Norwegian School of Economics
Bergen, Spring 2019

Predictive analytics with R

BAN404 - Project 1

Candidate no:

17

47

95

61

Professor: Jonas Andersson

Department of Business and Management Science

NORWEGIAN SCHOOL OF ECONOMICS

Introduction

In this paper, we will explore several prediction methods based on the data set *ceosal2* from the package **Wooldridge**. We will not consider the prediction coefficients, since we want to find the method that yields the most accurate predictions regardless of whether the coefficients may seem sensible or not. We will use an experimental approach, where we will look at the following three metrics in order to find the best method; *RMSE*, R^2 and *MAE*.

Root Mean Squared Error (*RMSE*) calculates the square of average differences between predicted and actual observations. Mean Absolute Error (*MAE*) calculates the average absolute differences between predictions and actual observations, where all individual differences have equal weight. The difference between these methods is how these metrics are effected by outliers. RMSE gives a relatively high weight to large errors, while MAE would be steady in such case. Furthermore, R-squared (R^2) is a statistical measure which is a proportion that explains the variation in the dependent variable that is going to be predicted.

Further in this paper, we will first present a descriptive statistics of the initial dataset *ceosal*. Next we will look at the different prediction methods we are going to use in order to predict the outcome variable *salary* and corresponding evaluation methods will also be presented. Thereby we will apply these methods by predicting and evaluating the results. Lastly, a summary of the best methods with regards to the metrics we use will be presented and a corresponding conclusion will be derived.

Contents

List of Tables

Task A

Descriptive statistics

The ceosal2 (CEO salary 2) data set consists of 15 variables and 177 observations. There are no missing values in the set. The dependent variable in our task is *salary*, which is the 1990 compensation in $1000s for 177 CEOs. There are 14 independent variables and they are of different types. The *college* and *grad* variables are dummies, 1 if attended college and 1 if attended graduate school, 0 otherwise. We also have log variables present, such as *lsalary* and *lsales*. The variables *comten* (years with company) and *ceoten* (years as ceo with company) are also present in squared form, as *comtensq* and *ceotensq*. The rest of the variables are *profits*, 1990 firm sales in millions, *mktval*, market value in the end of 1990 in millions and *profmarg*, profits as percent of sales, as well as the *age* of the CEOs. As the *lsalary* variable is the log variable of the variable we are trying to predict, this cannot be used as a predictor.

The CEOs are aged from 33 to 86 years, with a mean of just over 56. Their salaries in 1990 vary from 100 to 5 299, with a mean of 865.9 and a median of 707. From this, we deduce that the CEO that earned the most in compensation exceeds six times the mean of the CEO salaries. The second highest observation in this column is 2 792, close to 90% lower than the highest value, and thus the former appears like an outlier in our data set. The college and grad dummies have a mean of 0.97 and 0.53 respectively, which can be interpreted to 97% and 53% participation. From the profits vector we observe that nine of the values are negative. The different variables are a mixture of integers and numeric values. The included squared variables in the regression are testing the quadratic relationship between the said variables and salary.

The aforementioned observation in the salary column is worth looking into. There is a possibility that it is an error in data collection or recording, or that it is just an observation that considerably stands out. We are aware that removing an observation might lead to skewing and manipulating the data set in a disadvantageous way. However, care should be taken, since an outlier may instead indicate a deficiency with the model, such as a missing predictor (James, Witten, Hastie, & Tibshirani, 2013). We are aware of the consequences of excluding this observation, but conclude that the data set is a good representation of the actual data without this, and therefore we decide to exclude this observation.

We have also considered the profit margin observation in row 168. The observed value of -203.08 is removed due to the same reasons as presented above. It looks like an outlier in our data set as the value differs by a huge amount compared to the mean of 6.4 and median of 6.8, while the second lowest value is over four times smaller than the lowest.

The variable college is also worth mentioning. We argue that this variable may affect whether or not you are granted the CEO position, but does not necessarily help predict salary. When you have proven that you are worthy of serving as CEO, we claim that the degree you received years ago would not be significant in predicting salary. Nevertheless, we do not exclude this variable due to the risk of twisting the data.

Task B & C

Brief Explanation of Prediction Methods

We start by presenting a brief explanation of the prediction methods, how they are developed from standard linear regression, and the evaluation methods and we are using in this paper.

Linear Regression

Standard linear regression involves predicting a response Y on the basis of one or more predictors X; $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_p X_p$ where β_p is the average effect on Y if X_p increases by one unit (keeping everything else constant). To do this, we need to estimate the coefficients $(\beta_0, \beta_1, \beta_2, ..., \beta_p)$ and choose the ones that minimizes the Residual Sum of Squares (RSS). It is important to find out if there is a relationship between the response and predictors and only include the predictors that are associated with the response.

Subset Selection

Subset selection "involves identifying a subset of the p predictors that we believe to be related to the response. We then fit a model using least squares on the reduced set of variables" (James et al., 2013, p 204). In other words, subset selection is a method for choosing the most important set of predictors for the regression model.

We have considered the following subset selection methods: *Best Subset Selection*, *Forward Stepwise Selection* and *Backward Stepwise Selection*. The Best Subset method includes all possible models and outputs the best candidates. The Stepwise methods explore fewer models by adding or removing one predictor at a time. Hence, the Stepwise methods are less complex than the Best Subset method. However, Best Subset Selection cannot be used when the number of predictors is very high. Our data set includes only 13 predictors (after removing *lsalary*), in addition to the response, and we therefore choose to use the Best Subset Selection method.

Best Subset Selection

Best Subset Selection fits a least squares regression for every (possible) combination of the p predictors. In other words, we fit all p models that includes exactly one predictor, all $\binom{p}{2} = p(p-1)/2$ models that contain exactly two predictors, and so on. Further, we look at all the resulting models, aiming to recognize the best one. The problem appears when selecting the best model from Best Subset Selection. We want to choose the model with a low test error, based on different metrics, such as Adjusted R^2, Mallows's Cp, AIC and BIC. The three latter all have rigorous theoretical justifications, while adjusted R^2 is not as well motivated in statistical theory as Mallows's Cp, AIC and BIC (James et al., 2013, p 205).

Lasso and Ridge

When trying to fit a linear model to your data, we are minimizing the mean squared error between our model and the data. This is done by splitting the data into two parts, called train and test data. We fit the model to the training data, before assessing the model on the rest of the sample, known as the test. A way of evaluating how well this model fits with the data is done by the metric MSE. The model is better the smaller the MSE. One problem that may occur whilst doing this is that the MSE of training set is small, i.e. fits well, but does not fit as well on the test data. This matter is known as overfitting.

One way to approach overfitting is by adding a term - as developing from the standard linear regression – to the MSE that punishes large coefficients in our model. This is known as regularizations and thus shrinking the coefficient estimates could considerably reduce the variance. The two ways of dealing with this are called ridge and lasso. Where the former penalizes the sum of squared coefficients, the latter penalizes the sum of the absolute value of the coefficients. These processes result in regularized linear regression with fine tuned parameters that do not overfit, ending up with similar MSEs of the train and test set. Whilst an advantage of ridge is that it could reduce the variance (though increasing bias), and thus improve the predictive performance, it is incapable of shrinking the coefficients to exactly zero. However, in our data set we have primarily looked at RMSE, which is the root of MSE.

Evaluation Methods

We start by presenting a brief explanation of all the evaluation methods we are using in this paper.

Cross-Validation

Cross-Validation involves different validation techniques by estimating "the test error rate by holding out a subset of the training observations from the fitting process, and then applying the statistical learning method to those held out observations" (James et al., 2013, p 176). Hence, generalizing the results from statistical analysis to independent data. Cross-Validation outputs the test error for each model so that we can select the model with the lowest test error, hence estimate the accuracy of our prediction model. Further on, we will evaluate the predictions from Best Subset Selection using two different Cross-Validation methods; *The Validation Set Approach* and *K-fold Cross-Validation*.

The Validation Set Approach

The Validation Set Approach involves "randomly dividing the available set of observations into a training set and a validation set. The model is fit on the training set, and the fitted model is used to predict the responses for the observations in the validation set" (James et al., 2013, p 176).

K-fold Cross-Validation

K-fold Cross-Validation involves "randomly dividing the set of observations into k folds, of approximately equal size. The first fold is treated as a validation set, and the method is fit on the remaining *K - 1* folds. The MSE is then computed on the observations in the held-out fold. This procedure is repeated K times; each time, a different group of observations is treated as a validation set" (James et al., 2013, p 181). This method results in K different test errors, and the lowest one corresponds to the best model.

Standard Linear Regression

Since the aforementioned data set consists of multiple predictors, we use a standard multiple linear regression to predict the CEO salary. When fitting our linear regression model, we used the *lm* function from the **stats** package.

When training the model, we split the data into a test set and a train set, where the train set consists of 75% of observations in the total data set, while the test set consists of the remaining observations. The 75/25-split is chosen in order to include as many observations in the train set as possible, while still having enough observations in the test set to properly evaluate the fitted model.

We first fit our model on the train set by including all variables in the data, before we use the fitted model to predict on our test set. When doing so, we notice that the RMSE varies significantly, depending on the seed that was used when splitting up the data set. We therefore decide to run the model 10 000 times in order to ensure that our evaluation metrics were not influenced by the chosen seed. When we run the model 10 000 times with random seeds, the RMSE varies, with more than \$450, however, the mean of all the predictions gives us a RMSE of \$403.

Furthermore, we want to evaluate the linear regression model, with different variables, in order to see if the exclusion of some variables could have an effect on the model's predicting power. Ideally, we would try all different combinations of variables, but this is not possible, due to our limited processing power. We therefore choose to use the variables' correlation as basis for our exclusion of variables. By using the correlation from the correlation matrix (Appendix A), we decide to fit and ecaluate the model, removing all variables with a correlation above 0.7, one at a time. We then fit and evaluate the model 10 000 times, trying new combinations of variables, every time. The model (Appendix B), yielded the metrics in Table 01.

As can be seen in Table 01, the lowest average RMSE occurred when the variable *profits* was removed from the data set and all other variables were included. The same model also has the lowest MAE and we therefore conclude that the model with the best prediction power, out of the ones we have looked at, includes all variables besides *profits*. It should be noted however, that the differences between some of the models are fairly small and many have close to equal R^2.

In addition to the validation set approach, we use *k-fold cross-validation* to evaluate the linear

Table 01: Metrics from variable selection

	RMSE	R2	MAE
salary~.	402.90	0.35	296.45
comten, ceoten	407.69	0.33	297.05
comtensq, ceotensq	401.60	0.35	290.99
profits	395.99	0.36	289.25
lmktval	403.31	0.34	301.67
mktval	397.78	0.36	293.26
sales	398.91	0.36	292.73
lsales	397.12	0.36	293.02

regression model. Since the RMSE was lowest when the *profit* variable was extracted, the *k-fold CV* was fitted without the abovementioned variable. The *k-fold CV* approach resulted in an RMSE of $375.4. The remaining evaluation metrics can be found in Table 02 and the code can be found in Appendix F.

Table 02: Linear regression metrics from k-fold CV

	RMSE	R2	MAE
Regall	375.4	0.36	282.43

When comparing the metrics in the two methods, we see that the *k-fold CV* method yields a lower RMSE, the same R^2 and a lower MAE, than the *validation set approach.*

Best Subset Selection

When running the regression for subset selection, we use the function *regsubsets* from the **leaps** package and include the whole data set. The code can be found in Appendix C.

By running the code in , we see that the best one-variable model includes the variable *lmktval*, while the best four-variable model includes the variables *ceoten*, *mktval*, *lsales* and *comtensq*. In order to determine which of these models that is the best one, we plot the resulting RSS, *Adjusted* R^2, Mallows's Cp and BIC in Figure 01.

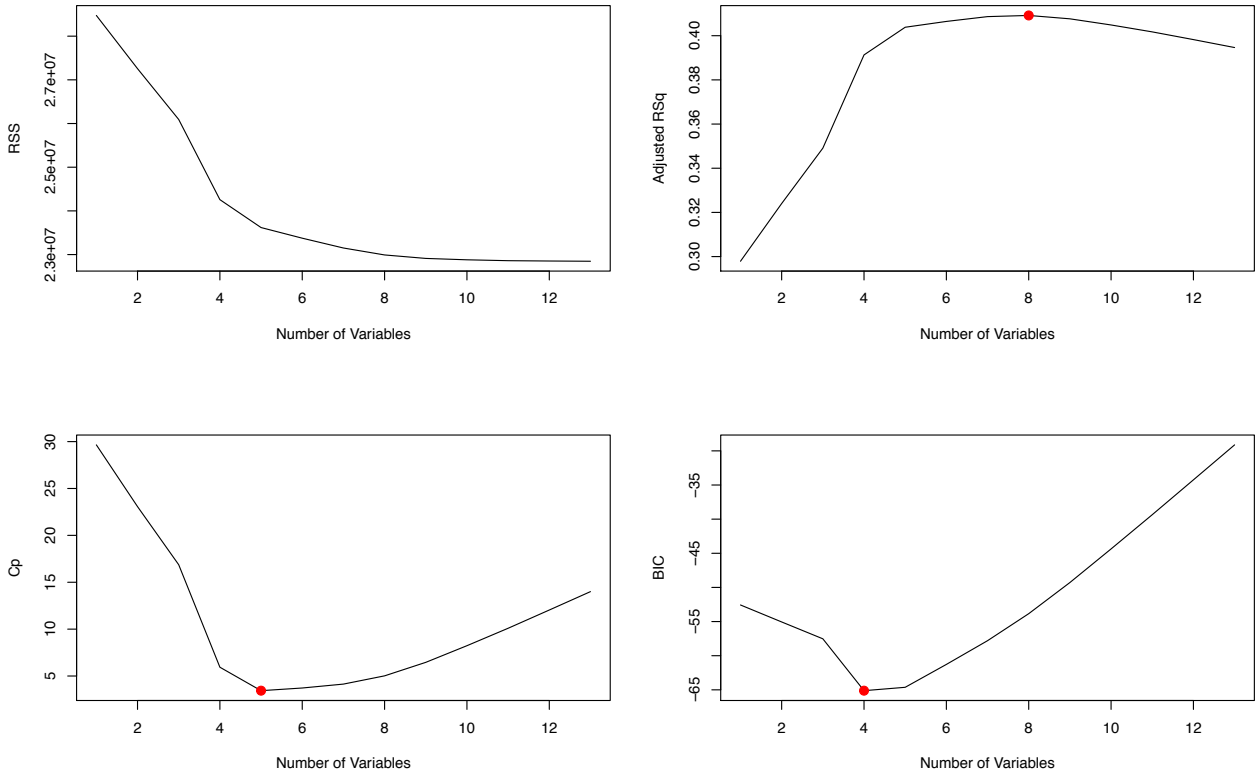


Figure 01: Plot Best Subset

We see that the RSS (Residual Sum of Squares) decreases with the number of variables. From one to four variables, the curve is very steep. From four to eight variables, the curve starts to flatten out, while beyond eight variables, it is almost completely flat. If we wanted to choose the model with the lowest training RSS (i.e. the lowest training error), we would choose a model that includes all the variables in the data set. However, we want a model with a low test error, which tends to be higher than the training error (James et al., 2013, p 205). Hence, we cannot use the RSS to choose the best model.

Adjusted R^2, Mallows's Cp and BIC are three different approaches for selecting among the different models. The adjusted R^2 is the regular R^2 adjusted for the number of predictors and should be as high as possible. The Mallows's Cp includes a penalty added to the RSS in order to adjust for the bias in the training error. Hence, Cp is an unbiased estimate of the test MSE (Mean Squared Error) and should therefore be as low as possible. The BIC is similar to Mallows's Cp but adds a heavier penalty on models with many variables. This approach will therefore result in a selection of a model with fewer variables than the Cp approach. We see that the adjusted R^2 chooses the eight-variable model as the best one, Mallows's Cp chooses the five-variable model, while BIC chooses the four-variable model.

When fitting the different models for prediction, we split the data using a 75/25 split. To avoid the problem with different results for different seeds, we run the code 10 000 times and compute the mean errors of every run.

We take a look at the best models according to the different metrics; Adjusted R^2, Mallows's Cp and BIC. We fit the model to the training set using least squares on the eight-variable model. We then predict *salary* using the fitted model on the test set. Finally, we use the **caret** package to find the RMSE, the R^2, the MAE, showed in Table 03.

Table 03: Results Best Subset

	RMSE	R2	MAE
Adj_Rsq	408.53	0.34	294.83
Cp	379.66	0.40	283.77
BIC	379.99	0.40	280.46

We see that the models seem almost equally good, but the Cp five-variable model is slightly more accurate than the others.

We see that the Cp and BIC models have similar RMSE and R^2, but the BIC model has a slightly lower MAE, which is desirable. In addition, the BIC model includes only four variables and thus is the simplest model.

The Validation Set Approach

Again, we run the code several times to avoid the problem with different results for different seeds. However, to avoid achieving higher dimensions than the dimension of the object, the run is limited to 12 times.

First, we perform Best Subset Selection on the training set, which yields a fitted model. Then the fitted model is used to compute the predictions and the corresponding test errors, as shown in 04.

Table 04: Results Best Subset VSA

	RMSE	R2	MAE
Val	504.47	0.2	343.8

The mean number of variables that resulted in the lowest test error, was $3.5 \approx 4$ variables. However, note that the included variables vary for each run meaning that the results are prone to errors.

K-fold Cross-Validation

When performing k-fold cross-validation, we divide the set of observations into 10 folds that each include 17 observations. To determine which model is the best in each round, we use the maximum adjusted R^2. Then we run a linear regression on the best model and compute the predictions of *salary*, and the results is depicted in Table 05.

Table 05: Results Best Subset k-fold CV

	RMSE	R2	MAE
K__fold	430.02	0.37	299.17

We also compute the mean number of variables that corresponds to the best models in each fold. On average, the best models in each fold has eight variables. Expanding on this, we find the total number of rounds each variable is included in the best model. For instance; *sales*, *comtensq* and *ceotensq* are included in every model (there are a total of six eight-variable models). The first one is the intercept, which naturally is included in every model. 06 shows how many times each variable is included, when we filter for models that include eight variables.

Table 06: Number of occurrences

intercept	age	college	grad	comten	ceoten	sales	profits	mktval	lsales	lmktval	comtensq	ceotensq	profmarg
6	1	1	4	0	6	0	1	5	5	5	6	6	2

Comparing the different methods

When we compare the test errors from the different methods, we see that the overall best method is the BIC-method, as it has the lowest RMSE, highest R^2 and lowest MAE. In addition, it also seems to be the simplest model in terms of number of variables included.

Table 07: Results Best Subset All

	RMSE	R2	MAE
Adj_Rsq	408.53	0.34	294.83
Cp	379.66	0.40	283.77
BIC	379.99	0.40	280.46
Val	504.47	0.20	343.80
K_fold	430.02	0.37	299.17

Validation set approach

When running the validation set approach, we are using functions from the *glmnet* package. For the intitial splitting of the data into training and test set we use a 75/25-split.

Ridge

As mentioned earlier, Ridge is similar to least squares, but in addition the method shrinks the coefficients close to zero, called a shrinkage penalty. The idea is to add a penalty equivalent to the square of the magnitude of coefficients (James et al., 2013).

As briefly looked into, the validation set approach randomly splits the data into training and test set, and uses the train data to fit a model to predict the output variable in the test data. Since this approach is highly variable and depends on the observations that are included in the train and test data, we choose to simulate the model 10 000 times and then calculate the mean of the average of the appropriate metrics we are investigating. The values in Table 08 are obtained.

Table 08: Ridge metrics from VSA

	RMSE	R2	MAE
Ridge VSA	389.2	0.38	289.59

Using the validation set approach the ridge model yields an RMSE of 389.2 and an R^2 of 0.38, and thus the model does not perform any better than the previous models investigated.

Lasso

In contrast to Ridge, Lasso will generally select the models that just involve a subset of the variables. As mentioned earlier, we also run this model 10 000 times to account for the differences in shuffling of train- and test data. When constructing the Lasso model, we obtain the results in Table 09.

Table 09: Lasso metrics from VSA

	RMSE	R2	MAE
Lasso VSA	393.45	0.36	290.52

We note that this particular approach yields a higher RMSE and lower R^2 than the other models that we have investigated, which is undesirable. The obtained results thus indicate that this model is less able to predict the test data.

K-Fold Cross-Validation

When running the k-fold cross-validation, we are using functions from the *caret* package. It should also be mentioned that these functions contain automatic tuning of relevant parameters such that the best models are built. Furthermore, when splitting the data into training and test set we here too use a 75/25-split.

Ridge

As we have mentioned previously, the k-fold cross-validation divides a set of observations into k groups of equal size and makes sure that each fold contains non-overlapping training and validation sets. When executing this particular model in order to predict the test data we obtain the results in Table 010.

Table 010: Ridge metrics from k-fold CV

	RMSE	R2	MAE
Ridge KCV	388.36	0.39	288.6

We note that Ridge with k-fold cross-validation has a lower RMSE than both of the results that were obtained under the validation set approach. This indicates that the square root of the average squared errors between actual and predicted values are lower. Furthermore, the MAE is also at a lower relative rate, which also is an informative metric as it gives the same weight to all errors between actual and predicted values. One should also mention that the R^2 is also at a higher rate, which is a desirable outcome, even though it is just slightly higher. With the above mentioned, this particular model seems to predict the actual test data in a better manner than the previous models.

Lasso

Lastly in this section, we use Lasso to predict the test data. The results in Table 010 are obtained.

Table 011: Lasso metrics from k-fold CV

	RMSE	R2	MAE
Lasso KCV	394.28	0.37	293.06

The results attained are quite similar to the results obtained under validation set approach. However, the model has higher RMSE than the Ridge model that was executed under k-fold cross-validation. As a consequence of the obtained results, the metrics indicate that this particular model is less suitable to predict the test data.

Task D

Additional prediction method: PCR

In principal components regression (PCR) principal components of the explanatory variables are used as predictors instead of the actual explanatory variables, as in the OLS method. This is due to an assumption that a small number of principal components suffice to explain a lot of the variance in the data, in addition to the relationship with the response. If this assumption holds, then fitting a least squares model to $Z_1, \cdots, Zm$ (linear combinations of our original p predictors) will lead to superior results than fitting a least squares model to $X_1, \cdots, X_p$, since most or all of the information in the data that relates to the response is contained in $Z_1, \cdots, Zm$. By estimating only $M < p$ coefficients we can diminish overfitting. It follows a dimension reduction technique that combine features rather than a feature selection method.

First, we divide the data set into a training set, that includes 75% of the observations, and a test set, that includes the remaining 25%. We then create the model by using the *pcr* function from the **pls** package and fit the model on the training set. The argument *scale* standardizes the data so that all the predictors are on the same scale. This is important in order to prevent skewing towards dominant (in absolute scale) predictors even if they are not the most relevant ones. Hence, predictors with high variance will influence the calculation of the principal components more.

Further, we examine the fitted model to see how many components that leads to the lowest mean square error of prediction (*MSEP*) and use that number of components when we predict on the test set. As previously mentioned, the valuation metrics are fluctuating, depending on the set seed. To resolve this, we use the same technique as with the linear regression model and run the model 10 000 times with different seeds. When doing so, we create a model that always chooses the number of components with the lowest *MSEP* when predicting. After running the model 10 000 times, we yield a RMSE of $397.52. The full results from the PCR-model can be seen in Table 012 and the model can be found in Appendix D.

Table 012: Metrics from PCR

	RMSE	R2	MAE
PCR	397.52	0.35	290.57

In addition to the validation set approach, we use *k-dold CV* to evaluate the PCR-model. By using a 10-fold cross validation Appendix F, we obtain a RMSE of $372.4.

Table 013: Metrics from PCR 10-fold CV

	RMSE	R2	MAE
PCR k-fold	372.4	0.38	282.18

When comparing the metrics in the two methods, we see that the *k-fold CV* method yields a lower RMSE, a higher R^2 and a lower MAE, than the *validation set approach*.

Task E

Conclusion

Through this paper, we have examined different prediction methods to predict *salary* in the dataset *ceosal* from the R-package *woolridge*. The methods have included linear regression, best subset, ridge, lasso and PCR. We have used the validation set approach and k-fold cross-validation to evaluate the methods, and used RMSE, R^2 and MAE to evaluate the predictions. The different prediction methods have yielded different results, as shown in 014.

Table 014: Summary of all models

	RMSE	R2	MAE
Regall	375.40	0.36	282.43
BIC	379.99	0.40	280.46
Ridge	388.36	0.39	288.60
PCR k-fold	372.40	0.38	282.18

Note that only the best performing models from each section is included in the table above. From the table we see that the PCR with k-fold cross-validation has the lowest RMSE, but the other metrics are not performing as well as RMSE. For instance, both the BIC model and Ridge model with k-fold cross validation has higher R^2. Furthermore, the BIC model has lower MAE than the PCR model. However, the most desirable prediction model is the one that has the lowest RMSE, and thus this is the model that performs the best among the methods we have used.

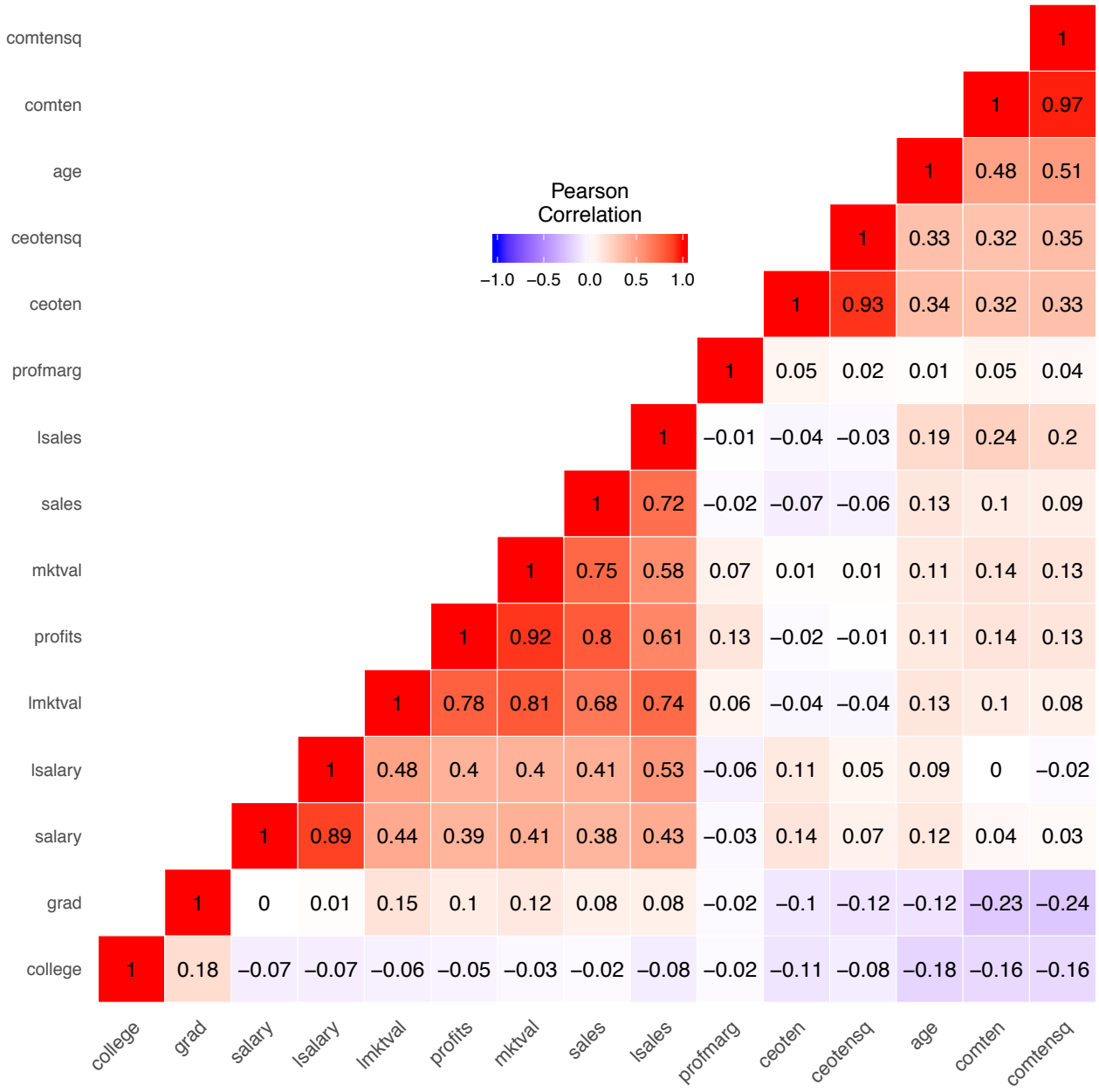
Finally, care should be taken in interpreting the results in this paper. The training and test data are taken as samples from the same dataset, which may be subject to concern.

References

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An introduction to statistical learning – with applications in r* (Vol. 103). New York: Springer. https://doi.org/10.1007/DOI

Appendices

Appendix A: Correlation Matrix



Appendix B: Simple variable selection, linear regression

```
stat_test <- data.frame(matrix(ncol = 4, nrow = 0))
colnames(stat_test) <- c("RMSE", "R2", "MAE", "Pred_error")

columns <- matrix(c(0,0,5,6,13,14,0,8,0,12,0,9,0,7,0,11), nrow = 8, ncol = 2,
```

```
                  byrow = T)

stat_variable_selection <- data.frame(matrix(ncol = 4, nrow = 0))
colnames(stat_variable_selection) <- c("RMSE", "R2", "MAE", "Pred_error")

t <- 10000

for (j in 1:nrow(columns)){
for (i in 1:t) {

#------------------------Split dataset test & train-------------------------

ceo <- ceosal2
# Remove outliers?
ceo <- ceo[-c(103,168),]
ceo <- subset(ceo, select = -c(lsalary,columns[j,]))

# Shuffle the dataset; build train and test
#set.seed(100)
n <- nrow(ceo)
shuffled <- ceo[sample(n),]
train <- shuffled[1:round(0.75 * n),]
test <- shuffled[(round(0.75 * n) + 1):n,]
#rm(list="shuffled")

#test_y <- test$salary
#test <- test[,-1]

#------------------------Modell-------------------------
fmla <- as.formula("salary~.")

mod <- lm(fmla, data = train)

#------------------------Prediction-------------------------
test$prediction <- predict(mod, newdata = test)
#test$salary <- test_y

stat_test[i,] <- (data.frame(RMSE = caret::RMSE(test$salary,test$prediction),
                             R2 = caret::R2(test$salary,test$prediction),
                             MAE = caret::MAE(test$salary,test$prediction),
                             Pred_error = caret::RMSE(test$salary,test$prediction)
                             /mean(test$salary)))

stat_variable_selection[j,] <- colMeans(stat_test)

if(i%%100==0){print(paste0(j,"-",(i)/(t*ncol(stat_variable_selection))*100))}
```

```
}
}

rownames(stat_variable_selection) <-
  c("None","comten,ceoten","comtensq,ceotensq",
    "profits","lmktval","mktval","sales","lsales")
```

Appendix C: Subset selection

```
library(leaps)
library(wooldridge)
library(dplyr)

ceosal2 = ceosal2
ceosal2 = select(ceosal2,-"lsalary")
ceosal2 = ceosal2[-103, ]
ceosal2 = ceosal2[-168, ]


# Best Subset
regfit.full = regsubsets(salary ~ .,
                         data = ceosal2, nvmax = 13)

reg.summary = summary(regfit.full)
save(reg.summary, file = "reg_summary.RData")

par(mfrow = c(2, 2))
plot(reg.summary$rss,
xlab = "Number of Variables",
ylab = "RSS",
type = "l")

plot(reg.summary$adjr2,
xlab = "Number of Variables",
ylab = "Adjusted RSq",
type = "l")
which.max(reg.summary$adjr2)
points(8,
reg.summary$adjr2[8],
col = "red",
cex = 2,
pch = 20)

plot(reg.summary$cp,
xlab = "Number of Variables",
ylab = "Cp",
```

```
type = "l")
which.min(reg.summary$cp)
points(5,
reg.summary$cp[5],
col = "red",
cex = 2,
pch = 20)
which.min(reg.summary$bic)
plot(reg.summary$bic,
xlab = "Numer of Varibles",
ylab = "BiC",
type = "l")
points(4,
reg.summary$bic[4],
col = "red",
cex = 2,
pch = 20)

coef(regfit.full, 8)
coef(regfit.full, 5)
coef(regfit.full, 4)

# Adjusted Rsq
library(caret)
stat_adjR = data.frame(matrix(ncol = 4, nrow = 0))
colnames(stat_adjR) = c("RMSE", "R2", "MAE", "Pred_error")

t = 10000
for (i in 1:t) {
n <- nrow(ceosal2)
shuffled <- ceosal2[sample(n), ]
train <- shuffled[1:round(0.75 * n), ]
test <- shuffled[(round(0.75 * n) + 1):n, ]

regfit.adjR = lm(salary ~ grad + ceoten + mktval +
lsales + comtensq + ceotensq +
profmarg,
data = train)
pred.adjR = data.frame(predict(regfit.adjR, newdata = test))

stat_adjR[i, ] <- round(
data.frame(
RMSE = caret::RMSE(test$salary, pred.adjR[, 1]),
R2 = caret::R2(test$salary, pred.adjR[, 1]),
MAE = caret::MAE(test$salary, pred.adjR[, 1]),
Pred_error = caret::RMSE(test$salary, pred.adjR[, 1]) /
  mean(test$salary)
```

```
),
digits = 2
)
print(i / t * 100)
}

Adj_Rsq = round(colMeans(stat_adjR), digits = 3)

# Cp
stat_cp <- data.frame(matrix(ncol = 4, nrow = 0))
colnames(stat_cp) <- c("RMSE", "R2", "MAE", "Pred_error")

t = 10000
for (i in 1:t) {
n <- nrow(ceosal2)
shuffled <- ceosal2[sample(n), ]
train <- shuffled[1:round(0.75 * n), ]
test <- shuffled[(round(0.75 * n) + 1):n, ]

regfit.cp = lm(salary ~ ceoten + mktval +
                 lsales + comtensq + ceotensq, data = train)
pred.cp = data.frame(predict(regfit.cp, newdata = test))

stat_cp[i, ] <-
round(
data.frame(
RMSE = caret::RMSE(test$salary, pred.cp[, 1]),
R2 = caret::R2(test$salary, pred.cp[, 1]),
MAE = caret::MAE(test$salary, pred.cp[, 1]),
Pred_error = caret::RMSE(test$salary, pred.cp[, 1]) /
mean(test$salary)
),
digits = 2
)
print(i / t * 100)
}

Cp = round(colMeans(stat_cp), digits = 3)

# BIC
stat_bic <- data.frame(matrix(ncol = 4, nrow = 0))
colnames(stat_bic) <- c("RMSE", "R2", "MAE", "Pred_error")

t = 10000
for (i in 1:t) {
n <- nrow(ceosal2)
shuffled <- ceosal2[sample(n), ]
```

```
train <- shuffled[1:round(0.75 * n), ]
test <- shuffled[(round(0.75 * n) + 1):n, ]

regfit.bic = lm(salary ~ ceoten + mktval +
                  lsales + comtensq, data = train)
pred.bic = data.frame(predict(regfit.bic, newdata = test))

(stat_bic[i, ] <-
round(
data.frame(
RMSE = caret::RMSE(test$salary, pred.bic[, 1]),
R2 = caret::R2(test$salary, pred.bic[, 1]),
MAE = caret::MAE(test$salary, pred.bic[, 1]),
Pred_error = caret::RMSE(test$salary, pred.bic[, 1]) /
mean(test$salary)
),
digits = 2
))

print(i / t * 100)
}

BIC = round(colMeans(stat_bic), digits = 3)

# Result
res1 = data.frame(rbind(Adj_Rsq = Adj_Rsq, Cp = Cp, BIC = BIC))
save(res1, file = "res1.RData")

# Validation Set Approach
stat_val <- data.frame(matrix(ncol = 4, nrow = 0))
colnames(stat_val) <- c("RMSE", "R2", "MAE", "Pred_error")

val.errors2 <- data.frame(matrix(ncol = 13, nrow = 0))

t = 12
for (i in 1:t) {
train = (sample(c(TRUE, FALSE), nrow(ceosal2), rep = TRUE))
test = ceosal2[!train, ]
train = ceosal2[train, ]

regfit.best = regsubsets(salary ~ ., data = train, nvmax = 13)
x.test = model.matrix(salary ~ ., data = test)

val.errors = rep(NA, 13)

for (j in 1:13) {
coefi = coef(regfit.best, id = j)
```

```
pred.val = x.test[, names(coefi)] %*% coefi
val.errors[j] = mean((test$salary - pred.val) ^ 2)

}
val.errors2[i, ] <- val.errors
(stat_val[i, ] <-
round(
data.frame(
RMSE = caret::RMSE(test$salary, pred.val),
R2 = caret::R2(test$salary, pred.val),
MAE = caret::MAE(test$salary, pred.val),
Pred_error = caret::RMSE(test$salary, pred.val) /
mean(test$salary)
),
digits = 2
))
}


Val = round(colMeans(stat_val), digits = 3)
res2 = data.frame(rbind(Val = Val))
save(res2, file = "res2.RData")

number.var1 = data.frame(matrix(ncol = 2, nrow = 12))
for (i in 1:t) {
number.var1[i, ] = data.frame(which(val.errors2 ==
                                      min(val.errors2[i, ]), arr.ind = TRUE))
}
number.var1 = data.frame(mean(number.var1$X2))

# K-fold Cross-Validation
testERR = data.frame(
regall = 0,
regbest = 0,
lasso = 0,
ridge = 0,
pcr = 0
)
stat_kfold <- data.frame(
RMSE = 0,
R2 = 0,
MAE = 0,
Pred_error = 0
)
indbest2 <- data.frame(matrix(ncol = 1, nrow = 0))
best.model2 = data.frame(matrix(ncol = 14, nrow = 0))
```

```
K = 10
n = nrow(ceosal2)
foldsize = floor(n / K)
kk = 1

for (i in 1:K) {
fold = kk:(kk + foldsize - 1)
train = ceosal2[-fold, ]
test = ceosal2[fold, ]
kk = kk + foldsize

best = regsubsets(salary ~ ., data = train, nvmax = 13)
bestsum = summary(best)
indbest = which.max(bestsum$adjr2)
best.model = bestsum$which[indbest, ]
xytrain = train[, best.model]
regbest = lm(salary ~ ., data = xytrain)
predbest = predict(regbest, newdata = test)

indbest2[i, ] = indbest
best.model2[i, ] = best.model
testERR[fold, "regbest"] = as.matrix(test["salary"]) - predbest
(stat_kfold[i, ] = round(
data.frame(
RMSE = caret::RMSE(test[, 1], predbest),
R2 = caret::R2(test[, 1], predbest),
MAE = caret::MAE(test[, 1], predbest),
Pred_error = caret::RMSE(test[, 1], predbest) /
mean(test[, 1])
),
digits = 2
))

}

number.var2 = mean(indbest2[1, ])

models = data.frame(matrix(ncol = 14, nrow = nrow(best.model2)))

for (i in 1:nrow(best.model2)) {
if (sum(as.numeric(best.model2[i, ])) == number.var2) {
models[i, ] <- best.model2[i, ]
}
}
models = na.omit(models)

which.var = t(data.frame(colSums(models)))
```

```
rownames(which.var) = "Number of occurrences"
colnames(which.var) = c("intercept", colnames(ceosal2[2:ncol(ceosal2)]))
save(which.var, file = "which_var.RData")

K_fold = round(colMeans(stat_kfold), digits = 3)
res3 = data.frame(rbind(K_fold = K_fold))
save(res3, file = "res3.RData")

# Result
res4 = data.frame(rbind(
Adj_Rsq = Adj_Rsq,
Cp = Cp,
BIC = BIC,
Val = Val,
K_fold = K_fold
))
```

Appendix D: PCR

```
stat_pcr_loop <- data.frame(matrix(ncol = 4, nrow = 0))

t <- 10000
for (i in 1:t) {
ceo <- ceosal2
ceo <- ceo[-c(103, 168), ]
ceo <- subset(ceo, select = -c(lsalary, profits))

#set.seed(100)
n <- nrow(ceo)
shuffled <- ceo[sample(n), ]
train_pcr <- shuffled[1:round(0.75 * n), ]
test_pcr <- shuffled[(round(0.75 * n) + 1):n, ]
rm(list = "shuffled")

pcr_fmla <- as.formula("salary~.")

test_pcr_y <- test_pcr$salary
pca_test_pcr <- subset(test_pcr, select = -c(salary))


# Create model
tryCatch({
pcr_model <-
pcr(pcr_fmla,
data = train_pcr,
scale = T,
```

```
validation = "CV")
}, error = function(e) {
cat()
})


# NCOMP
hei <- data.frame(RMSEP(pcr_model)[["val"]])
hei2 <- hei[2, ]
ncomp <- which.min(hei2) - 1


#summary(pcr_model)
#validationplot(pcr_model)
#validationplot(pcr_model, val.type= "MSEP")
# predplot(pcr_model)
# coefplot(pcr_model)

pcr_pred <- predict(pcr_model, test_pcr, ncomp = ncomp)
pcr_pred <- data.frame(cbind(pcr_pred, test_pcr_y))
pcr_pred$residuals <- pcr_pred$test_pcr_y - pcr_pred$pcr_pred


stat_pcr_loop[i, ] <- round(
data.frame(
RMSE =
caret::RMSE(pcr_pred$pcr_pred, pcr_pred$test_pcr_y),
R2 = caret::R2(pcr_pred$pcr_pred, pcr_pred$test_pcr_y),
MAE = caret::MAE(pcr_pred$pcr_pred, pcr_pred$test_pcr_y),
Pred_error = caret::RMSE(pcr_pred$pcr_pred, pcr_pred$test_pcr_y) /
mean(pcr_pred$test_pcr_y)
),
digits = 2
)


if (i %% 100 == 0) {
print(i / t * 100)
}

}

stat_pcr <-
t(data.frame(round(colMeans(stat_pcr_loop), digits = 3)))
rownames(stat_pcr) = "PCR"
colnames(stat_pcr) <- c("RMSE", "R2", "MAE", "Pred_error")
```

Appendix E: Lasso and Ridge

```
#------ FOR LOOP --------#

ridge_stats_ncomponents <-
  setNames(data.frame(matrix(ncol = 4, nrow = 0)),
           c("RMSE", "R2", "MAE", "Prediction error"))
lasso_stats_ncomponents <-
  setNames(data.frame(matrix(ncol = 4, nrow = 0)),
           c("RMSE", "R2", "MAE", "Prediction error"))

save(ridge_stats_ncomponents, file = "ridge_stats.Rdata")
save(lasso_stats_ncomponents, file = "lasso_stats.Rdata")

ridge <- data_frame(colMeans(ridge_stats_ncomponents))
lasoo <- colMeans(lasso_stats_ncomponents)

t = 10000


for (i in 1:t) {
#------ DATA PREPERATION ---#

  data("ceosal2", package = "wooldridge")

  ceosal2 <- ceosal2[-c(103, 168), ]
  ceosal2 <- subset(ceosal2, select = -c(lsalary))

  train = ceosal2 %>%
    sample_frac(0.75)

  test = ceosal2 %>%
    setdiff(train)

  x_train = model.matrix(salary ~ ., train)[, -1]
  x_test = model.matrix(salary ~ ., test)[, -1]

  y_train = train %>%
    dplyr::select(salary) %>%
    unlist() %>%
    as.numeric()

  y_test = test %>%
    dplyr::select(salary) %>%
    unlist() %>%
    as.numeric()
```

```r
#------ RIDGE ---------#



grid = 10 ^ seq(10, -2, length = 100)

cv.out = cv.glmnet(x_train, y_train, alpha = 0)
bestlam = cv.out$lambda.min

ridge_mod = glmnet( x_train,
                    y_train,
                    alpha = 0,
                    lambda = grid,
                    thresh = 1e-12)

ridge_pred = predict(ridge_mod, s = bestlam, newx = x_test)


ridge_stats_ncomponents[i, ] <- data.frame(
  RMSE = RMSE(ridge_pred, y_test),
  Rsquare = R2(ridge_pred, y_test),
  MAE = MAE(ridge_pred, y_test),
  PER = RMSE(ridge_pred, y_test) / mean(y_test)
)

coef(ridge_mod, s = bestlam) # Coefficients when lambda


#--- LASSO ----#

lasso_mod = glmnet(x_train,
                   y_train,
                   alpha = 1,
                   lambda = grid)


cv.out = cv.glmnet(x_train, y_train, alpha = 1) #

bestlam = cv.out$lambda.min # Se

lasso_pred = predict(lasso_mod, s = bestlam, newx = x_test)

#comparison of metrics
lasso_stats_ncomponents[i, ] <- data.frame(
  RMSE = RMSE(lasso_pred, y_test),
  Rsquare = R2(lasso_pred, y_test),
  MAE = MAE(lasso_pred, y_test),
```

```
    PER = RMSE(lasso_pred, y_test) / mean(y_test)
  )

  print((i / t) * 100)

}
```

Appendix F: K-fold cross-validation, loop

```
## EVALUATE BY K-FOLD CV
ceo <- ceosal2
ceo <- ceo[-c(103,168),]
ceo <- subset(ceo, select = -c(lsalary))

n=nrow(ceo)
testERR=data.frame(regall=0,regbest=0,lasso=0,ridge=0,pcr=0)
testR2=data.frame(regall=0,regbest=0,lasso=0,ridge=0,pcr=0)
testRMSE=data.frame(regall=0,regbest=0,lasso=0,ridge=0,pcr=0)
testMAE=data.frame(regall=0,regbest=0,lasso=0,ridge=0,pcr=0)
testPred_error=data.frame(regall=0,regbest=0,lasso=0,ridge=0,pcr=0)

X=as.matrix(ceo[,-1])
y=as.matrix(ceo[,1])
K=10
foldsize=floor(n/K)
kk=1
for(i in 1:K)
{
  fold=kk:(kk+foldsize-1)
  train=ceo[-fold,]
  test=ceo[fold,]
  kk=kk+foldsize
  # FULL REGRESSION
  trainlm=ceo[-fold,-8]
  testlm=ceo[fold,-8]
  regall=lm(salary~.,data=train)
  predall=predict(regall,newdata=test)
  # BEST SUBSET REGRESSION
  best=regsubsets(salary~.,data=train,nvmax=13)
  bestsum=summary(best)
  indbest=which.max(bestsum$adjr2)
  best.model=bestsum$which[indbest,]
  xytrain=train[,best.model]
  regbest=lm(salary~.,data=xytrain)
  predbest=predict(regbest,newdata=test)
  # LASSO
```

```
Xtrain=as.matrix(train[,-1])
ytrain=as.matrix(train[,1])
Xtest=as.matrix(test[,-1])
ytest=as.matrix(test[,1])
cvlambda=cv.glmnet(Xtrain,ytrain,alpha=1)
bestlambda=cvlambda$lambda.min
lassobest=glmnet(Xtrain,ytrain,family="gaussian",alpha=1,
                 lambda=bestlambda,standardize=TRUE)
predlasso=predict(lassobest,s=bestlambda,newx=Xtest)
# RIDGE
cvlambda=cv.glmnet(Xtrain,ytrain,alpha=0)
bestlambda=cvlambda$lambda.min
ridgebest=glmnet(Xtrain,ytrain,family="gaussian",alpha=0,
                 lambda=bestlambda,standardize=TRUE)
predridge=predict(ridgebest,s=bestlambda,newx=Xtest)
# PCR
pcr=pcr(salary~.,data=train,scale=TRUE,validation="CV")
hei <- data.frame(RMSEP(pcr)[["val"]])
hei2 <- hei[2,]
ncomp <- which.min(hei2)-1
predpcr=predict(pcr,test,ncomp=ncomp)
testERR[fold,"regbest"]=as.matrix(test["salary"])-predbest
testERR[fold,"regall"]=as.matrix(test["salary"])-predall
testERR[fold,"lasso"]=as.matrix(test["salary"])-predlasso
testERR[fold,"ridge"]=as.matrix(test["salary"])-predridge
testERR[fold,"pcr"]=as.matrix(test["salary"])-predpcr[,,1]

testR2[i,"regbest"]=caret::R2(test[,1],predbest)
testR2[i,"regall"]=caret::R2(test[,1],predall)
testR2[i,"lasso"]=caret::R2(test[,1],predlasso)
testR2[i,"ridge"]=caret::R2(test[,1],predridge)
testR2[i,"pcr"]=caret::R2(test[,1],predpcr)

testRMSE[i,"regbest"]=caret::RMSE(test[,1],predbest)
testRMSE[i,"regall"]=caret::RMSE(test[,1],predall)
testRMSE[i,"lasso"]=caret::RMSE(test[,1],predlasso)
testRMSE[i,"ridge"]=caret::RMSE(test[,1],predridge)
testRMSE[i,"pcr"]=caret::RMSE(test[,1],predpcr)

testMAE[i,"regbest"]=caret::MAE(test[,1],predbest)
testMAE[i,"regall"]=caret::MAE(test[,1],predall)
testMAE[i,"lasso"]=caret::MAE(test[,1],predlasso)
testMAE[i,"ridge"]=caret::MAE(test[,1],predridge)
testMAE[i,"pcr"]=caret::MAE(test[,1],predpcr)

testPred_error[i,"regbest"]=caret::RMSE(test[,1],predbest)/mean(test[,1])
testPred_error[i,"regall"]=caret::RMSE(test[,1],predall)/mean(test[,1])
```

```
  testPred_error[i,"lasso"]=caret::RMSE(test[,1],predlasso)/mean(test[,1])
  testPred_error[i,"ridge"]=caret::RMSE(test[,1],predridge)/mean(test[,1])
  testPred_error[i,"pcr"]=caret::RMSE(test[,1],predpcr)/mean(test[,1])
}

mse=function(x) mean(x^2)
testMSE=sapply(testERR,mse)

(testALL <- round(cbind(data.frame(RMSE = colMeans(testRMSE)),
                        data.frame(R2 = colMeans(testR2)),
                        data.frame(MAE = colMeans(testMAE)),
                        data.frame(Pred_error =
                                     colMeans(testPred_error))),
                  digits = 3))
```