Analysis report

Both protein.R and cancer.R data are divided into data.train and data.test.

At the beginning I carried out the analysis of the explanatory variables for both data. For both data sets, all variables are numeric variables.

1. DATA: protein.R

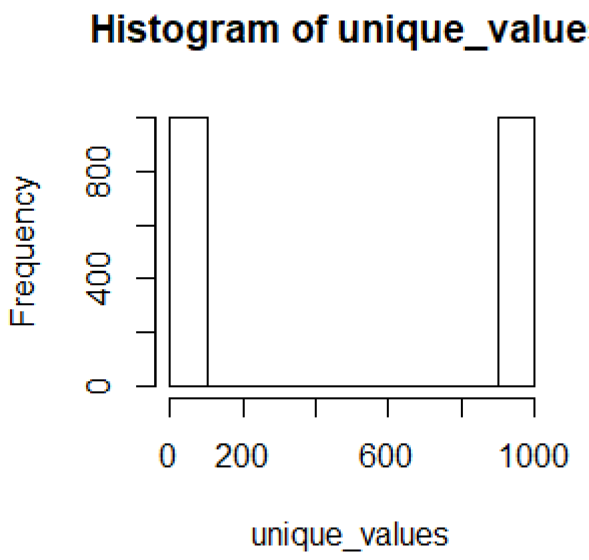
In case of **protein.R** we have:

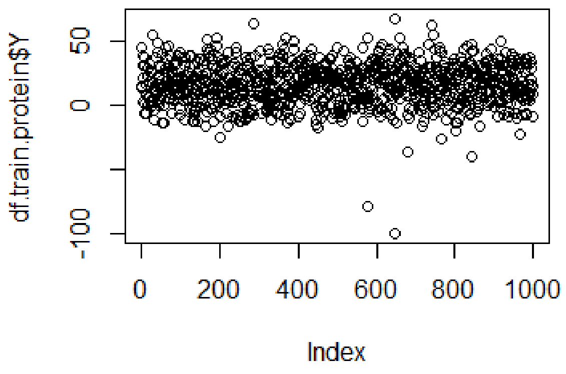
Number of explanatory variables : 2000

Number of explained variables: 1 ('Y')

The explained variable is the level of protein. From the description of the data from the task, it is not possible to clearly state / suppose what type of data we are dealing with. By analyzing the values of unique data, I concluded that some explanatory variables are qualitative variables.

```
unique_values <- apply(df.train.protein, 2, function(x){length(unique(x))})
hist(unique_values)
```





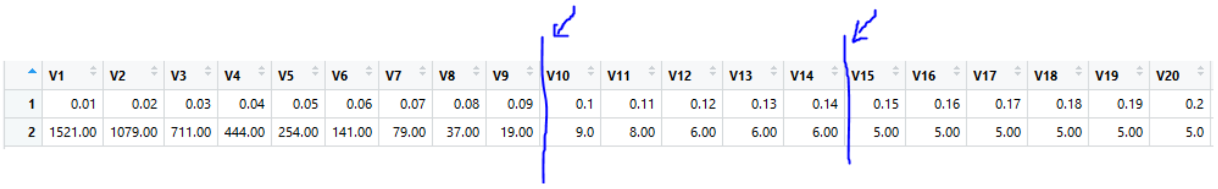
It turned out that there are variables that have two variables (even though the values at first glance are variables with continuous values). For this reason, I have included in the code a formula changing these values to qualitative values.

I also conducted a procedure for normalizing explanatory variables before proceeding with further analysis and model building.

The variable Y as the protein level is a continuous variable.
Minimum value: -100.3702

Maximum value: 67.79427

Considering the huge number of explanatory variables, I also conducted an analysis of the correlation of individual explanatory variables with the explanatory variable. It turned out that there are 9 variables with a correlation of 0.1 or more, and only 5 variables with corelation 0.2 or more.



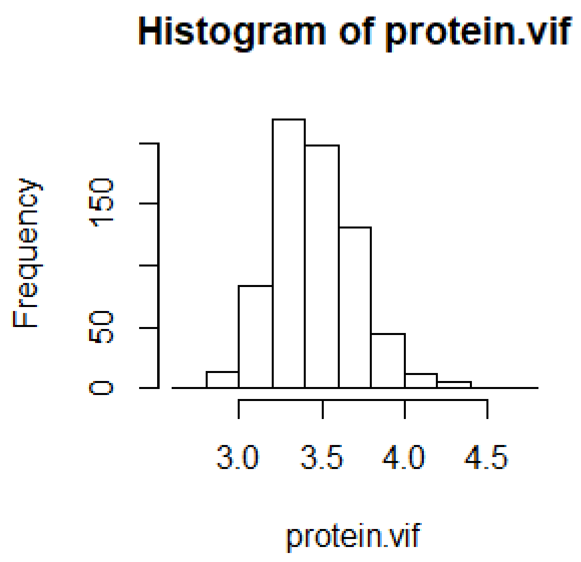
	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14	V15	V16	V17	V18	V19	V20
1	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09	0.1	0.11	0.12	0.13	0.14	0.15	0.16	0.17	0.18	0.19	0.2
2	1521.00	1079.00	711.00	444.00	254.00	141.00	79.00	37.00	19.00	9.0	8.00	6.00	6.00	6.00	5.00	5.00	5.00	5.00	5.00	5.0

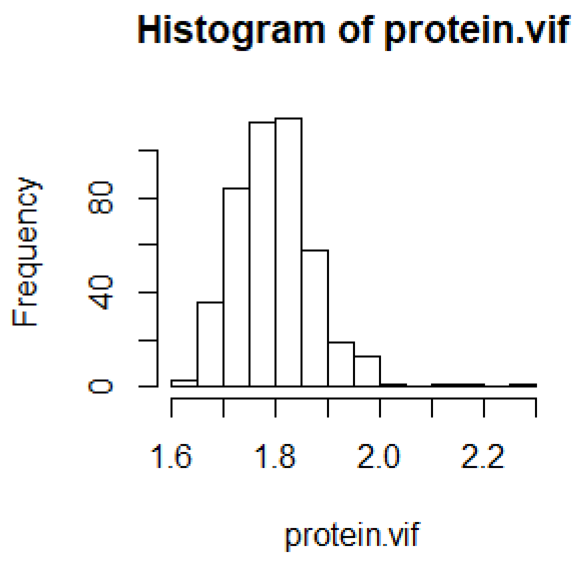
[I row: correlation lower band;
II row: number of variables having correlation equal or greater than the value above]

VIF

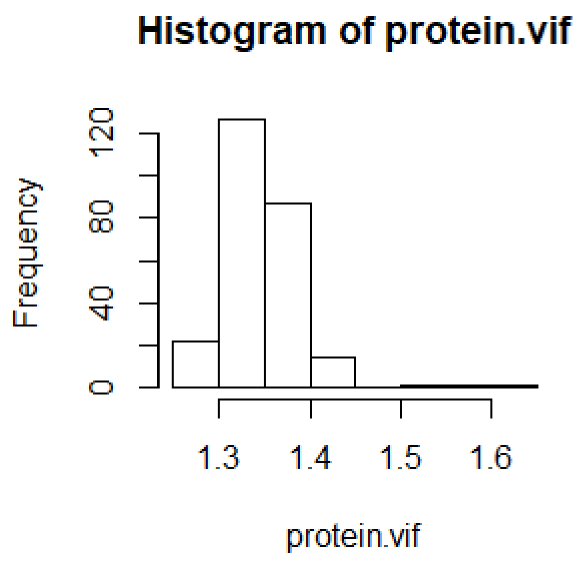
For VIF analysis, I considered two scenarios, choosing the correlation level **0.03**, **0.05, 0.1 i 0.15.** On the histograms, the values of this statistics are as follows:

0.03 **0.04**

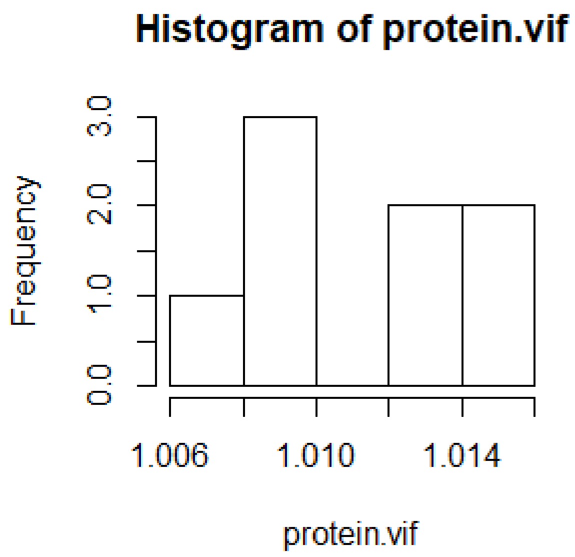




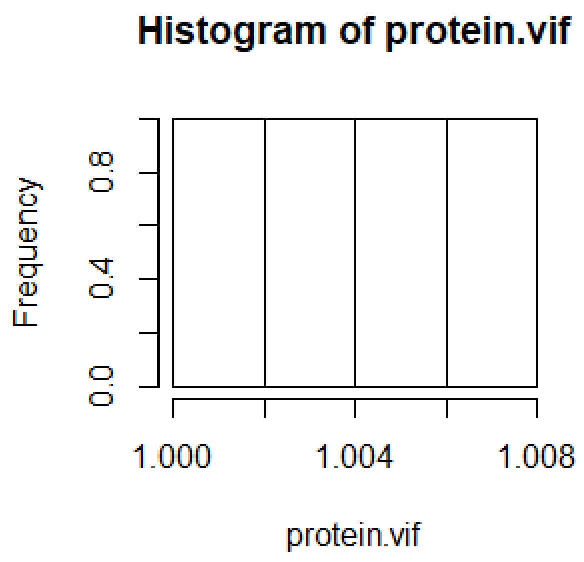
0.05


Histogram of protein.vif
Frequency
0
40
80
120
1.3
1.4
1.5
1.6
protein.vif

0.1


Histogram of protein.vif
Frequency
0.0
1.0
2.0
3.0
1.006
1.010
1.014
protein.vif

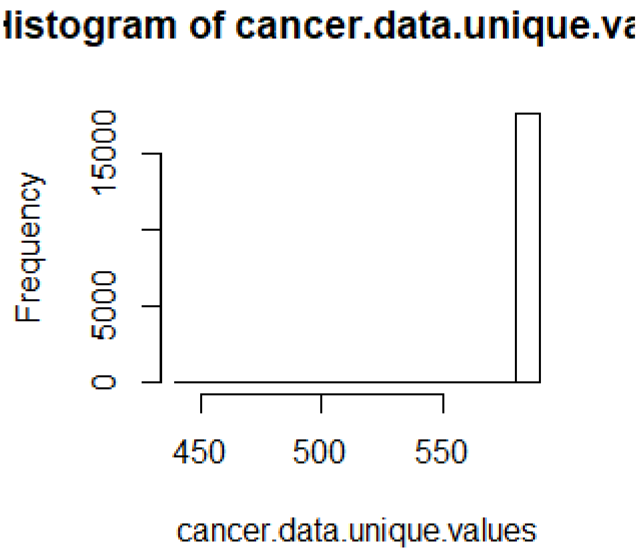
0.15

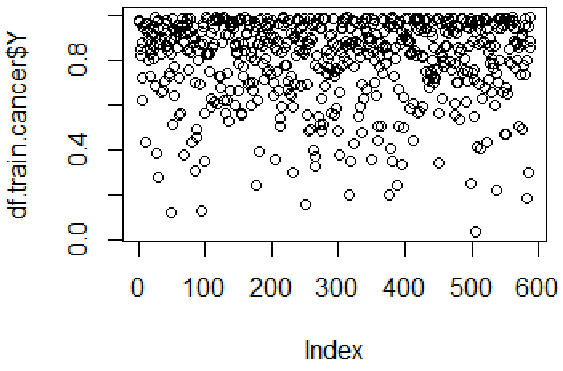

Histogram of protein.vif
Frequency
0.0
0.4
0.8
1.000
1.004
1.008
protein.vif

>>>>>>>...<<<<<<<<<

2. DATA: cancer.R

Similarly to what I did for the protein.R data, I did for the cancer.R data. The analysis of unique values showed that there are no qualitative variables.

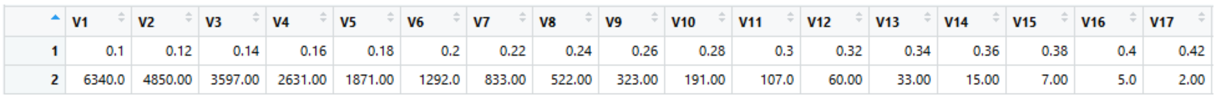




I also conducted a procedure for normalizing explanatory variables before proceeding with further analysis and model building.

The variable Y as the protein level is a continuous variable.
Maximum value: 0.031882
Minimum value: 0.998618

In the case of cancer.R data, there is also a problem of a large number of explanatory variables. Therefore, here I also carried out a correlation analysis of the explanatory variables with the explained variable and narrowed the set on which I then built models.



	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14	V15	V16	V17
1	0.1	0.12	0.14	0.16	0.18	0.2	0.22	0.24	0.26	0.28	0.3	0.32	0.34	0.36	0.38	0.4	0.42
2	6340.0	4850.00	3597.00	2631.00	1871.00	1292.0	833.00	522.00	323.00	191.00	107.0	60.00	33.00	15.00	7.00	5.0	2.00

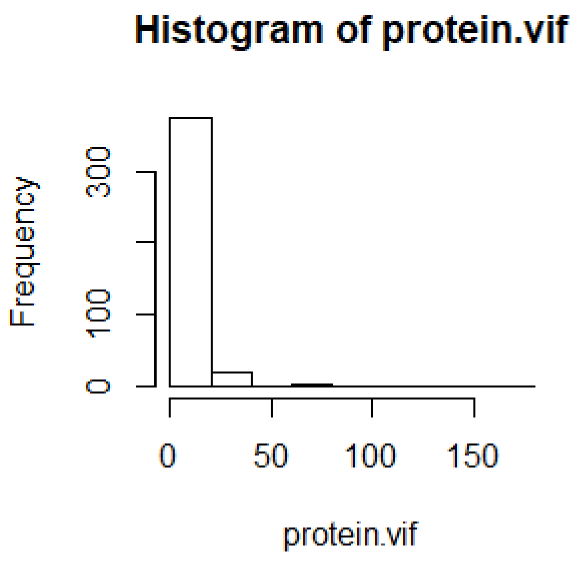
[I row: correlation lower band;
II row: number of variables having correlation equal or greater than the value above]

Due to the enormity of variables on the one hand (and the imperative to limit them) and the description of the data on the other (i.e. explanatory variables are levels of gene expressions and response to the oncological drug it can be assumed that this is a complicated problem at the molecular level and it can be so that the explained variable Y depends on many explanatory variables..

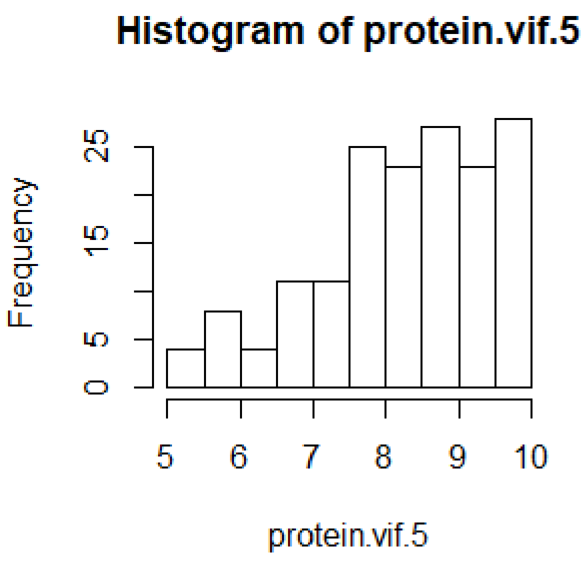
VIF

For VIF analysis, I took into account two scenarios, choosing a correlation level of 0.03, 0.05, 0.1 and 0.15. On the histograms, the values of this statistic are as follows:

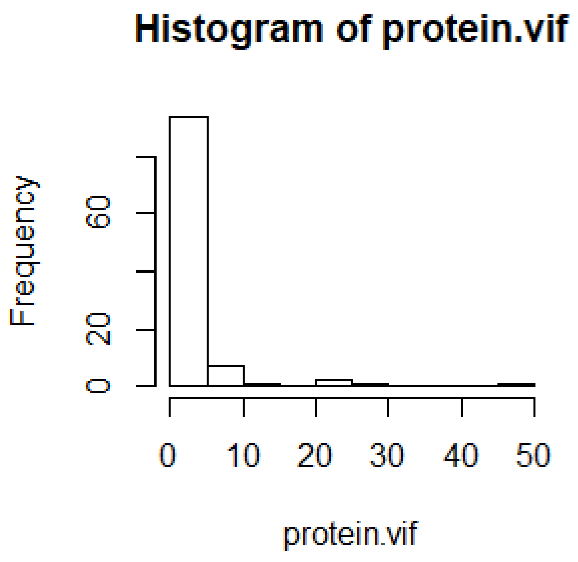
0.25



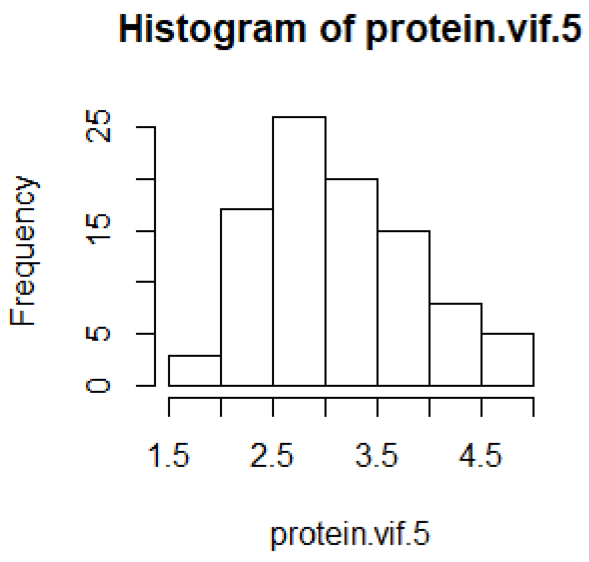
0.25 (II variant: VIFdistribution when VIF <10)



0.3



0.3 (II wariant: rozkład VIF gdy VIF <5)



3. **DESCRIPTION OF MACHINE LEARNING METHODS.**

   a. The first model I chose was the linear model with the selection of predictors by the greedy search method, measuring the significance of parameters with the F statistics. Starting from the null model, I iterated over all predictors I constructed the linear model with +1 predictor to the model (building on the previous iteration) and added the predictor for which the F statistic for the model was the highest. The stop condition was constructed when the F statistics were below 0.1.

   b. In the second model I decided to test is also greedy search using the BIC method (Schwarz information criterion). This method seeks to minimize this information criterion in the form:

$$BIC = (1/n)(RSS + \log(n)d\hat{\sigma}^2)$$

$$BIC = \log(n)k - 2\log(\hat{L})$$

k – number of estimated parameter of the model
L – maximum value of the likelihood function

This method, similar to the method C_p i AIC involves adding an additional penalty to RSS for the number of predictors added to the model. Thanks to such a procedure, we have a certain limitation, which inhibits the selection of the learned model - that is, in this case the model for which too many predictors will be selected. However, this is a stronger penalty (in the case of BIC) than in the case of AIC i C_p.

   c. The next two methods I used were LASSO and ELASTIC NET. I will start by describing the dorsal regression method, which is a combination of the LASSO method and the RIDGE REGRESSION method. In general, a penalty is added to all methods used here in the expression we minimize (i.e. RSS for classic linear regression).
As a rule, the RIDGE REGRESSION method is used for data in which it can be assumed that the explained variable depends on all or almost all explanatory variables. A classic example is genomic data in which a molecular phenomenon depends on many genes (and their level of expression). For this method, the expression we minimize by training the model is:

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}\beta_j^2 = RSS + \lambda\sum_{j=1}^{p}\beta_j^2$$

So using the norm $||\beta||_2 = \sqrt{\sum_{j=1}^{p}\beta_j^2}$ along with a properly selected control parameter, we reduce each beta parameter bit by bit.

The LASSO method is used for data in which we also have a lot of explanatory variables, but it can be presumed that the effect (explanatory variable) depends on a number of explanatory variables clearly smaller than the entire data set on which we work. The LASSO method helps to select for real predictors. For this model, we minimize the expression:

$$\sum_{i+1}^{n}\left(y_i-\beta_0-\sum_{j=1}^{p}\beta_j x_{ij}\right)^2+\lambda\sum_{j=1}^{p}|\beta_j| = RSS+\lambda\sum_{j=1}^{p}|\beta_j|.$$

So using the norm $||\beta||_1 = \sum|\beta_j|$ together with properly selected lambda, we will equate some irrelevant parameters with zero.

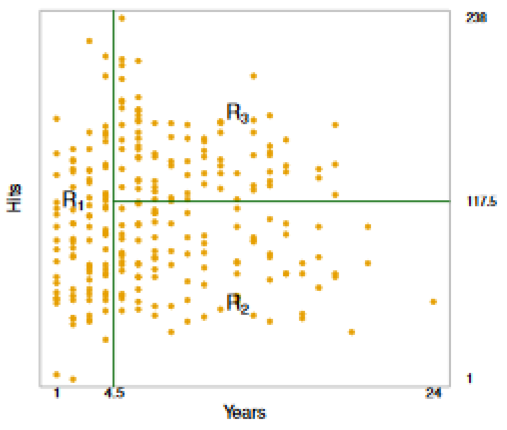
ELASTIC NET is a combination of these two approaches, which simultaneously eliminates irrelevant parameters, but slightly reduces the parameters while avoiding over-modeling.

By training both LASSO and ELASTIC NET models while extracting predictor values, the cv.glmnet () procedure provides two lambda parameters: lambda.min and lambda.1se. Lambda (i.e. the shrinking parameter) gives the best-matched parameters (with the lowest cross-validation MSE), but experimentally the lambda.1se parameter (i.e. we add 1 deviation to the lambda value) gives more conservative results (less relevant parameters).

d. In addition to testing in later steps the parameters obtained from the LASSO and ELASTIC NET methods, I pulled out the parameters and built a simple linear regression model for them using the function lm (Y ~ ...).

e. Random Forests was another method used in the analysis. Random forests work very well with data with a very large number of predictors (p) in relation to the number of observations. In the assumptions of this method, a large number of decision trees are constructed based on minimizing RSS for subspaces that divide our data into disjoint sets due to the value of some predictor.

(graphical example from the lecture)



$$R_1 = \{X \mid Years < 4.5\},\ R_2 = \{X \mid Years \geq 4.5, Hits < 117.5\},$$
$$R_3 = \{X \mid Years \geq 4.5, Hits \geq 117.5\}.$$

Random forests rely on the construction of trees according to the above logic, each time for a different set of variables and from bootstrapped samples from the whole data, which reduces overtraining the model. By constructing a certain number of trees each time with a slightly different set of predictors, you need to choose the parameter that tells you the number of parameters (so that it minimizes the test MSE). To this end, I carried out in the code the procedure for checking the test error for a sequence of parameters from 0.5 * ((number of parameters) ^ 0.5) to 2 * ((number of parameters) ^ 0.5) for a step of 10 and I chose the parameter that gave the smallest error. Then I used this parameter to build the model and check its MSE test error in 5-fold validation. The result of such a model is taken from all trees and averaged (for the continuous variable explained).

f. The last method I used was tree construction using the boosting method. Boosting trees as well as random forests are one of the methods of reducing the variance of the results obtained from the model. Slow learning is used here, i.e. instead of building one large model at a time, with a high risk of over-learning, we proceed in the next B steps, only teaching them a bit. Having the model from a given step, in the next we adjust the tree to the rest of this model and add the resulting tree to the model and update the rest. The parameters that we supply to the model are B (the number of trees, if it is too large, the phenomenon of overtraining may occur), lambda (tightening parameter - controls the rate of 'teaching' - usually they are in the order of 0.01 or 0.001), d (the number of divisions in the tree, controls tree depth). I also experimentally checked level B, which I set to 400.

Statistical learning models applied to the set protein.R

The data shows that we have many more predictors than observations (2000 explanatory variables + 1 explanatory variable x 1000 observations in training data). For this reason, some of the methods learned in the course of the class are not applicable, e.g. exhaustive search (due to the general 'enormity' of data) or backward search (by excluding irrelevant predictors due to t statistics).

Here, after attempting to build the model for all variables, I decided to limit myself to variables whose correlation coefficient is at least 0.05 which gave us a subset for further analysis with 254 variables. In order to train the models and then check the test MSE, I divided the data data.train into train.protein and valid.protein (0.8 observations fell into train. Protein, 0.2 into valid.protein).

I trained all 6 models on this data and then compared their MSE test. This gave me a first insight into how well the models perform in data estimation. Such trained models (predictors selected at this stage, and the corresponding values of their coefficients) of the greedy approach, i.e. model.F and model.BIC, and selected coefficients for model.RANDOM.FOREST and model.GMBTree I applied for the subsequent CV.MSE.Test calculation procedure . For model.ELASTIC.NET and model.LASSO, there was no such need and they learned over the course of subsequent cross-validation from scratch (but they gave the prediction for the same data as other models).

Additionally, in cross-validation, I treated the models: model.ELASTIC.NET and model.LASSO in two ways. Namely, in the first place, I treated them as a way to filter out important parameters and then training a simple linear regression on these lm parameters (Y ~ significant.parameters, data = ""). Despite this, I also used a different prediction to construct parameters that were trained during ELASTIC NET and LASSO procedures. In the data in the table it is reflected as:

"ELASTIC.NET.1se.FALSE", "ELASTIC.NET.min.FALSE", "LASSO.1se.FALSE", "LASSO.min.FALSE",

"ELASTIC.NET.1se.TRUE", "ELASTIC.NET.min.TRUE", "LASSO.1se.TRUE", "LASSO.min.TRUE"

(In addition, in each case I tested the prediction for lambda.min and lambda.1se, which gave us as many as 8 different results)

Statistical learning models applied to the set cancer.R

The models I used for the cancer set were the same as for the given protein.R, with the difference that I did not use the greedy approach using F statistics and the BIC criterion. This was motivated by the large number of explanatory variables (> 17k).

However, before starting to construct models, I decided to narrow down the input data only for those variables that have a correlation coefficient at least at the level of 0.25 (about 600 variables in the data)

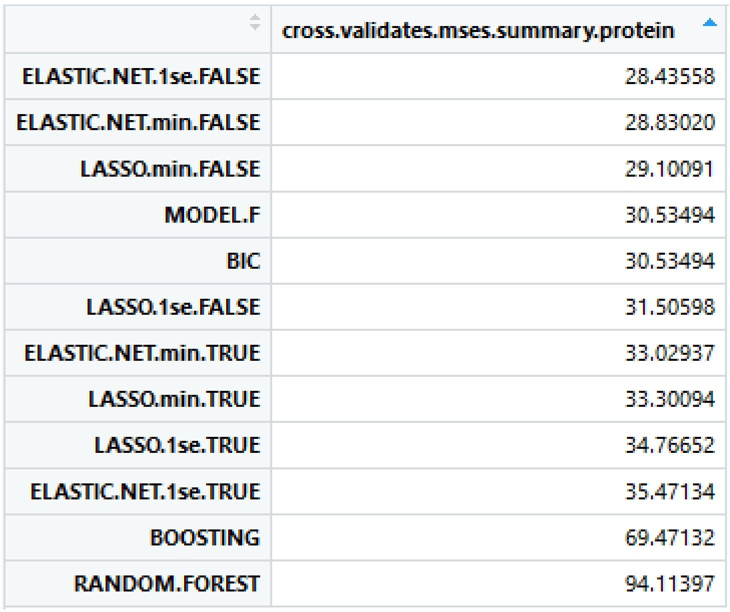
In the case of the cancer.R set, I also used a similar approach as for given proteins, i.e. I used different models and decided to choose the model for prediction only after receiving the CV.MSE errors.Testowy. The difference is that in this case I did not use greedy models (model.F and model.BIC). Here also, analytically for analysis for protein data. I trained the appropriate models for train.cancer data and MSE I checked for valid.cancer. Finally, I used such trained models with appropriately selected parameters to calculate CV.MSE.Testowy.

For random forests, the mtry parameter (the number of predictors used to create trees) was optimal for the level of 250, whereas for trees constructed using the boosting method, the lambda parameter, which minimized the validation MSE, was at the level of 0.16.

As a comment on the experience gained from the analysis of protein.R data I would like to add, that the trees had clearly worse results in relation to the other methods used, regardless of the parameters selected for them.

4. ESTIMATION OF THE TEST ERROR USING CROSS VALIDATION.

protein.R



	cross.validates.mses.summary.protein
ELASTIC.NET.1se.FALSE	28.43558
ELASTIC.NET.min.FALSE	28.83020
LASSO.min.FALSE	29.10091
MODEL.F	30.53494
BIC	30.53494
LASSO.1se.FALSE	31.50598
ELASTIC.NET.min.TRUE	33.02937
LASSO.min.TRUE	33.30094
LASSO.1se.TRUE	34.76652
ELASTIC.NET.1se.TRUE	35.47134
BOOSTING	69.47132
RANDOM.FOREST	94.11397

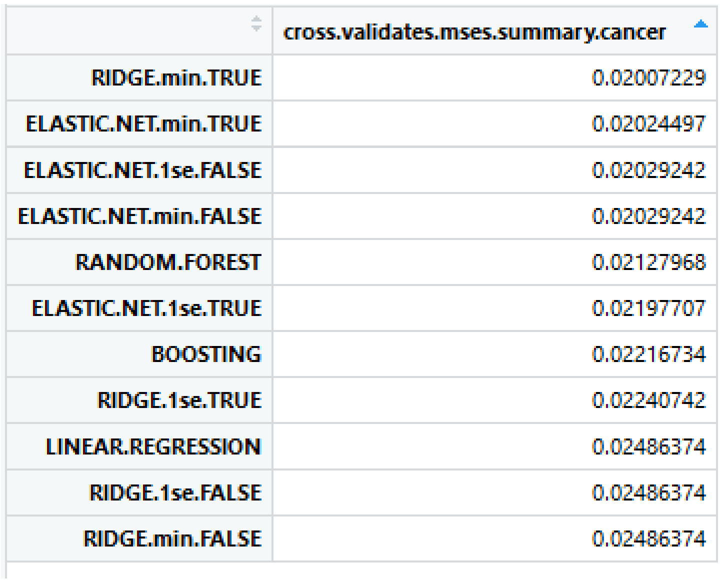
According to the above analysis, the lowest test error is obtained using a model, where I perform variable selection using ELASTIC.NET, and then I calculate the coefficient estimators again.

The top 5 predictors for given proteins are:

```
"x1623" "x420" "x1764" "x456" "x1028"
```

The selection process first consisted of narrowing the data set to those variables whose correlation coefficient was greater than 0.05, then, using several methods of building a linear model. Then calling the coefficients for a given model, I called only those whose coefficient was greater than 1.5. By doing so for both model.F, model.BIC and LASSO.1se, each time the set of these 5 predictors was confirmed, which I mentioned above.

cancer.R



	cross.validates.mses.summary.cancer
RIDGE.min.TRUE	0.02007229
ELASTIC.NET.min.TRUE	0.02024497
ELASTIC.NET.1se.FALSE	0.02029242
ELASTIC.NET.min.FALSE	0.02029242
RANDOM.FOREST	0.02127968
ELASTIC.NET.1se.TRUE	0.02197707
BOOSTING	0.02216734
RIDGE.1se.TRUE	0.02240742
LINEAR.REGRESSION	0.02486374
RIDGE.1se.FALSE	0.02486374
RIDGE.min.FALSE	0.02486374

The validation shows that the best estimated test error obtained with cross-validation gives the RidgeRegression model for the previously selected set of 100 best predictors (i.e. predictors with correlation greater than 0.25 and VIF less than 10).

The best 100 predictors for the cancer harvest are:

```
"ENSG00000011422" "ENSG00000021355" "ENSG00000026508" "ENSG00000049860" "ES
G00000064601" "ENSG00000072110" "ENSG00000073792" "ENSG00000075651" "ENSG00
000087253" "ENSG0000008730"  "ENSG00000089327" "ENSG00000090382" "ENSG00000
092208" "ENSG00000094916" "ENSG00000100027" "ENSG0000010050"  "ENSG00000101
439" "ENSG00000102265" "ENSG00000103540" "ENSG00000104368" "ENSG00000106397
" "ENSG00000107819" "ENSG00000108828" "ENSG00000109452" "ENSG00000110080" "
ENSG00000110628" "ENSG00000111321" "ENSG0000011187"  "ENSG00000112941" "ENS
G00000116127" "ENSG00000118113" "ENSG00000118508" "ENSG00000119986" "ENSG00
000120054" "ENSG00000120318" "ENSG00000120802" "ENSG00000120837" "ENSG00000
123146" "ENSG00000123892" "ENSG0000012457"  "ENSG00000126856" "ENSG00000129
226" "ENSG00000130958" "ENSG00000131981" "ENSG00000135318" "ENSG0000013540"
"ENSG00000135926" "ENSG00000136158" "ENSG00000137575" "ENSG00000138376" "EN
SG00000139289" "ENSG0000013931"  "ENSG00000141127" "ENSG00000143226" "ENSG0
0000143669" "ENSG00000145040" "ENSG00000147439" "ENSG0000014818"  "ENSG0000
0148655" "ENSG00000148834" "ENSG00000154493" "ENSG00000155330" "ENSG0000015
7227" "ENSG00000158747" "ENSG00000159445" "ENSG00000160307" "ENSG000016078
9" "ENSG00000161013" "ENSG00000161091" "ENSG00000162607" "ENSG00000163820"
"ENSG00000164062" "ENSG00000164733" "ENSG00000165113" "ENSG00000166068" "EN
SG0000016924"  "ENSG00000170275" "ENSG00000170485" "ENSG00000172687" "ENSG0
0000175832" "ENSG00000180626" "ENSG0000018138"  "ENSG00000181467" "ENSG0000
0181649" "ENSG00000182612" "ENSG00000183615" "ENSG00000183674" "ENSG0000018
423" "ENSG00000185022" "ENSG00000185664" "ENSG00000187098" "ENSG00000188662
" "ENSG00000197124" "ENSG0000019795" "ENSG00000197971" "ENSG00000203306" "E
NSG00000242852" "ENSG00000244405" "ENSG00000248905"
```