic factors

Based on prognostic factors detailed in Chapter 3.3.2 a number of design variables were created and included in the design matrix.

3.4.1 Continuous prognostic factors

This subsection deals with the derivation of design variables from continuous prognostic factors.

In addition to *age* and the binary variables of laboratory measurements indicating availability of a measurement in the prognostic factors harvesting window, other continuous variables were generated. The eGFR value was calculated from creatinine measure, age and sex using the CKD-EPI formula (1). The number of creatinine measurements was also a design variable.

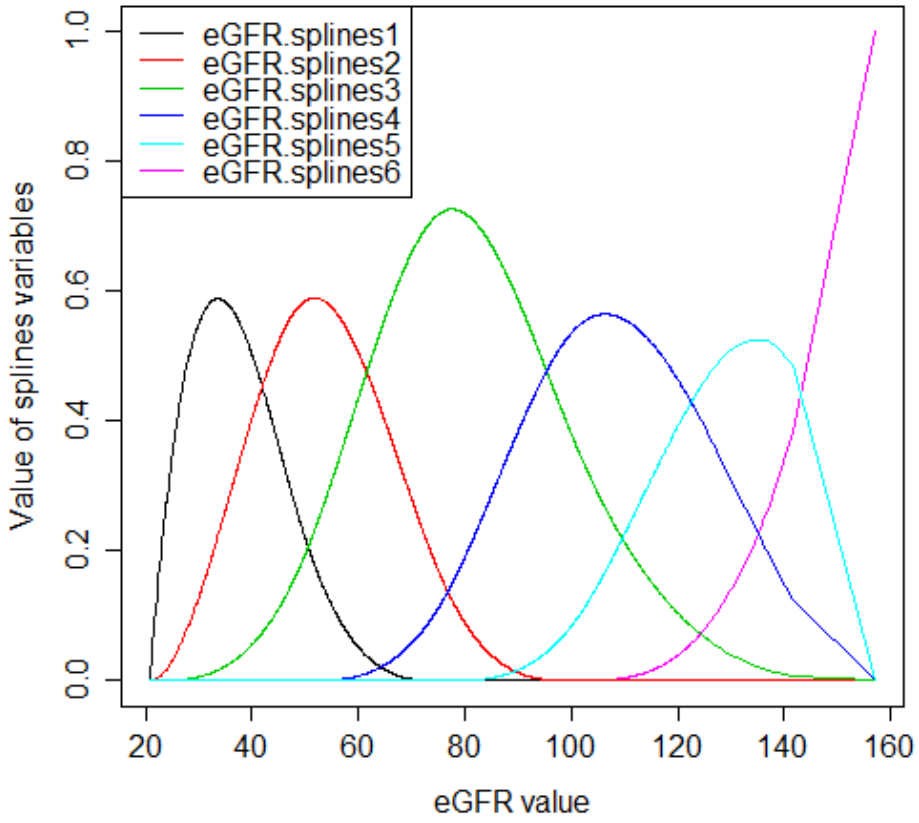
If a laboratory value was not available in the prognostic factors harvesting window, then the mean of the values of that variable was imputed. This enables the interpretation of the binary variable for a laboratory measurement as the expected difference in outcome (log odds of an event) between a patient with an average measured value and a patient without a measurement.

Moreover, implausible outliers were removed on a case-by-case basis. All outliers remaining were winsorized, i.e., all values below the 1^{st} percentile were set to the 1^{st} percentile, and all values above the 99^{th} percentile were set to the 99^{th} percentile.

In order to accommodate non-linear effects of *age*, *eGFR* and all laboratory values, B-splines were used (Hastie et al., 2009, pp. 186). For the care model, polynomials of the third degree with knots at the 0.1, 0.5 and 0.9 percentiles were generated for all previously mentioned prognostic factors. A second degree polynomial with knots at the 0.3 and 0.7 percentiles was used for the progression model in order to use less degrees of freedom

than in the care model because of fewer records. Figure 5 shows a splines example - the six spline bases for eGFR used in the care model, i.e., the six design variables used to represent a non-linear effect of eGFR in the care model.

Figure 5: **eGFR splines**



Since the LASSO approach can deal with multicollinear design variables, we also included the untransformed continuous variables to allow for the possibility of a linear effect. Thus, with this approach the LASSO can be used to add just local non-linearities to an assumed linear association with the outcome.

3.4.2 Binary variables

Many design variables were derived from binary prognostic factors.

Both the level-1 and the level-2 ATC codes were included as binary design variables. The binary values define whether the patient was taking a medication or not in the prognostic factors harvesting window.

Also diagnosis information was considered as binary design variables. A dummy variable was created for every diagnosis and states whether this diagnosis had been detected or not. If the disease was classified chronic the variable's value was set to 1 since it was detected the first time. Diabetes mellitus is a chronic disease and comprises ICD10 codes E10 to E14. These codes have ten equally defined subgroups, each concerns complications, and one variable per complication was included in the design matrix. E10 is the code for diabetes mellitus type I, so all type I diabetics were mapped to this variable. ICD10 codes N03 to N19 and N26 relate to kidney disease and fall into the "disease of the genitourinary system" category, whereby some of the diseases are chronic and others are acute. For each of the ICD10 codes a variable was created. Circulatory system disease span ICD10 codes I00, I10-I12, I15, I20, I21, I25, I42, I48, I50, I63, I63.8 and E78 and only I21 and I63 belongs to acute diseases. For each of these codes, a variable was again created.

In addition to the design variables of eGFR outlined in the previous subsection, we also used the clinical stages of eGFR (see Table 1) as an ordinal variable. An ordinal coding was used to define binary design variables (dummy variables). Three variables - *dummy.GFR1*, *dummy.GFR2* and *dummy.GFR3* - were used to compare the groups "0-2" versus "3a, 3b and 4", "0-2 and 3a" versus "3b and 4" and "0-2, 3a and 3b" versus "4". The same ordinal coding was applied to the duration of diabetes mellitus. Two variables - *dummy.diabetes1* and *dummy.diabetes2* - were created. The first one contrasts a diabetes duration of more than one year versus less than one year. The second variable dichotomized diabetes duration at 5 years.

To restrict the numerous binary design variables, a minimum prevalence for the effects was required. Different minimum prevalences were tried out and the results of model performance were compared. A minimal prevalence for binary design variables were chosen at 1 %.

3.4.3 Interaction terms

In addition to all previously described variables, we also defined interaction (product) terms. Interaction terms were composed of selected binary design variables of laboratory

values, ATC codes, and diagnoses. These design variables were also binary and were each built from two main effects. An interaction variable is 1 if both main effects were 1 (present), and it is 0 if only one main effect or none were 1.

The number of interaction terms was also restricted by a minimum prevalence whereby, first, different values were tried out, and finally, the decision fell on 5 %.

3.4.4 Model matrices

The meta parameter length of the prognostic factors harvesting window and of the outcome harvesting window, shift size and the minimum prevalences for binary variables and interaction terms have a great influence on the dimensions of the design matrix. We used different values of these meta parameters to evaluate whether these decisions had impact on final results.

As shown in Table 3, four matrices with different window parameters were built. The minimum prevalences remained fixed at 1 % for binary variables and 5 % for interaction terms.

Table 3: **Different window parameters**

	Matrix 1	Matrix 2	Matrix 3	Matrix 4
Harvesting window (in days)	365	365	730	1095
Outcome window (in days)	180	365	365	365
Shift size (in days)	730	730	1095	1460

Prevalences were also altered while the window parameters were fixed at 365 days for the harvesting and outcome windows (Table 4). The shift size was set to 730 days.

Table 4: **Different window**

	Matrix 1	Matrix 2	Matrix 3	Matrix 4
Minimum prevalence binary variables	10 %	5 %	1 %	1 %
Minimum prevalence interaction terms	20 %	10 %	5 %	1 %

Chapter 5 on results provides an overview of the comparison between the performances of the different design matrices.

3.5 List of design variables

Tables 5-8 list design variables created by different groups of prognostic factors and their description. According to medication, the number of design variables created by level-2 ATC codes is only stated in brackets in the table. Additionally, interaction terms mentioned in the previous chapter were considered as design variables and are also not listed. These design variables are combinations of two binary variables and are connected by "_", e.g. *M01_C*.

Table 5: **Design variables extracted from demographics**

Demographics	Design variable names	Description
sex	sex	1 male
		0 female
age	age	age in years at keydate
	age.splines1	value of first spline basis for age
	age.splines2	value of second spline basis for age
	age.splines3	value of third spline basis for age
	age.splines4	value of fourth spline basis for age
	age.splines5	value of fifth spline basis for age
	age.splines6	value of sixth spline basis for age

Table 6: **Design variables extracted from medication**

Medication level-1 ATC code	Design variable names	Description
A	A	1 drug A was prescribed
	(and 14 level-2 subgroups)	0 drug A was not prescribed
B	B	1 drug B was prescribed
	(and 5 level-2 subgroups)	0 drug B was not prescribed
C	C	1 drug C was prescribed
	(and 9 level-2 subgroups)	0 drug C was not prescribed
D	D	1 drug D was prescribed
	(and 11 level-2 subgroups)	0 drug D was not prescribed
G	G	1 drug G was prescribed
	(and 4 level-2 subgroups)	0 drug G was not prescribed
H	H	1 drug H was prescribed
	(and 5 level-2 subgroups)	0 drug H was not prescribed
J	J	1 drug J was prescribed
	(and 5 level-2 subgroups)	0 drug J was not prescribed
L	L	1 drug L was prescribed
	(and 4 level-2 subgroups)	0 drug L was not prescribed
M	M	1 drug M was prescribed
	(and 5 level-2 subgroups)	0 drug M was not prescribed
N	N	1 drug N was prescribed
	(and 7 level-2 subgroups)	0 drug N was not prescribed
P	P	1 drug P was prescribed
	(and 2 level-2 subgroups)	0 drug P was not prescribed
R	R	1 drug R was prescribed
	(and 5 level-2 subgroups)	0 drug R was not prescribed
S	S	1 drug S was prescribed
	(and 3 level-2 subgroups)	0 drug S was not prescribed
V	V	1 drug V was prescribed
	(and 2 level-2 subgroups)	0 drug V was not prescribed

Table 7: **Design variables extracted from laboratory measurements**

Laboratory measurements	Design variable names	Description
Calcium	calcium	numeric laboratory value of calcium
	calcium.binary	1 calcium was measured 0 calcium was not measured
	calcium.splines1	value of first spline basis for calcium
	calcium.splines2	value of second spline basis for calcium
	calcium.splines3	value of third spline basis for calcium
	calcium.splines4	value of fourth spline basis for calcium
	calcium.splines5	value of fifth spline basis for calcium (only available for care model)
	calcium.splines6	value of sixth spline basis for calcium (only available for care model)
Cholesterol	cholesterol	numeric laboratory value of cholesterol
	cholesterol.binary	1 cholesterol was measured 0 cholesterol was not measured
	cholesterol.splines1	value of first spline basis for cholesterol
	cholesterol.splines2	value of second spline basis for cholesterol
	cholesterol.splines3	value of third spline basis for cholesterol
	cholesterol.splines4	value of fourth spline basis for cholesterol
	cholesterol.splines5	value of fifth spline basis for cholesterol (only available for care model)
	cholesterol.splines6	value of sixth spline basis for cholesterol (only available for care model)
Creatinine	creatinine.binary	1 creatinine was measured 0 creatinine was not measured
	creatinine.number	number of creatinine measurements
	GFR	eGFR in $mL/min/1.73m^2$
	GFR.splines1	value of first spline basis for GFR
	GFR.splines2	value of second spline basis for GFR
	GFR.splines3	value of third spline basis for GFR
	GFR.splines4	value of fourth spline basis for GFR
	GFR.splines5	value of fifth spline basis for GFR (only available for care model)
	GFR.splines6	value of sixth spline basis for GFR (only available for care model)
	dummy.GFR1	1 if stage worse than stage 2
	dummy.GFR2	1 if stage worse than stage 3a
	dummy.GFR3	1 if stage worse than stage 3b
FBSL	FBSL	numeric laboratory value of FBSL
	FBSL.binary	1 FBSL was measured 0 FBSL was not measured
	FBSL.splines1	value of first spline basis for FBSL
	FBSL.splines2	value of second spline basis for FBSL
	FBSL.splines3	value of third spline basis for FBSL
	FBSL.splines4	value of fourth spline basis for FBSL
	FBSL.splines5	value of fifth spline basis for FBSL (only available for care model)
	FBSL.splines6	value of sixth spline basis for FBSL (only available for care model)
Haemoglobin	haemoglobin	numeric laboratory value of haemoglobin
	haemoglobin.binary	1 haemoglobin was measured 0 haemoglobin was not measured
	haemoglobin.splines1	value of first spline basis for haemoglobin
	haemoglobin.splines2	value of second spline basis for haemoglobin
	haemoglobin.splines3	value of third spline basis for haemoglobin
	haemoglobin.splines4	value of fourth spline basis for haemoglobin
	haemoglobin.splines5	value of fifth spline basis for haemoglobin (only available for care model)
	haemoglobin.splines6	value of sixth spline basis for haemoglobin (only available for care model)

Hematocrit	hematocrit	numeric laboratory value of hematocrit
	hematocrit.binary	1 hematocrit was measured
		0 hematocrit was not measured
	hematocrit.splines1	value of first spline basis for hematocrit
	hematocrit.splines2	value of second spline basis for hematocrit
	hematocrit.splines3	value of third spline basis for hematocrit
	hematocrit.splines4	value of fourth spline basis for hematocrit
	hematocrit.splines5	value of fifth spline basis for hematocrit
		(only available for care model)
	hematocrit.splines6	value of sixth spline basis for hematocrit
		(only available for care model)
Phosphate	phosphate	numeric laboratory value of phosphate
	phosphate.binary	1 phosphate was measured
		0 phosphate was not measured
	phosphate.splines1	value of first spline basis for phosphate
	phosphate.splines2	value of second spline basis for phosphate
	phosphate.splines3	value of third spline basis for phosphate
	phosphate.splines4	value of fourth spline basis for phosphate
	phosphate.splines5	value of fifth spline basis for phosphate
		(only available for care model)
	phosphate.splines6	value of sixth spline basis for phosphate
		(only available for care model)
Triglycerides	triglycerides	numeric laboratory value of triglycerides
	triglycerides.binary	1 triglycerides was measured
		0 triglycerides was not measured
	triglycerides.splines1	value of first spline basis for triglycerides
	triglycerides.splines2	value of second spline basis for triglycerides
	triglycerides.splines3	value of third spline basis for triglycerides
	triglycerides.splines4	value of fourth spline basis for triglycerides
	triglycerides.splines5	value of fifth spline basis for triglycerides
		(only available for care model)
	triglycerides.splines6	value of sixth spline basis for triglycerides
		(only available for care model)

Table 8: **Design variables extracted from diagnoses**

Diagnoses	Design variable names	Description
Diabetes	dummy.diabetes1	1 diabetes mellitus onset more than a year ago
		0 diabetes mellitus onset less than a year ago
	dummy.diabetes2	1 diabetes mellitus onset more than five years ago
		0 diabetes mellitus onset less than five years ago
E10	DM.type1	1 suffered from diabetes mellitus type I
		0 did not suffer from diabetes mellitus type I
E10.0-E10.6	DM.comp0	1 had coma
E11.0-E11.6		0 no coma
E14.0-E14.6	DM.comp2	1 had renal complications
		0 no renal complications
	DM.comp3	1 had ophthalmic complications
		0 no ophthalmic complications
	DM.comp4	1 had neurological complications
		0 no neurological complications
	DM.comp5	1 had peripheral circulatory complications
		0 no peripheral circulatory complications
	DM.comp6	1 had other specified complications
		0 no other specified complications
	DM.comp7	1 had multiple complications
		0 no multiple complications
	DM.comp9	1 no complications were described
		0 any complications were described

E78	e78	1 had disorders of lipoprotein metabolism and other lipidaemias
		0 no such diagnosis recorded
I00	i00	1 had rheumatic fever without mention of heart involvement
		0 no such diagnosis recorded
I10	i10	1 had essential (primary) hypertension
		0 no such diagnosis recorded
I11	i11	1 had hypertensive heart disease
		0 no such diagnosis recorded
I11.0	i11.0	1 had hypertensive heart disease with (congestive) heart failure
		0 no such diagnosis recorded
I12	i12	1 had hypertensive renal disease
		0 no such diagnosis recorded
I12.0	i12.0	1 had hypertensive renal disease with renal failure
		0 no such diagnosis recorded
I15	i15	1 had secondary hypertension
		0 no such diagnosis recorded
I20	i20	1 had angina pectoris
		0 no such diagnosis recorded
I21	i21	1 had acute myocardial infarction
		0 no such diagnosis recorded
I25	i25	1 had chronic ischaemic heart disease
		0 no such diagnosis recorded
I42	i42	1 had cardiomyopathy
		0 no such diagnosis recorded
I48	i48	1 had atrial fibrillation and flutter
		0 no such diagnosis recorded
I50	i50	1 had heart failure
		0 no such diagnosis recorded
I63	i63	1 had cerebral infarction
		0 no such diagnosis recorded
I63.8	i63.8	1 had other cerebral infarction
		0 no such diagnosis recorded
N03	n03	1 had chronic nephritic syndrome
		0 no such diagnosis recorded
N08	n08	1 had glomerular disorders in diseases classified elsewhere
		0 no such diagnosis recorded
N08.3	n08.3	1 had glomerular disorders in diabetes mellitus
		0 no such diagnosis recorded
N10	n10	1 had acute tubulo-interstitial nephritis
		0 no such diagnosis recorded
N11	n11	1 had chronic tubulo-interstitial nephritis
		0 no such diagnosis recorded
N13	n13	1 had obstructive and reflux uropathy
		0 no such diagnosis recorded
N17	n17	1 had acute renal failure
		0 no such diagnosis recorded
N18	n18	1 had chronic kidney disease
		0 no such diagnosis recorded
N18.1	n18.1	1 had chronic kidney disease, stage 1
		0 no such diagnosis recorded
N18.2	n18.2	1 had chronic kidney disease, stage 2
		0 no such diagnosis recorded
N18.3	n18.3	1 had chronic kidney disease, stage 3
		0 no such diagnosis recorded
N18.4	n18.4	1 had chronic kidney disease, stage 4
		0 no such diagnosis recorded
N18.5	n18.5	1 had chronic kidney disease, stage 5
		0 no such diagnosis recorded
N26	n26	1 had unspecified contracted kidney
		0 no such diagnosis recorded

3.6 Importance of prognostic factors

The importance of prognostic factors or groups of prognostic factors can be quantified by using the proportion of explained variation (PEV), which can be presented as marginal PEV and partial PEV. S is defined as the whole set of prognostic factors, and I is a subset. The marginal PEV is the explained variation, R^2, of a model with the variable subset I and is abbreviated by R^2_I. The partial PEV for the subset I is defined as $R^2_{I|\{S\setminus I\}} = R^2_S - R^2_{S\setminus I}$. (Heinze & Schemper, 2003, p. 156) Here R^2 refers to the squared correlation of cross-validation predicted probabilities and binary outcomes.

Another method to rank prognostic factors by their importance is to use standardized regression coefficients. This method is only available, if only one design variable for each prognostic factor is used. We will employ this method later to rank the prognostic importance of the design variables that were selected in a model.

4 Model validation

Model validation is used to determine whether a model returns adequate results with patients who were not involved in the model development process (Altman & Royston, 2000, p. 454). According to the evaluation of our two models, the c-index and the explained variation (R^2) were considered. An optimism corrected c-index was calculated using internal validation. This was done by a 20-fold cross-validation using a different random number seed for randomizing patients. In each of 20 sub-analyses, 19 folds of the data were used to perform exactly the same algorithm in order to determine the tuning parameter and to estimate the coefficients of the model. In addition, the model was applied to all records of the hold-out fold. Then the predictions for all records with their true outcome status could be compared, randomly sampling exactly one record per patient. Finally, a cross-validated c-index was derived for measuring model performance labeled "optimism corrected c-index". It was also possible to plot a calibration curve and its confidence intervals with this approach by using the function calibrate.plot() from the "gbm" package (Ridgeway, 2014).

Smith et al. (2014) compared several methods to avoid optimistic predictions using cross-validation. Among their methods were cross-validation without and with replication, leave-one-out cross-validation and leave-pair-out cross-validation. Our method corresponds to cross-validation without replication. All other approaches performed better than cross-validation without replication in the article (Smith et al., 2014, p. 323). In contrast to our data base, they used very small data sets. As our data set was huge, we chose cross-validation without replication as it was the least computationally expensive method and was deemed to be adequate.

5 Results

This section contains all details about our two models, care and progression.

5.1 Care model

First, the baseline characteristics are reported. Second, an overview presents the results of the different model matrices. Finally, the results of the model development and evaluation are presented.

5.1.1 Baseline characteristics

Baseline characteristics are described for Matrix 2 of the meta parameter combination in Table 3 which are the final chosen meta parameters. The prognostic factors and outcome harvesting window are both set at 365 days, and shift size is 730 days. The minimum prevalences for binary variables is set at 1 %, and for interaction terms it is set at 5 %. The baseline characteristics should provide the reader an impression on the data and a better understanding of the results. One record of each patient is randomly chosen in order to not distort the figures. Then this model can be applied to patients with similar characteristics.

With this meta parameter constellation 9960 patients were included. Most of them were women (54.24 %) and the elderly. The patients were on average 62 years old and the age of all patients spanned 20 to 85 years. Half of all patients were between 53 and 74 years old. It is not surprising that the patients are older than the population average, as diabetes is commonly a geriatric disease.

Table 9 shows facts about laboratory measurements. The fasting blood sugar level, haemoglobin, creatinine and cholesterol were measured most frequently. The boxplots in Figure 6 show that there are many triglycerides and fasting blood sugar values which are too high in comparison with the normal value (Wiener Krankenanstaltenverbund, 2014; Regents of the University of Minnesota, 2006), which is probably because all patients were diabetics. All other blood parameters tend to lie in a normal area except for some creatinine values (Wiener Krankenanstaltenverbund, 2014; Regents of the University of Minnesota, 2006). On average, each patient had 0.28 measurements of creatinine. Three patients, however, had 12 measurements within a year.

Table 9: **Overview of laboratory values - care model**

Laboratory value	Prevalence	Min	Mean	Max	Sd
Calcium	2.66 %	2.10	4.25	10.17	2.47
Cholesterol	17.39 %	115.20	212.30	340.60	44.49
Protein	1.07 %	5.70	7.07	8.79	0.71
Hematocrit	9.50 %	30.96	42.10	51.60	4.22
Haemoglobin	18.41 %	10.40	14.17	17.40	1.41
Creatinine	18.01 %	0.50	0.97	2.02	0.26
FBSL	23.50 %	67.70	132.90	318.60	49.81
Trigycerides	9.52 %	46.00	172.00	685.70	111.55

Figure 6: **Boxplots of laboratory values - care model**

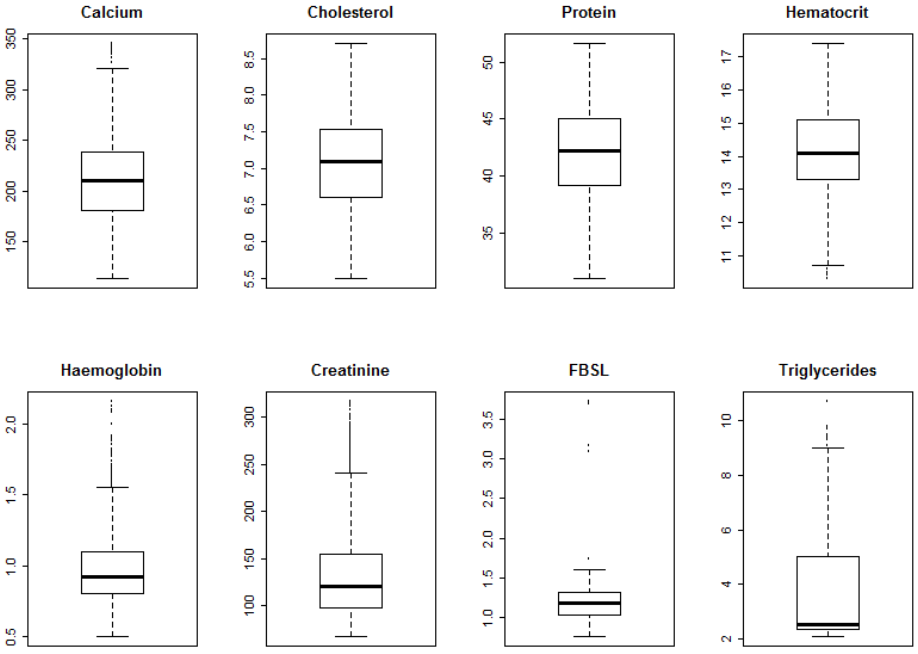


Table 10 lists the ten most frequent medications and their respective descriptions from the Website of WHO Collaborating Centre for Drug Statistics Methodology (2013). The prevalence of each individual medication is not high. M01 - antiinflamatory and antirheumatic products - is the leader of the table, which is understandable given that the patients tend to be older.

Table 10: **Most frequent medications - care model** (description reproduced from WHO Collaborating Centre for Drug Statistics Methodology, 2013)

ATC (level 2)	**Description level 1**	**Description level 2**	**Prevalence**
M01	Musculo-skeletal system	Antiinflamatory and antirheumatic products	11.25 %
J01	Antiinfectives for systemic use	Antibacterials for systemic use	8.92 %
A02	Alimentary tract and metabolism	Drugs for acid related disorders	6.24 %
M02	Musculo-skeletal system	Topical products for joint and muscular pain	5.19 %
R05	Respiratory system	Cough and cold preparations	5.10 %
C09	Cardio vascular system	Agents acting on the renin-angiotensin system	5.09 %
A10	Alimentary tract and metabolism	Drugs used in diabetes	4.34 %
N02	Nervous system	Analgesics	3.48 %
B01	Blood and blood forming organs	Antithrombotic agents	3.24 %
N06	Nervous system	Psychoanaleptics	3.03 %

Table 11 lists the prevalence of the different diagnoses. Patients frequently suffered from diabetes without complications, disorders of lipoprotein and other lipidemias (E78) and hypertension (I10). All other diagnoses were observed less often. The description of ICD10 refers to the homepage of World Health Organization (2010).

Table 12 shows that all records' eGFR values can be assigned to the first four groups of CKD stages, so no patient had a severe renal failure. Of note, the percentage of records with a measurement of creatinine is rather high in the groups 3a, 3b and 4.

Table 11: **Overview of diagnoses - care model** (description reproduced from World Health Organization, 2010)

Variable/ ICD code	**Description**	**Prevalence**
DM.type1	Type I diabetes	3.27 %
DM.comp2	Diabetes mellitus with renal complications	5.76 %
DM.comp3	Diabetes mellitus with ophthalmic complications	1.89 %
DM.comp4	Diabetes mellitus with neurological complications	4.81 %
DM.comp5	Diabetes mellitus with peripheral circulatory complications	1.51 %
DM.comp6	Diabetes mellitus with other complications	3.81 %
DM.comp7	Diabetes mellitus with multiple complications	3.14 %
DM.comp9	Diabetes mellitus without complications or any further information	86.07 %
N18	Chronic kidney disease	7.00 %
N18.1	Chronic kidney disease, stage 1	2.97 %
N08	Glomerular disorders in diseases classified elsewhere	1.03 %
E78	Disorders of lipoprotein metabolism and other lipidaemias	20.63 %
I00	Rheumatic fever without mention of heart involvement	2.00 %
I10	Essential (primary) hypertension	23.97 %
I20	Angina pectoris	2.32 %
I25	Chronic ischaemic heart disease	9.25 %
I42	Cardiomyopathy	2.79 %
I48	Atrial fibrillation and flutter	1.15 %
I50	Heart failure	1.26 %
I63.8	Cerebral infarction, other cerebral infarction	1.11 %

Table 12: **CKD stages by care outcome**

CKD stages	No care	Care	Total
0-2	28310 (82.5 %)	6010 (17.5 %)	34320 (96.1 %)
3a	284 (29.2 %)	689 (70.8 %)	973 (2.7 %)
3b	80 (22.9 %)	269 (77.1 %)	349 (1.0 %)
4	25 (29.8 %)	53 (70.2 %)	78 (0.2 %)
5	0	0	0

5.1.2 Comparison of different meta parameters

In this subsection different meta parameter combinations (see Chapter 3.4.4) are compared. Then the most appropriate constellation was chosen.

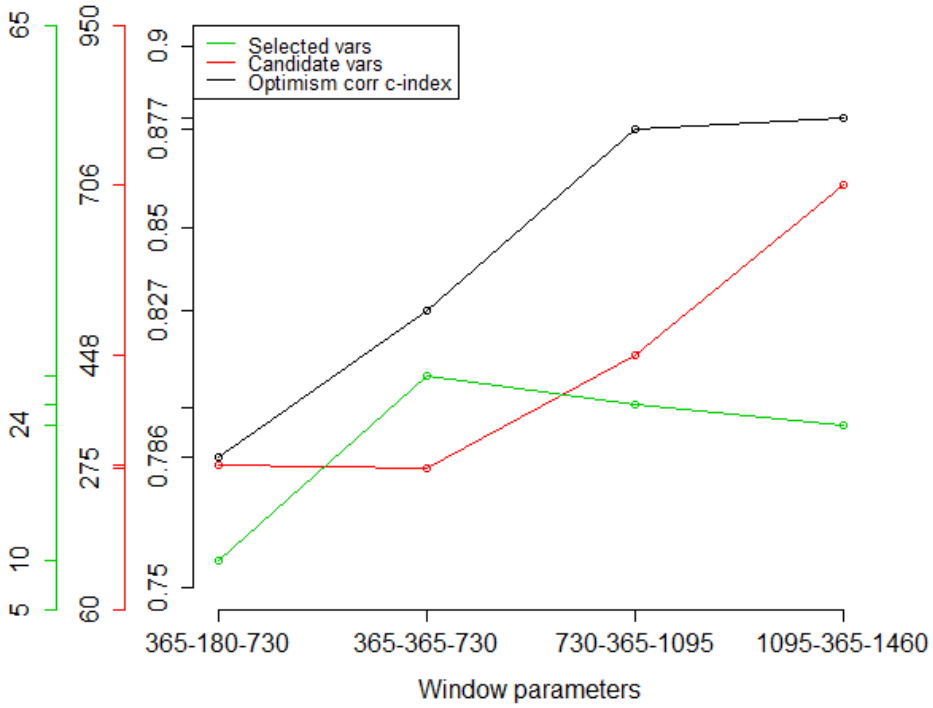
The following Table 13 shows an overview of the results of different meta parameters. The prevalances were fixed at 5 % for main effects and at 1 % for interaction effects.

Looking only at R^2 and the optimism corrected c-index in Table 13, Model 3 would be chosen but Model 2 is prefered because there are more patients in the design matrix and the measures remain good. All measures are illustrated by Figure 7. The different window parameters are on the x-axis. The first one is the size of the prognostic factors harvesting window in days, the second is the size of the outcome harvesting window in days and the last number is the shift size in days (see Figure 4). The optimism corrected c-index and the number of candidate design variables as well as the number of selected variables are displayed on the y-axis. The prognostically most important variables are ranked by their standardized coefficients. It is not surprising that the first two variables are an indicator for creatinine observed (*creatinine.binary*) and the number of creatinine measurements (*creatinine.number*). It is obviously more likely to find a measurement of creatinine when one already existed in the past. The many binary variables of laboratory values are noteworthy. This could be because doctors look at several blood parameters at the same time, so correlation may appear.

Table 13: **Comparison of results of different parameters - care model**

	Model 1	Model 2	Model 3	Model 4
Harvesting window (in days)	365	365	730	1095
Outcome window (in days)	180	365	365	365
Shift size (in days)	730	730	1095	1460
Number of patients	11949	9960	6504	4298
Number of records per patient	3.49	3.59	2.81	2.41
Percentage of care events	12 %	20 %	22 %	23 %
Number of candidate design variables	280	275	448	706
Number of selected design variables	10	29	26	24
Prognostically most important design variables (in decreasing order)	creatinine.binary creatinine.number i48_i63.8 i50_i63.8 hematocrit.binary_ i50	creatinine.binary creatinine.number cholesterol.binary hematocrit.binary_ creatinine.binary dummy.diabetes2	creatinine.number creatinine.binary cholesterol.binary haemoglobin.binary hematocrit.binary_ creatinine.binary	creatinine.number creatinine.binary haemoglobin.binary cholesterol.binary D06
R^2	0.210	0.336	0.400	0.385
C-index	0.795	0.830	0.882	0.886
Optimism corrected c-index	0.786	0.827	0.877	0.880

Figure 7: **Comparison of window prameters - care model**



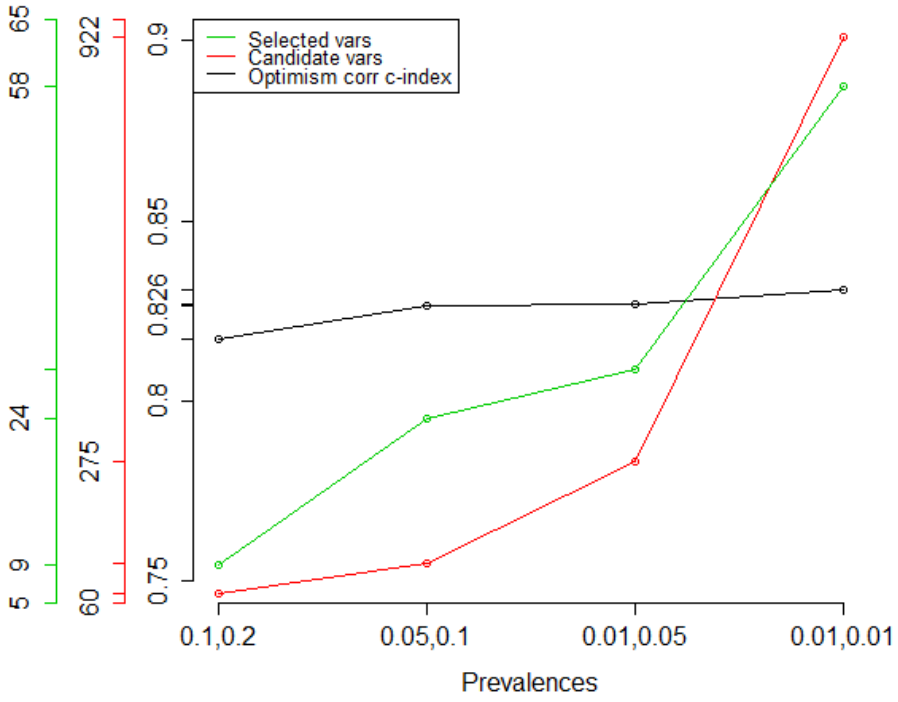
Various minimum prevalences were then analyzed for fixed window parameters. The prognostic factors harvesting window was 365 days, the outcome harvesting window was also 365 days and the shift size was 730 days. Table 14 and Figure 8 provide an overview of the results.

The minimum prevalences of 1 % for binary variables and of 5 % for interaction terms were chosen because the resulting model produced good results measured by R^2 and the optimism corrected c-index. It also does not include as many variables in the model as the model with more stringent restrictions. Figure 8 depicts the minimum prevalences of binary variables and interaction terms on the x-axis.

Table 14: **Comparison of results of different prevalences - care model**

Minimum prevalence binary variables	10 %	5 %	1 %	1 %
Minimum prevalence interaction terms	20 %	10 %	5 %	1 %
Number of candiate design variables	74	121	275	922
Number of selected design variables	9	24	29	58
Prognostically most important design variables (in decreasing order)	creatinine.binary creatinine.number haemoglobin.binary cholesterol.binary dummy.diabetes2	creatinine.binary creatinine.number cholesterol.binary dummy.diabetes2 hematocrit.binary_ haemoglobin.binary	creatinine.binary creatinine.number cholesterol.binary hematocrit.binary_ creatinine.binary dummy.diabetes2	creatinine.binary creatinine.number cholesterol.binary dummy.diabetes2 hematocrit.binary_ creatinine.binary
R^2	0.316	0.338	0.336	0.342
c-index	0.821	0.830	0.830	0.835
Optimism corrected c-index	0.817	0.826	0.827	0.831

Figure 8: **Comparison of prevalences - care model**



In the following, the meta parameter combinations of Model 2 of Table 13 was chosen and the analysis results are shown in the next subsection. In this constellation, the harvesting and outcome windows were set at 365 days, and the shift size was 730 days. The minimum prevalence for binary variables was set at 1 %, while the prevalences for interaction term was at least 5 %.

5.1.3 Presentation of the model

This subsection relates to model results and performance measures. As discussed in the beginning of this chapter, harvesting and outcome harvesting window were set at 365 days, and a shift size of 730 days was selected. The minimum prevalence of binary varialbes was set at 1 %, and the minimum prevalence of interaction terms was set at 5 %.

The resulting model achieved an R^2 of 0.336. The c-index was 0.830, and the optimism corrected c-index was 0.827. These measures indicate good predictive power. The optimism corrected ROC curve is plotted in Figure 9. The x-axis displays specificity (the false positive rate) and the y-axis shows sensitivity (the true positive rate). The specificity and sensitivity are plotted for different thresholds, and the c-index - the AUC - is the area under the curve.

Figure 9: **ROC curve - care model**

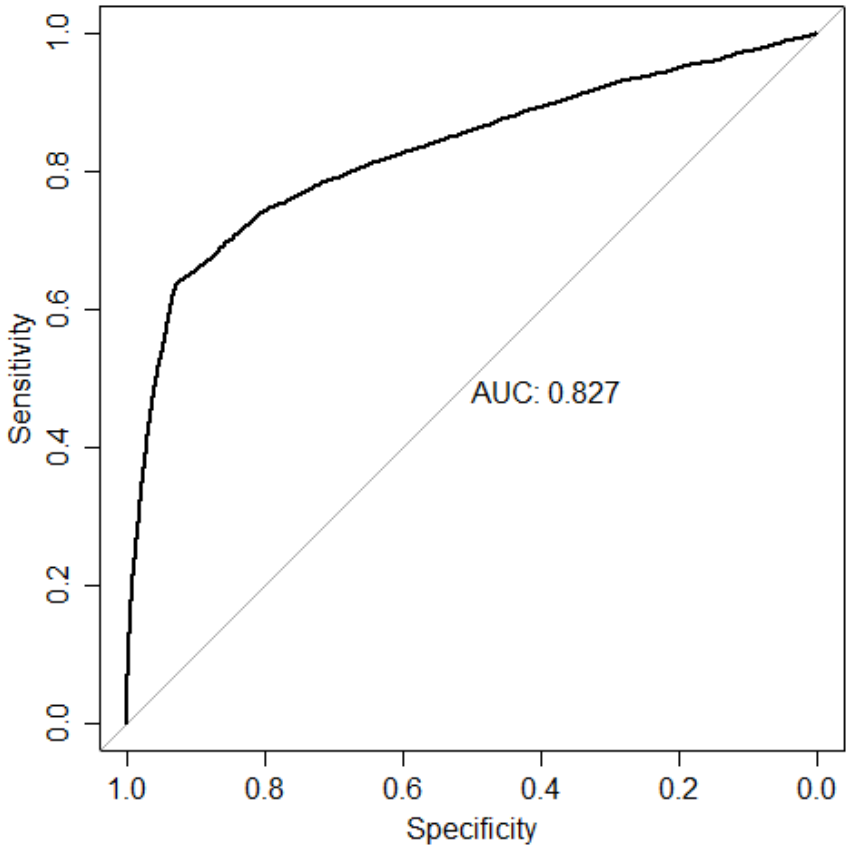


Figure 10 shows a boxplot for the true outcome versus the predicted probabilities of the model. Most predicted values are close to zero for the true outcome of "no measurement of creatinine", and the predicted values are much higher for "a measurement of creatinine" - a satisfactory result.

Figure 10: **Boxplot: true outcome vs. predicted values - care model**

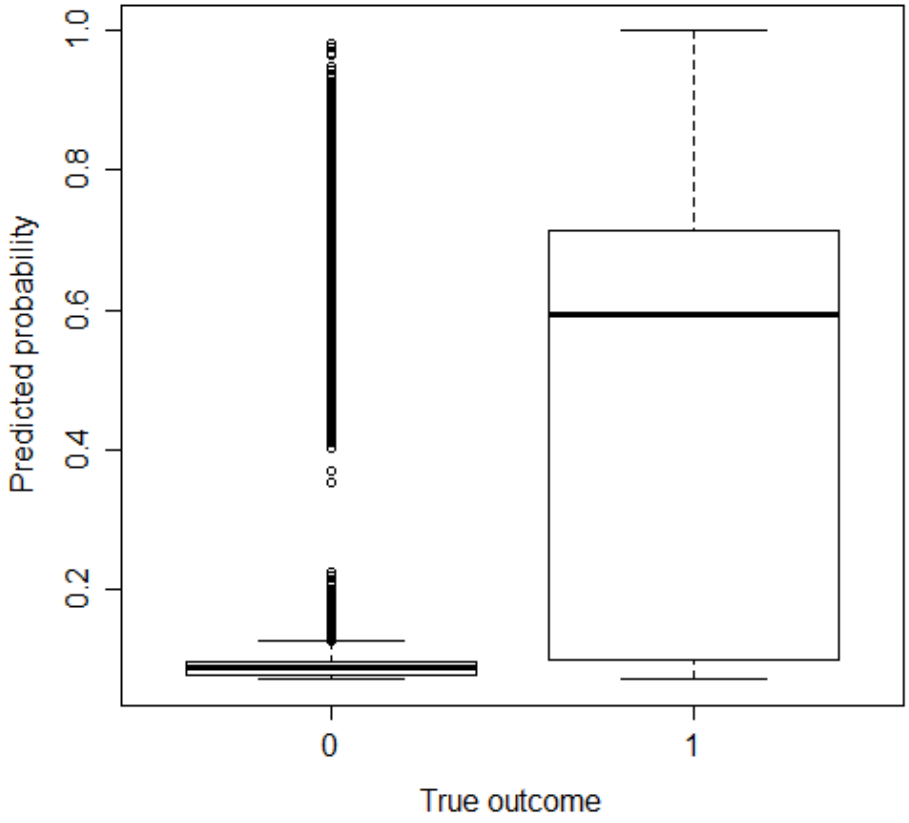


Figure 11 shows a calibration plot and a histogram of the predicted values. For this plot one record per patient was randomly sampled and included. Only few predicted values are between 0.2 and 0.4, and the model underestimates the probability for adequate care in this interval. Few predicted values fall near one, and the model slightly overestimates the measurement of creatinine. The yellow shaded area around the calibration curve marks confidential intervals. Although some discrepancies exist, the model accurately predicts the care outcome.

Figure 11: **Calibration plot - care model**

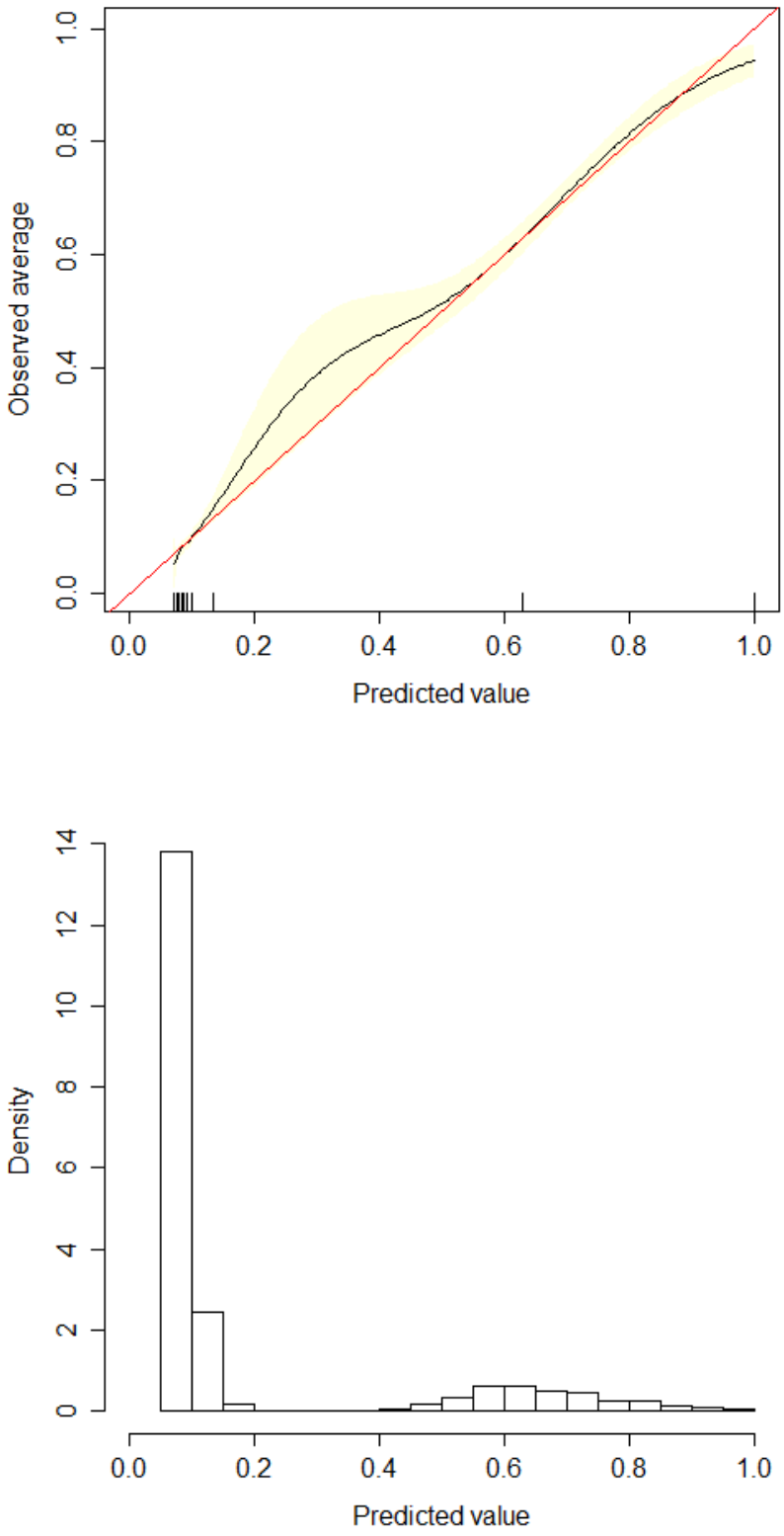


Table 15 shows the coefficients of the selected variables. They are displayed in decreasing order of the absolute standardized coefficients shown in the third column. The percentage of patients displaying the binary variables' characteristics is listed in the fourth column. *Creatinine.binary* has a large impact with a coefficient of 1.55, and almost 18 % of all patients were monitored for creatinine. *Creatinine.number* also has a high coefficient suggesting that having had several measurements in the previous year increases the probability to have one in the next period. Many binary variables of laboratory values and their combinations with other features are selected into the model. It is interesting that *dummy.diabetes2* has a relatively high standardized coefficient, which means that patients who have been diabetics for over 5 years have a higher probability of being measured for creatinine. The diagnosis of CKD in stage 1 and the D06 medication (dermatologicals - antibiotics and chemotherapeutics for dematological use) are within the ten most influential variables, although both had only very low prevalence of 2.2 % and 3 %, respectively.

Table 15: **Coefficients of care model**

	Explanation	Coefficients	Standardized coefficients	Prevalence
Intercept		-2.5190	-2.5190	
creatinine.binary	monitoring of creatinine	1.5540	0.6007	0.1829
creatinine.number	number of creatinine measurements	0.6158	0.4589	
cholesterol.binary	monitoring of cholesterol	0.3441	0.1311	0.1761
hematocrit.binary_creatinine.binary	monitoring of hematocrit & monitoring of creatinine	0.2786	0.0899	0.1181
dummy.diabetes2	diabetes for more than 5 years vs. diabetes for less than 5 years	0.1607	0.0802	0.4706
haemoglobin.binary	monitoring of haemoglobin	0.1298	0.0505	0.1856
D06	dermatologicals, antibiotics and chemotherapeutics for dematological use	0.2482	0.0364	0.0219
n18.1	CKD stage 1	0.1685	0.0287	0.0298
i50_i63.8	heart failure & other cerebral infarction	0.0643	0.0237	0.1623
C_DM.comp9	cardiovascular system & diabetes mellitus without complications	0.0502	0.0211	0.2300
DM.comp2	diabetes mellitus with renal complications	0.0831	0.0194	0.0579
i48_i63.8	atrial fibrillation and flutter & other cerebral infarction	0.0501	0.0188	0.1684
GFR.splines5	fifth spline basis of eGFR*	-0.5650	-0.0187	
N05	nervous system, psycholeptics	0.0807	0.0182	0.0541
GFR.splines2	second spline basis of eGFR*	0.1646	0.0152	
A_M	alimentary tract and metabolism & musculo-skeletal system	0.0380	0.0134	0.1449
hematocrit.binary	monitoring of hematocrit	0.0443	0.0131	0.0975
D	dermatologicals	0.0394	0.0129	0.1226
M01_C	musculo-skeletal system, antiinflammatory and antirheumatic products & cardiovascular system	0.0396	0.0117	0.0965
C01	cardiovascular system, cardiac therapy	0.0623	0.0113	0.0344
FBSL.binary	monitoring of FBSL	0.0265	0.0113	0.2402
age.splines1	first spline basis of age**	-0.0685	-0.0098	
M_creatinine.binary	musculo-skeletal system & monitoring of creatinine	0.0300	0.0090	0.1009
A_D	alimentary tract and metabolism & dermatologicals	0.0339	0.0078	0.0558
A	alimentary tract and metabolism	0.0142	0.0065	0.3030
age.splines4	fourth spline basis of age**	0.0256	0.0061	
C03	cardiovascular system, diuretics	0.0209	0.0041	0.0395
creatinine.binary_DM.comp9	monitoring of creatinine & diabetes mellitus without complications	0.0087	0.0035	0.1998
GFR.splines6	sixth spline basis of eGFR*	-0.0198	-0.0001	

*cubic B-splines; knots at 49.68, 76.05, 100.85 and boundary knots at 20.74, 157.10; no intercept
**cubic B-splines; knots at 41, 64, 79 and boundary knots at 20, 85; no intercept

Table 16 is an extraction of the coefficient path matrix. It lists variable names, the step at which a given variable is added to the model and the tuning parameter lambda at these steps. It also shows the deviance, the c-index, the misclassification error for each change and the degrees of freedom used. The deviance, the c-index and the misclassification error are measures to optimize lambda. The variable selection process stops after 40 steps because the largest value of lambda has been reached such that the deviance is within a standard error of its minimum. All variables selected up to the 40^{th} step constitute the final model. Considering the c-index for variable selection, the model would be completed after step 48, while using the misclassification error, the model could be completed after step 14 and would include only four variables.

Table 16: **Coefficient path matrix of care model**

	Step	Lambda	Deviance	c-index	Misclass. error	df
Intercept	1	0.2305296	0.99074	0.62919	0.19669	1
creatinine.binary	2	0.2100500	0.93787	0.78131	0.19669	2
creatinine.number	8	0.1201983	0.78587	0.78644	0.19664	3
i48_i50	14	0.0687819	0.73383	0.79221	0.12940	5
i50_i63.8	14	0.0687819	0.73383	0.79221	0.12940	5
cholesterol.binary	17	0.0520310	0.72160	0.79538	0.12940	6
i48_i63.8	18	0.0474087	0.71836	0.79344	0.12935	7
haemoglobin.binary	21	0.0358629	0.71116	0.79910	0.12929	8
hematocrit.binary	24	0.0271290	0.70666	0.79590	0.12873	9
i42_i50	24	0.0271290	0.70666	0.79590	0.12873	9
hematocrit.binary_ creatinine.binary	25	0.0247189	0.70552	0.80341	0.12876	9
dummy.diabetes2	30	0.0155242	0.70042	0.81785	0.12965	10
C_i48	31	0.0141451	0.69961	0.81848	0.12932	11
D06	32	0.0128885	0.69890	0.81910	0.12904	13
GFR.splines2	32	0.0128885	0.69890	0.81910	0.12904	13
C_M	33	0.0117435	0.69817	0.82121	0.12876	14
C1	34	0.0107002	0.69744	0.82309	0.12879	19
DM.comp2	34	0.0107002	0.69744	0.82309	0.12879	19
A_D	34	0.0107002	0.69744	0.82309	0.12879	19
A_M	34	0.0107002	0.69744	0.82309	0.12879	19
M_creatinine.binary	34	0.0107002	0.69744	0.82309	0.12879	19
n18.1	35	0.0097497	0.69671	0.82527	0.12876	20
N05	36	0.0088835	0.69596	0.82587	0.12870	24
D	36	0.0088835	0.69596	0.82587	0.12870	24
FBSL.binary	36	0.0088835	0.69596	0.82587	0.12870	24
M01_C	36	0.0088835	0.69596	0.82587	0.12870	24
GFR.splines5	36	0.0088835	0.69596	0.82587	0.12870	24
C01	37	0.0080943	0.69522	0.82639	0.12870	27
A	37	0.0080943	0.69522	0.82639	0.12870	27
age.splines	37	0.0080943	0.69522	0.82639	0.12870	27
C_DM.comp9	38	0.0073753	0.69453	0.82688	0.12898	28
age.splines4	38	0.0073753	0.69453	0.82688	0.12898	28
C03	39	0.0067201	0.69391	0.82726	0.12898	29
creatinine.binary_DM.comp9	40	0.0061231	0.69342	0.82744	0.12907	30
GFR.splines6	40	0.0061231	0.69342	0.82744	0.12907	30
A03	41	0.0055791	0.69304	0.82748	0.12932	34
N01	41	0.0055791	0.69304	0.82748	0.12932	34
i00	41	0.0055791	0.69304	0.82748	0.12932	34
age.splines3	41	0.0055791	0.69304	0.82748	0.12932	34

Figure 12 shows the ”coefficient path”, i.e., the magnitude of coefficients by different values of lambda. Each coloured line represents a variable, the x-axis is the value of the natural logarithm of lambda and the y-axis displays the values of the coefficients. For example, the upper pink line is *creatinine.binary*. It increases at the beginning and falls off at the end. The vertical dotted line represents lambda.1se - the largest value of lambda such that the deviance is within the one standard error of the minimum - and marks the optimal model. The coloured lines that are non-zero when crossing this dotted line represent the variables in the final model.

Figure 12: **Coefficients path - care model**

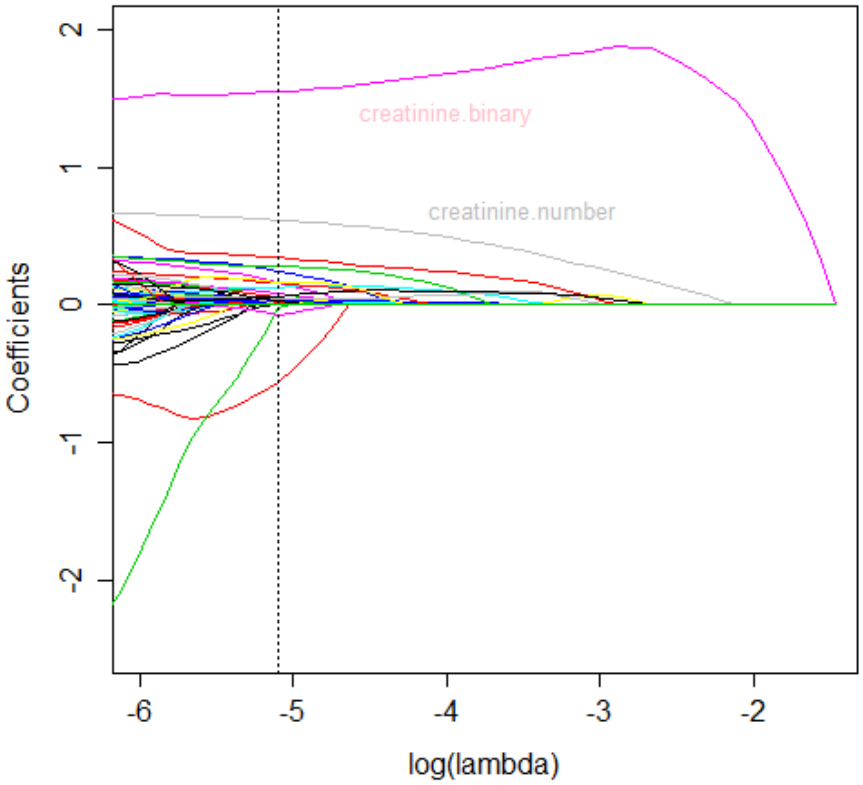
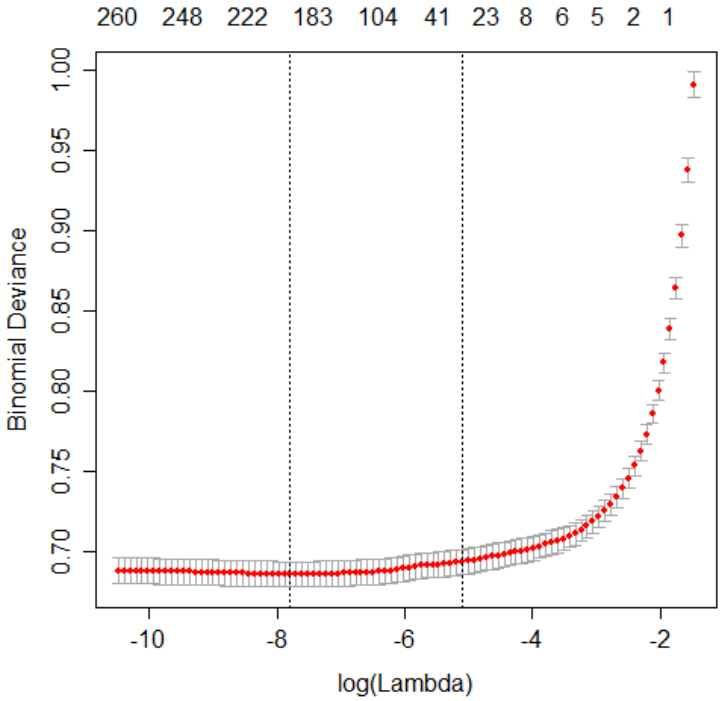


Figure 13 displays the choice of the optimal lambda value. The logarithm of lambda is shown on the x-axis, the y-axis shows the deviance of the model, and the top of the graph displays the number of variables selected for a given model. The red dots in the plot shows lambda values surrounded by error bars. The vertical dotted line on the right

hand side represents the lambda.1se and the second on the left is the lambda.min. If the lambda.min was chosen as the tuning parameter for variable selection, the final model would include over 200 variables. Lambda.1se leads to a final model with 29 variables and is a more cautious if overfitting should be avoided.

Figure 13: **Lambda curve - care model**



Response plots and the appropriate distribution for *eGFR* and *age* are shown in Figure 14 and 15. Response plots represent the estimated odds for a care event as non-linear function of a continuous variable, e.g. *eGFR* or *age*. They are used to demonstrate the non-linear effect which is based on the splines of these continuous variables. The graph is created, e.g. for eGFR, by calculating a vector product for every possible value of *eGFR* and plotting the results. This vector product is calculated between a vector with values of spline bases, the linear effect, and the ordinal coding variables of eGFR with all corresponding coefficient. The plot seems to be almost perfectly plausible for eGFR and age. As expected, patients with poorer kidney function had a higher probability of being monitored for creatinine, with a peak around an eGFR of 55 $ml/min/1.73m^2$, and patients with normal kidney function were almost never monitored. The response curve

increases beginning at $140\,ml/min/1.73m^2$ which is questionable but may be explained by the fact that few records exist in this interval. The same applies to age, as there are few young patients the splines do not offer a plausible fit for them. The highest probability of being measured for creatinine is at an age between 60 and 80.

Figure 14: **Response plot of eGFR and histogram - care model**

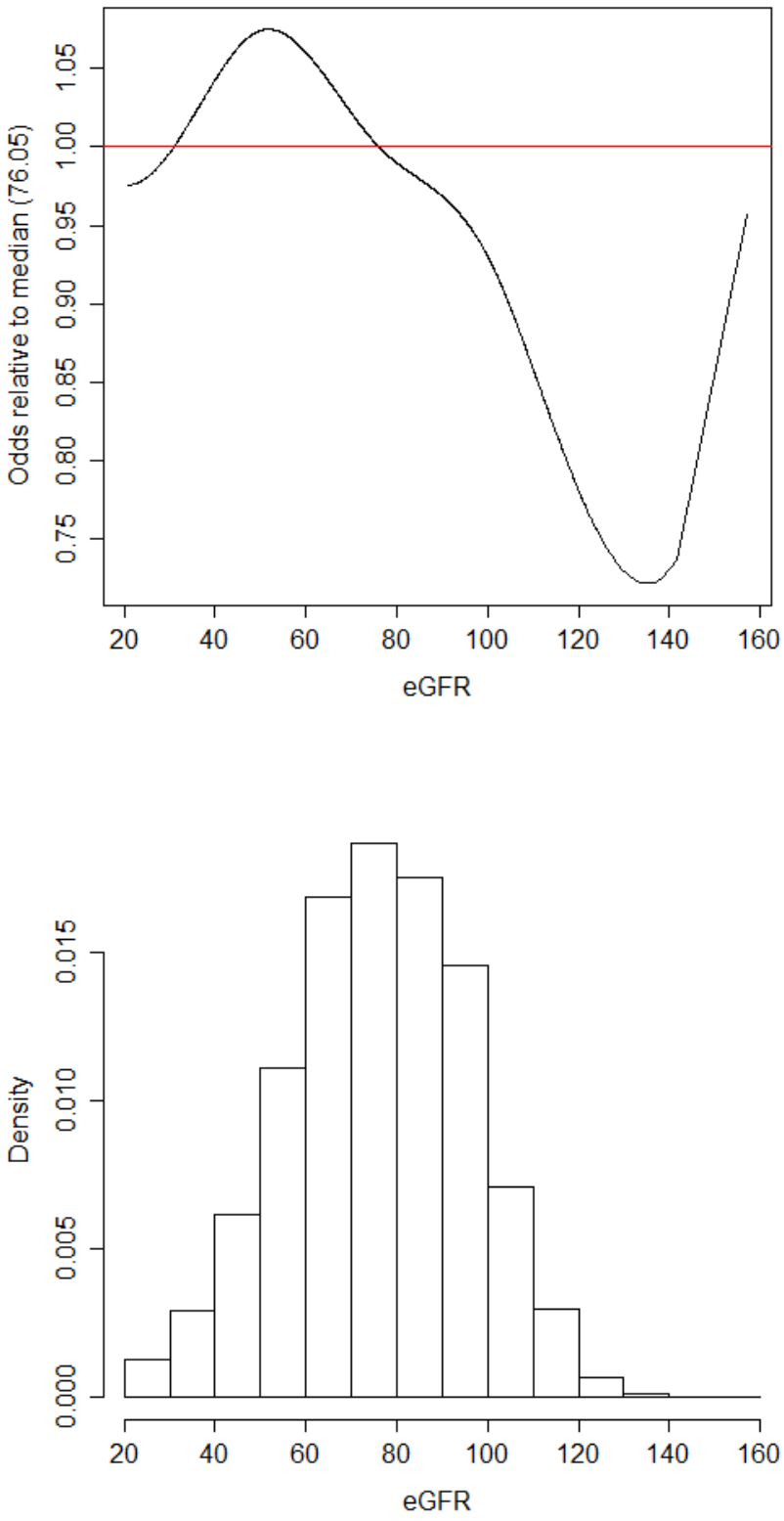
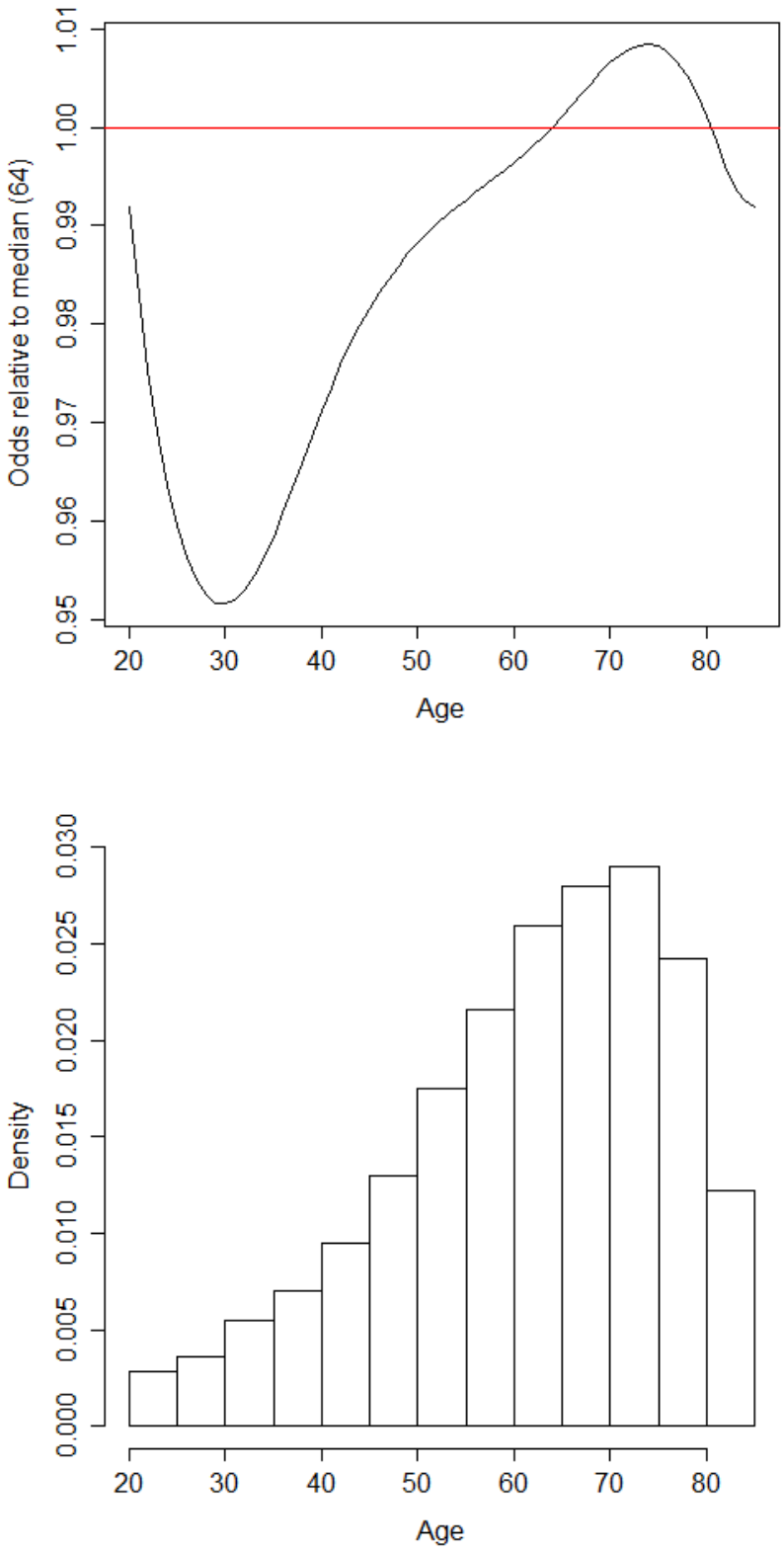


Figure 15: **Response plot of age and histogram - care model**



The prognostic factors were analyzed for importance using the proportion of explained variation (PEV). The marginal and partial PEV for different groups of prognostic factors were calculated and shown in Table 17. These groups include laboratory measurements, medication, diagnosis and demographic information. They should not be interpreted in the standard manner (e.g. as interpreted in linear regression). When we omit some variables out of in a model, LASSO may choose other variables that were not included originally. That is, looking at partial PEV, the explained variation of a model is increased if demographic prognostic factors were not considered at all. The same applies to the prognostic factor group about diagnoses, although that group achieves a very high R^2 (marginal PEV) when used alone for a model development. As both prognostic factors about diagnoses and about laboratory measurements achieve a very high R^2 - almost as high as R^2 of the complete model - when looking at marginal PEV, the two groups can give the same information on the care outcome.

Table 17: **Importance of prognostic factors - care model**

Prognostic factors created by	Partial PEV	Marginal PEV
Laboratory values	0.011	0.294
Medication	0.024	0.017
Diagnoses	-0.001	0.322
Demographics	-0.002	0.000
Model (all variables)		0.336

5.2 Progression model

First, the baseline characteristics are described. Second, the results of the different model matrices are presented, and finally, the results of the model development and evaluation are reported.

5.2.1 Baseline characteristics

Baseline characteristics are detailed for Matrix 2 of the meta parameter combination in Table 3 in this subsection and may be used to better understand insights gained from the prediction model. One record of each patient is randomly chosen for this purpose. The prognostic factors and outcome harvesting window are both set at 365 days and shift size is 730 days. The minimum prevalences for binary variables is set at 1 % and for interaction terms it is set at 5 %.

For the model development are 2250 patients available, 55.03 % of them were women, and the average patient age was 65 years. 50 % of all patients were between 58 and 74 years old, which is understandable given that diabetes tends to affect older people.

Table 18 shows facts about laboratory measurements. The fasting blood sugar level, cholesterol and haemoglobin were measured most frequently. The prevalence of creatinine measurements is 100 % because patients were required to have a creatinine measurement to be considered in this design matrix. The boxplots of most blood parameters in Figure 16 are unremarkable except for FBSL, triglycerides - as in the design matrix for the care model - and some high phosphate values. Approximately 25 % of all records have values of FBSL and triglycerides that are higher than the normal value (Wiener Krankenanstaltenverbund, 2014; Regents of the University of Minnesota, 2006). The average number of creatinine measurements was 1.8, and one person even had twenty.

Table 18: **Overview of laboratory values - progression model**

Laboratory value	Prevalence	Min	Mean	Max	Sd
Calcium	16.53 %	2.10	3.66	10.17	2.12
Cholesterol	84.14 %	115.20	210.00	340.60	45.27
Protein	10.59 %	5.70	6.96	8.79	0.67
Hematocrit	43.33 %	30.96	41.85	51.60	4.27
Haemoglobin	59.41 %	10.40	14.13	17.40	1.39
Creatinine	100 %	0.50	0.98	2.02	0.25
FBSL	92.29 %	67.00	130.20	318.60	44.91
Phosphate	5.67 %	0.72	1.17	3.70	0.37
Triglycerides	40.72 %	46.00	167.50	685.70	103.09

Figure 16: **Boxplots of laboratory values - progression model**

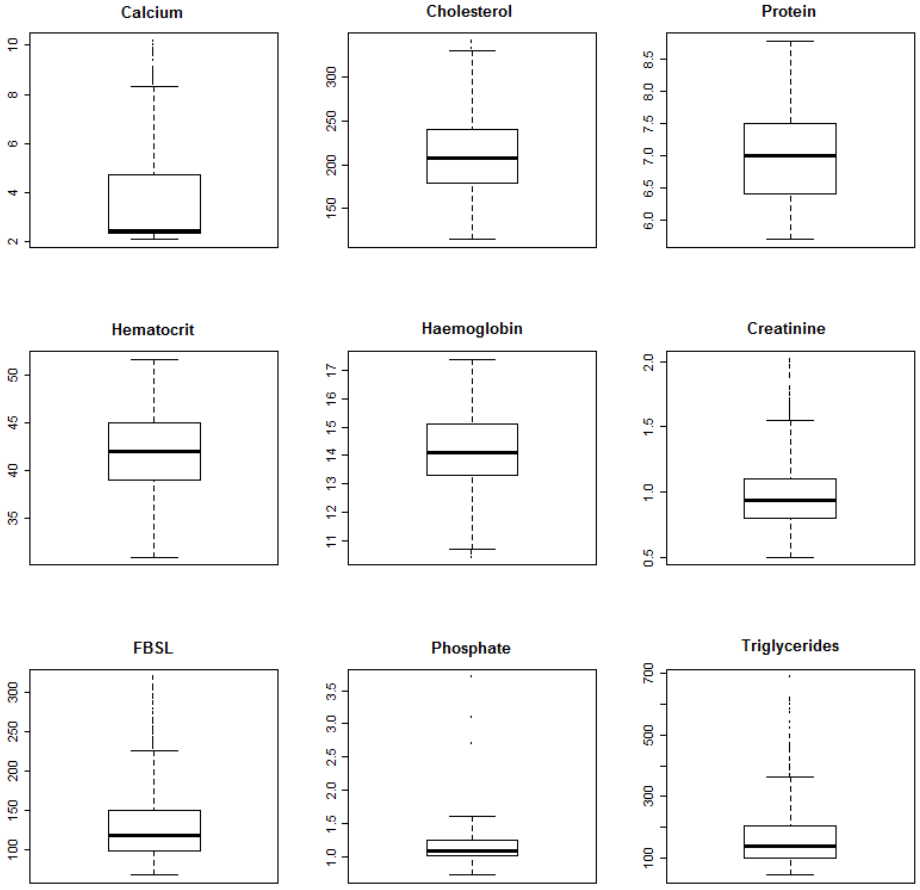


Table 19 shows the 10 most frequent medications and their respective description gained from the Website of WHO Collaborating Centre for Drug Statistics Methodology (2013).

Table 19: **Most frequent medications - progression model** (description reproduced from WHO Collaborating Centre for Drug Statistics Methodology, 2013)

ATC (level 2)	**Description level 1**	**Description level 2**	**Prevalence**
M01	Musculo-skeletal system	Antiinflamatory and antirheumatic products	10.02 %
J01	Antiinfectives for systemic use	Antibacterials for systemic use	7.79 %
A02	Alimentary tract and metabolism	Drugs for acid related disorders	6.16 %
M02	Musculo-skeletal system	Topical products for joint and muscular pain	4.53 %
C09	Cardio vascular system	Agents acting on the renin-angiotensin system	4.47 %
A10	Alimentary tract and metabolism	Drugs used in diabetes	4.33 %
R05	Respiratory system	Cough and cold preparations	3.82 %
C10	Cardio vascular system	Lipid modifying agent	3.30 %
B01	Blood and blood forming organs	Antithrombotic agents	3.25 %
C05	Cardio vascular system	Vasoprotectives	2.96 %

The most frequent medications are similar to those of the care model although more cardiovascular medicines are found on this list. This might be because cardiovascular diseases influence the renal function and must therefore be kept in check.

Table 20 shows an overview of the diagnoses. Diabetes without complications, hypertension, and disorders of lipoprotein metabolism were the most frequent diagnoses. Over 10 % of patients suffered from chronic ischaemic heart disease, diabetes with renal complications and CKD. All these diseases may impact the renal function.

Table 20: **Overview of diagnoses - progression model** (description reproduced from World Health Organization, 2010)

Variable/ ICD-code	**Description**	**Prevalence**
DM.type1	Type I diabetes	3.81 %
DM.comp2	Diabetes mellitus with renal complications	12.41 %
DM.comp3	Diabetes mellitus with ophthalmic complications	2.97 %
DM.comp4	Diabetes mellitus with neurological complications	8.06 %
DM.comp5	Diabetes mellitus with peripheral circulatory complications	1.51 %
DM.comp6	Diabetes mellitus with other complications	5.23 %
DM.comp7	Diabetes mellitus with multiple complications	4.83 %
DM.comp9	Diabetes mellitus without complications or any further information	76.96 %
N18	Chronic kidney disease	10.86 %
N18.1	Chronic kidney disease, stage 1	4.08 %
N08	Glomerular disorders in diseases classified elsewhere	2.44 %
E78	Disorders of lipoprotein metabolism and other lipidaemias	36.33 %
I00	Rheumatic fever without mention of heart involvement	2.35 %
I10	Essential (primary) hypertension	37.57 %
I20	Angina pectoris	3.94 %
I25	Chronic ischaemic heart disease	15.24 %
I42	Cardiomyopathy	4.43 %
I48	Atrial fibrillation and flutter	1.60 %
I50	Heart failure	1.28 %
I63.8	Cerebral infarction, other cerebral infarction	2.35 %

Most of the records belong to stages 0-2 and 8.5 % experienced a decline in kidney function. 13.5 % of patients in stage 3a had a decline in kidney function, while those in stage 3b and 4 did not experience a decline (Table 21).

Table 21: CKD stages by progression outcome

CKD stages	No progression	progression	Total
0-2	3159 (91.5 %)	293 (8.5 %)	3452 (77.2 %)
3a	596 (86.5 %)	93 (13.5 %)	689 (15.4 %)
3b	269 (100.0 %)	0 (0.0 %)	269 (6.0 %)
4	59 (100.0 %)	0 (0.0 %)	59 (1.3 %)
5	0	0	0

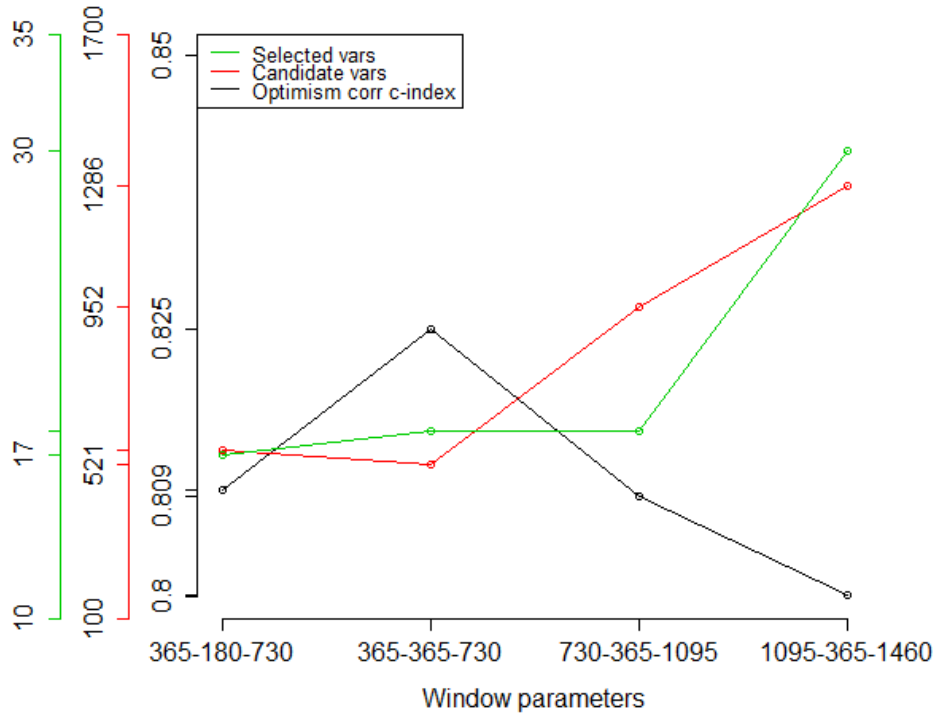
5.2.2 Comparison of different meta parameters

Table 22 shows four models with different window parameters. The minimum prevalences were fixed at 1 % for binary variables and 5 % for interaction terms. Fewer records are used for this model, as patients were required to have their creatinine measured within both the prognostic factors harvesting window and the outcome harvesting window to determine whether a decline in kidney function occured. Figure 17 provides a short overview. Model 2 was determined to be the optimal model, as R^2 and the optimism corrected c-index were the best and the number of patients was highest. eGFR and *age* play the most important roles in the model and the corresponding design variables are detailed in the next subsection.

Table 22: **Comparison of results of different parameters - progression model**

	Model 1	Model 2	Model 3	Model 4
Harvesting window (in days)	365	365	730	1095
Outcome window (in days)	180	365	365	365
Shift size (in days)	730	730	1095	1460
Number of patients	1934	2250	1807	1295
Number of records per patient	1.73	1.99	1.69	1.51
Percentage of progression event	8 %	9 %	10 %	10 %
Number of candidate design variables	563	521	952	1286
Number of selected design variables	17	18	18	30
Prognostically most important design variables (in decreasing order)	GFR.splines3 dummy.GFR2 age GFR.splines1 dummy.GFR1	GFR.splines1 dummy.GFR1 dummy.GFR2 age GFR.splines3	age GFR.splines3 dummy.GFR2 dummy.GFR1 GFR.splines1	age GFR.splines3 dummy.GFR2 creatinine.binary_ FBSL.binary sex
R^2	0.044	0.066	0.048	0.057
c-index	0.829	0.836	0.828	0.843
Optimism corrected c-index	0.810	0.825	0.809	0.800

Figure 17: **Comparison of window parameters - progression model**

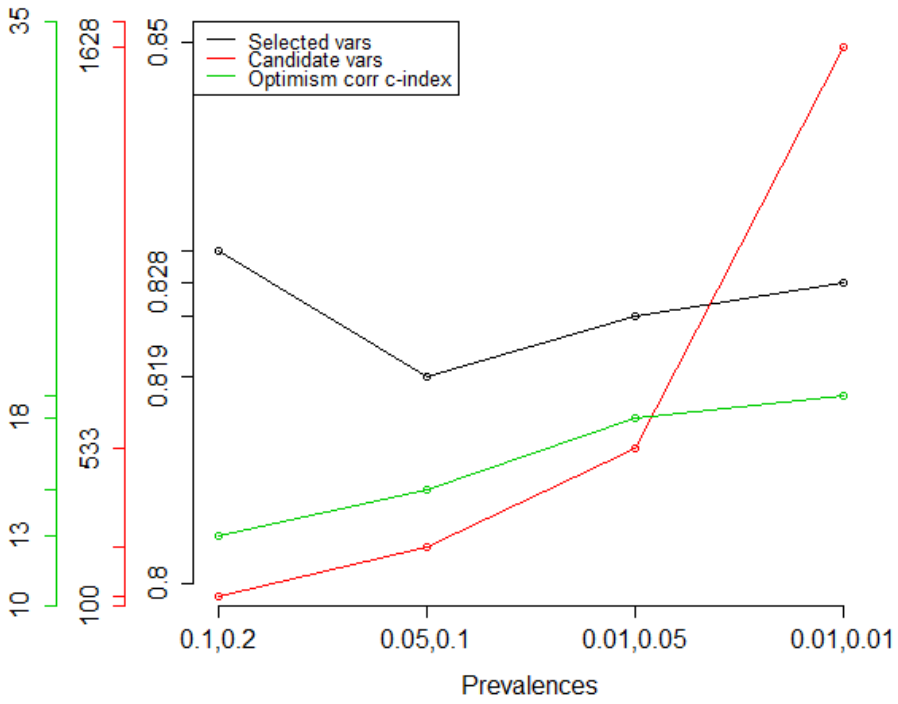


Various window parameters were analyzed and different percentages for the prevalence of main and interaction effects were tried out. The comparison of these different parameters is shown in Table 23 and in Figure 18. Model 2 and its window parameters proved to be most suitable. Minimum prevalence for binary variables was set at 1%, and for interaction terms it was set at 5 %. This set of prevalences allowed for a variable selection that included more variables than the best model (prevalences set at 10 % and 20 %), while also not giving up on too much in terms of performance.

Table 23: **Comparison of results of different prevalences - progression model**

Minimum prevalence binary variables	10 %	5 %	1 %	1 %
Minimum prevalence interaction terms	20 %	10 %	5 %	1 %
Number of candidate design variables	107	248	521	1616
Number of selected design variables	13	15	18	19
Prognostically most important design variables (decreasing)	GFR.splines1 dummy.GFR1 GFR.splines3 age dummy.GFR2	GFR.splines1 dummy.GFR1 dummy.GFR2 GFR.splines3 age	GFR.splines1 dummy.GFR1 dummy.GFR2 age GFR.splines3	GFR.splines1 GFR.splines3 dummy.GFR2 dummy.GFR1 age.splines3
R^2	0.071	0.064	0.066	0.065
c-index	0.839	0.829	0.836	0.842
Optimism corrected c-index	0.831	0.819	0.825	0.828

Figure 18: **Comparison of prevalences - progression model**



5.2.3 Presentation of the model

The chosen model's performance is detailed, and final coefficients are presented. Prognostic factors are also compared.

The model's R^2 was 0.066, and it achieved a c-index of 0.836 and an optimism corrected c-index of 0.825. The measures point to this model's good performance. The ROC curve is shown in Figure 19.

Figure 19: **ROC curve - progression model**

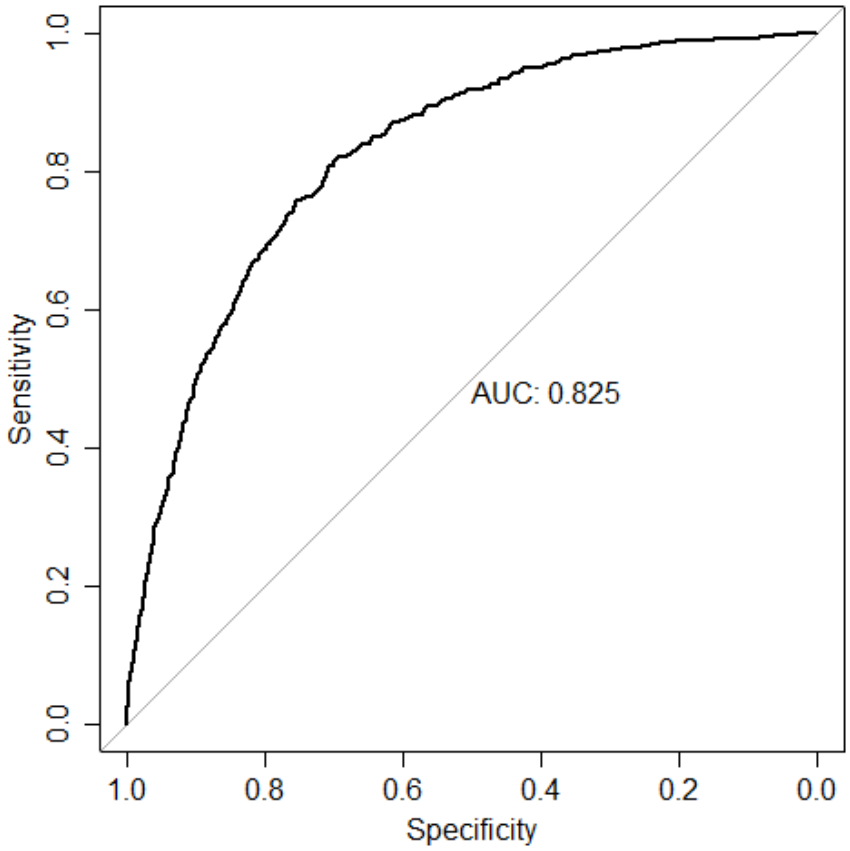


Figure 20 shows boxplots of the true outcome versus the predicted values of the progression model. The predicted values for no decline are much lower than the predicted values for a decline in kidney function. All predicted probabilities never exceed 45 %.

Figure 20: **Boxplot: true outcome vs. predicted values - progression model**

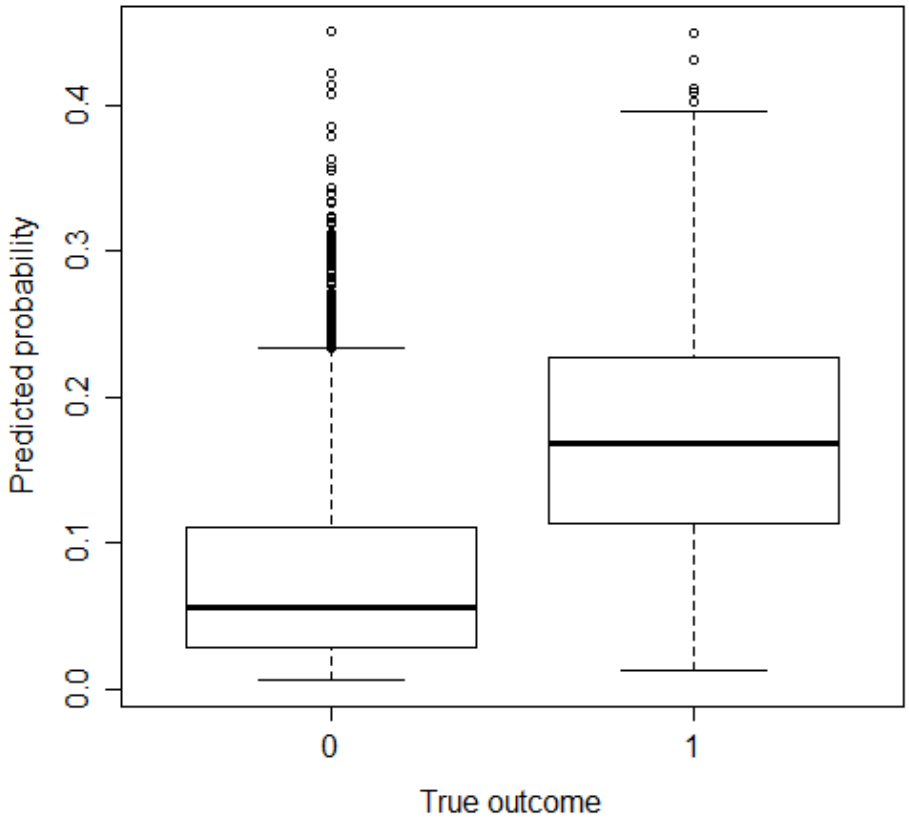


Figure 21 shows a calibration plot of the chosen model and a corresponding histogram. From the plots, one sees that the chosen model underestimates the probability for the top 20 % of patients, while the probability for other patients is overestimated. This inaccuracy might be a result from the severely imbalanced outcome distribution (only 9 % of all cases experienced a decline in kidney function).

Figure 21: **Calibration plot - progression model**

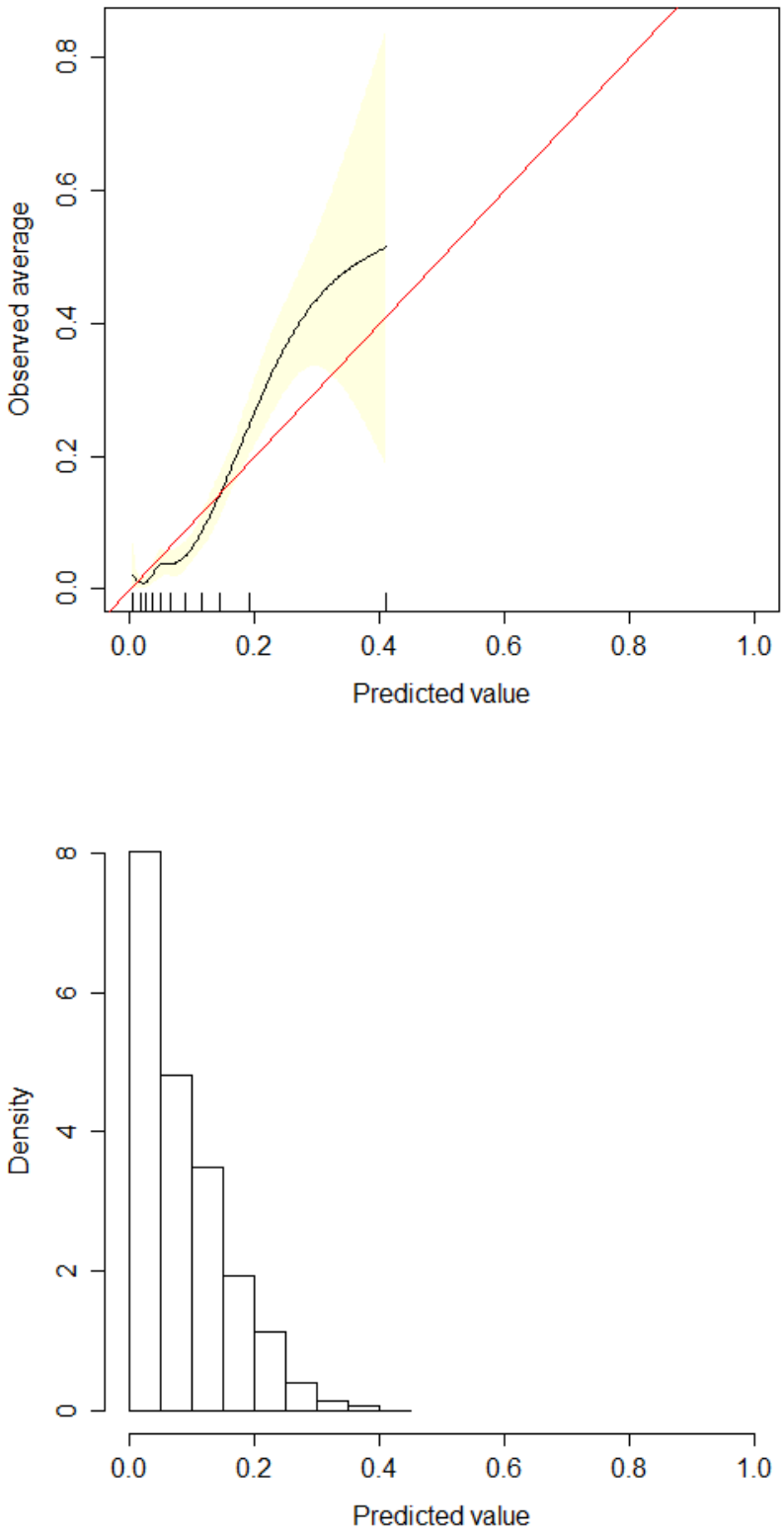


Table 24 consists of the variables included in the chosen model, their coefficients and descriptions taken from World Health Organization (2010) and WHO Collaborating Centre for Drug Statistics Methodology (2013). The variables are ordered by the absolute value of the standardized coefficients and the last column shows the percentage of patients displaying the binary variables characteristics. Some of the most significant variables concern eGFR and medication as well as age. Medication for the cardiovascular system (denoted "C" in the variable name) is the most frequent form of medicine included in the model. Medication for the musculo-skeletal system (denoted "M"), the alimentary tract and metabolism (denoted "A") in combination with other variables are also highly related to the outcome.

Table 24: **Coefficients of progression model**

	Explanation	Coefficients	Standardized coefficients	Prevalence
Intercept		-2.1347	-2.1347	
GFR.splines1	first spline basis of eGFR*	2.7312	0.5908	
dummy.GFR1	eGFR stage 0-2 vs. all others	-1.0382	-0.4353	0.4193
dummy.GFR2	eGFR stage 0-2 and 3a vs. all others	-1.6206	-0.4227	0.2608
age	age of a patient	0.0254	0.2981	
GFR.splines3	third spline basis of eGFR*	-1.3119	-0.2768	
haemoglobin	numeric value of haemoglobin	-0.1689	-0.2184	
age.splines3	third spline basis of age**	0.5713	0.1458	
C07	cardiovascular system, beta blocking agents	0.2969	0.0784	0.0754
N05_C	nervous system, psycholeptics & cardiovascular system	0.1956	0.0431	0.0512
i42_creatinine.binary	cardiomyopathy & monitoring of creatinine	0.1907	0.0421	0.0515
M01_C	musculo-skeletal system, antiinflammatory and antiheumatic products & cardiovascular system	0.0848	0.0319	0.1705
C01	cardiovascular system, cardiac therapy	0.1089	0.0264	0.0629
M02_C	musculo-skeletal system, & cardiovascular system	0.0917	0.0258	0.0864
C09_FBSL.binary	cardiovascular system,agents acting on the renin-angiotensin system & montioring of FBSL	0.0617	0.0210	0.1340
A02_M01	alimentary tract and metabolism, drugs for acid related disorders & musculo-skeletal system, antiinflammatory and antiheumatic products	0.0420	0.0126	0.0996
A02_i10	alimentary tract and metabolism, drugs for acid related disorders & essential hypertension	0.0221	0.0063	0.0904
sex	sex of a patient	0.0122	0.0061	
C01_creatinine.binary	cardiovascular system, cardiac therapy & monitoring of creatinine	0.0119	0.0029	0.0629

*quadratic B-splines; knots at 64.17, 85.69 and boundary knots at 20.88, 135.97; no intercept

**quadratic B-splines; knots at 60, 73 and boundary knots at 20, 84; no intercept

The coefficient path matrix is listed in Table 25. It consists of variable names, the steps at which the variables were entered into the model, lambda, three measures for parameter optimization and the degrees of freedom at each step. The measures for parameter optimization are the model deviance, the c-index and misclassification error. The deviance is used for variable selection, and suggests the final model created after 18 steps that is found in Table 25. Using the c-index for variable selection, the selection process stops after step 19. If one were to use the missclassification error, a model consisting only of an intercept would result.

Table 25: **Coefficient path matrix of progression model**

	Step	Lambda	Deviance	c-index	Misclass. error	Df
Intercept	1	0.0535476	0.58832	0.59027	0.08637	1
age	2	0.0487906	0.58261	0.70602	0.08637	3
GFR.splines3	2	0.0487906	0.58261	0.70602	0.08637	3
age.splines3	3	0.0444561	0.57580	0.72461	0.08637	4
GFR.splines1	4	0.0405068	0.56945	0.72665	0.08637	5
dummy.GFR2	7	0.0306419	0.55304	0.74335	0.08637	6
haemoglobin	9	0.0254395	0.53886	0.77010	0.08637	7
C07	12	0.0192440	0.52179	0.78715	0.08637	9
C07_creatinine.binary	12	0.0192440	0.52179	0.78715	0.08637	9
C01	13	0.0175344	0.51708	0.79079	0.08637	11
C01_creatinine.binary	13	0.0175344	0.51708	0.79079	0.08637	11
dummy.GFR1	13	0.0175344	0.51708	0.79079	0.08637	11
M01_C	14	0.0159767	0.50966	0.80159	0.08637	14
N05_C	14	0.0159767	0.50966	0.80159	0.08637	14
M02_C	15	0.0145574	0.50250	0.81129	0.08637	16
i42_creatinine.binary	15	0.0145574	0.50250	0.81129	0.08637	16
C09_FBSL.binary	17	0.0120858	0.49128	0.82077	0.08637	17
A02_M01	18	0.0110121	0.48716	0.82289	0.08637	19
A02_i10	18	0.0110121	0.48716	0.82289	0.08637	19
sex	19	0.0100338	0.48376	0.82425	0.08637	20
haemoglobin.binary	20	0.0091425	0.48093	0.82546	0.08637	25
J01_i10	20	0.0091425	0.48093	0.82546	0.08637	25
N02_cholesterol.binary	20	0.0091425	0.48093	0.82546	0.08637	25
N05_FBSL.binary	20	0.0091425	0.48093	0.82546	0.08637	25
n18_FBSL.binary	20	0.0091425	0.48093	0.82546	0.08637	25
N05_DM.comp9	21	0.0083303	0.47860	0.82671	0.08660	29
DM.comp9_FBSL.binary	21	0.0083303	0.47860	0.82671	0.08660	29
haemoglobin.splines	21	0.0083303	0.47860	0.82671	0.08660	29
haemoglobin.splines	21	0.0083303	0.47860	0.82671	0.08660	29

Figure 22 shows the variable selection process. Each coloured line represents a variable, the x-axis shows the natural logarithm of lambda and the y-axis displays the values of the coefficients. For example, the upper red line is *GFR.splines1*, and the lower blue line is *dummy.GFR2*. The dotted vertical line represents lambda.1se, and all coefficents that are non-zero at this point are included in the model.

Figure 22: **Coefficients path - progression model**

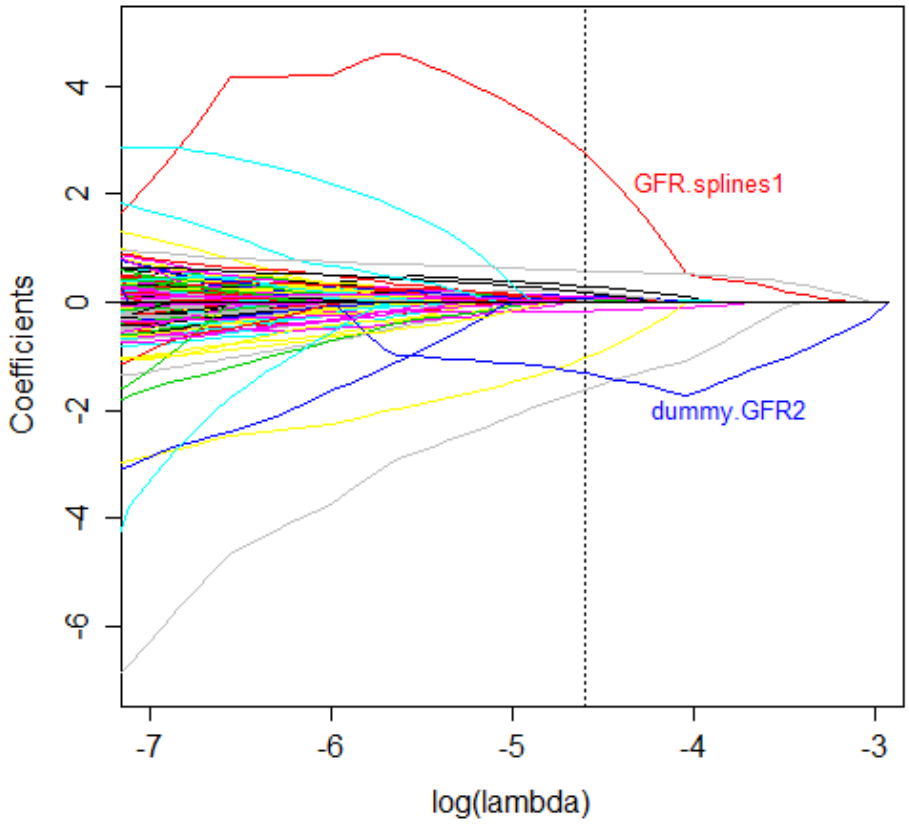


Figure 23 illustrates the values of lambda during the LASSO process. Here the minimum of lambda can be seen clearly. The dotted line on the left marks lambda.min and the dotted line on the right is lambda.1se. Lambda.1se was used for the final model, as it suggests a model with fewer variables.

Figure 23: **Lambda curve - progression model**

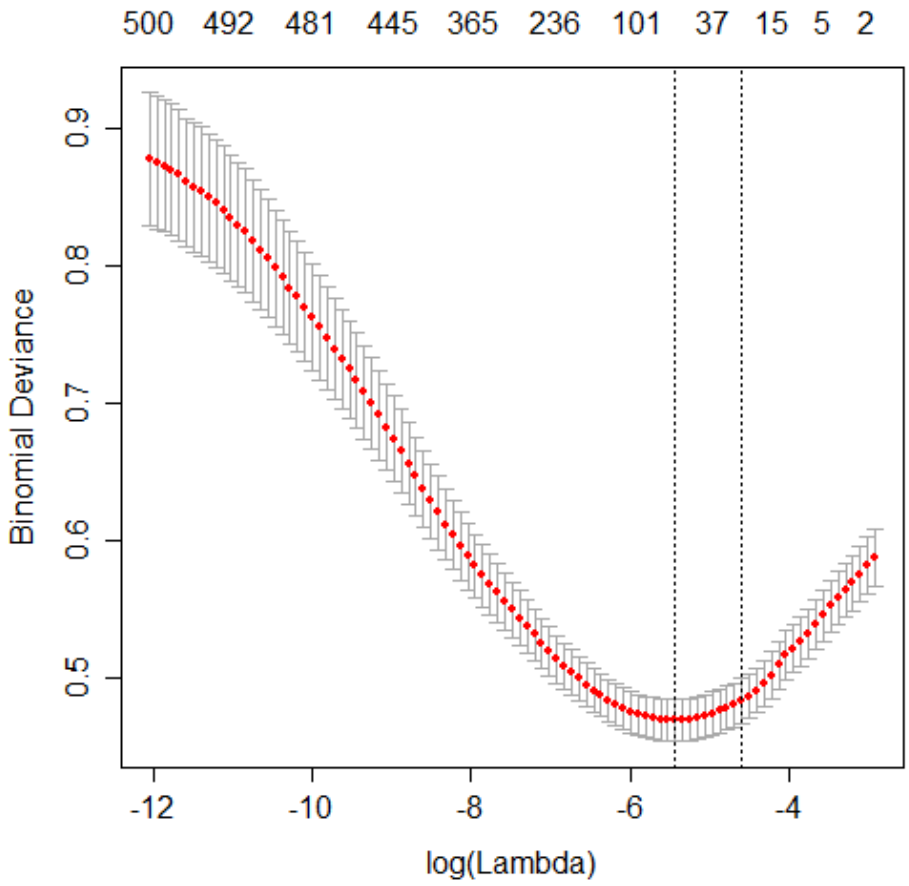


Figure 24 illustrates the response curve for eGFR and a histogram with eGFR distribution. The response curve reveals that patients with an eGFR value between 60 and 70 have a higher risk for a decline with the peak at 60 $ml/min/1.73m^2$ which is very plausible. Also patients with an eGFR value between 45 and 55 are at higher risk. Patients with an lower value than 45 $ml/min/1.73m^2$ can expect a lower probability for a decline. The ascent following 110 $ml/min/1.73m^2$ does not appear to be accurate, and an explanation for this might be the lack of records starting at that measurement.

Figure 25 shows the response curve of age and the variable's histogram. Patients older than 65 are more prone to having a decline in renal function, and those around 75 years have the highest risk for a decline.

Figure 24: **Response plot of eGFR and histogram - progression model**

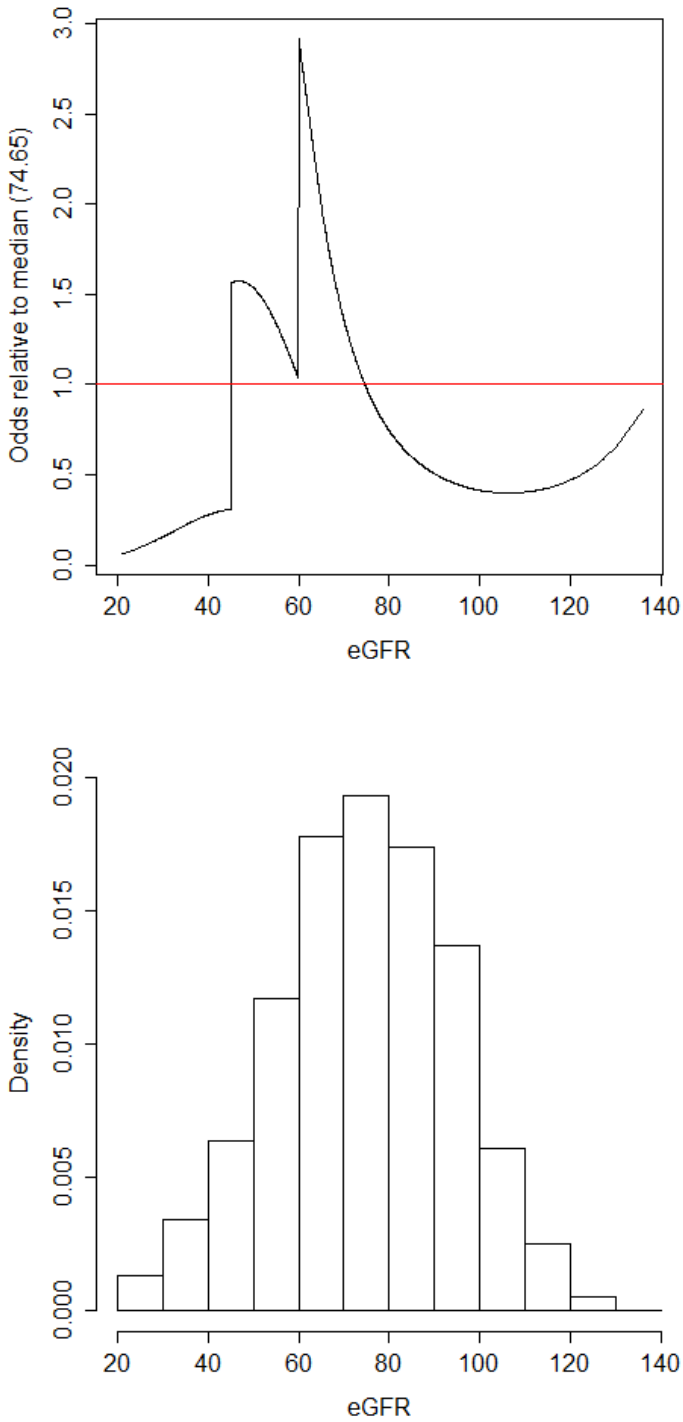


Figure 25: **Response plot of age and histogram - progression model**

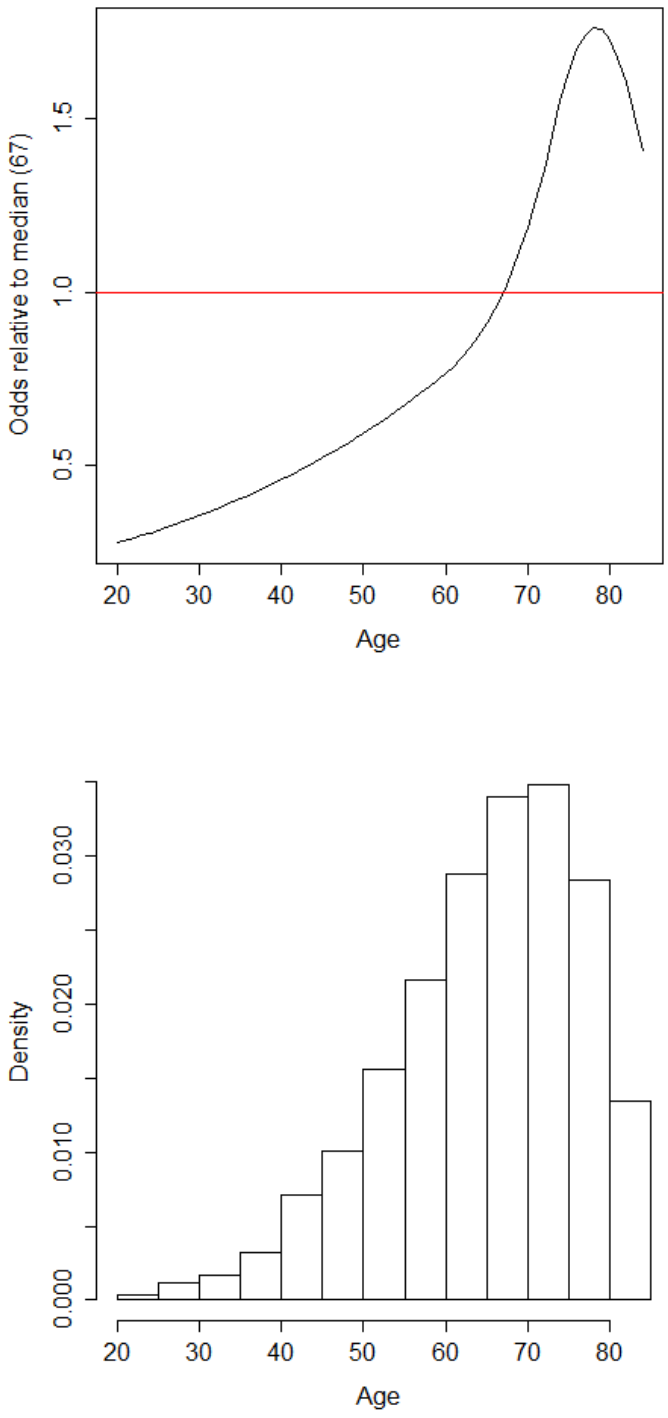


Table 26 lists the importance of different prognostic factors grouped into laboratory measurements, eGFR, medication, diagnoses and demographics. Unlike in the care model, eGFR prognostic factors were treated as a separate group because they play a significant role in the final model. The R^2 performance measure is worsened by medication prognostic factors, as can be seen by looking at partial PEV. Choosing only the laboratory measurements group, the explained variation would be higher than providing the LASSO with all variables for selection. The marginal PEV of eGFR prognostic factors give the same result, i.e., eGFR related prognostic factors cover the same information as the superset laboratory measurements. eGFR's partial PEV indicates that the eGFR variables can explain the outcome almost as well as all variables combined. Medication, diagnoses and demographics have little to no predictive power.

Table 26: **Importance of prognostic factors - progression model**

prognostic factors created by	partial PEV	marginal PEV
eGFR	0.056	0.068
laboratory values	0.002	0.068
medication	-0.014	0.006
diagnoses	0.000	0.000
demographics	0.015	0.007
model (all variables)		0.066

6 Discussion

Two models concerning renal function were developed in this paper with binary outcomes. The models' outcomes were defined as adequate renal care (denoted as care model) and a decline in kidney function (denoted as progression model). A data base including 700,000 health records stemming from 60 primary care physicians offices in Austria was used to develop these models. The data information was limited because the data base does not contain any death dates, only visits of doctors' offices were recorded. The models were developed for diabetics type I and type II, therefore, the data was restricted to 31,000 patients. LASSO regression was the method used for model development. Before applying LASSO, several preprocessing steps were necessary. A combination of innovative approaches for dynamic modeling, non-linear modeling and constructing interaction terms were used in a comprehensive framework for variable selection. With dynamic modeling it was possible to use several records from a patient by floating baseline dates, i.e., each record has its own individual baseline date. This dependency between records was especially considered in the LASSO process. Non-linear modeling was taken into account for continuous variables. B-splines with a second or third polynomial degree were created for every continuous variable. Thus, with LASSO a combination of a linear and a non-linear effects was possible, and results were illustrated with response plots. Additionally, interaction terms were created by combining two binary variables. The number of binary design variables and interaction terms was restricted by a minimum prevalence. For model performance evaluation, cross-validation was used for optimism correction. The similarity of the model-based c-index and the optimism corrected c-index in both models indicated that the models were not overfitted, which could be a consequence of the built-in cross-validation in LASSO. Also the calibration plots, constructed using only one randomly sampled record per patient, showed adequate agreement of predicted and observed outcomes.

According to the results, the care model has a high predictive power - explained variation is 0.336 and the optimism corrected c-index is 0.827. The most important factors determined by absolute standardized coefficients are *creatinine.binary* and *creatinine.number* which seems plausible. The calibration plot shows an almost perfect agreement of predicted and observed probabilities for creatinine measurement. This prediction model is useful for investigating which patients did not receive adequate renal care at primary care physicians. It may be applied to people who are diabetics and who are between 20

and 85 years old. Similar models do not exist for the analysis of renal care, which makes this model particularly valuable.

The progression model has a relatively low explained variation of 0.066 and a high optimism corrected c-index of 0.825. The most important covariates are *GFR.splines1*, *dummy.GFR1* and *dummy.GFR2*. The calibration plot shows that the model severely underestimates the probability of a decline in kidney function for patients with a high probability which is probably because of the imbalanced outcome distribution. The patients with no decline could be downweighted or downsampled in order to obtain a better fit for patients with higher risk and, thus, improve the model. One could also recalibrate the model. Unlike the models of Dunkler et al. and Halbesma et al., the models decribed in this paper were developed with "real-life data" from primary care physicians and can be applied to diabetics aged between 20 and 85.

In summary, this thesis demonstrates how innovative approaches to statistical modeling can be applied to derive scientifically plausible conclusions from relatively unstructured and partly uncoded routine patient records. Although there are still considerable limitations and our results should be cautiously interpreted, it provides a general roadmap for similar analyses. In the near future, the increasing availability of electronically collected information will further stimulate the retrospective analysis of medical histories and care pathways. This will generate big challenges for statistical methodology, and more studies, similar to the one presented here, will be needed to accumulate further experience with such data.

References

Akaike, H. (1973). Information theory and an extension of the maximum likelihood principle. *Second International Symposium on Information Theory*, 267-281.

Altman, D. G., & Royston, P. (2000). What do we mean by validating a prognostic model? *Statistics in Medicine*, *19*, 453-473.

Daugirdas, J. T. (2011). *Handbook of chronic kidney disease management* (first ed.). Philadelphia, PA: Lippincott Williams & Wilkins.

de Bruijne, M. H., Sijpkens, Y. W., Paul, L. C., Westendorp, R. G., van Houwelingen, H. C., & Zwinderman, A. H. (2003). Predicting kidney graft failure using time-dependent renal function covariates. *Journal of Clinical Epidemiology*, *56*, 448-455.

Dunkler, D., Clase, C., Gerstein, H., Heinze, G., Kohl, M., Mann, J. F., Theo, K. K., Yusuf, S., & Oberbauer, R. (2014). Risk prediction of early chronic kidney disease: net benefit in individuals with type 2 diabetes. *submitted*.

Fox, C, S., Gona, P., Larson, M. G., Selhub, J., Tofler, G., Hwang, S. J., Meigs, J. B., Levy, D., Wang, T. J., Jacques P. F., Benjamin E. J., & Vasan, R. S. (2010). A multi-marker approach to predict incident ckd and microalbuminuria. *The Journal of the American Society of Nephrology*, *21*, 2143-2149.

Friedman, J., Hastie, T., Simon, N., & Tibshirani, R. (2014). *Lasso and elastic-net regularized generalized linear models.* Retrieved June 28, 2014, from `http://cran.r-project.org/web/packages/glmnet/glmnet.pdf`.

Gilbert, S., & Weiner, D. E. (2013). *National kidney foundation's primer on kidney diseases* (sixth ed.). Philadelphia, PA: Elsevier Health Sciences; Saunders.

Halbesma, N., Jansen, D. F., Heymans, M. W., Stolk, R. P., de Jong, P. E., & Gansevoort, R. T. (2011). Predicting kidney graft failure using time-dependent renal function covariates. *Clinical Journal of the American Society of Nephrology*, *6*, 1731-1738.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning* (second ed.). New York, NY: Springer Science+Business Media.

Heinze, G. (2012). *Application of the R package glmnet for development of prediction models and for identification of biomarkers for CKD incidence or progression, SysKID internal report.*

Heinze, G., & Schemper, M. (2003). Comparing the importance of prognostic factors in Cox and logistic regression using SAS. *Computer Methods and Programs in Biomedicine*, *71*, 155-163.

Hunsicker, L. G., Adler, S., Caggiula, A., England, B. K., Greene, T., Kusek, J. W., Rogers, N. L., & Teschan, P. E. (1997). A predictive model for progression of chronic kidney disease to kidney failure. *Kidney International*, *51*, 1908-1919.

James, M. T., Hemmelgarn, B. R., & Tonelli, M. (2010). Early recognition and prevention of chronic kidney disease. *The Lancet*, *375*, 1296 - 1309.

Lerma, E. V., & Nissenson, A. R. (2011). *Nephrology secrets* (third ed.). Philadelphia, PA: Elsevier Health Sciences; Mosby.

Lewis, R. (2012). *Understanding chronic kidney disease: A guide for the non-specialist* (first ed.). Keswick: M&K Update Limited.

McClellan, W. M., Schoolwerth, A. C., & Gehr, T. (2006). *Clinical management of chronic kidney disease* (first ed.). Professional Communications. Retrieved April 13, 2014, from `http://www.renal.org/information-resources/the-uk-eckd-guide/ckd-stages#sthash.SNTWTtru.dpbs`.

Meinshausen, N., & Bühlmann, P. (2010). Stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *72*, 417–473.

Regents of the University of Minnesota. (2006). *Normal Lab Values.* Retrieved July 29, 2014, from `http://www.student.med.umn.edu/wardmanual/normallabs.php`.

Ridgeway, G. (2014). *Package 'gbm'.* Retrieved October 2, 2014, from `http://cran.r-project.org/web/packages/gbm/gbm.pdf`.

Schwarz, G. (1978). Estimating the dimension of a model. *The Annals of Statistics*, *6*(2), 461-464.

Skali, H., Uno, H., Levey, A. S., Inker, L. A., Pfeffer, M. A., & Solomon, S. D. (2001). Prognostic assessment of estimated glomerular filtration rate by the new chronic kidney disease epidemiology collaboration equation in comparison with the modification of diet in renal disease study equation. *American Heart Journal*, *162*(3), 548 - 554.

Smith, G. C. S., Seaman, S. R., Wood, A. M., Roystock, P., & White, I. R. (2014). Correcting for Optimistic Prediction in Small Data Sets. *American Journal of Epidemiology*, *180*, 318-324.

SysKID consortium. (2014a). *About us.* Retrieved April 19, 2014, from `http://www.syskid.eu/about-us`.

SysKID consortium. (2014b). *Flyer.* Retrieved April 19, 2014, from `http://www.syskid.eu/images/stories/12-0-19_flyer_syskid.pdf`.

SysKID consortium. (2014c). *Scientfic Concept.* Retrieved April 19, 2014, from `http://www.syskid.eu/scientific-concept`.

Tangri, N., Stevens, L. A., Griffith, J., Tighiouart, H., Djurdjev, O., Naimark, D., Levin, A., & Levey, A. S. (2011). A predictive model for progression of chronic kidney disease to kidney failure. *The Journal of the American Medical Association*, *305*, 1553-1559.

The Renal Association. (2013). *CKD stages.* Retrieved April 13, 2014, from `http://www.renal.org/information-resources/the-uk-eckd-guide/ckd-stages#sthash.SNTWTtru.dpbs`.

Tibshirani, R. (1996). Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society*, *58*(1), 267 - 288.

WHO Collaborating Centre for Drug Statistics Methodology. (2011). *ATC - Structure and Principles.* Retrieved May 6, 2014, from `http://www.whocc.no/atc/structure_and_principles/`.

WHO Collaborating Centre for Drug Statistics Methodology. (2013). *ATC/DDD Index 2014.* Retrieved July 4, 2014, from `http://www.whocc.no/atc_ddd_index/`.

Wiener Krankenanstaltenverbund. (2014). *Referenzwerte.* Retrieved July 29, 2014, from `http://www.akhwien.at/default.aspx?pid=3986`.

World Health Organization. (2010). *ICD-10 Version:2010.* Retrieved July 29, 2014, from `http://apps.who.int/classifications/icd10/browse/2010/e`.

World Health Organization. (2014). *International classification of diseases (icd) information sheet.* Retrieved May 6, 2014, from `http://www.who.int/classifications/icd/factsheet/en/`.

Zou, H., Hastie, T., & Tibshirani, R. (2007). On the "degrees of freedom" of the lasso. *The Annals of Statistics*, *35*(5), 2173-2192.

Appendix

Code

Model matrix

```
library(caTools)
library(parallel)

setwd("~/Studium/Magarbeit")
patsample <- read.csv("~/Studium/Magarbeit/GP_diabetics/
patsample", sep=";")
medatc <- read.csv("~/Studium/Magarbeit/GP_diabetics/
medatc", sep=";")
tabllabor <- read.csv("~/Studium/Magarbeit/GP_diabetics/
tabllabor", sep=";")
tbldiagnosis <- read.csv("~/Studium/Magarbeit/GP_diabetics/
tbldiagnosis", sep=";")
gpsas.tbllaborkey<-read.csv("~/Studium/Magarbeit/
GP_diabetics/gpsas.tbllaborkey", sep=";")
icdcodes<-read.csv("~/Studium/Magarbeit/pd_icd_trans_diab/
pd_icd_trans_diab", sep=";")
colnames(icdcodes)<-gsub("_",".",colnames(icdcodes))


### unique tabllabor
tabllabor<-unique(tabllabor)

### add names of laboratory values
tabllabor<-merge(tabllabor, gpsas.tbllaborkey[1:9,], by="LabKey", all.x=T )

### dates
patsample$GebDat[nchar(patsample$GebDat)==7]<-paste("0",
patsample$GebDat[nchar(patsample$GebDat)==7], sep="")
patsample$GebDat<-as.Date(patsample$GebDat,format="%d%m%Y")
```

```
tabllabor$SystemDate<-as.Date(tabllabor$SystemDate)
medatc$SystemDate<-as.Date(medatc$SystemDate)
tbldiagnosis$SystemDate<-as.Date(tbldiagnosis$SystemDate)

###set ICD-codes to 0 instead of NA
icdcodes[is.na(icdcodes)]<-0
colnames(icdcodes)[1]<-"PatID"

###clear rows without birthdate
patsample<-subset(patsample, !is.na(patsample$GebDat))



### clear dates with no sex
table(patsample$Geschlecht)
patsample<-subset(patsample, patsample$Geschlecht!="" &
patsample$Geschlecht!="X")
patsample$Geschlecht<-droplevels(patsample$Geschlecht)

### clear unplausible creatinine values
table(tabllabor$Wert[tabllabor$LabKey==6])
tabllabor<-tabllabor[-which(tabllabor$Wert==0 & tabllabor$LabKey==6),]
tabllabor<-tabllabor[-which(tabllabor$Wert==36 & tabllabor$LabKey==6),]
table(tabllabor$Wert[tabllabor$LabKey==1])
table(tabllabor$Wert[tabllabor$LabKey==2])
tabllabor<-tabllabor[-which(tabllabor$Wert==0 & tabllabor$LabKey==2),]
tabllabor<-tabllabor[-which(tabllabor$Wert==102245 &
tabllabor$LabKey==2),]
table(tabllabor$Wert[tabllabor$LabKey==3])
tabllabor<-tabllabor[-which(tabllabor$Wert>=20 & tabllabor$LabKey==3),]
table(tabllabor$Wert[tabllabor$LabKey==4])
tabllabor<-tabllabor[-which(tabllabor$Wert==0 & tabllabor$LabKey==4),]
table(tabllabor$Wert[tabllabor$LabKey==5])
tabllabor<-tabllabor[-which(tabllabor$Wert>30 &tabllabor$LabKey==5),]
table(tabllabor$Wert[tabllabor$LabKey==7])
```

```
table(tabllabor$Wert[tabllabor$LabKey==8])
table(tabllabor$Wert[tabllabor$LabKey==9])
tabllabor<-tabllabor[-which(tabllabor$Wert==0 & tabllabor$LabKey==9),]


### length of window
window<-function(patID, patID_date1, date1, patID_date2, date2,
patID_date3, date3){
  mini<-sapply(patID, function(x) min(min(date1[patID_date1==x]),
                                      min(date2[patID_date2==x]),
                                      min(date3[patID_date3==x])))
  maxi<-sapply(patID,function(x) max(max(date1[patID_date1==x]),
                                     max(date2[patID_date2==x])
                                     ,max(date3[patID_date3==x])))
  mat<-as.data.frame(patID)
  mat<-as.data.frame(cbind(mat,as.Date(mini, origin="1970-01-01"),
  as.Date(maxi, origin="1970-01-01")))
  #because R's origin is 1970-01-01
  mat<-mat[!is.infinite(mat[,2]),]
  mat<-mat[!is.na(mat[,2]),]
  range<-mat[,3]-mat[,2]
  mat<-cbind(mat,range)
  colnames(mat)<-c("PatID", "Min", "Max", "range")
  return(mat)
}

windowrange<-window(patsample$PatID,tabllabor$PatID,
tabllabor$SystemDate, medatc$PatID, medatc$SysemDate,
tbldiagnosis$PatID, tbldiagnosis$SystemDate)


age.calc <- function(first, second)
{
  age <- as.numeric(format(second,format="%Y")) -
  as.numeric(format(first,format="%Y"))
```

```
  first<-as.Date(ifelse(substring(first,5,10)=="-02-29",first+1,first),
  origin="1970-01-01")
  help<- as.Date(paste(format(second,format="%Y"),"-",
  format(first,format="%m-%d"),sep=""),format="%Y-%m-%d")
  age[help > second] <- age[help > second] - 1
  return(age)
}
patsample<-merge(windowrange,patsample, by="PatID", all.x=T)


### interesting columns of med
med<-medatc[c("PatID", "SystemDate", "ATCCodeGruppe")]

### 1-level ATC
med<-cbind(med, substr(med$ATCCodeGruppe,1,1))
colnames(med)[4]<-"ATC1"

### diabetes
diabetes<-as.data.frame(cbind(icdcodes[,c(1,54,2)],matrix(0,
nrow=dim(icdcodes)[1],ncol=8)))
colnames(diabetes)[3:11]<-c("DM.type1", "DM.comp0","DM.comp2",
"DM.comp3","DM.comp4","DM.comp5","DM.comp6","DM.comp7","DM.comp9")

diabetes$DM.comp0<-apply(icdcodes[,c("e10.0","e11.0", "e14.0")],1,sum)
diabetes$DM.comp2<-apply(icdcodes[,c("e10.2","e11.2", "e14.2")],1,sum)
diabetes$DM.comp3<-apply(icdcodes[,c("e10.3","e11.3", "e14.3")],1,sum)
diabetes$DM.comp4<-apply(icdcodes[,c("e10.4","e11.4", "e14.4")],1,sum)
diabetes$DM.comp5<-apply(icdcodes[,c("e10.5","e11.5", "e14.5")],1,sum)
diabetes$DM.comp6<-apply(icdcodes[,c("e10.6","e11.6")],1,sum)
diabetes[apply(diabetes[,4:9],1,sum)>1,"DM.comp7"]<-1
diabetes[apply(diabetes[,4:9],1,sum)==0,"DM.comp9"]<-1

### kidney diseases
kidney<-icdcodes[,c(1,54,38:51)]
kidney[kidney[,"n18.1"]==1 | kidney[,"n18.2"]==1 |
```

```
          kidney[,"n18.3"]==1 | kidney[,"n18.4"]==1 |
          kidney[,"n18.5"]==1,"n18"]<-1
kidney[kidney[,"n08.3"]==1, "n08"]<-1

## chronic n03, n11, n18
kidney.chron<- kidney[,c("PatID", "diag.date","n03","n11","n18","n18.1"
,"n18.2","n18.3","n18.4", "n18.5")]
kidney.acute<-kidney[,c("PatID", "diag.date","n08", "n08.3","n10","n13",
"n17","n26")]
kidney.chron<-kidney.chron[apply(kidney.chron[,-(1:2)], 1, sum)>0,]
kidney.acute<-kidney.acute[apply(kidney.acute[,-(1:2)], 1, sum)>0,]

### heart disease
heart<-icdcodes[,c(1,54, 22:37)]
heart.chron<-heart[,c("PatID", "diag.date", "e78","i00", "i10",
"i11", "i11.0", "i12", "i12.0", "i15", "i20", "i25", "i42", "i48",
"i50", "i63.8")]
heart.acute<-heart[,c("PatID", "diag.date", "i21", "i63")]
heart.chron<-heart.chron[apply(heart.chron[,-(1:2)], 1, sum)>0,]
heart.acute<-heart.acute[apply(heart.acute[,-(1:2)], 1, sum)>0,]

### rename laboratory values
labor.values<-c("calcium","cholesterol", "protein", "hematocrit",
"haemoglobin", "creatinine", "FBSL", "phosphate", "triglycerides")
levels(tabllabor$Bez)<-labor.values


### rename sex
levels(patsample$Geschlecht)<-c("M", "F")


### data for datamatrix
PatID<- windowrange[,"PatID"]
```

```
### function to create model matrix
dataset<-function(PatID, age.min, age.max, windowsize, windowsize.future,
shift.size,prop.main.GFR, prop.int.GFR, prop.main.decline,
prop.int.decline){
  #windowsize in days

  ###choose keydate
  windowrange<-merge(data.frame(PatID=PatID), windowrange, by="PatID")
  PatID<-PatID[windowrange$range>windowsize+windowsize.future]
  patsample<-merge(data.frame(PatID=PatID), patsample, by="PatID",
          all.x=T)

  windowrange<-merge(data.frame(PatID=PatID), windowrange, by="PatID",
          all.x=T)
  keydate.min<-as.Date(windowrange$Min+windowsize)
  keydate.max<-as.Date(windowrange$Max-windowsize.future)

  GebDat.aux<-as.Date(ifelse(substr(patsample$GebDat,5,10)=="-02-29",
          patsample$GebDat+1, patsample$GebDat), origin="1970-01-01")
  keydate.geb.min<-as.Date(paste(as.numeric(substr(GebDat.aux,1,4))+
          age.min, substr(GebDat.aux,5,10), sep=""),format="%Y-%m-%d")
  keydate.geb.max<-as.Date(paste(as.numeric(substr(GebDat.aux,1,4))+
          age.max, substr(GebDat.aux,5,10), sep=""),format="%Y-%m-%d")

  key.min<-as.Date(ifelse(keydate.min<keydate.geb.min, keydate.geb.min,
          keydate.min), origin="1970-01-01")
  key.max<-as.Date(ifelse(keydate.max>keydate.geb.max, keydate.geb.max,
          keydate.max), origin="1970-01-01")

  window<-as.data.frame(cbind(PatID,key.min, key.max))[key.min<key.max,]
  window<-window[window$key.max-window$key.min>=windowsize+
          windowsize.future,]
  PatID<-window$PatID

  shift<-(window$key.max-window$key.min)/shift.size +1
```

```
number.shift<-floor(shift)
vary.window<-(shift-number.shift)*shift.size

pats.keydates<-NULL
for (i in (1:length(PatID))){
  start<-sample(seq(from=as.Date(window$key.min,
          origin="1970-01-01")[i],
          to=as.Date((window$key.min+vary.window),
          origin="1970-01-01")[i],1), 1)
  as.Date(window$key.max[i], origin="1970-01-01")
  rest<-NULL
  for (j in (1:(number.shift[i]-1))){
    rest[j]<-start+j*shift.size
  }
  pats.keydates<-rbind(pats.keydates,cbind(rep(PatID[i],
          number.shift[i]),
  c(start, rest)))
}
colnames(pats.keydates)<-c("PatID", "keydate")

patsample<-merge(pats.keydates, patsample, by="PatID", all.x=T)
patsample$keydate<-as.Date(as.numeric(patsample$keydate),
        origin="1970-01-01")
keydate<-as.Date(pats.keydates[,2], origin="1970-01-01")

# calculate age
patsample<-cbind(patsample,age=age.calc(patsample$GebDat,
patsample$keydate))

#med data
med.curr<-merge(x=pats.keydates,y=med, by="PatID",all.x=T )
med.curr<-med.curr[!is.na(med.curr$SystemDate),]
med.curr$keydate<-as.Date(med.curr$keydate, origin="1970-01-01")
falt.med<-med.curr["SystemDate"]<=med.curr["keydate"]&
  med.curr["SystemDate"]>=med.curr["keydate"]-windowsize
```

```
med.curr<-med.curr[falt.med,]


#labor data
lab.curr<-merge(x=pats.keydates,y=tabllabor, by="PatID", all.x=T)
lab.curr<-lab.curr[!is.na(lab.curr$SystemDate),]
lab.curr$keydate<-as.Date(lab.curr$keydate, origin="1970-01-01")
falt.lab<-lab.curr["SystemDate"]<=lab.curr["keydate"] &
  lab.curr["SystemDate"]>lab.curr["keydate"]-windowsize
lab.curr<-lab.curr[falt.lab,]

#data diabetes
diabetes.curr<-merge(x=pats.keydates, y=diabetes, by="PatID", all.x=T)
diabetes.curr$keydate<-as.Date(diabetes.curr$keydate,
        origin="1970-01-01")
diabetes.curr$diag.date<-as.Date(diabetes.curr$diag.date,
        origin="1960-01-01")
diabetes.curr<-diabetes.curr[diabetes.curr$keydate>=
        diabetes.curr$diag.date &
        diabetes.curr$diag.date>diabetes.curr$keydate-windowsize,]

#data kidney
kidney.chron.curr<-merge(x=pats.keydates, y=kidney.chron, by="PatID")
kidney.chron.curr$keydate<-as.Date(kidney.chron.curr$keydate,
        origin="1970-01-01")
kidney.chron.curr$diag.date<-as.Date(kidney.chron.curr$diag.date,
        origin="1960-01-01")
kidney.chron.curr<-kidney.chron.curr[kidney.chron.curr$keydate>=
        kidney.chron.curr$diag.date &
        kidney.chron.curr$diag.date>kidney.chron.curr$keydate-windowsize
        & !is.na(kidney.chron.curr$diag.date),]

kidney.acute.curr<-merge(x=pats.keydates, y=kidney.acute, by="PatID",
        all.x=T)
kidney.acute.curr$keydate<-as.Date(kidney.acute.curr$keydate,
```

```
        origin="1970-01-01")
kidney.acute.curr$diag.date<-as.Date(kidney.acute.curr$diag.date,
        origin="1960-01-01")
kidney.acute.curr<-kidney.acute.curr[kidney.acute.curr$keydate>=
        kidney.acute.curr$diag.date &
        kidney.acute.curr$diag.date>kidney.acute.curr$keydate-windowsize
        & !is.na(kidney.acute.curr$diag.date),]


#data heart
heart.chron.curr<-merge(x=pats.keydates, y=heart.chron, by="PatID",
        all.x=T)
heart.chron.curr$keydate<-as.Date(heart.chron.curr$keydate,
        origin="1970-01-01")
heart.chron.curr$diag.date<-as.Date(heart.chron.curr$diag.date,
        origin="1960-01-01")
        heart.chron.curr<-heart.chron.curr[heart.chron.curr$keydate>=
heart.chron.curr$diag.date &
        heart.chron.curr$diag.date>heart.chron.curr$keydate-windowsize
        & !is.na(heart.chron.curr$diag.date),]

heart.acute.curr<-merge(x=pats.keydates, y=heart.acute, by="PatID",
all.x=T)
heart.acute.curr$keydate<-as.Date(heart.acute.curr$keydate,
        origin="1970-01-01")
heart.acute.curr$diag.date<-as.Date(heart.acute.curr$diag.date,
        origin="1960-01-01")
heart.acute.curr<-heart.acute.curr[heart.acute.curr$keydate>=
        heart.acute.curr$diag.date &
        heart.acute.curr$diag.date>heart.acute.curr$keydate-windowsize
        & !is.na(heart.acute.curr$diag.date),]


print("finished data")
```

```
#create variables

##ATC level3
med.curr$ATCCodeGruppe<-med.curr$ATCCodeGruppe[drop=T]
var.med<-levels(factor(med.curr$ATCCodeGruppe))
var.med.orig<-var.med

ATC.mat<-matrix(0, nrow=dim(pats.keydates)[1],ncol=length(var.med))
colnames(ATC.mat)<-var.med

for(i in (1:(dim(pats.keydates)[1]))){
  ATC.mat[i, unique(med.curr$ATCCodeGruppe[med.curr$PatID==
          pats.keydates[i,"PatID"]&
          med.curr$keydate==pats.keydates[i, "keydate"]])]<-1
}


##ATC level1
med.curr$ATC1<-med.curr$ATC1[drop=T]
var.med1<-levels(factor(med.curr$ATC1))
var.med1.orig<-var.med1

ATC1.mat<-matrix(0, nrow=dim(pats.keydates)[1],ncol=length(var.med1))
colnames(ATC1.mat)<-var.med1

for(i in (1:(dim(pats.keydates)[1]))){
  ATC1.mat[i,
          unique(med.curr$ATC1[med.curr$PatID==pats.keydates[i,"PatID"]
          & med.curr$keydate==pats.keydates[i, "keydate"]])]<-1
}


##labor binary
lab.curr$Bez<-lab.curr$Bez[drop=T]
var.lab<-levels(factor(lab.curr$Bez))
```

```
var.lab.binary<-paste(var.lab, ".binary",sep="")
var.lab.binary.orig<-var.lab.binary

lab.binary.mat<-matrix(0, nrow=dim(pats.keydates)[1],
        ncol=length(var.lab))
colnames(lab.binary.mat)<-var.lab

for(i in (1:(dim(pats.keydates)[1]))){
  lab.binary.mat[i, unique(lab.curr$Bez[lab.curr$PatID==
          pats.keydates[i,"PatID"]&
          lab.curr$keydate==pats.keydates[i, "keydate"]])]<-1
}

colnames(lab.binary.mat)<-var.lab.binary
head(lab.binary.mat)

##diabetes
var.diabetes<-colnames(diabetes)[3:dim(diabetes)[2]]
diabetes.mat<-matrix(0,
        nrow=dim(pats.keydates)[1],ncol=length(var.diabetes))

for (i in (1:dim(pats.keydates)[1])){
  auxi<-as.matrix(diabetes.curr[diabetes.curr$PatID==
          pats.keydates[i,"PatID"]&
          diabetes.curr$keydate<=pats.keydates[i,"keydate"],
          4:dim(diabetes.curr)[2]])
  diabetes.mat[i,]<-apply(auxi, 2,sum)
}
diabetes.mat[diabetes.mat>1]<-1
colnames(diabetes.mat)<-var.diabetes
diabetes.mat[apply(diabetes.mat[,2:7],1,sum)>1,"DM.comp7"]<-1
diabetes.mat[apply(diabetes.mat[,2:7],1,sum)<=1,"DM.comp7"]<-0
diabetes.mat[apply(diabetes.mat[,2:8],1,sum)==0,"DM.comp9"]<-1
diabetes.mat[apply(diabetes.mat[,2:8],1,sum)>0,"DM.comp9"]<-0
var.diabetes.orig<-var.diabetes
```

```
##kidney
var.kidney.chron<-colnames(kidney.chron)[3:dim(kidney.chron)[2]]
kidney.chron.mat<-matrix(0,
nrow=dim(pats.keydates)[1],ncol=length(var.kidney.chron))

for (i in (1:dim(pats.keydates)[1])){
  auxi<-as.matrix(kidney.chron.curr[kidney.chron.curr$PatID==
          pats.keydates[i,"PatID"]&
          kidney.chron.curr$keydate<=pats.keydates[i, "keydate"],
          4:dim(kidney.chron.curr)[2]])
  kidney.chron.mat[i,]<-apply(auxi, 2,sum)
}
kidney.chron.mat[kidney.chron.mat>1]<-1
colnames(kidney.chron.mat)<-var.kidney.chron
var.kidney.chron.orig<-var.kidney.chron


var.kidney.acute<-colnames(kidney.acute)[3:dim(kidney.acute)[2]]
kidney.acute.mat<-matrix(0, nrow=dim(pats.keydates)[1],
        ncol=length(var.kidney.acute))

for (i in (1:dim(pats.keydates)[1])){
  auxi<-as.matrix(kidney.acute.curr[kidney.acute.curr$PatID==
          pats.keydates[i,"PatID"]&
          kidney.acute.curr$keydate==pats.keydates[i, "keydate"],
          4:dim(kidney.acute.curr)[2]])
  kidney.acute.mat[i,]<-apply(auxi, 2,sum)
}
kidney.acute.mat[kidney.acute.mat>1]<-1
colnames(kidney.acute.mat)<-var.kidney.acute
var.kidney.acute.orig<-var.kidney.acute
```

```
##heart
var.heart.chron<-colnames(heart.chron)[3:dim(heart.chron)[2]]
heart.chron.mat<-matrix(0, nrow=dim(pats.keydates)[1],
ncol=length(var.heart.chron))

for (i in (1:dim(pats.keydates)[1])){
  auxi<-as.matrix(heart.chron.curr[heart.chron.curr$PatID==
          pats.keydates[i,"PatID"]&
          heart.chron.curr$keydate<=pats.keydates[i, "keydate"],
          4:dim(heart.chron.curr)[2]])
  heart.chron.mat[i,]<-apply(auxi, 2,sum)
}
heart.chron.mat[heart.chron.mat>1]<-1
colnames(heart.chron.mat)<-var.heart.chron
var.heart.chron.orig<-var.heart.chron


var.heart.acute<-colnames(heart.acute)[3:dim(heart.acute)[2]]
heart.acute.mat<-matrix(0, nrow=dim(pats.keydates)[1]
,ncol=length(var.heart.acute))

for (i in (1:dim(pats.keydates)[1])){
  auxi<-as.matrix(heart.acute.curr[heart.acute.curr$PatID==
          pats.keydates[i,"PatID"]&
          heart.acute.curr$keydate==pats.keydates[i, "keydate"],
          4:dim(heart.acute.curr)[2]])
  heart.acute.mat[i,]<-apply(auxi, 2,sum)
}
heart.acute.mat[heart.acute.mat>1]<-1
colnames(heart.acute.mat)<-var.heart.acute
var.heart.acute.orig<-var.heart.acute



##creatinine number of measurements
```

```
creatinine.number<-apply(pats.keydates,1,function(x)
sum(lab.curr$PatID==x[1] & lab.curr$Bez=="creatinine" &
lab.curr$keydate==x[2]))

###labor values
lab.orig.mat<-lab.binary.mat

for(i in (1:dim(lab.binary.mat)[1])){
  for(j in (1:dim(lab.binary.mat)[2])){
    lab.orig.mat[i,j]<-ifelse(lab.binary.mat[i,j]==1,
              tail(lab.curr$Wert[lab.curr$PatID==pats.keydates[i,1]&
              lab.curr$keydate==pats.keydates[i,2]&
              lab.curr$Bez==var.lab[j]],n=1),NA)
  }
}

colnames(lab.orig.mat)<-var.lab



labs.adapted<-apply(lab.orig.mat,2, function(x){
  ifelse(x<quantile(x,prob=0.01, na.rm=T),
         quantile(x,prob=0.01, na.rm=T),
         ifelse(x>quantile(x,prob=0.99, na.rm=T),
         quantile(x,prob=0.99, na.rm=T),x))
})

lab.mat<- apply(labs.adapted, 2, function(x){
  ifelse(is.na(x),mean(x, na.rm=T),x)
})

##dummy's diabetes onset
auxi.diabetes<-merge(pats.keydates, unique(icdcodes[,c(1,52)]),
by="PatID")
diabetes.days<-as.Date(auxi.diabetes[,2],
```

```
origin="1970-01-01")-as.Date(auxi.diabetes[,3], origin="1960-01-01")

dummy.diabetes<-matrix(0, ncol=2, nrow=dim(pats.keydates)[1])
colnames(dummy.diabetes)<-paste("dummy.diabetes",1:2, sep="")
for(i in (1:dim(pats.keydates)[[1]])){
  if(diabetes.days[i]>1826){
    dummy.diabetes[i,1]<-1
    dummy.diabetes[i,2]<-1
  }
  if(365<=diabetes.days[i]&diabetes.days[i]<1826){
    dummy.diabetes[i,1]<-1
    dummy.diabetes[i,2]<-0
  }
}
dummy.diabetes<-as.data.frame(dummy.diabetes)
dummy.diabetes[,1]<-as.factor(dummy.diabetes[,1])
dummy.diabetes[,2]<-as.factor(dummy.diabetes[,2])



## original Matrix
colnames(labs.adapted)<-paste(colnames(labs.adapted),".adapted",sep="")
orig.datenmatrix<-as.data.frame(cbind(PatID=pats.keydates[,1],
ATC.mat, ATC1.mat,diabetes.mat, kidney.chron.mat, kidney.acute.mat ,
heart.chron.mat, heart.acute.mat, lab.binary.mat, labs.adapted,
lab.orig.mat,creatinine.number))
orig.datenmatrix<-cbind(orig.datenmatrix,dummy.diabetes,
patsample[,c("PatID","Geschlecht","age","Bundesland", "OrdID")])

orig.datenmatrix$Bundesland<-as.factor(orig.datenmatrix$Bundesland)
orig.datenmatrix$OrdID<-as.factor(orig.datenmatrix$OrdID)



## model matrix
datenmatrix.decline<-as.data.frame(cbind(PatID=pats.keydates[,1],
```

```
ATC.mat, ATC1.mat,diabetes.mat, kidney.chron.mat, kidney.acute.mat,
heart.chron.mat, heart.acute.mat, lab.binary.mat,
lab.mat,creatinine.number, dummy.diabetes))
creatinine<-lab.mat[,colnames(lab.mat)=="creatinine"]

min.main.GFR<-prop.main.GFR*dim(pats.keydates)[1]
ATC.mat<-ATC.mat[,apply(ATC.mat,2,sum)>=min.main.GFR]
var.med<-colnames(ATC.mat)

ATC1.mat<-ATC1.mat[,apply(ATC1.mat,2,sum)>=min.main.GFR]
var.med1<-colnames(ATC1.mat)

lab.binary.mat<-lab.binary.mat[,apply(lab.binary.mat,2,sum)>=
min.main.GFR]
var.lab.binary<-colnames(lab.binary.mat)

lab.mat<-lab.mat[,substring(colnames(lab.binary.mat),1,
nchar(colnames(lab.binary.mat))-7)]  # use only appearing labs
var.lab<-colnames(lab.mat)

diabetes.mat<-diabetes.mat[,apply(diabetes.mat,2,sum)>=min.main.GFR]
var.diabetes<-colnames(diabetes.mat)

kidney.chron.mat<-kidney.chron.mat[,apply(kidney.chron.mat,2,sum)>=
min.main.GFR]
var.kidney.chron<-colnames(kidney.chron.mat)

kidney.acute.mat<-kidney.acute.mat[,apply(kidney.acute.mat,2,sum)>=
min.main.GFR]
var.kidney.acute<-colnames(kidney.acute.mat)


heart.chron.mat<-heart.chron.mat[,apply(heart.chron.mat,2,sum)>=
min.main.GFR]
var.heart.chron<-colnames(heart.chron.mat)
```

```
heart.acute.mat<-heart.acute.mat[,apply(heart.acute.mat,2,sum)>=
min.main.GFR]
var.heart.acute<-colnames(heart.acute.mat)



datenmatrix<-as.data.frame(cbind(pats.keydates[,1], ATC.mat, ATC1.mat,
diabetes.mat,kidney.chron.mat, kidney.acute.mat, heart.chron.mat,
heart.acute.mat, lab.binary.mat, lab.mat,creatinine.number))
colnames(datenmatrix)[1]<-"PatID"

# if creatinine is not there
datenmatrix$creatinine.binary<-as.factor(
orig.datenmatrix$creatinine.binary)
datenmatrix$creatinine<-creatinine

datenmatrix<-cbind(datenmatrix, dummy.diabetes,
patsample[,c("Geschlecht","age","OrdID")])

datenmatrix$OrdID<-as.factor(datenmatrix$OrdID)

var.bld<-paste("province",unique(patsample$Bundesland), sep="")
bld.mat<-matrix(0, nrow=dim(pats.keydates)[1],ncol=length(var.bld))
for(i in (1:(dim(pats.keydates)[1]))){
  bld.mat[i, patsample$Bundesland[i]]<-1
}
bld.mat<-as.data.frame(bld.mat)
for(i in (1:length(bld.mat))){
  bld.mat[,i]<-as.factor(bld.mat[,i])
}
colnames(bld.mat)<-var.bld
datenmatrix<-cbind(datenmatrix,bld.mat)

print("created variables")
```

```
## interaction effects

allcombs<-combs(c(var.med,var.med1,var.lab.binary,var.diabetes,
var.kidney.chron, var.kidney.acute, var.heart.chron,
var.heart.acute),2)

var.int<-apply(allcombs,1,function(x){paste(x[1],x[2],sep="_")})

aux.var<-c(var.med,var.med1, var.diabetes, var.lab.binary,
var.kidney.chron, var.kidney.acute, var.heart.chron, var.heart.acute)

inter.mat<-matrix(0, nrow=dim(pats.keydates)[1],ncol=length(var.int),
                  dimnames=list(NULL,var.int))

combis<-list(NULL)
for(i in (1:(dim(pats.keydates)[1]))){
  if(sum(datenmatrix[i,1:length(aux.var)]==1)>1){
    combis[[i]]<-combs(aux.var[datenmatrix[i,2:(length(aux.var)+1)]==1]
    ,2)
    combi.var<-c(paste(combis[[i]][,1],combis[[i]][,2],sep="_"),
    paste(combis[[i]][,2],combis[[i]][,1],sep="_"))
    combi.var<-combi.var[(combi.var %in% var.int)]
    inter.mat[i,combi.var]<-1
  }
}


inter.mat<-inter.mat[,-which(nchar(colnames(inter.mat))==5 &
substr(colnames(inter.mat),1,1)==substr(colnames(inter.mat),5,5))]

inter.mat<-as.data.frame(inter.mat)
if(sum(colnames(inter.mat)=="n18_n18.1")>0)
        inter.mat<-inter.mat[,-which(colnames(inter.mat)=="n18_n18.1")]
if(sum(colnames(inter.mat)=="n18_n18.2")>0
```

```
        inter.mat<-inter.mat[,-which(colnames(inter.mat)=="n18_n18.2")]
if(sum(colnames(inter.mat)=="n18_n18.3")>0)
        inter.mat<-inter.mat[,-which(colnames(inter.mat)=="n18_n18.3")]
if(sum(colnames(inter.mat)=="n18_n18.4")>0)
        inter.mat<-inter.mat[,-which(colnames(inter.mat)=="n18_n18.4")]
if(sum(colnames(inter.mat)=="i11_i11.0")>0
        inter.mat<-inter.mat[,-which(colnames(inter.mat)=="i11_i11.0")]
if(sum(colnames(inter.mat)=="i12_i12.0")>0
        inter.mat<-inter.mat[,-which(colnames(inter.mat)=="i12_i12.0")]


var.int.orig<-colnames(inter.mat)
orig.datenmatrix<-cbind(orig.datenmatrix,lapply(inter.mat,factor))

min.int.GFR<-prop.int.GFR*dim(pats.keydates)[1] ### restrict interactions

inter.mat<-inter.mat[,lapply(inter.mat,sum)>min.int.GFR]

if(length(inter.mat)>0){
  var.int<-colnames(inter.mat)
  datenmatrix<-cbind(datenmatrix,lapply(inter.mat, factor))
}

print("created interaction effects")

# keydate GFR
GFR.orig<-apply(orig.datenmatrix,1,function(x)
        {ifelse(x["Geschlecht"]=="M",
                144*min(as.numeric(x["creatinine.adapted"])/0.9,1)^(-0.411)*
              max(as.numeric(x["creatinine.adapted"])/0.9,1)^(-1.209)
                *0.993^as.numeric(x["age"]),
                144*min(as.numeric(x["creatinine.adapted"])/0.7,1)^(-0.329)*
                max(as.numeric(x["creatinine.adapted"])/0.7,1)^(-1.209)*
                0.993^as.numeric(x["age"])*1.018)})
```

```
GFR<-GFR.orig
GFR[is.na(GFR.orig)]<- mean(GFR.orig, na.rm=T)

datenmatrix<-cbind(datenmatrix,GFR)

# GFR groups
datenmatrix$GFR.groups<-ifelse(GFR>=60, 1, ifelse(45<=GFR & GFR<60, 2,
                ifelse(30<=GFR & GFR<45, 3,
                ifelse(15<=GFR & GFR<30, 4, 5))))

datenmatrix$GFR.groups<-factor(datenmatrix$GFR.groups, levels=c(1:5),
                        labels=c("0-2","3a","3b","4","5"))



#GFR 0/1 future
lab.curr.all<-merge(x=pats.keydates,y=tabllabor, by="PatID", all.x=T)
lab.curr.all<-lab.curr.all[!is.na(lab.curr.all$SystemDate),]
lab.curr.all$keydate<-as.Date(lab.curr.all$keydate,
origin="1970-01-01")

falt.gfr<- lab.curr.all["keydate"]+windowsize.future>
        lab.curr.all["SystemDate"]&
        lab.curr.all["keydate"]<lab.curr.all["SystemDate"]&
        lab.curr.all["LabKey"]==6
        falt.gfr[is.na(falt.gfr)]<-FALSE
lab.curr.all<-lab.curr.all[falt.gfr,]

datenmatrix$GFR.binary.future<-rep(0, dim(datenmatrix)[1])

for (i in (1:dim(lab.curr.all)[1])){
  datenmatrix$GFR.binary.future[which(datenmatrix$PatID==
          lab.curr.all[i,"PatID"] &
          pats.keydates[,"keydate"]==lab.curr.all[i,"keydate"])]<-1
}
```

```
datenmatrix$GFR.binary.future<-as.factor(datenmatrix$GFR.binary.future)

#GFR future
krea.future.orig<-apply(datenmatrix,1,function(x){
  ifelse(x["GFR.binary.future"]==1,
         tail(lab.curr.all$Wert[lab.curr.all$PatID==x[1]
         &lab.curr.all$keydate==pats.keydates[,"keydate"]],
          n=1),NA)})

krea.future.orig<-rep(NA, times=dim(datenmatrix)[1])
for(i in (1: dim(datenmatrix)[1])){
if(datenmatrix[i,"GFR.binary.future"]==1){
  krea.future.orig[i]<-
  tail(lab.curr.all$Wert[lab.curr.all[,"PatID"]==datenmatrix[i,1]
     & lab.curr.all[,"keydate"]==pats.keydates[i,"keydate"]],n=1)
  }
}



krea.future.adapted<-ifelse(krea.future.orig<quantile(krea.future.orig,
        prob=0.05, na.rm=T),quantile(krea.future.orig,prob=0.05,
        na.rm=T),
        ifelse(krea.future.orig>quantile(krea.future.orig,prob=0.95,
        na.rm=T), quantile(krea.future.orig,prob=0.95,
        na.rm=T),krea.future.orig))

krea.future<- ifelse(is.na(krea.future.adapted),
mean(krea.future.adapted, na.rm=T),krea.future.adapted)

orig.datenmatrix<-cbind(orig.datenmatrix,krea.future.orig)
datenmatrix<-cbind(datenmatrix, krea.future)

GFR.future<-apply(cbind(datenmatrix, krea.future.adapted),1,
function(x) {ifelse(x["Geschlecht"]=="M",
144*min(as.numeric(x["krea.future.adapted"])/0.9,1)^(-0.411)*
```

```
max(as.numeric(x["krea.future.adapted"])/0.9,1)^(-1.209)*
0.993^as.numeric(x["age"]),
144*min(as.numeric(x["krea.future.adapted"])/0.7,1)^(-0.329)*
max(as.numeric(x["krea.future.adapted"])/0.7,1)^(-1.209)*
0.993^as.numeric(x["age"])*1.018)})
GFR.future.orig<-GFR.future
GFR.future[is.na(GFR.future)]<- mean(GFR.future, na.rm=T)

datenmatrix<-cbind(datenmatrix, GFR.future)

datenmatrix$GFR.groups.future<-ifelse(GFR.future>=60, 1,
ifelse(45<=GFR.future & GFR.future<60, 2,
ifelse(30<=GFR.future & GFR.future<45, 3,
ifelse(15<=GFR.future & GFR.future<30, 4,5))))

datenmatrix$GFR.groups.future<-factor(datenmatrix$GFR.groups.future,
levels=c(1:5), labels=c("0-2","3a","3b","4","5"))

#original matrix
orig.datenmatrix<-cbind(orig.datenmatrix,GFR.orig)
orig.datenmatrix$GFR.groups<-ifelse(GFR.orig>=60, 1,
ifelse(45<=GFR.orig & GFR.orig<60, 2,
ifelse(30<=GFR.orig & GFR.orig<45, 3,
ifelse(15<=GFR.orig & GFR.orig<30, 4, 5))))

orig.datenmatrix$GFR.groups<-factor(orig.datenmatrix$GFR.groups,
levels=c(1:5), labels=c("0-2","3a","3b","4","5"))

orig.datenmatrix<-cbind(orig.datenmatrix,GFR.future.orig)

orig.datenmatrix$GFR.groups.future<-ifelse(GFR.future.orig>=60, 1,
ifelse(45<=GFR.future.orig & GFR.future.orig<60, 2,
ifelse(30<=GFR.future.orig & GFR.future.orig<45, 3,
ifelse(15<=GFR.future.orig & GFR.future.orig<30, 4,5))))
```

```
orig.datenmatrix$GFR.groups.future<-
factor(orig.datenmatrix$GFR.groups.future, levels=c(1:5),
labels=c("0-2","3a","3b","4","5"))

orig.datenmatrix$decline<-as.factor(apply(orig.datenmatrix,1,
function(x){ifelse(x["GFR.groups.future"]>x["GFR.groups"],1,0)}))


for(i in (2:(length(var.med.orig)+length(var.med1.orig)+
        length(var.lab.binary.orig)+length(var.diabetes.orig)+
        length(var.kidney.chron.orig)+ length(var.kidney.acute.orig)+
        length(var.heart.chron.orig)+length(var.heart.acute.orig)+1))){
                orig.datenmatrix[i]<-as.factor(orig.datenmatrix[[i]])
}

#decline
datenmatrix$decline<-as.factor(apply(datenmatrix,1,function(x){
  ifelse(x["GFR.groups.future"]>x["GFR.groups"],1,0)}))


for(i in (2:(length(var.med)+length(var.med1)+length(var.lab.binary)+
        length(var.diabetes) + length(var.kidney.chron) +
        length(var.kidney.acute)+ length(var.heart.chron) +
        length(var.heart.acute)+1))){
          datenmatrix[i]<-as.factor(datenmatrix[[i]])
}

print("finished GFR matrix")


### model matrix for progression
for(i in (2:(length(var.med.orig)+length(var.med1.orig)+
        length(var.lab.binary.orig)+length(var.diabetes.orig)+
        length(var.kidney.chron.orig)+length(var.kidney.acute.orig)+
        length(var.heart.chron.orig)+length(var.heart.acute.orig)+1))){
```

```
          datenmatrix.decline[i]<-as.factor(datenmatrix.decline[[i]])
}

datenmatrix.decline<-datenmatrix.decline[
datenmatrix$creatinine.binary==1 &datenmatrix$GFR.binary.future==1,]
orig.datenmatrix.decline<-
orig.datenmatrix[datenmatrix$creatinine.binary==1 &
datenmatrix$GFR.binary.future==1,]

min.main.decline<-prop.main.decline*dim(datenmatrix.decline)[1]
min.int.decline<-prop.int.decline*dim(datenmatrix.decline)[1]

aux.mat<-orig.datenmatrix[datenmatrix$creatinine.binary==1 &
datenmatrix$GFR.binary.future==1,]
aux.var<-c(var.med.orig, var.med1.orig, var.diabetes.orig,
        var.kidney.chron.orig, var.kidney.acute.orig,
        var.heart.chron.orig, var.heart.acute.orig,var.lab.binary.orig)

#reduce main effects
FT.main<-NULL
for(i in (1:length(aux.var))){
  FT.main[i]<-sum(as.numeric(aux.mat[,aux.var[i]])-1)<
  min.main.decline
}

reduce.main<-aux.var[FT.main]

if(length(reduce.main)>0 & sum(nchar(reduce.main)>8)>1)
{
  aux.red<-reduce.main[nchar(reduce.main)>8]
  reduce.main.lab<-NULL

  for(i in (1:length(aux.red))) {
  reduce.main.lab[i]<-substring(aux.red[i],1, nchar(aux.red[i])-7)}
  datenmatrix.decline<-datenmatrix.decline[,-c(match(reduce.main,
```

```
          colnames(datenmatrix.decline)),
          match(reduce.main.lab,colnames(datenmatrix.decline)))]
}else{
  if(length(reduce.main)>0)
  {
    datenmatrix.decline<-datenmatrix.decline[,-match(reduce.main,
    colnames(datenmatrix.decline))]
  }
}

# if creatinine.binary is not here
datenmatrix.decline$creatinine.binary<-as.factor(
aux.mat$creatinine.binary)
datenmatrix.decline$creatinine<-creatinine[
datenmatrix$creatinine.binary==1 &datenmatrix$GFR.binary.future==1]

if(which(colnames(datenmatrix.decline)=="creatinine.number")>3){
  main.var<-colnames(datenmatrix.decline[,2:
          (which(colnames(datenmatrix.decline)=="creatinine.number")
          -1)])

  datenmatrix.decline<-cbind(datenmatrix.decline,
  patsample[datenmatrix$creatinine.binary==1 &
          datenmatrix$GFR.binary.future==1
          ,c("Geschlecht","age","OrdID")])
  datenmatrix.decline$OrdID<-as.factor(datenmatrix.decline$OrdID[
          datenmatrix$creatinine.binary==1
          & datenmatrix$GFR.binary.future==1])
  bld.mat.decline<-bld.mat[datenmatrix$creatinine.binary==1 &
          datenmatrix$GFR.binary.future==1,]
  datenmatrix.decline<-cbind(datenmatrix.decline,bld.mat.decline)

  ##interaction effects
  var.med.main<-var.med.orig[var.med.orig %in% main.var]
  var.med1.main<-var.med1.orig[var.med1.orig %in% main.var]
```

```
var.diabetes.main<-var.diabetes.orig[var.diabetes.orig %in% main.var]
var.kidney.chron.main<-var.kidney.chron.orig[
var.kidney.chron.orig %in% main.var]
var.kidney.acute.main<-var.kidney.acute.orig[
var.kidney.acute.orig %in% main.var]
var.heart.chron.main<-var.heart.chron.orig[
var.heart.chron.orig %in% main.var]
var.heart.acute.main<-var.heart.acute.orig[
var.heart.acute.orig %in% main.var]

allcombs<-combs(main.var,2)

var.int<-apply(allcombs,1,function(x){paste(x[1],x[2],sep="_")})

inter.mat<-matrix(0, nrow=dim(datenmatrix.decline)[1],ncol=
        length(var.int), dimnames=list(NULL,var.int))

combis<-list(NULL)
for(i in (1:(dim(datenmatrix.decline)[1]))){
  if(sum(aux.mat[i,1:(sum(aux.var%in%main.var)+1)]==1)>1){
    combis[[i]]<-combs(aux.var[aux.mat[i,2:(length(aux.var)+1)]==1],
            2)
    combi.var<-c(paste(combis[[i]][,1],combis[[i]][,2],sep="_"),
            paste(combis[[i]][,2],combis[[i]][,1],sep="_"))
    combi.var<-combi.var[(combi.var %in% var.int)]
    inter.mat[i,combi.var]<-1
  }
}
colnames(datenmatrix.decline)
apply(inter.mat,2,sum)

inter.mat<-inter.mat[,-which(nchar(colnames(inter.mat))==5 &
        substr(colnames(inter.mat),1,1)==substr(colnames(inter.mat),
        5,5))]
```

```
  inter.mat<-as.data.frame(inter.mat)
  inter.mat<-inter.mat[,apply(inter.mat,2,sum)>min.int.decline]
  if(sum(colnames(inter.mat)=="n18_n18.1")>0)
          inter.mat<-inter.mat[,-which(colnames(inter.mat)=="n18_n18.1")]
  if(sum(colnames(inter.mat)=="n18_n18.2")>0)
          inter.mat<-inter.mat[,-which(colnames(inter.mat)=="n18_n18.2")]
  if(sum(colnames(inter.mat)=="n18_n18.3")>0)
          inter.mat<-inter.mat[,-which(colnames(inter.mat)=="n18_n18.3")]
  if(sum(colnames(inter.mat)=="n18_n18.4")>0)
          inter.mat<-inter.mat[,-which(colnames(inter.mat)=="n18_n18.4")]
  if(sum(colnames(inter.mat)=="i11_i11.0")>0)
          inter.mat<-inter.mat[,-which(colnames(inter.mat)=="i11_i11.0")]
  if(sum(colnames(inter.mat)=="i12_i12.0")>0)
          inter.mat<-inter.mat[,-which(colnames(inter.mat)=="i12_i12.0")]


  if(length(inter.mat)>0){datenmatrix.decline<-cbind(
          datenmatrix.decline,lapply(inter.mat,factor))}
        }


datenmatrix.decline<-cbind(datenmatrix.decline,GFR=GFR[
        datenmatrix$creatinine.binary==1 &
        datenmatrix$GFR.binary.future==1],
        GFR.groups=datenmatrix$GFR.groups[
        datenmatrix$creatinine.binary==1
        & datenmatrix$GFR.binary.future==1],
        GFR.binary.future= datenmatrix$GFR.binary.future[
        datenmatrix$creatinine.binary==1 &
        datenmatrix$GFR.binary.future==1],
        krea.future=krea.future[datenmatrix$creatinine.binary==1
        & datenmatrix$GFR.binary.future==1],
        GFR.groups.future=datenmatrix$GFR.groups.future[
        datenmatrix$creatinine.binary==1 &
        datenmatrix$GFR.binary.future==1],
```

```
          decline=datenmatrix$decline[datenmatrix$creatinine.binary==1
          & datenmatrix$GFR.binary.future==1])


 ### change names
 colnames(datenmatrix)[colnames(datenmatrix)=="Geschlecht"]<-"sex"
 colnames(datenmatrix.decline)[colnames(datenmatrix.decline)==
         "Geschlecht"]<-"sex"


  return(list(datenmatrix, datenmatrix.decline,orig.datenmatrix,
          orig.datenmatrix.decline))
}


## model matrices

start<-Sys.time()
daten.shift365.365.730.1<-dataset(PatID, 20,85,
        365,365,730,0.1,0.2,0.1,0.2)
daten.shift365.365.730.2<-dataset(PatID, 20,85,
        365,365,730,0.05,0.1,0.05,0.1)
daten.shift365.365.730.3<-dataset(PatID, 20,85,
        365,365,730,0.01,0.01,0.01,0.01)
daten.shift365.180.730<-dataset(PatID, 20,85, 365,180,
        730,0.01,0.05,0.01,0.05)
daten.shift365.365.730<-dataset(PatID, 20,85, 365,365,
        730,0.01,0.05,0.01,0.05)
daten.shift730.365.1095<-dataset(PatID, 20,85, 730,365,
        1095,0.01,0.05,0.01,0.05)
daten.shift1095.365.1460<-dataset(PatID,20,85,1095,365,
        1460,0.01,0.05,0.01,0.05)
end<-Sys.time()
end-start
```

care model and progression model

```
library(glmnet)
library(pROC)
library(ROCR)
library(splines)
library(caTools)
library(gbm)
library(doParallel)
registerDoParallel(4)

for(i in 1:4){
  colnames(daten.shift365.365.730.1[[i]])[colnames(
  daten.shift365.365.730.1[[i]])=="diabetes.mat"]<-"DM.comp9"
  colnames(daten.shift365.365.730.2[[i]])[colnames(
  daten.shift365.365.730.2[[i]])=="diabetes.mat"]<-"DM.comp9"
  colnames(daten.shift365.365.730.1[[i]])[colnames(
  daten.shift365.365.730.1[[i]])=="kidney.acute.mat"]<-"n08"
  colnames(daten.shift365.365.730.2[[i]])[colnames(
  daten.shift365.365.730.2[[i]])=="kidney.acute.mat"]<-"n08"
  colnames(daten.shift365.365.730.3[[i]])[colnames(
  daten.shift365.365.730.3[[i]])=="kidney.acute.mat"]<-"n08"
  colnames(daten.shift365.365.730[[i]])[colnames(
  daten.shift365.365.730[[i]])=="kidney.acute.mat"]<-"n08"
  colnames(daten.shift365.180.730[[i]])[colnames(
  daten.shift365.180.730[[i]])=="kidney.acute.mat"]<-"n08"
  colnames(daten.shift730.365.1095[[i]])[colnames(
  daten.shift730.365.1095[[i]])=="kidney.acute.mat"]<-"n08"
  colnames(daten.shift1095.365.1460[[i]])[colnames(
  daten.shift1095.365.1460[[i]])=="kidney.acute.mat"]<-"n08"
}


datenmat<-list(daten.shift365.180.730[[1]],
        daten.shift365.365.730.1[[1]], daten.shift365.365.730.2[[1]],
        daten.shift365.365.730.3[[1]], daten.shift365.365.730[[1]],
```

```
        daten.shift730.365.1095[[1]], daten.shift1095.365.1460[[1]])

datenmat.orig<-list(daten.shift365.180.730[[3]],
        daten.shift365.365.730.1[[3]], daten.shift365.365.730.2[[3]],
        daten.shift365.365.730.3[[3]], daten.shift365.365.730[[3]],
        daten.shift730.365.1095[[3]], daten.shift1095.365.1460[[3]])

datenmat.decline<-list(daten.shift365.180.730[[2]],
        daten.shift365.365.730.1[[2]], daten.shift365.365.730.2[[2]],
        daten.shift365.365.730.3[[2]], daten.shift365.365.730[[2]],
        daten.shift730.365.1095[[2]], daten.shift1095.365.1460[[2]])

datenmat.decline.orig<-list(daten.shift365.180.730[[4]],
        daten.shift365.365.730.1[[4]], daten.shift365.365.730.2[[4]],
        daten.shift365.365.730.3[[4]], daten.shift365.365.730[[4]],
        daten.shift730.365.1095[[4]], daten.shift1095.365.1460[[4]])


names(datenmat)<-c("daten.shift365.180.730",
        "daten.shift365.365.730.1", "daten.shift365.365.730.2",
        "daten.shift365.365.730.3",        "daten.shift365.365.730",
        "daten.shift730.365.1095", "daten.shift1095.365.1460")
names(datenmat.orig)<-c("daten.shift365.180.730",
        "daten.shift365.365.730.1", "daten.shift365.365.730.2",
        "daten.shift365.365.730.3", "daten.shift365.365.730",
        "daten.shift730.365.1095", "daten.shift1095.365.1460")
names(datenmat.decline)<-c("daten.shift365.180.730",
        "daten.shift365.365.730.1", "daten.shift365.365.730.2",
        "daten.shift365.365.730.3","daten.shift365.365.730",
        "daten.shift730.365.1095", "daten.shift1095.365.1460")
names(datenmat.decline.orig)<-c("daten.shift365.180.730",
        "daten.shift365.365.730.1", "daten.shift365.365.730.2",
        "daten.shift365.365.730.3", "daten.shift365.365.730",
        "daten.shift730.365.1095", "daten.shift1095.365.1460")
```

```
cindex<-function(x,y, abs=TRUE) {
  c1<-wilcox.test(x~y)$statistic/sum(y==0)/sum(y==1)
  if(abs){
      if(c1<0.5) c1<-1-c1
      }
  return(c1)
  }

path<-function(obj,byvar=TRUE,...){
#     coef(obj$fit)[,obj$fit$lambda==obj$cv.fit$lambda.min]
      cat("\nAnalysis of cumulative inclusion of covariates:\n\n")
      nam<-rownames(coef(obj$glmnet.fit))
      nvar<-length(nam)
      if(!byvar){
        mat<-data.frame(step=1:length(obj$glmnet.fit$lambda),
        lambda=obj$glmnet.fit$lambda, cvm=obj$cvm)
        mat.added=rep("",length(obj$glmnet.fit$lambda))
        mat.optimal=rep("",length(obj$glmnet.fit$lambda))
        for(i in 2:dim(coef(obj$glmnet.fit))[2]){
          added<-""
          for(j in 1:length(nam)) if(coef(obj$glmnet.fit)[j,i]!=0 &
          coef(obj$glmnet.fit)[j,i-1]==0) added<-paste(added,nam[j],
          sep="    ")
          if(obj$glmnet.fit$lambda[i]==obj$lambda.min)
          mat.optimal[i]<-"*******"
          if(added!="") mat.added[i]<-added
        }
        mat$optimal<-mat.optimal
        mat$added.variables.at.step<-mat.added
        return(mat[mat$added.variables.at.step!="" | mat$optimal!="",])
        } else {
        mat<-matrix(0, nvar, 4)
        colnames(mat)<-c("step","lambda", "cvm", "df")
        rownames(mat)<-nam
```

```
        nlamb<-length(obj$lambda)
        for(j in 1:length(nam)){
            if(sum(coef(obj$glmnet.fit)[j,]!=0)==0) i<-NULL
            else  i<-head((1:nlamb)[coef(obj$glmnet.fit)[j,]!=0],1)
            if(is.null(i)) {
              mat[j,1]<-nlamb+1
              mat[j,2]<-NA
              mat[j,3]<-NA
              mat[j,4]<-nvar
              } else {
              mat[j,1]<-i
              mat[j,2]<-obj$glmnet.fit$lambda[i]
              mat[j,3]<-obj$cvm[i]
              mat[j,4]<-sum(coef(obj$glmnet.fit)[,i]!=0)
              }
            }
        mat<-mat[order(mat[,1]),]
        return(mat)
      }
    }


### care model

Lasso.GFR01<-datenmat
for(k in c(1:length(datenmat))){

  set.seed(1234)

  ## create dummy variables for GFR
  dummy.GFR<-matrix(0, ncol=3, nrow=dim(datenmat[[k]])[1])
  colnames(dummy.GFR)<-paste("dummy.GFR",1:3, sep="")
  for(i in (1:dim(datenmat[[k]])[[1]])){
    if(datenmat[[k]]$GFR.groups[i]=="3a"){
      dummy.GFR[i,1]<-1
```

```
  }
  if(datenmat[[k]]$GFR.groups[i]=="3b"){
    dummy.GFR[i,1]<-1
    dummy.GFR[i,2]<-1
  }
  if(datenmat[[k]]$GFR.groups[i]=="4"){
    dummy.GFR[i,1]<-1
    dummy.GFR[i,2]<-1
    dummy.GFR[i,3]<-1
  }
}
dummy.GFR<-as.data.frame(dummy.GFR)
dummy.GFR[,1]<-as.factor(dummy.GFR[,1])
dummy.GFR[,2]<-as.factor(dummy.GFR[,2])
dummy.GFR[,3]<-as.factor(dummy.GFR[,3])

## create splines
GFR.splines<-as.matrix(bs(datenmat[[k]]$GFR,
        knots=quantile(datenmat.orig[[k]]$GFR.orig,probs=c(0.1,0.5,0.9),
        na.rm=T)))
colnames(GFR.splines)<-paste("GFR.splines",1:6, sep="")

lab.values<-c("hematocrit", "protein", "calcium", "triglycerides",
        "FBSL","cholesterol", "haemoglobin", "phosphat")
lab.splines<-NULL
name<-NULL
for(i in (1:length(lab.values))){
  if(sum(lab.values[i]==colnames(datenmat[[k]]))==1){
    lab.splines<-cbind(lab.splines,as.matrix(
            bs(datenmat[[k]][,lab.values[i]],
            knots=quantile(datenmat.orig[[k]][,paste(lab.values[i],
            ".adapted",sep="")],probs=c(0.1,0.5,0.9), na.rm=T))))
    name<-c(name, paste(lab.values[i],".splines",1:6, sep=""))
  }
}
```

```
colnames(lab.splines)<-name

age.splines<-as.matrix(bs(datenmat[[k]]$age,
        knots=quantile(datenmat.orig[[k]]$age,probs=c(0.1,0.5,0.9),
        na.rm=T)))
colnames(age.splines)<-paste("age.splines",1:6, sep="")



## clear unnecessary columns
if(length(which(lapply(apply(datenmat[[k]],2,table),length)==1))>0){
  datenmat[[k]]<-datenmat[[k]][,-which(lapply(apply(datenmat[[k]],
  2,table),length)==1)]
}

end<-which(colnames(datenmat[[k]])=="GFR.groups")-1
form<-formula(paste('~', paste(c(colnames(datenmat[[k]]
)[c(2:(which(colnames(datenmat[[k]])=="creatinine")-1),
    (which(colnames(datenmat[[k]])=="creatinine")+1):
    (which(colnames(datenmat[[k]])=="OrdID")-1),
    (which(colnames(datenmat[[k]])=="province9")+1):end)] ,
colnames(dummy.GFR),
colnames(GFR.splines), colnames(lab.splines), colnames(age.splines)),
collapse="+")))


if(is.integer(dim(lab.splines)[2])){
  xmat<-model.matrix(form,cbind(datenmat[[k]],
  dummy.GFR,GFR.splines, lab.splines, age.splines))[,-1]
}else{
  xmat<-model.matrix(form,cbind(datenmat[[k]]
  dummy.GFR,GFR.splines, age.splines))[,-1]
}

##create foldids
```

```
folds<-datenmat[[k]]$PatID-floor(datenmat[[k]]$PatID/20)*20+1
Pats.folds<-unique(cbind(datenmat[[k]]$PatID,folds))
Pats.folds<-cbind(Pats.folds,sample(Pats.folds[,2],
length(Pats.folds[,2])))
colnames(Pats.folds)<-c("PatID", "folds.old", "folds.new")
folds<-merge(datenmat[[k]][,1:2],Pats.folds, by="PatID", all.x=T)[,4]

##model
cvfitL1<-cv.glmnet(y=datenmat[[k]][,"GFR.binary.future"],
                   x=xmat,
                   family="binomial", foldid=folds, alpha=1,
                   parallel=T)
fitL1<-cvfitL1$glmnet.fit
cvfitL1.class<-cv.glmnet(y=datenmat[[k]][,"GFR.binary.future"],
                         x=xmat, type.measure="class",
                         family="binomial", foldid=folds, alpha=1,
                         parallel=T)
cvfitL1.auc<-cv.glmnet(y=datenmat[[k]][,"GFR.binary.future"],
                       x=xmat, type.measure="auc",
                       family="binomial", foldid=folds, alpha=1,
                       parallel=T)
pathmat<-cbind(path(cvfitL1)[,1:2],deviance=
        round(path(cvfitL1)[,3],5),cindex=round(path(cvfitL1.auc)[,3],5),
        misclass=round(path(cvfitL1.class)[,3],5),df=path(cvfitL1)[,4])
lambdas1se<- c(cvfitL1$lambda.1se, cvfitL1.auc$lambda.1se,
        cvfitL1.class$lambda.1se)
pathmat<-cbind(pathmat, cutoff=NA)
pathmat[max(which(pathmat[,"lambda"]>=lambdas1se[1])),"cutoff"]<-
        "deviance"
pathmat[max(which(pathmat[,"lambda"]>=lambdas1se[2])),"cutoff"]<-
       "c-index"
pathmat[max(which(pathmat[,"lambda"]>=lambdas1se[3])),"cutoff"]<-
       "missclass"

print("models created")
```

```
preds.orig<-predict(fitL1, xmat, s=cvfitL1$lambda.1se,
type="response")

coeffs<-cbind(coef(cvfitL1)@Dimnames[[1]][(coef(cvfitL1)@i+1)],
coef(cvfitL1)@x)

AUC<-cindex(preds.orig,datenmat[[k]][,"GFR.binary.future"])

## standardized coefficients
coeffs.sd<-as.vector(lapply(datenmat[[k]], sd))
coeffs.sd<-c(1,coeffs.sd[c(2:(which(colnames(datenmat[[k]])==
        "creatinine")-1),(which(colnames(datenmat[[k]])=="creatinine")
        +1):(which(colnames(datenmat[[k]])=="OrdID")-1),
        (which(colnames(datenmat[[k]])=="province9")+1):end)],
  apply(dummy.GFR,2,sd),
  apply(GFR.splines,2,sd), apply(lab.splines,2,sd),
  apply(age.splines,2,sd))


mat.coeffs<-cbind(as.matrix(round(coef(cvfitL1,s=cvfitL1$lambda.1se)
        ,digits=4)),as.numeric(paste(round(as.numeric(coeffs.sd)*
        coef(cvfitL1,s=cvfitL1$lambda.1se),digits=4), coll="", sep="")))
colnames(mat.coeffs)<-c("coefficients", "standardized coefficients")
mat.coeffs<-mat.coeffs[mat.coeffs[,2]!=0,]
mat.coeffs<-mat.coeffs[order(abs(mat.coeffs[,2]), decreasing=T),]

tf<-which(substring(rownames(mat.coeffs),nchar(rownames(mat.coeffs))
        ,nchar(rownames(mat.coeffs)))=="1")
aux.bin<-substring(rownames(mat.coeffs)[tf],1,
        (nchar(rownames(mat.coeffs)[tf])-1))
binary.var<-aux.bin[substring(aux.bin,
        nchar(aux.bin)-6,nchar(aux.bin))!="splines"]

if(sum(grepl("dummy.GFR",aux.bin))>0){
```

```
  dummy.GFR.sd<-apply(dummy.GFR,2,sd)
  prev<-NULL
  prev[grepl("dummy.GFR",binary.var)]<-round(dummy.GFR.sd[
          binary.var[grepl("dummy.GFR",binary.var)]],4)
  prev[!grepl("dummy.GFR",binary.var)] <-
          round(apply(datenmat[[k]][,binary.var[-which(grepl(
          "dummy.GFR",binary.var)==TRUE)]],2,table)[2,]
          /dim(datenmat[[k]])[1],4)
}else{
  prev<-round(apply(datenmat[[k]][,binary.var],2,table)[2,]/
          dim(datenmat[[k]])[1],4)
}


mat.coeffs<-cbind(mat.coeffs, prevalence=NaN)
mat.coeffs[paste(binary.var,1, sep=""),3]<-prev

number.pat<-dim(unique(datenmat[[k]][1]))[1]
outcome.table<-rbind(number=table(datenmat[[k]][,
        "GFR.binary.future"]),percentage=100*round(prop.table(
        table(datenmat[[k]][,"GFR.binary.future"])), digits=2))

kreuztab.gfr<-table(datenmat[[k]][,"GFR.groups"],
        datenmat[[k]][,"GFR.binary.future"])
kreuztabelle<-cbind(kreuztab.gfr, round(prop.table(kreuztab.gfr,
        margin=1), digits=4),
        total=round(prop.table(table(datenmat[[k]][,"GFR.groups"])),
        digits=4)
)

#R squared
R.quad<-var(preds.orig)/var(datenmat[[k]]$GFR.binary.future)

#cross-validation c-index
Pats.folds<-unique(cbind(datenmat[[k]]$PatID,folds))
```

```
Pats.folds<-cbind(Pats.folds,sample(Pats.folds[,2],
length(Pats.folds[,2])))
colnames(Pats.folds)<-c("PatID", "folds.old", "folds.new")
folds<-merge(datenmat[[k]][,1:2],Pats.folds, by="PatID",
all.x=T)[,4]

preds<-NULL
cvfit.cross<-NULL
for(i in (1:20)){
  folds.new<-folds
  folds.new[folds>i]<-folds[folds>i]-1
  cvfit.cross<-cv.glmnet(y=datenmat[[k]][folds!=i,
          "GFR.binary.future"], x=xmat[folds!=i,], family="binomial",
          foldid=folds.new[folds!=i], alpha=1, parallel=T)
  preds<-c(preds,predict(cvfit.cross$glmnet.fit, xmat[folds==i,],
          s=cvfitL1$lambda.1se, type="response"))
}
AUC.corr<-cindex(preds,datenmat[[k]][order(folds),"GFR.binary.future"])

## add together
Lasso.GFR01[[k]]<-list(number.pat=number.pat,outcome.table=
        outcome.table, crosstable.groups.outcome=kreuztabelle,AUC=AUC,
                AUC.corr=AUC.corr, R.quadrat=R.quad, coefficients=
                mat.coeffs, pathmatrix=pathmat)


par(mfrow=c(1,1))
dev.copy(png,paste(names(datenmat)[k],"_GFR01_folds",20,
        "_part1.png", sep=""),width=500, height=500)
        plot(cvfitL1)
dev.off()

dev.copy(png,paste(names(datenmat)[k],"_GFR01_folds",20,
        "_part2.png", sep=""),width=500, height=500)
       plot(log(fitL1$lambda), fitL1$beta[1,], main="coefficients plot",
```

```
                xlab="log(lambda)", ylim=c(-2.5,2), type="l", col=1,
                ylab="coefficients", xlim=c(-6,-1.5))
        for(j in 2:nrow(fitL1$beta)) lines(log(fitL1$lambda),
                fitL1$beta[j,], col=j)
        abline(v=log(cvfitL1$lambda.1se), lty=3)
        text(-4,1.4, "creatinine.binary", col="pink", cex=0.8)
        text(-3.5,0.7, "creatinine.number", col="grey", cex=0.8)
dev.off()


 dev.copy(png,paste(names(datenmat)[k],"_GFR01_folds",20,
         "_part3.png", sep=""),width=500, height=500)
         ##ROC curve with corrected AUC
         plot.roc(datenmat[[k]][order(folds),"GFR.binary.future"],preds,
                 print.auc=T, main="ROC curve")
 dev.off()

dev.copy(png,paste(names(datenmat)[k],"_GFR01_folds",20,
        "_part4.png", sep=""),width=500, height=500)
         boxplot(preds~datenmat[[k]][order(folds),"GFR.binary.future"],
                 xlab="true outcome", ylab="predicted probability",
                 main="observed vs. predicted values")
 dev.off()

 dev.copy(png,paste(names(datenmat)[k],"_GFR01_folds",20,
         "_part5.png", sep=""),width=500, height=500)
         ## plot GFR splines
         plot(sort(datenmat[[k]]$GFR),GFR.splines[order(
                 datenmat[[k]]$GFR),6], type="l", main="splines of eGFR",
                 col=6)
         for(i in 1:5){
           lines(sort(datenmat[[k]]$GFR),
           GFR.splines[order(datenmat[[k]]$GFR),i], type="l", col=i)
         }
         legend("topleft",paste("eGFR.splines", 1:6, sep=""),
```

```
        col=c(1:6), lty=1)
 dev.off()



set.seed(1234)
number.random<-NULL
for(i in 1:length(unique(datenmat[[k]]$PatID))){
  number.random[i]<-sample(which(datenmat[[k]][order(folds),"PatID"]
          ==unique(datenmat[[k]]$PatID)[i]),1)
}
 dev.copy(png,paste(names(datenmat)[k],"_GFR01_folds",20,
        "_part6.png", sep=""),width=500, height=1000)
        ## calibration plot
        par(mfrow=c(2,1))
        x.aux<-datenmat[[k]][order(folds),"GFR.binary.future"][
                number.random]
        y.aux<-preds[number.random]
        calibrate.plot(x.aux,y.aux, main="calibration plot",xlim=c(0,1))
        hist(preds[number.random], main="", xlab="predicted value",
        prob=T, xlim=c(0,1))
 dev.off()



   ## response plots
 dev.copy(png,paste(names(datenmat)[k],"_GFR01_folds",20,
        "_part7.png", sep=""),width=500, height=1000)
   par(mfrow=c(2,1))
   GFR.coeffs<-coef(cvfitL1,s=cvfitL1$lambda.1se)[grepl(
           "GFR.splines",rownames(coef(cvfitL1,s=cvfitL1$lambda.1se)))]
   spl<-bs(datenmat[[k]]$GFR, knots=
           quantile(datenmat.orig[[k]]$GFR.orig,probs=c(0.1,0.5,0.9),
           na.rm=T))
   spl.median<-predict(spl,median(datenmat.orig[[k]]$GFR.orig,
           na.rm=T))
```

```
  plot(sort(datenmat[[k]][,"GFR"]),
          exp((GFR.splines%*%GFR.coeffs+datenmat[[k]][,"GFR"]*
          coef(cvfitL1,s=cvfitL1$lambda.1se)["GFR",]-
      rep(spl.median%*%GFR.coeffs,dim(GFR.splines)[1]))[order(
      datenmat[[k]][,"GFR"])]-rep(median(datenmat.orig[[k]][,
      "GFR.orig"],na.rm=T)*coef(cvfitL1,s=cvfitL1$lambda.1se
      )["GFR",],dim(GFR.splines)[1])) ,
      type="l", main="response curve for eGFR", xlab="eGFR",
      ylab=paste("odds relative to median (",
      round(median(datenmat.orig[[k]][,"GFR.orig"],na.rm=T),2),
      ")",sep=""), xlim=c(min(datenmat.orig[[k]][,"GFR.orig"],
      na.rm=T),max(datenmat.orig[[k]][,"GFR.orig"] na.rm=T)))
  abline(h=1, col="red")
  hist(datenmat.orig[[k]][,"GFR.orig"], main="",prob=T, xlab="eGFR",
       xlim=c(min(datenmat.orig[[k]][,"GFR.orig"],
       na.rm=T),max(datenmat.orig[[k]][,"GFR.orig"], na.rm=T)))
  dev.off()

dev.copy(png,paste(names(datenmat)[k],"_GFR01_folds",20,
        "_part8.png", sep=""),width=500, height=1000)
        par(mfrow=c(2,1))
        age.coeffs<-coef(cvfitL1,s=cvfitL1$lambda.1se)[grepl(
                "age.splines",rownames(coef(cvfitL1,s=
                cvfitL1$lambda.1se)))]
                spl<-bs(datenmat[[k]]$age,
                knots=quantile(datenmat[[k]]$age,probs=c(0.1,0.5,0.9),
                na.rm=T))
        spl.median<-predict(spl,median(datenmat[[k]]$age,na.rm=T))
        plot(sort(datenmat[[k]][,"age"]),
                exp((age.splines%*%age.coeffs+datenmat[[k]][,"age"]*
                coef(cvfitL1,s=cvfitL1$lambda.1se)["age",]-
                rep(spl.median%*%age.coeffs,dim(age.splines)[1]))[
                order(datenmat[[k]][,"age"])]-rep(
                median(datenmat.orig[[k]][,"age"],na.rm=T)*coef(cvfitL1,
                s=cvfitL1$lambda.1se)["age",])),
```

```
                type="l", main="response curve for age", xlab="age",
                ylab=paste("odds relative to median (",
                round(median(datenmat.orig[[k]][,"age"],na.rm=T),2),")",
                sep=""), xlim=c(20,max(datenmat.orig[[k]][,"age"],
                rm.na=T)))
        abline(h=1, col="red")
        hist(datenmat.orig[[k]][,"age"], main="",prob=T, xlab="age",
                xlim=c(20,max(datenmat.orig[[k]][,"age"],rm.na=T)))
dev.off()


used.lab.splines<-colnames(lab.splines)[colnames(lab.splines)%in%
rownames(Lasso.GFR01[[k]][[7]])]
if(length(used.lab.splines)>0){
  lab.splines.names<-unique(substring(used.lab.splines,1,
          nchar(used.lab.splines)-1))
  lab.values<-unique(substring(used.lab.splines,1,
          nchar(used.lab.splines)-9))
  par(mfrow=c(2,1))
  for(i in 1:length(lab.values)){
    dev.copy(png,paste(names(datenmat)[k],"_GFR01_folds",20,
    "_part",8+i,".png", sep=""),width=500, height=1000)
            lab.coeffs<-coef(cvfitL1,s=cvfitL1$lambda.1se)[grepl(
                    lab.splines.names[i],rownames(coef(cvfitL1,
                    s=cvfitL1$lambda.1se))),]
            spl<-bs(datenmat[[k]][,lab.values[i]],
            knots=quantile(datenmat.orig[[k]][,lab.values[i]]
            ,probs=c(0.1,0.5,0.9), na.rm=T))
    spl.median<-predict(spl,median(
    datenmat.orig[[k]][,lab.values[i]],na.rm=T))
    plot(sort(datenmat[[k]][,lab.values[i]]),
            exp((spl%*%lab.coeffs+datenmat[[k]][,lab.values[i]]*
            coef(cvfitL1,s=cvfitL1$lambda.1se)[lab.values[i],]-
            rep(spl.median%*%lab.coeffs,times=dim(lab.splines)[1]))[
            order(datenmat[[k]][,lab.values[i]])]-
```

```
              rep(median(datenmat.orig[[k]][,paste(lab.values[i],".adapted",
              sep="")],na.rm=T)*coef(cvfitL1,s=cvfitL1$lambda.1se)[
              lab.values[i],],)),
              type="l", main=paste("response curve for", lab.values[i]),
              xlab=lab.values[i], ylab=paste("odds relative to median (",
              round(median(datenmat.orig[[k]][,paste(lab.values[i],
              ".adapted" ,sep="")],na.rm=T),2),")",sep=""),
              xlim=c(min(datenmat.orig[[k]][,paste(lab.values[i],
              ".adapted",sep="")],na.rm=T),
              max(datenmat.orig[[k]][,paste(lab.values[i],
              ".adapted",sep="")], na.rm=T)))
      abline(h=1, col="red")
      hist(datenmat.orig[[k]][,paste(lab.values[i],
              ".adapted",sep="")], main="", xlab=lab.values[i], prob=T,
              xlim=c(min(datenmat.orig[[k]][,paste(lab.values[i],
              ".adapted",sep="")],na.rm=T),
              max(datenmat.orig[[k]][,paste(lab.values[i],
              ".adapted",sep="")], na.rm=T)))
      dev.off()
    }
  }
print(paste("finished",k))
}


### progression

Lasso.decline<-datenmat.decline

for(k in c(1:length(datenmat.decline))){

  set.seed(1234)

  ## create dummy variables for GFR
  dummy.GFR<-matrix(0, ncol=3, nrow=dim(datenmat.decline[[k]])[1])
```

```
colnames(dummy.GFR)<-paste("dummy.GFR",1:3, sep="")
for(i in (1:dim(datenmat.decline[[k]])[[1]])){
  if(datenmat.decline[[k]]$GFR.groups[i]=="3a"){
    dummy.GFR[i,1]<-1
  }
  if(datenmat.decline[[k]]$GFR.groups[i]=="3b"){
    dummy.GFR[i,1]<-1
    dummy.GFR[i,2]<-1
  }
  if(datenmat.decline[[k]]$GFR.groups[i]=="4"){
    dummy.GFR[i,1]<-1
    dummy.GFR[i,2]<-1
    dummy.GFR[i,3]<-1
  }
}
dummy.GFR<-as.data.frame(dummy.GFR)
dummy.GFR[,1]<-as.factor(dummy.GFR[,1])
dummy.GFR[,2]<-as.factor(dummy.GFR[,2])
dummy.GFR[,3]<-as.factor(dummy.GFR[,3])

## create splines
GFR.splines<-as.matrix(bs(datenmat.decline[[k]]$GFR, degree=2,
        knots=quantile(datenmat.decline.orig[[k]]$GFR.orig,probs=
        c(0.3,0.7),na.rm=T)))
colnames(GFR.splines)<-paste("GFR.splines",1:4, sep="")

lab.values<-c("hematocrit", "protein", "calcium", "triglycerides",
        "FBSL","cholesterol", "haemoglobin", "phosphate")
lab.splines<-NULL
name<-NULL
for(i in (1:length(lab.values))){
  if(sum(lab.values[i]==colnames(datenmat.decline[[k]]))==1){
    lab.splines<-cbind(lab.splines,as.matrix(bs(
    datenmat.decline[[k]][,lab.values[i]], degree=2,
    knots=quantile(datenmat.decline.orig[[k]][,paste(lab.values[i],
```

```
    ".adapted", sep="")],probs=c(0.3,0.7), na.rm=T))))
    name<-c(name, paste(lab.values[i],".splines",1:4, sep=""))
  }
}
colnames(lab.splines)<-name

age.splines<-as.matrix(bs(datenmat.decline[[k]]$age, degree=2,
        knots=quantile(datenmat.decline.orig[[k]]$age,
        probs=c(0.3,0.7), na.rm=T)))
colnames(age.splines)<-paste("age.splines",1:4, sep="")

##clear unnecessary columns
if(length(which(lapply(apply(datenmat.decline[[k]],2,table),
length)==1))>0){
  datenmat.decline[[k]]<-datenmat.decline[[k]][,-which(lapply(
  apply(datenmat.decline[[k]],2,table),length)==1)]
}

end<-which(colnames(datenmat.decline[[k]])=="GFR.groups")-1
form<-formula(paste('~', paste(c(colnames(datenmat.decline[[k]]
)[c(2:(which(colnames(datenmat.decline[[k]])=="creatinine")-1),
    (which(colnames(datenmat.decline[[k]])=="creatinine")+1):
    (which(colnames(datenmat.decline[[k]])=="OrdID")-1),
    (which(colnames(datenmat.decline[[k]])=="province9")+1):end)] ,
colnames(dummy.GFR),
colnames(GFR.splines), colnames(lab.splines),
colnames(age.splines)), collapse="+")))


if(is.integer(dim(lab.splines)[2])){
  xmat<-model.matrix(form,cbind(datenmat.decline[[k]], dummy.GFR,
          GFR.splines, lab.splines, age.splines))[,-1]
}else{
  xmat<-model.matrix(form,cbind(datenmat.decline[[k]],
          dummy.GFR,GFR.splines, age.splines))[,-1]
```

```
}

##create foldids
folds<-datenmat.decline[[k]]$PatID-floor(
        datenmat.decline[[k]]$PatID/20)*20+1
Pats.folds<-unique(cbind(datenmat.decline[[k]]$PatID,folds))
Pats.folds<-cbind(Pats.folds,sample(Pats.folds[,2],
        length(Pats.folds[,2])))
colnames(Pats.folds)<-c("PatID", "folds.old", "folds.new")
folds<-merge(datenmat.decline[[k]][,1:2],Pats.folds, by="PatID",
        all.x=T)[,4]

## model
cvfitL1<-cv.glmnet(y=datenmat.decline[[k]][,"decline"],
                   x=xmat,
                   family="binomial", foldid=folds, alpha=1,
                   parallel=T)
fitL1<-cvfitL1$glmnet.fit
cvfitL1.class<-cv.glmnet(y=datenmat.decline[[k]][,"decline"],
                         x=xmat, type.measure="class",
                         family="binomial", foldid=folds, alpha=1,
                         parallel=T)
cvfitL1.auc<-cv.glmnet(y=datenmat.decline[[k]][,"decline"],
                       x=xmat, type.measure="auc",
                       family="binomial", foldid=folds, alpha=1,
                       parallel=T)
pathmat<-cbind(path(cvfitL1)[,1:2],deviance=round(path(cvfitL1)[,3],
                5),cindex=round(path(cvfitL1.auc)[,3],5),
                misclass=round(path(cvfitL1.class)[,3],5),
                df=path(cvfitL1)[,4])
lambdas1se<- c(cvfitL1$lambda.1se, cvfitL1.auc$lambda.1se,
        cvfitL1.class$lambda.1se)
pathmat<-cbind(pathmat, cutoff=NA)
pathmat[max(which(pathmat[,"lambda"]>=lambdas1se[1])),"cutoff"]<-
"deviance"
```

```
pathmat[max(which(pathmat[,"lambda"]>=lambdas1se[2])),"cutoff"]<-
"c-index"
pathmat[max(which(pathmat[,"lambda"]>=lambdas1se[3])),"cutoff"]<-
"missclass"

preds.orig<-predict(fitL1, xmat, s=cvfitL1$lambda.1se, type="response")

AUC<-cindex(preds.orig,datenmat.decline[[k]][,"decline"])

## standardized coefficients
coeffs.sd<-as.vector(lapply(datenmat.decline[[k]], sd))
coeffs.sd<-c(1,coeffs.sd[c(2:(which(colnames(
        datenmat.decline[[k]])=="creatinine")-1),
        (which(colnames(datenmat.decline[[k]])=="creatinine")+1):
        (which(colnames(datenmat.decline[[k]])=="OrdID")-1),
        (which(colnames(datenmat.decline[[k]])=="province9")+1):end)],
        apply(dummy.GFR,2,sd),apply(GFR.splines,2,sd),
        apply(lab.splines,2,sd), apply(age.splines,2,sd))

mat.coeffs<-cbind(as.matrix(round(coef(cvfitL1,s=cvfitL1$lambda.1se)
        ,digits=4)), as.numeric(paste(round(as.numeric(coeffs.sd)*
        coef(cvfitL1,s=cvfitL1$lambda.1se),digits=4), coll="", sep="")))
colnames(mat.coeffs)<-c("coefficients", "standardized coefficients")
mat.coeffs<-mat.coeffs[mat.coeffs[,2]!=0,]
mat.coeffs<-mat.coeffs[order(abs(mat.coeffs[,2]), decreasing=T),]

tf<-which(substring(rownames(mat.coeffs),nchar(rownames(mat.coeffs))
        ,nchar(rownames(mat.coeffs)))=="1")
aux.bin<-substring(rownames(mat.coeffs)[tf],1,
        (nchar(rownames(mat.coeffs)[tf])-1))
binary.var<-aux.bin[substring(aux.bin,
        nchar(aux.bin)-6,nchar(aux.bin))!="splines"]

if(sum(grepl("dummy.GFR",aux.bin))>0){
  dummy.GFR.sd<-apply(dummy.GFR,2,sd)
```

```
  prev<-NULL
  prev[grepl("dummy.GFR",binary.var)]<-round(dummy.GFR.sd[
          binary.var[grepl("dummy.GFR",binary.var)]],4)
  prev[!grepl("dummy.GFR",binary.var)] <-
          round(apply(datenmat.decline[[k]][,binary.var[
          -which(grepl("dummy.GFR",binary.var)==TRUE)]],2,table)[2,]
          /dim(datenmat.decline[[k]])[1],4)
}else{
  prev<-round(apply(datenmat.decline[[k]][,binary.var],2,table)[2,]
          /dim(datenmat.decline[[k]])[1],4)
}

mat.coeffs<-cbind(mat.coeffs,prevalence=NaN)
mat.coeffs[paste(binary.var,1, sep=""),3]<-prev


number.pat<-dim(unique(datenmat.decline[[k]][1]))[1]
outcome.table<-rbind(number=table(datenmat.decline[[k]][
        ,"decline"]) ,percentage=100*round(prop.table(table(
        datenmat.decline[[k]][,"decline"])), digits=2))

kreuztab.gfr<-table(datenmat.decline[[k]][,"GFR.groups"],
        datenmat.decline[[k]][,"decline"])
kreuztabelle<-cbind(kreuztab.gfr, round(prop.table(kreuztab.gfr,
        margin=1), digits=4), total=round(prop.table(
        table(datenmat.decline[[k]][,"GFR.groups"])), digits=4))


#R squared
R.quad<-var(preds.orig)/var(datenmat.decline[[k]]$decline)

#cross-validation c-index
Pats.folds<-unique(cbind(datenmat.decline[[k]]$PatID,folds))
Pats.folds<-cbind(Pats.folds,sample(Pats.folds[,2],
        length(Pats.folds[,2])))
```

```
colnames(Pats.folds)<-c("PatID", "folds.old", "folds.new")
folds<-merge(datenmat.decline[[k]][,1:2],Pats.folds, by="PatID",
        all.x=T)[,4]

preds<-NULL
cvfit.cross<-NULL
for(i in (1:20)){
  folds.new<-folds
  folds.new[folds>i]<-folds[folds>i]-1
  cvfit.cross<-cv.glmnet(y=datenmat.decline[[k]][folds!=i,"decline"]
          , x=xmat[folds!=i,], family="binomial",
          foldid=folds.new[folds!=i], alpha=1, parallel=T)
  preds<-c(preds,predict(cvfit.cross$glmnet.fit, xmat[folds==i,],
          s=cvfitL1$lambda.1se, type="response"))
}
AUC.corr<-cindex(preds,datenmat.decline[[k]][
        order(folds),"decline"])

## add together
Lasso.decline[[k]]<-list(number.pat=number.pat,outcome.table=
        outcome.table, crosstable.groups.outcome=kreuztabelle,AUC=AUC,
        AUC.corr=AUC.corr,R.quadrat=R.quad,coefficients=mat.coeffs,
        pathmatrix=pathmat)



par(mfrow=c(1,1))
dev.copy(png,paste(names(datenmat.decline)[k],"_decline_folds",
20,"_part1.png", sep=""),width=500, height=500)
        plot(cvfitL1)
dev.off()

dev.copy(png,paste(names(datenmat.decline)[k],"_decline_folds",
20,"_part2.png", sep=""),width=500, height=500)
        plot(log(fitL1$lambda), fitL1$beta[1,], main="coefficients
                plot",xlab="log(lambda)", ylim=c(-7, 5), type="l", col=1,
```

```
                ylab="coefficients",xlim=c(-7,-3))
        for(j in 2:nrow(fitL1$beta)) lines(log(fitL1$lambda),
                fitL1$beta[j,], col=j)
        abline(v=log(cvfitL1$lambda.1se), lty=3)
        text(-3.9,2.2, "GFR.splines1", col="red", cex=0.8)
        text(-3.8,-2, "dummy.GFR2", col="blue", cex=0.8)
dev.off()

dev.copy(png,paste(names(datenmat.decline)[k],"_decline_folds",
20,"_part3.png", sep=""),width=500, height=500)
##ROC curve with corrected AUC
        plot.roc(datenmat.decline[[k]][order(folds),"decline"],preds,
                print.auc=T, main="ROC curve")
dev.off()

dev.copy(png,paste(names(datenmat.decline)[k],"_decline_folds",
20,"_part4.png", sep=""),width=500, height=500)
        boxplot(preds~datenmat.decline[[k]][order(folds),"decline"],
        xlab="true outcome", ylab="predicted probability"
  ,main="observed vs. predicted values")
dev.off()

dev.copy(png,paste(names(datenmat.decline)[k],"_decline_folds",
20,"_part5.png", sep=""),width=500, height=500)
## plot GFR splines
        plot(sort(datenmat.decline[[k]]$GFR),
                GFR.splines[order(datenmat.decline[[k]]$GFR),4], type="l",
                main="splines of eGFR", col=6)
        for(i in 1:3){
          lines(sort(datenmat.decline[[k]]$GFR),
          GFR.splines[order(datenmat.decline[[k]]$GFR),i], type="l",
          col=i)
        }
        legend("topleft",paste("eGFR.splines", 1:6, sep=""), col=c(1:6),
        lty=1)
```

```
dev.off()


set.seed(1234)
number.random<-NULL
for(i in 1:length(unique(datenmat.decline[[k]]$PatID))){
  number.random[i]<-sample(which(datenmat.decline[[k]][
  order(folds),"PatID"]==unique(datenmat.decline[[k]]$PatID)[i]),1)
}
dev.copy(png,paste(names(datenmat.decline)[k],"_decline_folds",
20,"_part6.png", sep=""),width=500, height=1000)
## calibration plot
        par(mfrow=c(2,1))
        calibrate.plot(datenmat.decline[[k]][order(folds),"decline"][
        number.random],preds[number.random], main="calibration plot",
        xlim=c(0,1))
        hist(preds[number.random], main="", xlab="predicted value",
        xlim=c(0,1))
dev.off()


##response plots
dev.copy(png,paste(names(datenmat.decline)[k],"_decline_folds",
20,"_part7.png", sep=""),width=500, height=1000)
        par(mfrow=c(2,1))
        GFR.coeffs<-coef(cvfitL1,s=cvfitL1$lambda.1se)[grepl(
                "GFR.splines",rownames(coef(cvfitL1,s=
                cvfitL1$lambda.1se)))]
        spl<-bs(datenmat.decline[[k]]$GFR,degree=2,
                knots=quantile(datenmat.decline.orig[[k]]$GFR.orig,
                probs=c(0.3,0.7), na.rm=T))
        spl.median<-predict(spl,median(
                datenmat.decline.orig[[k]]$GFR.orig,na.rm=T))
        plot(sort(datenmat.decline[[k]][,"GFR"]),
```

```
                exp((GFR.splines%*%GFR.coeffs+datenmat.decline[[k]][
                ,"GFR"]*coef(cvfitL1,s=cvfitL1$lambda.1se)["GFR",]-
      rep(spl.median%*%GFR.coeffs,dim(GFR.splines)[1]))[
      order(datenmat.decline[[k]][,"GFR"])]-
      rep(coef(cvfitL1,s=cvfitL1$lambda.1se)["GFR",]*
      median(datenmat.decline.orig[[k]]$GFR.orig,na.rm=T),
      times=dim(GFR.splines)[1])+(matrix(as.numeric(as.matrix(
              dummy.GFR)),ncol=3)%*%GFR.dummy.coeffs)[order(
              datenmat.decline[[k]][,"GFR"])]),
      type="l", main="response curve for eGFR", xlab="GFR",
      ylab=paste("odds relative to median (",
      round(median(datenmat.decline.orig[[k]]$GFR.orig,na.rm=T),
      2),")",sep=""),
             xlim=c(min(datenmat.decline.orig[[k]][,"GFR.orig"],
             rm.na=T),max(datenmat.decline.orig[[k]][,"GFR.orig"],
             rm.na=T)))
        abline(h=1, col="red")
        hist(datenmat.decline.orig[[k]][,"GFR.orig"], main="",
                xlab="GFR", xlim=c(min(
                datenmat.decline.orig[[k]][,"GFR.orig"],
                rm.na=T),max(datenmat.decline.orig[[k]][,"GFR.orig"],
                rm.na=T)), prob=T)
dev.off()

dev.copy(png,paste(names(datenmat.decline)[k],"_decline_folds",
20,"_part8.png", sep=""),width=500, height=1000)
        par(mfrow=c(2,1))
        age.coeffs<-coef(cvfitL1,s=cvfitL1$lambda.1se)[grepl(
                "age.splines",rownames(coef(cvfitL1,s=cvfitL1$lambda.1se
                )))]
        spl<-bs(datenmat.decline[[k]]$age,
                degree=2,knots=quantile(datenmat.decline[[k]]$age,probs=
                c(0.3,0.7), na.rm=T))
        spl.median<-predict(spl,median(datenmat.decline[[k]]$age,
                na.rm=T))
```

```
        plot(sort(datenmat.decline[[k]][,"age"]),
                exp((age.splines%*%age.coeffs+datenmat.decline[[k]][
                ,"age"]*coef(cvfitL1,s=cvfitL1$lambda.1se)["age",]-
          rep(spl.median%*%age.coeffs,dim(age.splines)[1]))[order(
          datenmat.decline[[k]][,"age"])]-
          rep(coef(cvfitL1,s=cvfitL1$lambda.1se)["age",]*
          median(datenmat.decline.orig[[k]][,"age"],na.rm=T),
          times=dim(age.splines)[1])),
          type="l", main="response curve for age", xlab="age",
          ylab=paste("odds relative to median (",
          round(median(datenmat.decline.orig[[k]][,"age"],na.rm=T),2),
          ")",sep=""),xlim=c(20,max(datenmat.decline.orig[[k]][,"age"],
          rm.na=T)))
        abline(h=1, col="red")
        hist(datenmat.decline.orig[[k]][,"age"], main="", xlab="age",
                prob=T, xlim=c(20,max(datenmat.decline.orig[[k]][,"age"],
                rm.na=T)))
dev.off()

used.lab.splines<-colnames(lab.splines)[colnames(lab.splines)%in%
        rownames(Lasso.decline[[k]][[7]])]
if(length(used.lab.splines)>0){
  lab.splines.names<-unique(substring(used.lab.splines,1,
          nchar(used.lab.splines)-1))
  lab.values<-unique(substring(used.lab.splines,1,
          nchar(used.lab.splines)-9))
  for(i in 1:length(lab.values)){
    dev.copy(png,paste(names(datenmat.decline)[k],"_decline_folds",
    20,"_part", 8+i,".png", sep=""),width=500, height=1000)
            par(mfrow=c(2,1))
            lab.coeffs<-coef(cvfitL1,s=cvfitL1$lambda.1se)[grepl(
                    lab.splines.names[i],rownames(coef(cvfitL1,s=
                    cvfitL1$lambda.1se))),]
            spl<-bs(datenmat.decline[[k]][,lab.values[i]], degree=2,
            knots=quantile(datenmat.decline.orig[[k]][
```

```
            ,lab.values[i]],probs=c(0.3,0.7), na.rm=T))
      spl.median<-predict(spl,median(datenmat.decline.orig[[k]][
            ,lab.values[i]],na.rm=T))
      plot(sort(datenmat.decline[[k]][,lab.values[i]]),
            exp((spl%*%lab.coeffs+
        datenmat.decline[[k]][,lab.values[i]]*coef(cvfitL1,s=
        cvfitL1$lambda.1se)[lab.values[i],]
        -rep(spl.median%*%lab.coeffs,times=dim(lab.splines)[1]))[
        order(datenmat.decline[[k]][,lab.values[i]])]
        -rep(coef(cvfitL1,s=cvfitL1$lambda.1se)[lab.values[i],]*
        median(datenmat.decline.orig[[k]][,paste(lab.values[i],
        ".adapted",sep="")],na.rm=T), times=dim(lab.splines)[1])),
        type="l", main=paste("response curve for", lab.values[i]),
        xlab=lab.values[i], ylab=paste("odds relative to median (",
        round(median(datenmat.decline.orig[[k]][,paste(lab.values[i],
        ".adapted",sep="")],na.rm=T),2),")",sep=""),xlim=c(min(
        datenmat.decline.orig[[k]][,paste(lab.values[i],".adapted",
        sep="")],na.rm=T),
        max(datenmat.decline.orig[[k]][,paste(lab.values[i],
        ".adapted",sep="")], na.rm=T)))
      abline(h=1,col="red")
      hist(datenmat.decline.orig[[k]][,paste(lab.values[i],
            ".adapted",sep="")],main="", xlab=lab.values[i],prob=T,
         xlim=c(min(datenmat.decline.orig[[k]][,paste(lab.values[i],
         ".adapted",sep="")], na.rm=T),
         max(datenmat.decline.orig[[k]][,paste(lab.values[i],
         ".adapted",sep="")], na.rm=T)))
      dev.off()
      }
    }
  print(paste("finished",k))
}
```

Model comparison

```
### model comparison  care

# number of patients
for(i in (c(1,5:7))){
  print(Lasso.GFR01[[i]][[1]])
}
#number of records
for(i in (c(1,5:7))){
  print(dim(datenmat[[i]])[1]/Lasso.GFR01[[i]][[1]])
}

#percentage of outcome
for(i in (c(1,5:7))){
  print(Lasso.GFR01[[i]][[2]])
}
#number of variables in the model matrix
for(i in (c(1,5:7))){
  print(dim(Lasso.GFR01[[i]][[8]])-1)
}
#number of variables in the model
for(i in (c(1,5:7))){
  print(dim(Lasso.GFR01[[i]][[7]])-1)
}
#most powerful variables
for(i in (c(1,5:7))){
  print(rownames(head(Lasso.GFR01[[i]][[7]])))
}
#R^2
for(i in (c(1,5:7))){
  print(Lasso.GFR01[[i]][[6]])
}
#c-index
for(i in (c(1,5:7))){
```

```
  print(Lasso.GFR01[[i]][[4]])
}
#optimism corrected c-index
for(i in (c(1,5:7))){
  print(Lasso.GFR01[[i]][[5]])
}

## comparison window parameters
par(mfrow=c(1,1))
dev.copy(png,"comparison_window_care.png",width=600, height=500)
AUC<-NULL
taken.var<-NULL
numb.var<-NULL
for(i in c(1,5:7)){
  AUC<-cbind(AUC,Lasso.GFR01[[i]][[5]])
  taken.var<-cbind(taken.var, dim(Lasso.GFR01[[i]][[7]])[1]-1)
  numb.var<-cbind(numb.var,dim(Lasso.GFR01[[i]][[8]])[1]-1)
}
par(mar=c(5, 8, 4, 4) + 0.1)
plot(1:4,AUC,type = "l", axes=F, ylab="",xlab="window parameters",
     main="comparison of different window parameters", ylim=c(0.75,0.9))
points(1:4,AUC)
axis(2,c(0.75,0.8,0.85,0.90,AUC),c(0.75,0.8,0.85,0.90,
        round(AUC,digits=3)), line=0)
par(new = TRUE)
plot(1:4, numb.var, yaxs = "i", axes=F,type = "l", ann = FALSE,
        col = 2, ylim=c(60,950))
points(1:4, numb.var,col=2)
axis(2, c(60,950, numb.var),c(60,950,numb.var), line=3, col=2)
par(new = TRUE)
plot(1:4, taken.var,yaxs="i", axes=F, type="l", col=3,
        ann = FALSE, ylim=c(5,65))
points(1:4, taken.var,col=3)
axis(2, c(5,65,taken.var), c(5,65,taken.var), line=6, col=3)
axis(1, 1:4,c("365-180-730","365-365-730",
```

```
        "730-365-1095", "1095-365-1460"))
legend("topleft", legend=c("optimism corr c-index","vars in
        matrix","vars in model"),col=1:3,lty=1, cex=0.8)
dev.off()


##fixed window parameters
#number of variables in the model matrix
for(i in (c(2,3,5,4))){
  print(dim(Lasso.GFR01[[i]][[8]])-1)
}
#number of variables in the model
  for(i in (c(2,3,5,4))){
    print(dim(Lasso.GFR01[[i]][[7]])-1)
  }
#most powerful variables
for(i in (c(2,3,5,4))){
  print(rownames(head(Lasso.GFR01[[i]][[7]])))
}
#R^2
for(i in (c(2,3,5,4))){
  print(Lasso.GFR01[[i]][[6]])
}
#c-index
for(i in (c(2,3,5,4))){
  print(Lasso.GFR01[[i]][[4]])
}
#optimism corrected c-index
for(i in (c(2,3,5,4))){
  print(Lasso.GFR01[[i]][[5]])
}

setwd("C:\\Users\\Christine\\Documents\\Studium\\Magarbeit\\Arbeit")
## comparison prevalences
par(mfrow=c(1,1))
```

```
dev.copy(png,"comparison_prevalences_care.png",width=600, height=500)
AUC<-NULL
taken.var<-NULL
numb.var<-NULL
for(i in (2:5)){
  AUC[i-1]<-Lasso.GFR01[[i]][[5]]
  taken.var[i-1]<-dim(Lasso.GFR01[[i]][[7]])[1]-1
  numb.var[i-1]<-dim(Lasso.GFR01[[i]][[8]])[1]-1
}
par(mar=c(5, 8, 4, 4) + 0.1)
plot(1:4,AUC[c(1:2,4:3)],type = "l", axes=F, ylab="",xlab="prevalences",
     main="comparison of different prevalences", ylim=c(0.75,0.9))
points(1:4,AUC[c(1:2,4:3)])
axis(2,c(0.75,0.8,0.85,0.9,AUC[c(1:2,4:3)]),c(0.75,0.8,0.85,0.9,
        round(AUC[c(1:2,4:3)],digits=3)), line=0)
par(new = TRUE)
plot(1:4, numb.var[c(1:2,4:3)], yaxs = "i", axes=F,type = "l",
        ann = FALSE, col = 2, ylim=c(60,950))
points(1:4, numb.var[c(1:2,4:3)],col=2)
axis(2,c(60,950,numb.var[c(1:2,4:3)]),c(60,950,
        numb.var[c(1:2,4:3)]), line=3, col=2)
par(new = TRUE)
plot(1:4, taken.var[c(1:2,4:3)],yaxs="i", axes=F, type="l", col=3,
        ann = FALSE,ylim=c(5,65))
points(1:4, taken.var[c(1:2,4:3)],col=3)
axis(2,c(5,65,taken.var[c(1:2,4:3)]),c(5,65, taken.var[c(1:2,4:3)]),
        line=6, col=3)
axis(1, 1:4,c("0.1,0.2","0.05,0.1", "0.01,0.05","0.01,0.01"))
legend("topleft", legend=c("optimism corr c-index","vars in
        matrix","vars in model"),col=1:3,lty=1, cex=0.8)
dev.off()



###### models progression
```

```
# number of patients
for(i in (c(1,5:7))){
  print(Lasso.decline[[i]][[1]])
}
#number of records
for(i in (c(1,5:7))){
  print(dim(datenmat.decline[[i]])[1]/Lasso.decline[[i]][[1]])
}

#percentage of outcome
for(i in (c(1,5:7))){
  print(Lasso.decline[[i]][[2]])
}

#number of variables in the model matrix
for(i in (c(1,5:7))){
  print(dim(Lasso.decline[[i]][[8]])-1)
}

#number of variables in the model
for(i in (c(1,5:7))){
  print(dim(Lasso.decline[[i]][[7]])-1)
}

#most powerful variables
for(i in (c(1,5:7))){
  print(rownames(head(Lasso.decline[[i]][[7]])))
}
#R^2
for(i in (c(1,5:7))){
  print(Lasso.decline[[i]][[6]])
}
#c-index
for(i in (c(1,5:7))){
```

```
  print(Lasso.decline[[i]][[4]])
}
  #optimism corrected c-index
for(i in (c(1,5:7))){
  print(Lasso.decline[[i]][[5]])
}


## comparison window parameters
par(mfrow=c(1,1))
dev.copy(png,"comparison_window_progression.png",width=600, height=500)
AUC<-NULL
taken.var<-NULL
numb.var<-NULL
for(i in c(1,5:7)){
  AUC<-cbind(AUC,Lasso.decline[[i]][[5]])
  taken.var<-cbind(taken.var, dim(Lasso.decline[[i]][[7]])[1]-1)
  numb.var<-cbind(numb.var,dim(Lasso.decline[[i]][[8]])[1]-1)
}
par(mar=c(5, 8, 4, 4) + 0.1)
plot(1:4,AUC,type = "l", axes=F, ylab="",xlab="window parameters",
     main="comparison of different window parameters", ylim=c(0.8,0.85))
points(1:4,AUC)
axis(2,c(0.8,0.85,AUC),c(0.8,0.85,round(AUC,digits=3)), line=0)
par(new = TRUE)
plot(1:4, numb.var, yaxs = "i", axes=F,type = "l", ann = FALSE,
        col = 2, ylim=c(100,1700))
points(1:4, numb.var,col=2)
axis(2,c(100,numb.var, 1700), c(100, numb.var,1700), line=3, col=2)
par(new = TRUE)
plot(1:4, taken.var,yaxs="i", axes=F, type="l", col=3, ann = FALSE,
        ylim=c(10,35))
points(1:4, taken.var,col=3)
axis(2,c(10,taken.var,35), c(10,taken.var,35), line=6, col=3)
axis(1, 1:4,c("365-180-730","365-365-730",
```

```
        "730-365-1095", "1095-365-1460"))
legend("topleft", legend=c("optimism corr c-index","vars in
        matrix","vars in model"),col=1:3,lty=1, cex=0.8)
dev.off()


##fixed window parameters
#number of variables in the model matrix
for(i in (c(2,3,5,4))){
  print(dim(Lasso.decline[[i]][[8]])-1)
}
#number of variables in the model
for(i in (c(2,3,5,4))){
  print(dim(Lasso.decline[[i]][[7]])-1)
}
#most powerful variables
for(i in (c(2,3,5,4))){
  print(rownames(head(Lasso.decline[[i]][[7]])))
}
#R^2
for(i in (c(2,3,5,4))){
  print(Lasso.decline[[i]][[6]])
}
#c-index
for(i in (c(2,3,5,4))){
  print(Lasso.decline[[i]][[4]])
}
#optimism corrected c-index
for(i in (c(2,3,5,4))){
  print(Lasso.decline[[i]][[5]])
}

### comparison prevalences
par(mfrow=c(1,1))
dev.copy(png,"comparison_prevalences_progression.png",width=600,
```

```
        height=500)
AUC<-NULL
taken.var<-NULL
for(i in (2:5)){
  AUC[i-1]<-Lasso.decline[[i]][[5]]
  taken.var[i-1]<-dim(Lasso.decline[[i]][[7]])[1]-1
}
numb.var<-(as.numeric(lapply(datenmat.decline, length))-9+50)[2:5]
par(mar=c(5, 8, 4, 4) + 0.1)
plot(1:4,AUC[c(1:2,4:3)],type = "l", axes=F, ylab="",xlab="prevalences",
     main="comparison of different prevalences", ylim=c(0.8,0.85))
points(1:4,AUC[c(1:2,4:3)])
axis(2,c(0.8,0.85,AUC[c(1:2,4:3)]),c(0.8,0.85,round(AUC[c(1:2,4:3)],
        digits=3)), line=0)
par(new = TRUE)
plot(1:4, numb.var[c(1:2,4:3)], yaxs = "i", axes=F,type = "l",
        ann = FALSE, col = 2, ylim=c(100,1700))
points(1:4, numb.var[c(1:2,4:3)],col=2)
axis(2,c(100, numb.var[c(1:2,4:3)],1700),
        c(100,numb.var[c(1:2,4:3)],1700), line=3, col=2)
par(new = TRUE)
plot(1:4, taken.var[c(1:2,4:3)],yaxs="i", axes=F, type="l", col=3,
        ann = FALSE, ylim=c(10,35))
points(1:4, taken.var[c(1:2,4:3)],col=3)
axis(2,c(10,35,taken.var[c(1:2,4:3)]),
        c(10,35,taken.var[c(1:2,4:3)]), line=6, col=3)
axis(1, 1:4,c("0.1,0.2","0.05,0.1",  "0.01,0.05","0.01,0.01"))
legend("topleft", legend=c("optimism corr c-index","vars in
        matrix","vars in model"),col=1:3,lty=1, cex=0.8)
dev.off()
```

Descriptive statistics

```
### care

number.random<-NULL
set.seed(1234)
for(i in 1:length(unique(daten.shift365.365.730[[1]]$PatID))){
  number.random[i]<-sample(which(daten.shift365.365.730[[1]][
          ,"PatID"]==unique(daten.shift365.365.730[[1]]$PatID)[i]),1)
}

length(number.random)
colnames(daten.shift365.365.730[[1]])

age<-summary(daten.shift365.365.730[[1]]$age[number.random])
sd(daten.shift365.365.730[[1]]$age[number.random])
age

sex<-rbind(table(daten.shift365.365.730[[1]]$sex[number.random]),
        round(100*prop.table(table(daten.shift365.365.730[[1]]$sex[
                number.random])),2))
sex
atc<-rbind(sort(apply(daten.shift365.365.730[[1]][number.random,2:41],
        2, function(x)sum(as.numeric(x))), decreasing=T),
        round(100*prop.table(sort(apply(
        daten.shift365.365.730[[1]][number.random,2:41],2,
        function(x)sum(as.numeric(x))), decreasing=T)),2))
atc[2,1:10]

labor.binary<-apply(daten.shift365.365.730[[1]][number.random,73:80]
        ,2, function(x) round(100*prop.table(table(x)),2))
        labor.binary[2,]
colnames(daten.shift365.365.730[[3]][1:200])
labor<-apply(daten.shift365.365.730[[3]][number.random,146:154],2,
```

```
        summary)
labor[c(1,4,6),]
apply(daten.shift365.365.730[[3]][number.random,146:154],2,
        function(x) sd(x,na.rm=T))

dev.copy(png,"labvalues_care.png",width=800, height=600)
par(mfrow=c(2,4))
choice<-c(146:152,154)
name<-c("Calcium", "Cholesterol", "Protein",
"Hematocrit", "Haemoglobin", "Creatinine", "FBSL",
"Triglycerides")
for(i in 1:8){
  boxplot(daten.shift365.365.730[[3]][number.random,choice[i]],
  main=name[i])
}
dev.off()

which(daten.shift365.365.730[[3]]$protein==100)

krea.number<-summary(daten.shift365.365.730[[1]]$creatinine.number[
        number.random])
krea.number
sd(daten.shift365.365.730[[1]]$creatinine.number[number.random])
which(daten.shift365.365.730[[1]]$creatinine.number[number.random]==
        12)

colnames(daten.shift365.365.730[[1]][1:200])
diags<-apply(daten.shift365.365.730[[1]][number.random,53:72],2,
        function(x) round(100*prop.table(table(x)),2))
diags[2,]

number.random.care<-number.random

### progression
```

```
number.random<-NULL
set.seed(1234)
for(i in 1:length(unique(daten.shift365.365.730[[2]]$PatID))){
  number.random[i]<-sample(which(daten.shift365.365.730[[2]][
  ,"PatID"]==unique(daten.shift365.365.730[[2]]$PatID)[i]),1)
}

length(number.random)
colnames(daten.shift365.365.730[[2]])

age<-summary(daten.shift365.365.730[[2]]$age[number.random])
sd(daten.shift365.365.730[[2]]$age[number.random])
age

sex<-rbind(table(daten.shift365.365.730[[2]]$sex[number.random]),
        round(100*prop.table(table(daten.shift365.365.730[[2]]$sex[
        number.random])),2))
sex
atc<-rbind(sort(apply(daten.shift365.365.730[[2]]
        [number.random,2:48],2, function(x)sum(as.numeric(x))),
        decreasing=T),
    round(100*prop.table(sort(apply(daten.shift365.365.730[[2]][
    number.random,2:48],2, function(x)sum(as.numeric(x))),
    decreasing=T)),2))
atc[,1:10]

labor.binary<-apply(daten.shift365.365.730[[2]][
        number.random,81:89],2, function(x)
        round(100*prop.table(table(x)),2))
labor.binary
labor<-apply(daten.shift365.365.730[[4]][number.random,146:154],2,
        summary)
labor
colnames(daten.shift365.365.730[[4]])[1:160]
round(apply(daten.shift365.365.730[[4]][number.random,146:154],2,
```

```
        function(x) sd(x,na.rm=T)),2)


dev.copy(png,"labvalues_progression.png",width=800, height=800)
par(mfrow=c(3,3))
choice<-c(146:154)
name<-c("Calcium", "Cholesterol", "Protein",
"Hematocrit", "Haemoglobin", "Creatinine", "FBSL",
"Phosphate", "Triglycerides")
for(i in 1:9){
  boxplot(daten.shift365.365.730[[4]][number.random,choice[i]],
  main=name[i])
}
dev.off()

krea.number<-summary(daten.shift365.365.730[[2]]$creatinine.number[
        number.random])
krea.number
sd(daten.shift365.365.730[[2]]$creatinine.number[number.random])
which(daten.shift365.365.730[[2]]$creatinine.number[number.random]
        ==24)

diag<-apply(daten.shift365.365.730[[2]][number.random,61:80],2,
        function(x) round(100*prop.table(table(x)),2))
diag[2,]

age
atc
labor.binary
DM
```

Proportion of explained variation

```
library(glmnet)
library(doParallel)
```

```
registerDoParallel(4)



### care

set.seed(1234)

# laboratory values
xmat.no.lab<-xmat[,-c(81:88,grep("binary", colnames(xmat)))]
xmat.lab<-xmat[,c(73:88,187:188,195,220:269)]

cvfitL1.no.lab<-cv.glmnet(y=datenmat[[k]][,"GFR.binary.future"],
                    x=xmat.no.lab,
                    family="binomial", foldid=folds, alpha=1,
                    parallel=T)
fitL1.no.lab<-cvfitL1.no.lab$glmnet.fit

preds.no.lab<-predict(fitL1.no.lab, xmat.no.lab,
        s=cvfitL1.no.lab$lambda.1se, type="response")
R.no.lab<-var(preds.no.lab)/var(datenmat[[k]]$GFR.binary.future)

R.quad-R.no.lab

cvfitL1.lab<-cv.glmnet(y=datenmat[[k]][,"GFR.binary.future"],
                           x=xmat.lab,
                           family="binomial", foldid=folds, alpha=1,
                           parallel=T)
fitL1.lab<-cvfitL1.lab$glmnet.fit

preds.lab<-predict(fitL1.lab, xmat.lab, s=cvfitL1.lab$lambda.1se,
        type="response")
R.lab<-var(preds.lab)/var(datenmat[[k]]$GFR.binary.future)

R.lab
```

```
# diagnoses

xmat.no.diag<-xmat[,-c(53:72,89:90,96,98,99,101,108:111,
        119:125,126:128,131,140:144,146,155:159,161,167:168,
        173:177,181:182,185:186,189:194,196:217)]
xmat.diag<-xmat[,c(53:72,89:90,203:217)]

cvfitL1.no.diag<-cv.glmnet(y=datenmat[[k]][,"GFR.binary.future"],
                           x=xmat.no.diag,
                           family="binomial", foldid=folds, alpha=1,
                           parallel=T)
fitL1.no.diag<-cvfitL1.no.diag$glmnet.fit

preds.no.diag<-predict(fitL1.no.diag, xmat.no.diag,
        s=cvfitL1.no.diag$lambda.1se, type="response")
R.no.diag<-var(preds.no.diag)/var(datenmat[[k]]$GFR.binary.future)

R.quad-R.no.diag

cvfitL1.diag<-cv.glmnet(y=datenmat[[k]][,"GFR.binary.future"],
                        x=xmat.diag,
                        family="binomial", foldid=folds, alpha=1,
                        parallel=T)
fitL1.diag<-cvfitL1.diag$glmnet.fit

preds.diag<-predict(fitL1.diag, xmat.diag, s=cvfitL1.diag$lambda.1se,
        type="response")
R.diag<-var(preds.diag)/var(datenmat[[k]]$GFR.binary.future)

R.diag


# medication
```

```
xmat.no.med<-xmat[,-c(1:52,93:186)]
xmat.med<-xmat[,c(1:52,93:95,97,100,102:107,113:117,125,
        129:130,132:137,145,147:151,160,162:164, 169:170,178)]

cvfitL1.no.med<-cv.glmnet(y=datenmat[[k]][,"GFR.binary.future"],
                          x=xmat.no.med,
                          family="binomial", foldid=folds, alpha=1,
                          parallel=T)
fitL1.no.med<-cvfitL1.no.med$glmnet.fit

preds.no.med<-predict(fitL1.no.med, xmat.no.med,
        s=cvfitL1.no.med$lambda.1se, type="response")
R.no.med<-var(preds.no.med)/var(datenmat[[k]]$GFR.binary.future)

R.quad-R.no.med

cvfitL1.med<-cv.glmnet(y=datenmat[[k]][,"GFR.binary.future"],
                       x=xmat.med,
                       family="binomial", foldid=folds, alpha=1,
                       parallel=T)
fitL1.med<-cvfitL1.med$glmnet.fit

preds.med<-predict(fitL1.med, xmat.med, s=cvfitL1.med$lambda.1se,
        type="response")
R.med<-var(preds.med)/var(datenmat[[k]]$GFR.binary.future)

R.med


# demographic variables

xmat.no.demo<-xmat[,-c(91:92,270:275)]
xmat.demo<-xmat[,c(91:92,270:275)]
```

```
cvfitL1.no.demo<-cv.glmnet(y=datenmat[[k]][,"GFR.binary.future"],
                           x=xmat.no.demo,
                           family="binomial", foldid=folds, alpha=1,
                           parallel=T)
fitL1.no.demo<-cvfitL1.no.demo$glmnet.fit

preds.no.demo<-predict(fitL1.no.demo, xmat.no.demo,
        s=cvfitL1.no.demo$lambda.1se, type="response")
R.no.demo<-var(preds.no.demo)/var(datenmat[[k]]$GFR.binary.future)

R.quad-R.no.demo

cvfitL1.demo<-cv.glmnet(y=datenmat[[k]][,"GFR.binary.future"],
                        x=xmat.demo,
                        family="binomial", foldid=folds, alpha=1,
                        parallel=T)
fitL1.demo<-cvfitL1.demo$glmnet.fit

preds.demo<-predict(fitL1.demo, xmat.demo, s=cvfitL1.demo$lambda.1se,
        type="response")
R.demo<-var(preds.demo)/var(datenmat[[k]]$GFR.binary.future)

R.demo


overview<-matrix(c(R.quad-R.no.lab, R.quad-R.no.med, R.quad-R.no.diag,
        R.quad-R.no.demo,R.lab,R.med, R.diag, R.demo), ncol=2)
overview


### progression


# laboratory values
```

```
xmat.no.lab<-xmat[,-c(90:98,grep("binary", colnames(xmat)))]
xmat.lab<-xmat[,c(82:98,458:517)]

cvfitL1.no.lab<-cv.glmnet(y=datenmat.decline[[5]][,"decline"],
                          x=xmat.no.lab,
                          family="binomial", foldid=folds, alpha=1,
                          parallel=T)
fitL1.no.lab<-cvfitL1.no.lab$glmnet.fit

preds.no.lab<-predict(fitL1.no.lab, xmat.no.lab,
        s=cvfitL1.no.lab$lambda.1se, type="response")
R.no.lab<-var(preds.no.lab)/var(datenmat.decline[[k]]$decline)

R.quad-R.no.lab

cvfitL1.lab<-cv.glmnet(y=datenmat.decline[[5]][,"decline"],
                       x=xmat.lab,
                       family="binomial", foldid=folds, alpha=1,
                       parallel=T)
fitL1.lab<-cvfitL1.lab$glmnet.fit

preds.lab<-predict(fitL1.lab, xmat.lab, s=cvfitL1.lab$lambda.1se,
        type="response")
R.lab<-var(preds.lab)/var(datenmat.decline[[k]]$decline)

R.lab

# diagnoses

xmat.no.diag<-xmat[,-c(62:81,99,110:112,127:129,136,144:146,163:164,
        180:182,191:192,207:209,225:229,239:241,251,258,266,275:277,
        296:301,312:314,326:331,343:345,358:360,369:373,383:385,392:394,
        401,406:457)]
xmat.diag<-xmat[,c(62:81,99,406:408,422:424,435:436,443)]
```

```
cvfitL1.no.diag<-cv.glmnet(y=datenmat.decline[[k]][,"decline"],
                           x=xmat.no.diag,
                           family="binomial", foldid=folds, alpha=1,
                           parallel=T)
fitL1.no.diag<-cvfitL1.no.diag$glmnet.fit

preds.no.diag<-predict(fitL1.no.diag, xmat.no.diag,
        s=cvfitL1.no.diag$lambda.1se, type="response")
R.no.diag<-var(preds.no.diag)/var(datenmat.decline[[k]]$decline)

R.quad-R.no.diag

cvfitL1.diag<-cv.glmnet(y=datenmat.decline[[k]][,"decline"],
                        x=xmat.diag,
                        family="binomial", foldid=folds, alpha=1,
                        parallel=T)
fitL1.diag<-cvfitL1.diag$glmnet.fit

preds.diag<-predict(fitL1.diag, xmat.diag, s=cvfitL1.diag$lambda.1se,
        type="response")
R.diag<-var(preds.diag)/var(datenmat.decline[[k]]$decline)

R.diag


# medication


xmat.no.med<-xmat[,-c(1:61,101:405)]
xmat.med<-xmat[,c(1:61,101:109,125:126,141:143,160:162,177:179,
        189:190,199:206,216:224,237:238,249:250,256:257,263:265,271:274,
        289:295,309:311,321:325,339:342,355:357,367:368,382)]

cvfitL1.no.med<-cv.glmnet(y=datenmat.decline[[k]][,"decline"],
                          x=xmat.no.med,
```

```
                          family="binomial", foldid=folds, alpha=1,
                          parallel=T)
fitL1.no.med<-cvfitL1.no.med$glmnet.fit

preds.no.med<-predict(fitL1.no.med, xmat.no.med,
        s=cvfitL1.no.med$lambda.1se, type="response")
R.no.med<-var(preds.no.med)/var(datenmat.decline[[k]]$decline)

R.quad-R.no.med

cvfitL1.med<-cv.glmnet(y=datenmat.decline[[k]][,"decline"],
                       x=xmat.med,
                       family="binomial", foldid=folds, alpha=1,
                       parallel=T)
fitL1.med<-cvfitL1.med$glmnet.fit

preds.med<-predict(fitL1.med, xmat.med, s=cvfitL1.med$lambda.1se,
        type="response")
R.med<-var(preds.med)/var(datenmat.decline[[k]]$decline)

R.med


# demographic variables

xmat.no.demo<-xmat[,-c(100:101,518:521)]
xmat.demo<-xmat[,c(100:101,518:521)]


cvfitL1.no.demo<-cv.glmnet(y=datenmat.decline[[k]][,"decline"],
                           x=xmat.no.demo,
                           family="binomial", foldid=folds, alpha=1,
                           parallel=T)
fitL1.no.demo<-cvfitL1.no.demo$glmnet.fit
```

```
preds.no.demo<-predict(fitL1.no.demo, xmat.no.demo,
        s=cvfitL1.no.demo$lambda.1se, type="response")
R.no.demo<-var(preds.no.demo)/var(datenmat.decline[[k]]$decline)

R.quad-R.no.demo

cvfitL1.demo<-cv.glmnet(y=datenmat.decline[[k]][,"decline"],
                        x=xmat.demo,
                        family="binomial", foldid=folds, alpha=1,
                        parallel=T)
fitL1.demo<-cvfitL1.demo$glmnet.fit

preds.demo<-predict(fitL1.demo, xmat.demo, s=cvfitL1.demo$lambda.1se,
        type="response")
R.demo<-var(preds.demo)/var(datenmat.decline[[k]]$decline)

R.demo


### eGFR

xmat.no.GFR<-xmat[,-c(478:485)]
xmat.GFR<-xmat[,c(478:485)]


cvfitL1.no.GFR<-cv.glmnet(y=datenmat.decline[[k]][,"decline"],
                          x=xmat.no.GFR,
                          family="binomial", foldid=folds, alpha=1,
                          parallel=T)
fitL1.no.GFR<-cvfitL1.no.GFR$glmnet.fit

preds.no.GFR<-predict(fitL1.no.GFR, xmat.no.GFR,
        s=cvfitL1.no.GFR$lambda.1se, type="response")
R.no.GFR<-var(preds.no.GFR)/var(datenmat.decline[[k]]$decline)
```

```
R.quad-R.no.GFR

cvfitL1.GFR<-cv.glmnet(y=datenmat.decline[[k]][,"decline"],
                       x=xmat.GFR,
                       family="binomial", foldid=folds, alpha=1,
                       parallel=T)
fitL1.GFR<-cvfitL1.GFR$glmnet.fit

preds.GFR<-predict(fitL1.GFR, xmat.GFR, s=cvfitL1.GFR$lambda.1se,
        type="response")
R.GFR<-var(preds.GFR)/var(datenmat.decline[[k]]$decline)

R.GFR


overview<-matrix(c(R.quad-R.no.lab, R.quad-R.no.med, R.quad-R.no.diag,
        R.quad-R.no.demo, R.lab,R.med, R.diag, R.demo), ncol=2)
overview
```

Lebenslauf

Christine Wallisch

christine.wallisch@univie.ac.at

Studium

seit 2012	Magisterstudium: Statistik an der Universität Wien Titel der Magisterarbeit: „Development of a prediction model for the decline in kidney function based on a retrospective analysis of 700,000 electronic health records" Leistungsstipendium für 2013
2009–2012	Bachelorstudium: Statistik an der Universität Wien am 13.07.2012 mit Auszeichnung abgeschlossen Leistungsstipendium für 2010 und 2012

Schulische Ausbildung

Schulabschluss	maturiert am 10.06.2009 mit ausgezeichnetem Erfolg
2004–2009	Bundeshandelsakademie Gänserndorf
2000–2004	Bundesrealgymnasium Gänserndorf
1996–2000	Volksschule in Lassee

Berufliche Laufbahn

seit Okt. 2010	Studienassistentin für „Grundzüge der Wirtschaftsmathematik" bzw. "Mathematik 1" an der Universität Wien
Okt. 2013–Feb. 2014	Tutorin für „Wahrscheinlichkeitsrechnung" an der Universität Wien
August 2010	Ferialpraktikum bei Siemens AG, Abteilung: Corporate Finance
2010	Nachhilfe-Trainerin, effeff language services, Gänserndorf
Juli 2009	Ferialpraktikum bei LBG Wirtschaftstreuhand- u BeratungsgesmbH, Gänserndorf
Juli 2008	Ferialpraktikum bei Raiffeisenbank Wien, Filiale Seilergasse
Juli 2007	Ferialpraktikum bei Raiffeisen Leasing GmbH, Abteilung: Bilanzierung und Konzernsteuerung

Kenntnisse und Interessen

Sprachkenntnisse	Deutsch – Muttersprache Englisch – fließend in Wort und Schrift Spanisch – Grundkenntnisse
EDV-Kenntnisse	Microsoft Office: Word, Excel, PowerPoint, Access, Publisher Buchhaltung: SAP, BMD Statistik: R, SPSS, SAS
Führerschein	Klasse B und F
Interessen	Kochen, Radfahren, Tanzen, Gärtnern

Oktober 2014

Abstract

The SysKID project concentrates on chronic kidney disease (CKD) and kidney function. Two prediction models were developed to better understand kidney disease and function in this master thesis. The first model - the care model - predicts whether a patient receives adequate renal care. Adequate renal care is defined as a minimum of one creatinine measurement. The second prediction model estimates the probability for a decline in kidney function and is called the progression model. A data base containing patient information from 60 primary care physicians' offices was available with demographic facts, laboratory measurements, medication and diagnoses. In the models only patients with diabetes mellitus were considered. As the data base includes longitudinal data, the time aspect was taken into account and treated with dynamic modeling. Thus, it was possible to consider some patients with a long obeservation period several times in the design matrix. Dynamic modeling was combined with non-linear modeling. A preselection of binary variables and interaction terms was done by using a minimum prevalence. A logistic regression with LASSO for variable selection was then used to develop the two models. The care model achieved an R^2 of 0.336 and a c-index of 0.827 and was optimism corrected by a 20-fold cross-validation. The calibration plot showed an almost perfect curve. The progression model had an explained variation of 0.066 and an optimism corrected c-index of 0.825. The calibration plot underestimates the probability for a decline in kidney function for patients who are in the uppermost quintile according to this probability. This is probably because the outcome was unbalanced - only 9 % of the patients experienced a decline in kidney function. Both models can be applied to diabetics between 20 and 85 years who have been monitored for at least one year. The advantage of these models described in this paper is that they were developed using "real-life data" and can therefore easily be applied without requiring data beyond routine examinations. In summary, this thesis demonstrates how innovative approaches to statistical modeling can be applied to derive scientifically plausible conclusions from relatively unstructured and partly uncoded routine patient records.

Zusammenfassung

Das Projekt SysKID beschäftigt sich vor allem mit der chronischen Nierenerkrankung und untersucht diese, um sie besser zu verstehen und deren Progression verlangsamen zu können. Aus diesem Grund werden in dieser Magisterarbeit zwei Vorhersagemodelle erstellt. Das erste betrifft die adäquate Vorsorge der Nierenfunktion und -erkrankung und wurde "care model" genannt. Die adäquate Vorsorge wird hierbei durch die Durchführung einer Messung des Kreatinin-Wertes definiert. Das zweite Vorhersagemodell ("progression model") schätzt die Wahrscheinlichkeit für eine Verschlechterung der Nierenfunktion. Die Daten, die für diese Analysen verwendet wurden, wurden von 60 österreichischen Hausärzten zur Verfügung gestellt. Die Datenbank umfasste 700.000 Patienten und enthielt Informationen zur Demographie, zu gemessenen Laborwerten, zu der Medikation der Patienten und deren Diagnosen. In dieser Magisterarbeit wurden nur Patienten mit der Erkrankung Diabetes Mellitus zur Analyse herangezogen. Da die Patienteninformationen datiert wurden, musste der zeitliche Aspekt durch dynamisches Modellieren berücksichtigt werden. Einige Patienten konnten damit öfters in die Designmatrix aufgenommen werden, da sich ihr Beobachtungszeitraum über mehrere Jahre erstreckte. Zusätzlich wurde dynamische Modellierung hier mit nicht-linearer Modellierung kombiniert. Die Anzahl binärer Variablen und deren Interaktionen wurden durch Prävalenzen von zu mindest 1 % bzw. 5 % beschränkt. Für die Modellentwicklung wurde eine logistische Regression mit LASSO als Variablenselektion verwendet. Die Analyse ergab, dass das "care model" ein Bestimmtheitsmaß, R^2, von 0.336 und einen c-index von 0.827 erreicht. Der c-index wurde zusätzlich durch eine 20-fache Kreuzvalidierung korrigiert. Der Kalibrierungsplot dieses Modells zeigt eine fast perfekte Kurve. Das "progression model" erreichte nur ein R^2 von 0.066. Der korrigierte c-index ist jedoch relativ hoch mit einem Wert von 0.825. Der Kalibierungsplot unterschätzt leider die Wahrscheinlichkeit für eine Verschlechterung der Nierenfunktion bei den 20 % der Patienten mit schlechtester Prognose. Dies ist wahrscheinlich die Konsequenz aus der unbalancierten Verteilung der Response-Variable. Die beiden Vorhersagemodelle sind so erstellt worden, dass sie auf Diabetiker im Alter von 20 bis 85 Jahren, welche zumindest ein Jahr beobachtet wurden, angewendet werden können. Der große Vorteil dieser Modelle ist, dass mit realen Daten aus dem Alltag gearbeitet wurde, welche nicht eigens für eine Studie gesammelt wurde. Somit sind zur Anwendung der Modelle keine Informationen, die über eine routinemäßige Aufzeichnung hinausgehen, notwendig. In dieser Magisterarbeit wird somit demonstriert, wie innovative Ansätze statistischen Modellierens dabei helfen, wissenschaftlich plausible Schlussfolgerungen von relative unstrukturierten und teilweise

unkodierten Routinepatientendaten zu erhalten.