MSc.IDS - Machine Learning I

Report of Group Work 'House Sales in King County, USA'

Stefan Berchtold, Georg Oberer

08.01.2021

Contents

Support Vector Machines

In a fist step id and date are removed from the data set.

```
data <- read.csv("kc_house_data.csv", sep = ",", header = TRUE, colClasses = c("view" = "factor", "wate
df <- data[, c(-1, -2)]
```

In a second step the model is build since we have a regression problem Support Vector Regression is used with a linear kernel.

```
set.seed(134616)
library(e1071)
library(caret)
library(doParallel)
library(ModelMetrics)


indexes <- createDataPartition(df$price, p = .9, list = F)
train <- df[indexes, ]
test <- df[-indexes, ]

svmfit_reg <- svm(price~., data = train, kernel = "linear", cost = 0.1 )
```

In a third step predictions are made and the errors are computed.

```
pred <- predict(svmfit_reg, test)

print(mse(pred, test$price))
```

```
## [1] 35411605942
```

```
print(RMSE(pred, test$price))
```

```
## [1] 188179.7
```

```
rsquared <- cor(pred,test$price)^2
print(rsquared)
```

```
## [1] 0.7072669
```

```
r_squared_adj <- 1-((1-rsquared)*(nrow(df)-1)/(nrow(df)-ncol(df)-1))
print(r_squared_adj)
```

```
## [1] 0.7070094
```

To see if the model can be improved the tune fucntion is used other tries tuning the model indicates that a higher c parameter leads to better results so only the c parameter of the first model is used and higher c parameters namely 1 and 5. In a Support Vector Regression it is also important to tune the epsilon parameter which defines a margin of tolerance where no penalty is given to the errors.

```
#tuned_model <- tune(svm, price~., data = train, kernel = "linear", ranges = list(epsilon = seq(0,1,0.2

#bestmodel <- tuned_model$best.model
#summary(bestmodel)
```

The summary of best model indicates that a model with cost = 5 and epsilon = 0.1 leads to the best result. So in the next model those two parameters are used and predictions are made and the errors are calculated.

```
svmfit_reg1 <- svm(price~., data = train, kernel = "linear", cost = 5, epsilon = 0.1)
```

```
pred1 <- predict(svmfit_reg1, test)

print(mse(pred1, test$price))
```

```
## [1] 35159159030
```

```
print(RMSE(pred1, test$price))
```

```
## [1] 187507.8
```

```
rsquared <- cor(pred1,test$price)^2
print(rsquared)
```

```
## [1] 0.7079777
```

```
r_squared_adj <- 1-((1-rsquared)*(nrow(df)-1)/(nrow(df)-ncol(df)-1))
print(r_squared_adj)
```

```
## [1] 0.7077207
```