Machine Learning Applied to Finance: Interest Rate Curve in Mexico

Jose Velasco

25/11/2020

Machine Learning Applied to Finance: Interest Rate Curve in Mexico

For the final capstone project of the HarvardX - PH125.9x, this paper examines the interest rates market in Mexico. Specifically, it models the 10-year rate in the swap interest rates market. Since different models are used, the accuracy of each model will be assessed using Root Mean Squared Error (RMSE).

1 Introduction

Interest rate markets are important to the economy because they shed light on economic matters. For example, by examining the overnight rate and the monetary policy statements of the different central banks, an idea can be formed about the current economic outlook (inflation and growth) and its expected course.

The economic theory also tell us the importance of the shape of the yield curve (commonly comprised of the overnight rate to the 10-year rate). As the US Central Bank (Federal Reserve or Fed for short) noted https://www.chicagofed.org/publications/chicago-fed-letter/2018/404 the spread between the overnight rate and the long term rates are often regarded as predictors of recessions.

This paper will attempt to generate models from different approaches in order to obtain a sufficiently good prediction (measured using the RMSE loss function) for the 10yr rate.

First, an exploratory analysis will be conducted. Second, a series of models will be derived, starting from simple models (linear approaches) to machine learning ones (PCA, KNN and Random Forest). Third, the results will be presented comparing the different RMSEs. Last, conclusions and final remarks will be presented.

2 Methods and Analysis

2.1 Data Exploration

First of all, the data is loaded from the provided file. The information is extracted from Bloomberg.

```
library("readxl")
library("zoo")
library("forecast")
library("tidyverse")
library("dplyr")
library("TTR")
library("GGally")
library(tidyverse)
```

```r
library(caret)
library(data.table)
library(stringr)
library(xts)
library(TSstudio)
```

```r
#loading the dataset
dat <- read_excel("ratesmx.xlsx", sheet = 1, col_names = FALSE)
dat <- dat[-c(1,2,4,5,6),] #we remove useless rows
colnames(dat) <- dat[1,] #assign names to columns
colnames(dat)[1] <- "date"
names <- colnames(dat) #store names on a separate vector
dat <- dat[-c(1),] #remove names
dat <- as.matrix(dat) #transform to matrix
num_col <- ncol(dat) #extract the number of columns
num_row <- nrow(dat) # extract the number of rows
dat <- mapply(dat, FUN=as.numeric, nrow = length(dat)) #transform to numeric
#using extracted number of rows to preserve dimension
dat <- matrix(dat, ncol=num_col, nrow=num_row) #create a matrix again
#using extracted number of columns and rows to preserve dimension
dat <- data.frame(dat) # as data frame
dat[,1] <- as.Date(dat[,1], origin = "1899-12-30") #transform column one to dates
colnames(dat) <- names #restore names
#now we can work with the data
```

A quick summary of the data reveals the following:

```r
summary(dat)
```

```
##       date                irs3m           irs6m           irs9m      
##  Min.   :2010-01-01   Min.   :3.250   Min.   :3.246   Min.   :3.270  
##  1st Qu.:2012-09-04   1st Qu.:4.173   1st Qu.:4.245   1st Qu.:4.290  
##  Median :2015-05-08   Median :4.855   Median :4.881   Median :4.910  
##  Mean   :2015-05-08   Mean   :5.415   Mean   :5.437   Mean   :5.465  
##  3rd Qu.:2018-01-09   3rd Qu.:7.289   3rd Qu.:7.228   3rd Qu.:7.135  
##  Max.   :2020-09-11   Max.   :8.635   Max.   :8.725   Max.   :8.795  
##      irs1yr          irs2yr          irs3yr          irs4yr     
##  Min.   :3.313   Min.   :3.740   Min.   :4.165   Min.   :4.335  
##  1st Qu.:4.332   1st Qu.:4.460   1st Qu.:4.760   1st Qu.:5.090  
##  Median :4.954   Median :5.115   Median :5.350   Median :5.580  
##  Mean   :5.501   Mean   :5.645   Mean   :5.832   Mean   :6.025  
##  3rd Qu.:7.117   3rd Qu.:6.986   3rd Qu.:6.845   3rd Qu.:6.895  
##  Max.   :8.865   Max.   :8.880   Max.   :8.900   Max.   :8.925  
##      irs5yr          irs7yr         irs10yr        overnight    
##  Min.   :4.459   Min.   :4.810   Min.   :5.120   Min.   :3.000  
##  1st Qu.:5.350   1st Qu.:5.835   1st Qu.:6.221   1st Qu.:3.750  
##  Median :5.800   Median :6.230   Median :6.670   Median :4.500  
##  Mean   :6.212   Mean   :6.550   Mean   :6.864   Mean   :5.083  
##  3rd Qu.:7.135   3rd Qu.:7.360   3rd Qu.:7.550   3rd Qu.:7.000  
##  Max.   :8.975   Max.   :9.095   Max.   :9.285   Max.   :8.250  
##       fed             us10yr           tiie28            mxn       
##  Min.   :0.2500   Min.   :0.5069   Min.   :3.274   Min.   :11.50  
##  1st Qu.:0.2500   1st Qu.:1.8680   1st Qu.:4.066   1st Qu.:12.95  
```

```
##  Median :0.2500   Median :2.2940   Median :4.840   Median :15.35
##  Mean   :0.7265   Mean   :2.2974   Mean   :5.404   Mean   :16.03
##  3rd Qu.:1.2500   3rd Qu.:2.7220   3rd Qu.:7.298   3rd Qu.:18.94
##  Max.   :2.5000   Max.   :3.9859   Max.   :8.600   Max.   :25.36
##       g1yr            g2yr            g3yr            g5yr
##  Min.   :2.985   Min.   :3.217   Min.   :3.860   Min.   :3.954
##  1st Qu.:3.956   1st Qu.:4.175   1st Qu.:4.705   1st Qu.:5.125
##  Median :4.558   Median :4.828   Median :5.167   Median :5.615
##  Mean   :5.189   Mean   :5.370   Mean   :5.683   Mean   :6.013
##  3rd Qu.:6.763   3rd Qu.:6.753   3rd Qu.:6.774   3rd Qu.:6.881
##  Max.   :8.652   Max.   :8.690   Max.   :8.908   Max.   :9.020
##      g10yr           g20yr           g30yr
##  Min.   :4.426   Min.   :4.910   Min.   :5.325
##  1st Qu.:5.963   1st Qu.:6.634   1st Qu.:6.821
##  Median :6.349   Median :7.106   Median :7.338
##  Mean   :6.596   Mean   :7.164   Mean   :7.368
##  3rd Qu.:7.307   3rd Qu.:7.707   3rd Qu.:7.863
##  Max.   :9.257   Max.   :9.793   Max.   :9.838
```

The data has more than 2,500 observations spanning from January 2010 to September 2020 (more than 10 years of daily observations). Furthermore, there are no missing values.

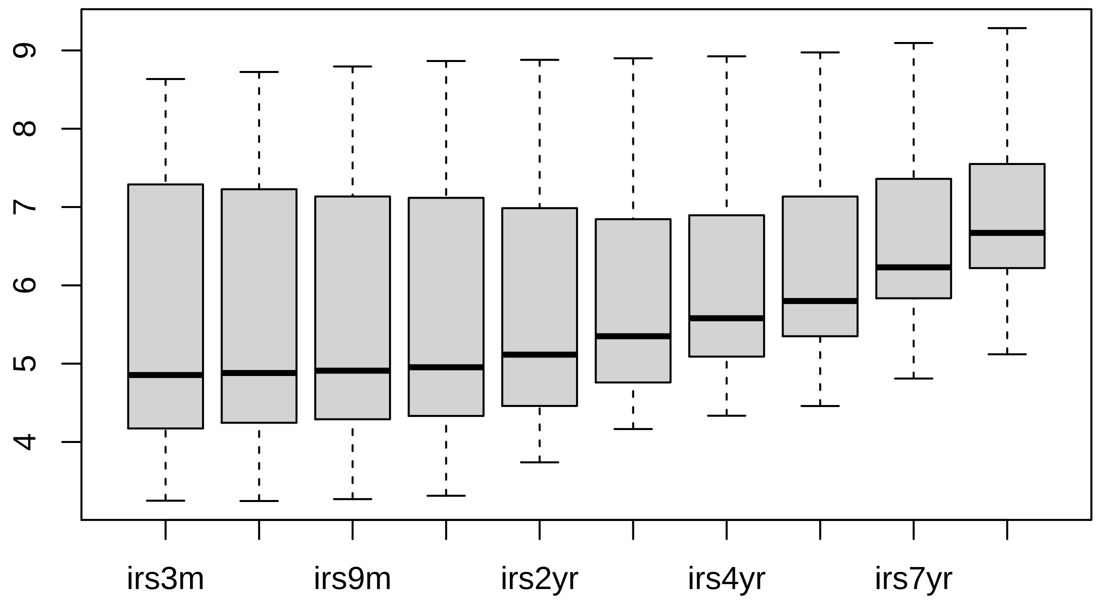
Here is a description of the variables:

- Index: dates
- irs3m: 3 months interest rate swap
- irs6m: 6 months interest rate swap
- irs9m: 9 months interest rate swap
- irs1yr: 1 year interest rate swap
- irs2yr: 2 years interest rate swap
- irs3yr: 3 years interest rate swap
- irs4yr: 4 years interest rate swap
- irs5yr: 5 years interest rate swap
- irs6yr: 7 years interest rate swap
- irs10yr: 10 years interest rate swap
- overnight: official Mexico central bank interest overnight rate (base rate)
- fed: official US central bank interest overnight rate (base rate)
- us10yr: maturity constant 10 years government bond rate
- tiie28: interbank one month rate for Mexico (base rate for swaps)
- mxn: exchange rate between us dollar and mexican peso
- g1yr to g30yr: constant maturity government bond rates for Mexico

The database provides two 10yr rates for Mexico: the IRS (interest rate swaps) and the government 10yr rate. The first one will be chosen for the model because irs rates present some advantages over government bonds. First of all, they are traded with constant maturity. This means that the observed rates are the actual level for a specific date on that instrument. In contrast, government bonds have specific maturity dates, for example, the current 10yr bond (or on the run bond) has a maturity date of May 2029, which means that it cannot be used as the 10yr bond if the analysis starts in 2010, since it would be the 20yr bond. For this reason, bloomberg calculates a theoretical 10yr government bond rate, taking into account the "on the run bond", which changes from time to time. This means that the g1yr to g30yr are theoretical levels for that instruments. Hence, the choice of the IRS over the government bonds.

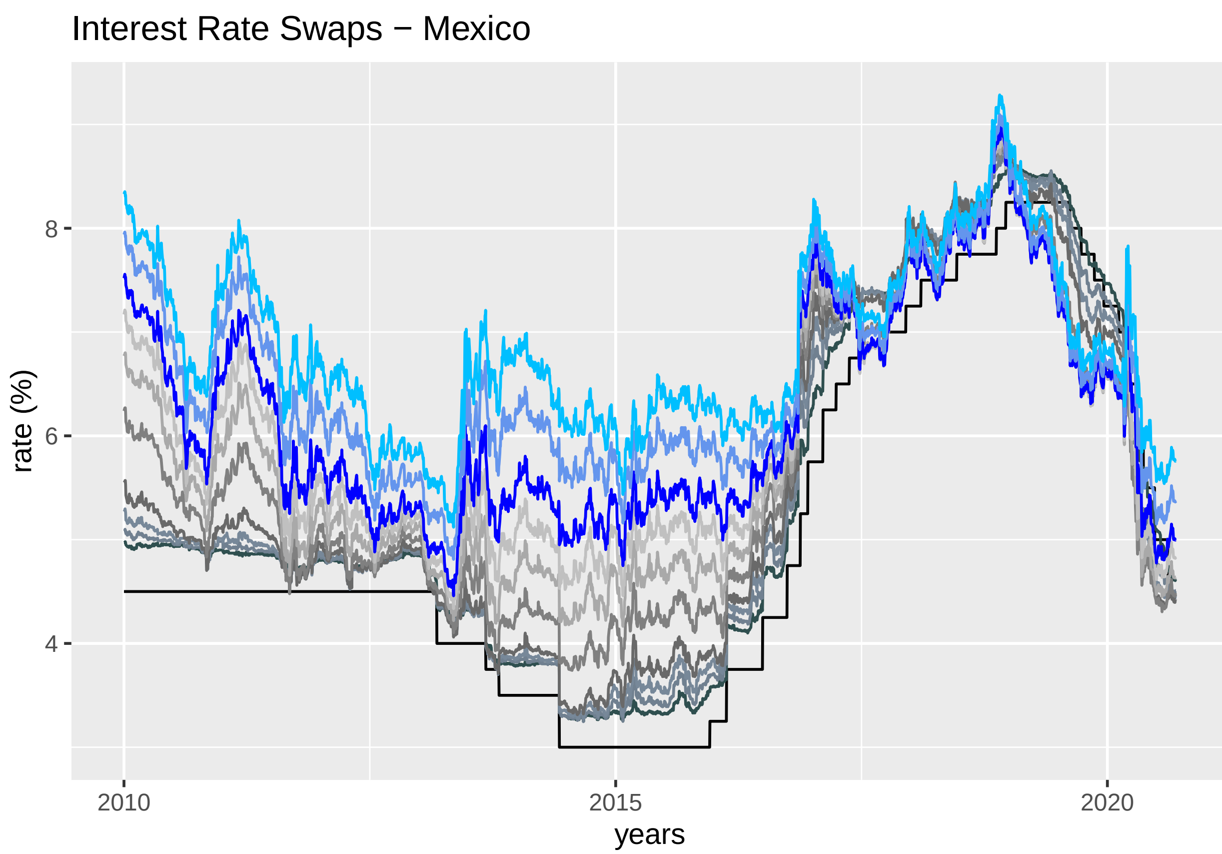
Regarding the IRS curve, we have information for 10 different nodes, spanning from 3 months to 10 years.

```
boxplot((dat[,2:11])) #boxplot of the IRS curve
```



The following visual examination shows the relationship between term and spread. The longer the maturity of the asset, the higher the premium vs the overnight rate. Note that this relation tends to hold under normal conditions. Under stress coditions or tightening monetary policy cycles (i.e. policy rate cuts) as in 2018-2019, the relationship tends to inverse.

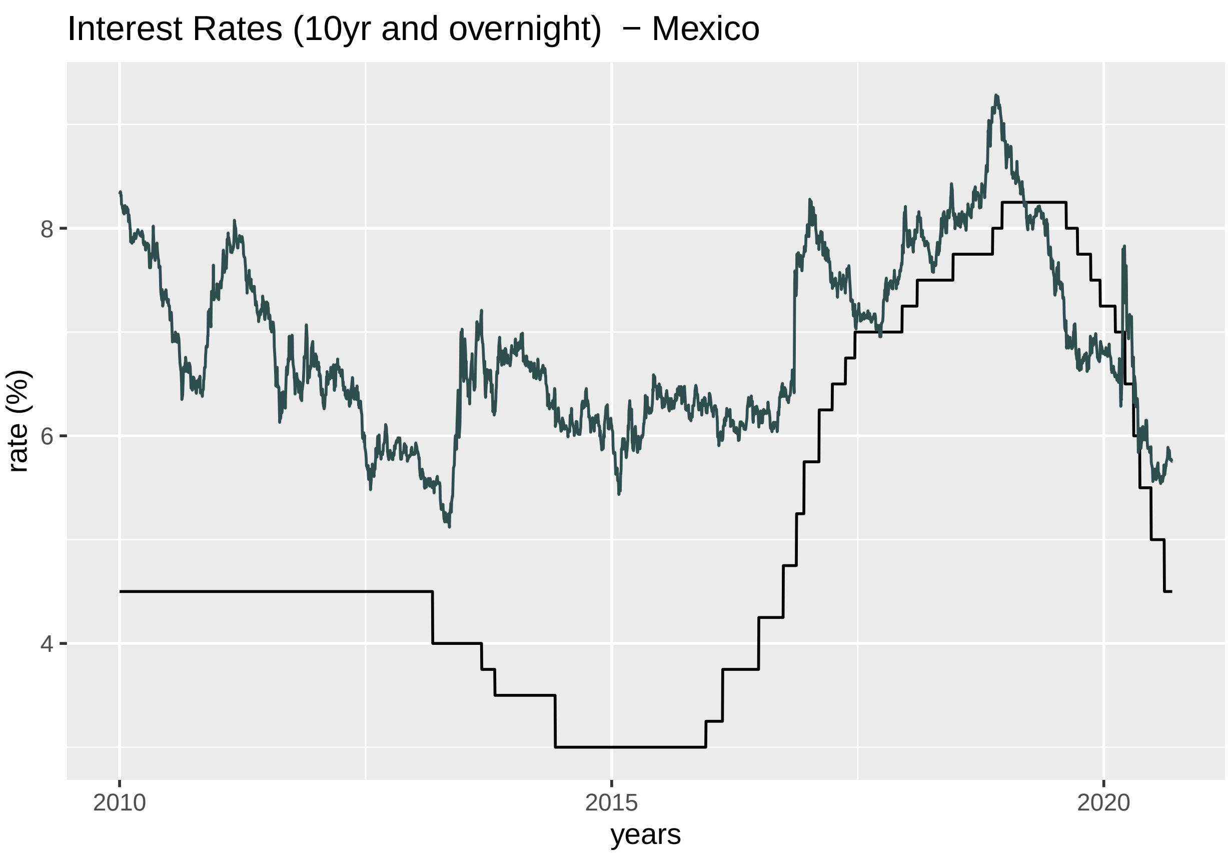
```
#graph of the historical values of the IRS curve
dat %>%
  ggplot(aes(x = date)) +
  geom_line(aes(y = overnight), colour="#000000") +
  geom_line(aes(y = irs3m), colour="#2F4F4F") +
  geom_line(aes(y = irs6m), colour="#708090") +
  geom_line(aes(y = irs9m), colour="#778899") +
  geom_line(aes(y = irs1yr), colour="#696969") +
  geom_line(aes(y = irs2yr), colour="#808080") +
  geom_line(aes(y = irs3yr), colour="#A9A9A9") +
  geom_line(aes(y = irs4yr), colour="#C0C0C0") +
  geom_line(aes(y = irs5yr), colour="#0000FF") +
  geom_line(aes(y = irs7yr), colour="#6495ED") +
  geom_line(aes(y = irs10yr), colour="#00BFFF") +
  xlab("years") + ylab("rate (%)") +
  ggtitle("Interest Rate Swaps - Mexico")
```



In the following graphs, a relationship can be observed between the 10yr IRS and three other variables.

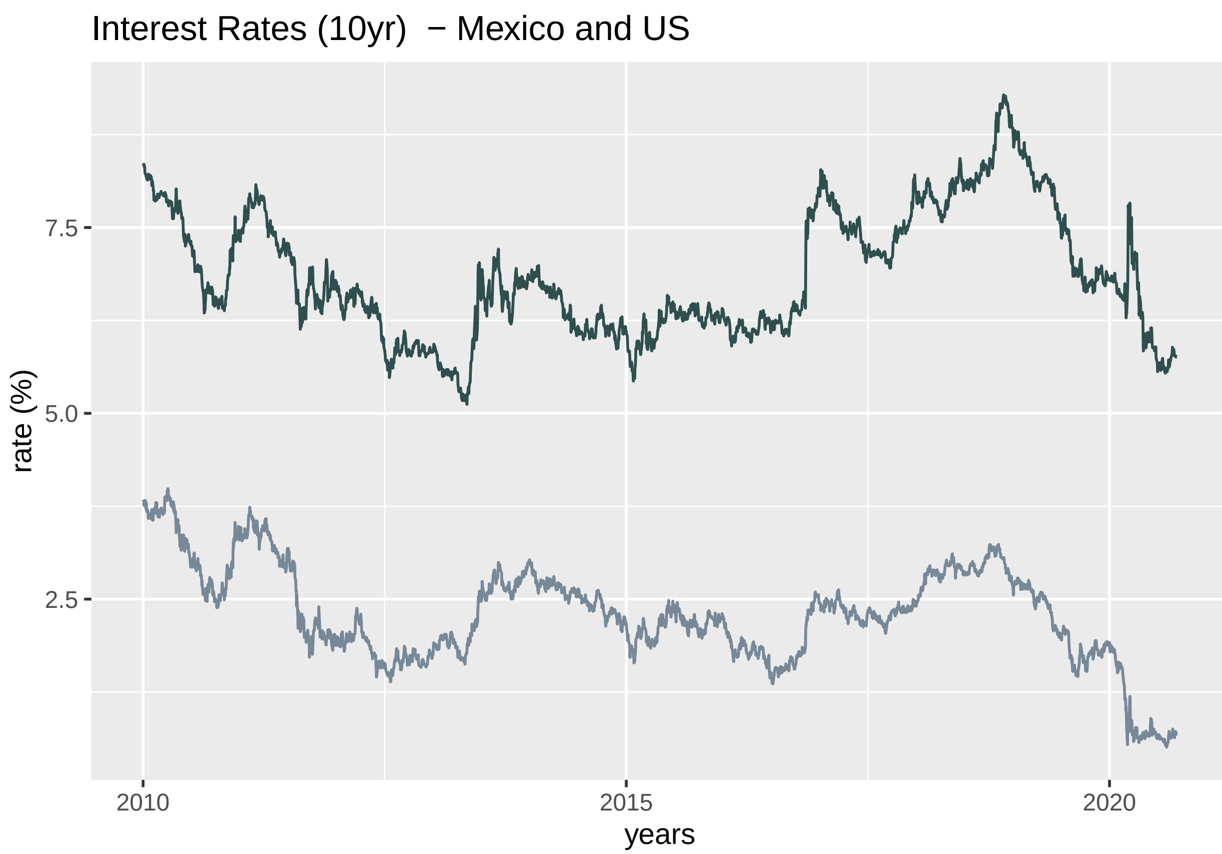
First, it can be noted that the 10yr rate has a relation with the overnight rate. This makes sense as the one day rate is the base rate for the other ones as it is the one set by policy-makers.

```
#graph of the historical values of the overnight rate and the 10yr swap
dat %>%
  ggplot(aes(x = date)) +
  geom_line(aes(y = overnight), colour="#000000") +
  geom_line(aes(y = irs10yr), colour="#2F4F4F") +
  xlab("years") + ylab("rate (%)") +
  ggtitle("Interest Rates (10yr and overnight)  - Mexico")
```



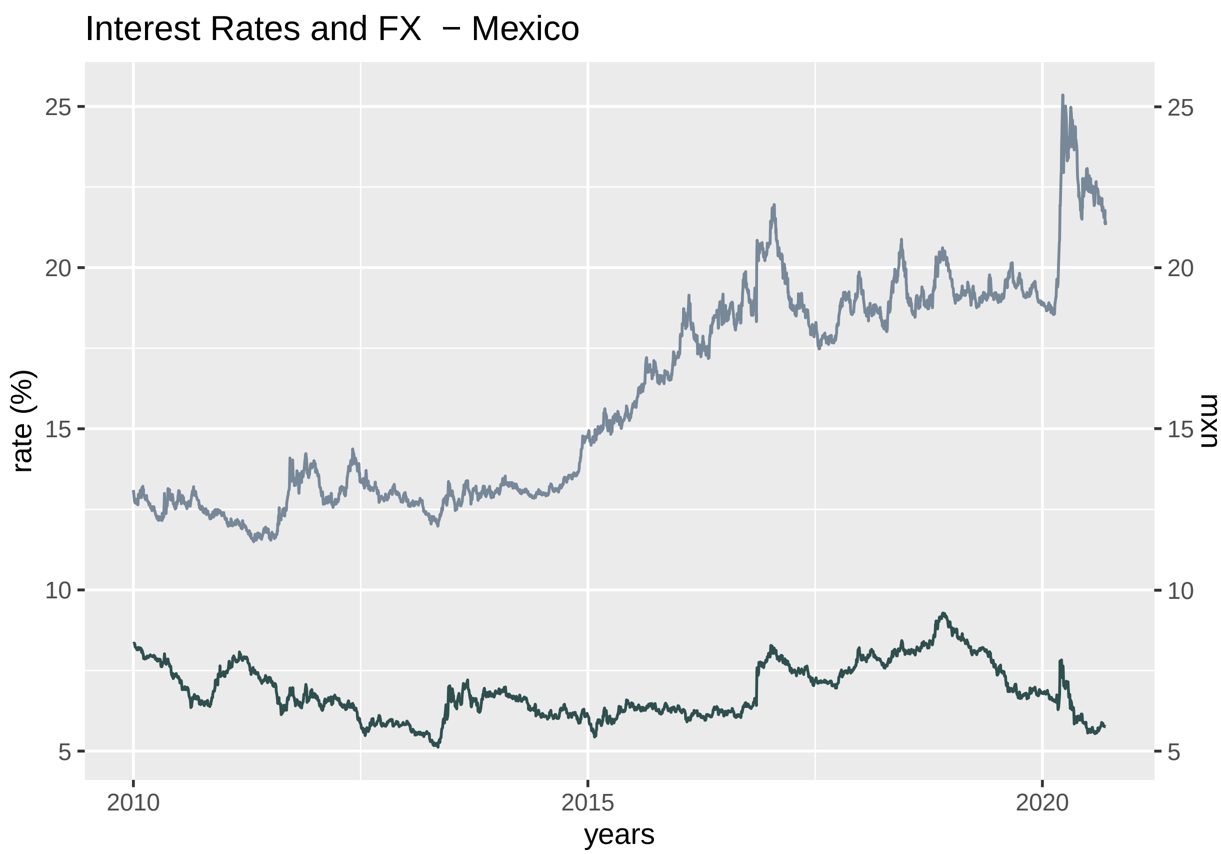
Second, a relationship appears between 10yrs from Mexico and US. Mexico's economy is dependent on US economy and so is the monetary policy. The Bank of Mexico watches closely US rates (and also inflation, growth, and other factors) in order to set the monetary policy rate.

```
dat %>%
  ggplot(aes(x = date)) +
  geom_line(aes(y = irs10yr), colour="#2F4F4F") +
  geom_line(aes(y = us10yr), colour="#778899") +
  xlab("years") + ylab("rate (%)") +
  ggtitle("Interest Rates (10yr)  - Mexico and US")
```



Finally, a positive relation appears between the 10yr rate and the foreign exchange rate (USDMXN). Both variables are often regarded as a measure of risk for the Mexican assets. When the MXN appreciates against the dollar, it goes down in the graph. The same is true for the interest rate.

```
dat %>%
  ggplot(aes(x = date)) +
  geom_line(aes(y = irs10yr), colour="#2F4F4F") +
  geom_line(aes(y = mxn), colour="#778899") +
  xlab("years") +
  ggtitle("Interest Rates and FX  - Mexico") +
  scale_y_continuous(
    name = "rate (%)",
    sec.axis = sec_axis(trans=~., name ="mxn") #for the secondary axis
  )
```



These relationships will be useful later when the models are derived.

2.2 Data Preparation

Now, the dataset is partitioned in training and dataset. The first one will comprise 80% of the dataset.

```
#we will create a data partition in training and testing sets
#that is suitable for time series analysis, the train set comprises
#the first 80% of the data time ordered

set.seed(31416, sample.kind = "Rounding")
test_index <- createDataPartition(dat$irs3m, times = 1, p = 0.2, list = FALSE) # create a 20% test set
test_set <- dat[test_index,]
train_set <- dat[-test_index,]
```

2.3 Model Evaluation

The model performance is going to be evaluated through the RMSE (residual mean square error). The function is defined as follows:

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{t}\left(\hat{y}_t - y_t\right)^2}$$

Let y_t be defined as the rate at time t and denote our prediction with $\hat{y}_t$, with N being the number of time/node combinations and the sum occurring over all the series.

The RMSE is similar to a standard deviation: it is the typical error that is made when predicting a given node for a specific date. It translates in basis points (bps). For example, if the RMSE is 0.10, it means that the error made is from 10bps, which means an estimate of 5.50 instead of the true value of 5.60.

2.4 Base Model

After analysing the data, it is possible to define a base model. The one that will be the benchmark for the other, more complex, models.

Since we explored the relation between 10yr rates and overnight rate. A model which considers this plus a constant spread will be proposed. In this manner, the relationship could be explained with nothing more than a base rate, a constant.

Thus, this relationship can be modelled as follows:

$$\hat{y}_t = \beta_0 + \beta_{1,t} x_{1,t} + \varepsilon_t$$

Let $\hat{y}_t$ be our estimate of the 10yr irs rate based on a constant β_0, a linear term where $\beta_{1,t}$ is the effect derived from the overnight rate and ε_t is the error term due to randomness.

```
#base model
set.seed(1, sample.kind = "Rounding")
train_lm_1 <- train(irs10yr ~ overnight
                    , method = "lm", data = train_set)
#a linear model with only one variable and with intercept using the training set
```

The coefficients of the first model are presented:

```
#we observe the coefficients of the first model
train_lm_1[["finalModel"]][["coefficients"]]
```

```
## (Intercept)   overnight
##   5.1488317   0.3376162
```

The 10yr rate is basically explained by a fixed number and a spread vs the overnight rate.

Now, we extract the RMSE on the train set and observe a rather big error of 0.62. This means that the model does not do a good job predicting the 10yr interest rate. The error would be of 62 basis points (bps), which is huge for interest rates. However, now we have a starting point.

```
train_lm_1[["results"]][["RMSE"]] #displays the RMSE for the triaing set
```

```
## [1] 0.6214588
```

2.5 Economic Model

As we noted above, the 10-year Mexican rate is correlated to the US 10-year rates, to the overnight Mexican rate and to the USDMXN exchange rate. In this section we will derive a model that tries to capture this information:

We do not use the intercept since we are trying to explain the variable using exclusively other variables.

```r
set.seed(1, sample.kind = "Rounding")
#linear model using 4 variables and without intercept
train_lm_2 <- train(irs10yr ~ overnight + fed + us10yr + mxn
                    , method = "lm", data = train_set,
                    tuneGrid  = expand.grid(intercept = FALSE))
```

The coefficients of the second model are presented:

```r
#displays the coefficients of the second model
train_lm_2[["finalModel"]][["coefficients"]]
```

```
##  overnight        fed     us10yr        mxn
##  0.3560172 -0.5173115  1.2518984  0.1586797
```

In this model, the overnight Mexican rate, the 10yr US rate and exchange rate have a positive correlation with the 10yr IRS rate, whereas the Fed funds rate has a negative impact.

The RMSE from the train set is presented: 29bps, this model is twice as good as the base model according to the RMSE.

```r
#get RMSE
train_lm_2[["results"]][["RMSE"]]
```

```
## [1] 0.2919127
```

2.6 Linear Model

In this model, the 10yr IRS rate is estimated using all the data available.

```r
#liner model using all the variables and without intercept
set.seed(1, sample.kind = "Rounding")
train_lm_3 <- train(irs10yr ~ .
                  ,method = "lm", data = train_set,
                  tuneGrid  = expand.grid(intercept = FALSE))
```

The coefficients of the third model are presented:

```r
#this will display the coefficients of all the variables
train_lm_3[["finalModel"]][["coefficients"]]
```

```
##          date         irs3m         irs6m         irs9m        irs1yr
##  2.660671e-07 -2.828782e-02  1.488440e-02  3.453592e-02 -1.418992e-02
##        irs2yr        irs3yr        irs4yr        irs5yr        irs7yr
##  8.534550e-02 -1.376729e-01 -1.026570e-02 -3.747744e-01  1.416160e+00
##     overnight           fed        us10yr        tiie28           mxn
##  4.361767e-03  5.094678e-02  2.462347e-02 -1.518824e-02  6.565777e-03
##          g1yr          g2yr          g3yr          g5yr         g10yr
## -1.085549e-02  2.239708e-02 -4.531583e-02 -4.305660e-02 -4.709839e-02
##         g20yr         g30yr
##  2.765568e-03  1.168624e-01
```

The train set RMSE has a substantial improvement to 2.45bps

```r
#glance at the model
train_lm_3[["results"]][["RMSE"]]
```

```
## [1] 0.02453963
```

2.7 Principal Components Analysis

The fourth model that will be presented is a Principal Component Analysis (PCA) using only the IRS curve. The correlation between all the variables is presented here.

```r
#correlation between all IRS
cor(train_set[,2:11])
```

```
##             irs3m     irs6m     irs9m    irs1yr    irs2yr    irs3yr    irs4yr
## irs3m   1.0000000 0.9973198 0.9920647 0.9837829 0.9535671 0.9182620 0.8825155
## irs6m   0.9973198 1.0000000 0.9985261 0.9939301 0.9706264 0.9392932 0.9057553
## irs9m   0.9920647 0.9985261 1.0000000 0.9983204 0.9808655 0.9531236 0.9218081
## irs1yr  0.9837829 0.9939301 0.9983204 1.0000000 0.9899089 0.9672499 0.9394269
## irs2yr  0.9535671 0.9706264 0.9808655 0.9899089 1.0000000 0.9930330 0.9768448
## irs3yr  0.9182620 0.9392932 0.9531236 0.9672499 0.9930330 1.0000000 0.9949320
## irs4yr  0.8825155 0.9057553 0.9218081 0.9394269 0.9768448 0.9949320 1.0000000
## irs5yr  0.8423381 0.8670265 0.8846882 0.9050067 0.9522917 0.9804330 0.9950531
## irs7yr  0.7740446 0.8000799 0.8194087 0.8427583 0.9016492 0.9425550 0.9701041
## irs10yr 0.7111108 0.7368423 0.7564341 0.7809931 0.8463054 0.8956391 0.9326913
##            irs5yr    irs7yr   irs10yr
## irs3m   0.8423381 0.7740446 0.7111108
## irs6m   0.8670265 0.8000799 0.7368423
## irs9m   0.8846882 0.8194087 0.7564341
## irs1yr  0.9050067 0.8427583 0.7809931
## irs2yr  0.9522917 0.9016492 0.8463054
## irs3yr  0.9804330 0.9425550 0.8956391
## irs4yr  0.9950531 0.9701041 0.9326913
## irs5yr  1.0000000 0.9888837 0.9622021
## irs7yr  0.9888837 1.0000000 0.9915305
## irs10yr 0.9622021 0.9915305 1.0000000
```

As we can see, the correlation between all the IRS is very high. We will limit the model to the IRS curve since an economic explanation can be applied to the first three components.

A dimension reduction algorithm is applied to the data.

```r
#pca
x <- train_set[,2:11] %>% as.matrix() #we set the IRS curve as an x matrix
pca <- prcomp(x)
summary(pca)
```

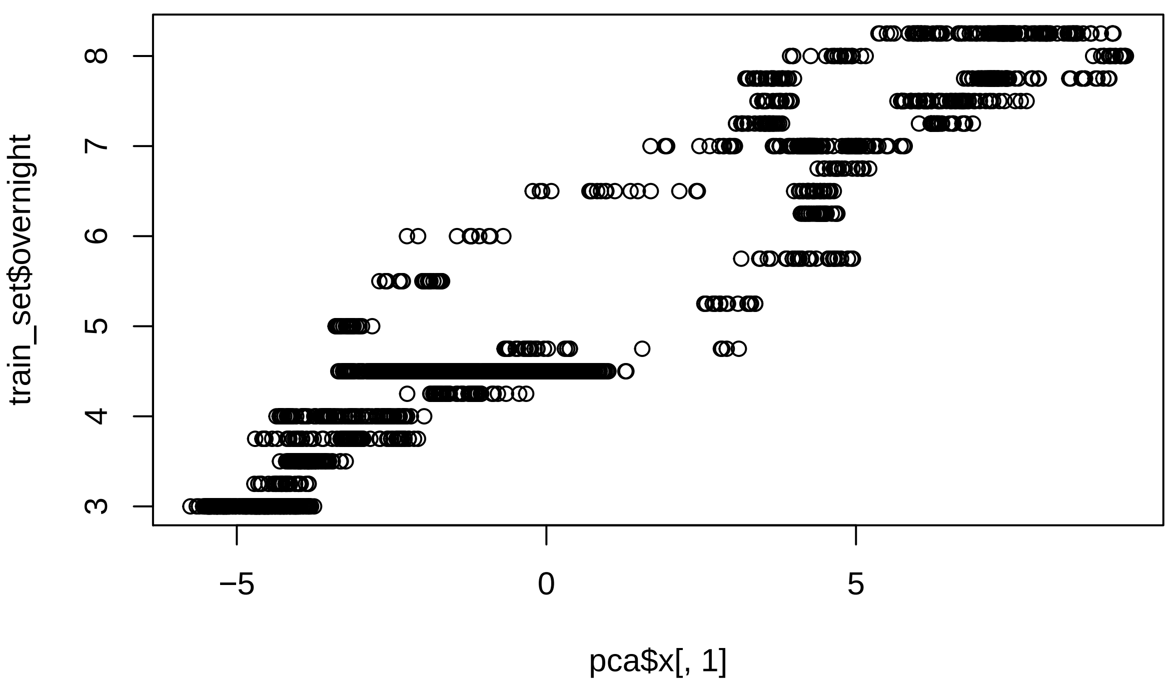
```
## Importance of components:
##                           PC1     PC2     PC3     PC4     PC5     PC6     PC7
## Standard deviation     4.1699 0.95156 0.25453 0.09338 0.02875 0.01776 0.01439
## Proportion of Variance 0.9466 0.04929 0.00353 0.00047 0.00005 0.00002 0.00001
## Cumulative Proportion  0.9466 0.99591 0.99944 0.99991 0.99996 0.99997 0.99998
##                            PC8      PC9     PC10
```

```
## Standard deviation     0.01147 0.009645 0.008379
## Proportion of Variance 0.00001 0.000010 0.000000
## Cumulative Proportion  0.99999 1.000000 1.000000
```

It can be observed that the first three principal components explain 99.944% of the variation of the total data. As supported by the literature,

The PC1 one correspond to the level of the curve, which means that it approximates the funding rate (Banco de Mexico policy rate or overnight rate).

```r
plot(pca$x[,1], train_set$overnight)
```

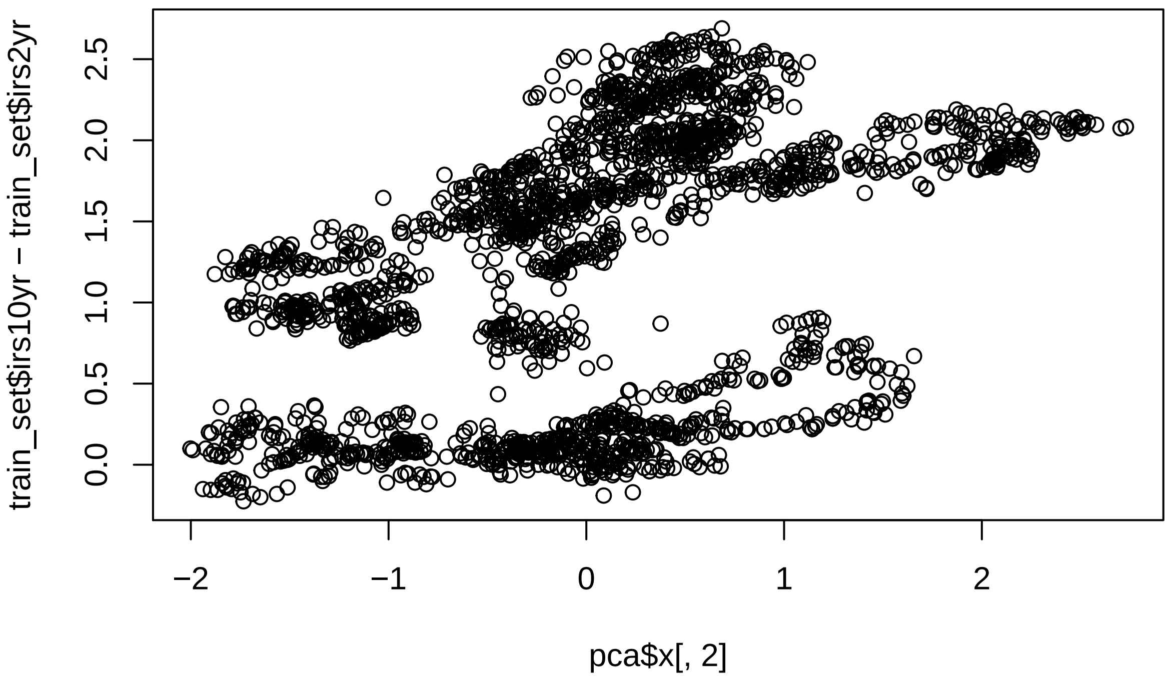



```r
cor(pca$x[,1], train_set$overnight)
```

```
## [1] 0.9545597
```

The PC2 correspond to the slope of the curve. In this case, the slope between the 10yr and 2yr instruments is graphed along with the correlation. It is not as strong as the previous PC1 relation to the overnight rate.

```r
plot(pca$x[,2], train_set$irs10yr - train_set$irs2yr)
```

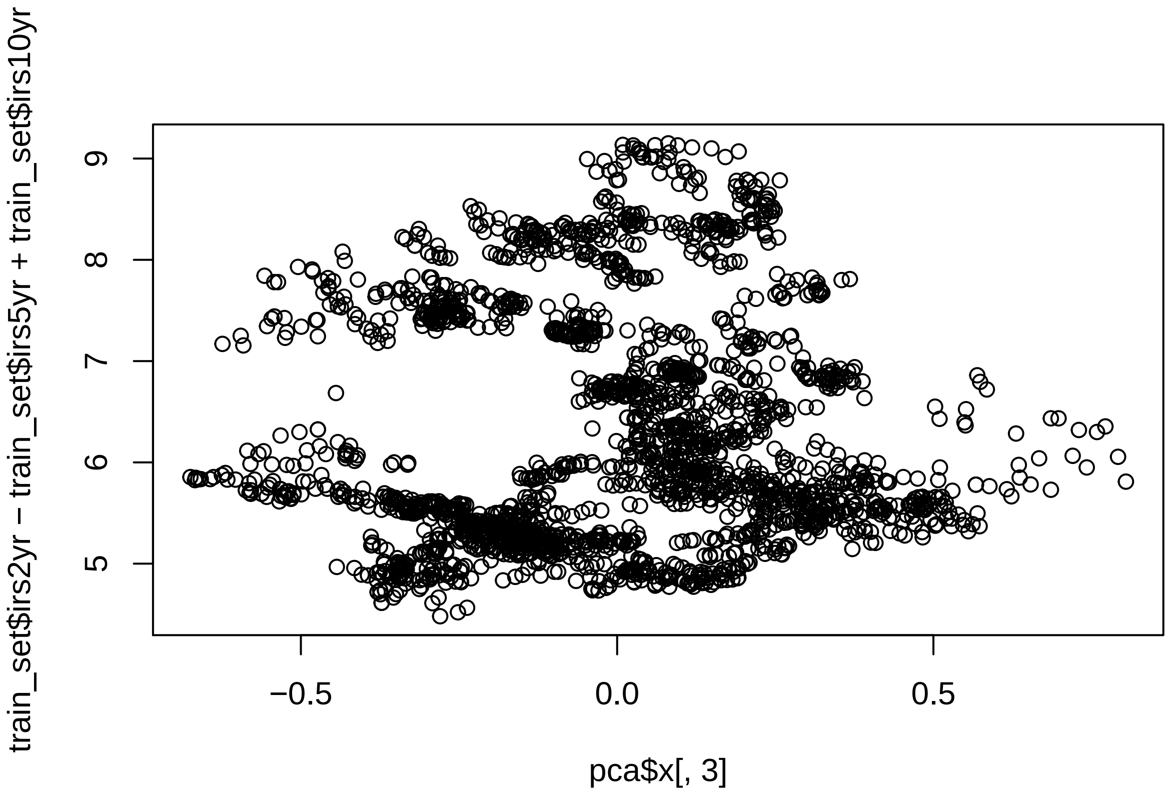



```
cor(pca$x[,2], train_set$irs10yr - train_set$irs2y)
```

```
## [1] 0.4601129
```

The PC3 correspond to the convexity of the curve. This one is traditionally approximated by a strategy called a fly, which implies buying the 2yr and 10yr instruments and selling the 5yr instrument. As the correlation and graph shows, it is not a good approximation of the PC3.

```
plot(pca$x[,3], train_set$irs2yr - train_set$irs5yr + train_set$irs10yr)
```




```r
cor(pca$x[,3], train_set$irs2yr - train_set$irs5yr + train_set$irs10yr)
```

```
## [1] 0.02497903
```

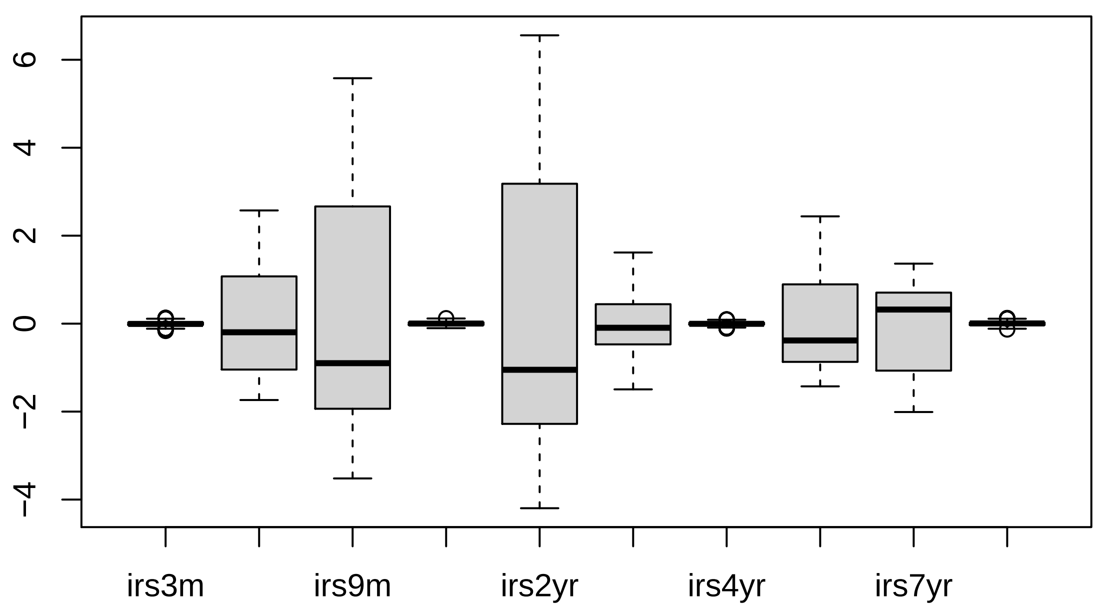
2.7.1 Residuals In order to compare this model to the previous ones, a RMSE should be computed.

```r
# we will extract the pcas and the rotation matrix
pc1to3 <- pca$x[,1:3]
pc_rot <- pca$rotation[,1:3]


#create a function to compute the reconstructed curve with the pca
curvapca <- function(i) {
  rowSums(pc1to3[i,]*(pc_rot))+pca$center
}

# create a matrix of the curve
curva_con_pca <-  data.frame(t(sapply(seq(1:nrow(train_set)), curvapca)))

#then we can compute the squared residuals
pca_rmse <- sqrt(colSums((x - curva_con_pca)^2)/nrow(train_set))
pca_rmse
```

```
##       irs3m       irs6m       irs9m      irs1yr      irs2yr      irs3yr      irs4yr
```

```
## 0.04627066 1.15811065 2.61059092 0.04238561 3.11366575 0.64873808 0.03373044
##     irs5yr     irs7yr    irs10yr
## 1.02295273 1.01195821 0.04163830
```

```r
#now we compute the dispersion of the data
dispersion <- (x - curva_con_pca)

#this show us the residuals
boxplot(dispersion)
```



As can be noted in the boxplot of the residuals and in the computation of the residual mean square error (RMSE), there is large variability in the following nodes: 6m, 9m, 2yr, 5yr and 7yr.

However, the model does provide a good estimate for the 10yr node. This could be explained by various reasons, but one of them is that the 10yr is a benchmark node and hence it is highly liquid in the market, whereas other nodes are not so liquid. The RMSE in the training set is about 4bps.

```r
pca_rmse[10] #display only the residual por the 10yr irs
```

```
##   irs10yr
## 0.0416383
```

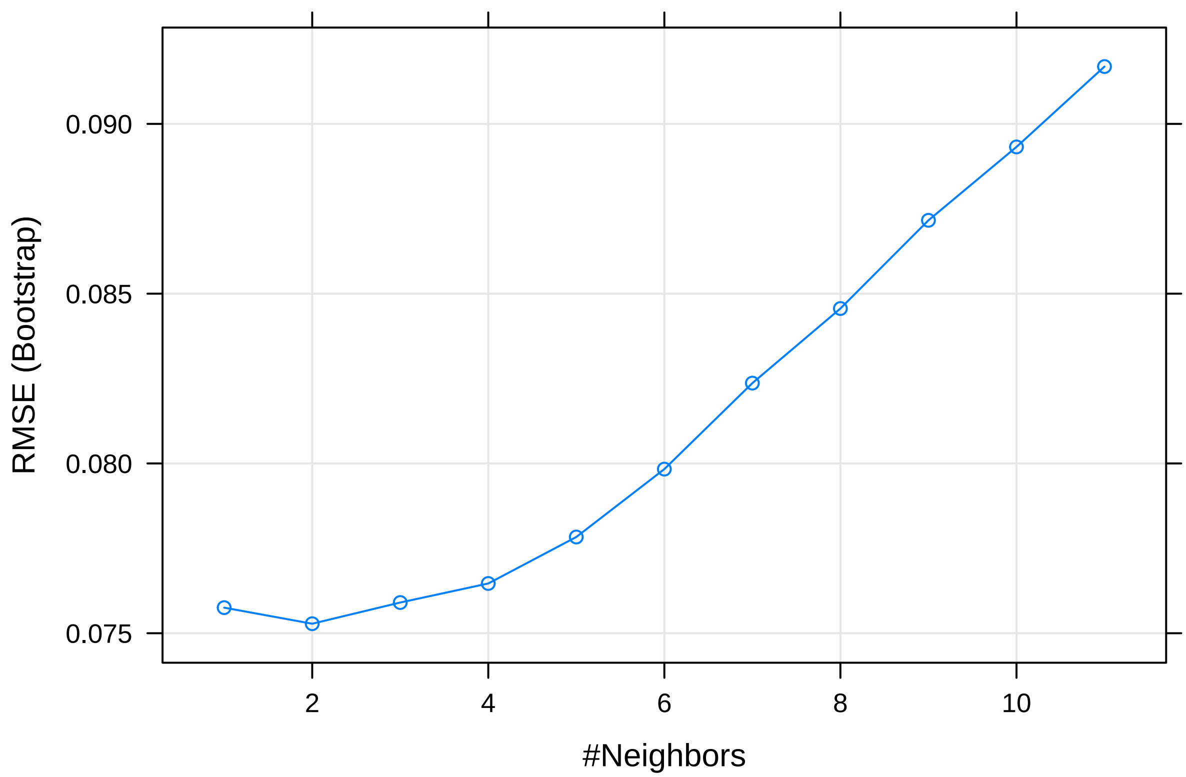
2.8 K Nearest Neighbours

The next model we will implement is a K-Nearest Neighbours approach.

```r
# we will extract the pcas and the rotation matrix
set.seed(1, sample.kind = "Rounding")
train_knn <- train(irs10yr ~ ., method = "knn", data = train_set,
                   tuneGrid = data.frame(k = seq(1, 11 , 1)))
train_knn$bestTune
```

```
##   k
## 2 2
```

```r
plot(train_knn)
```



In order to get the best fit, cross-validation is used to find that when we set k = 2 we minimise our loss function to 7.5bps in the training set.

```r
#we minimise the RMSE
min(train_knn[["results"]][["RMSE"]])
```

```
## [1] 0.07527954
```

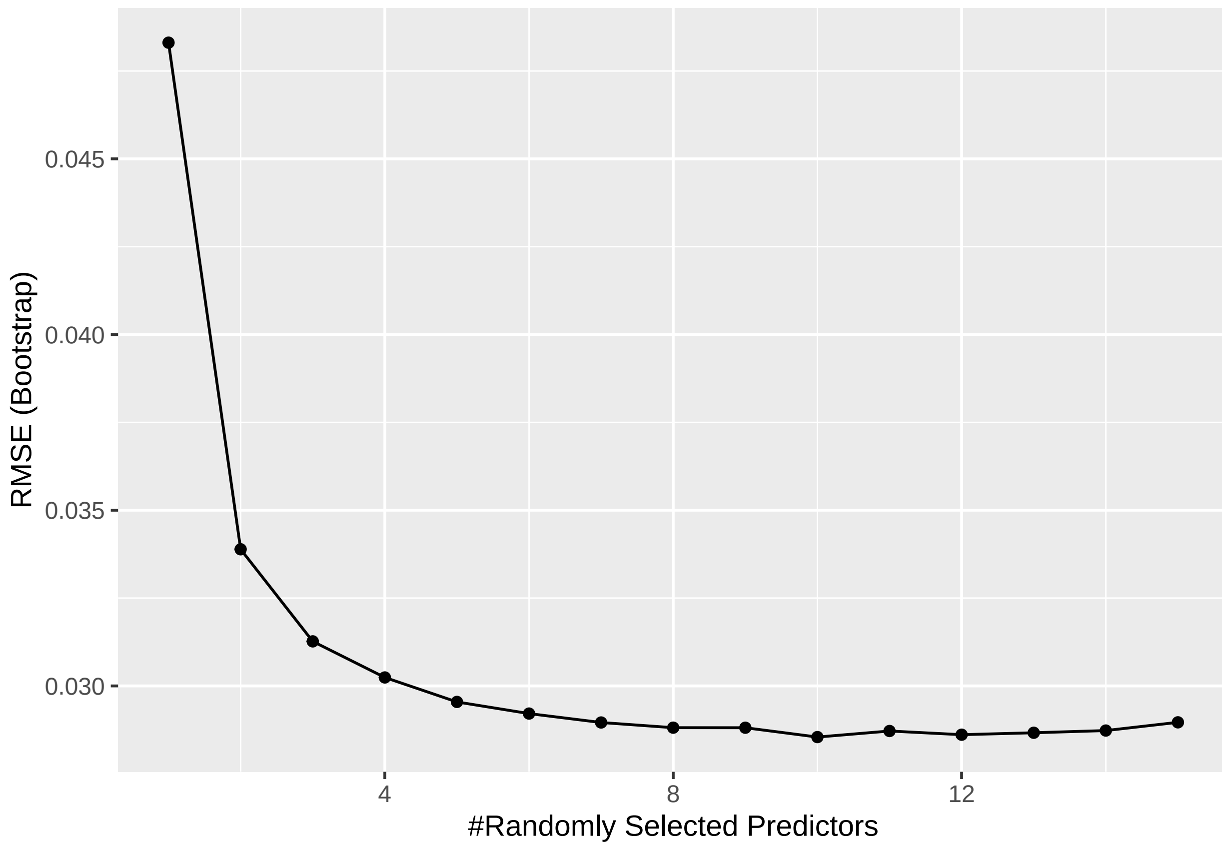
2.9 Random Forest

Finally, a random forest model is implemented.

```r
####random forest######
set.seed(1, sample.kind = "Rounding")
train_rf <- train(irs10yr ~ ., method = "rf", data = train_set,
                  tuneGrid = data.frame(mtry = seq(1, 15, 1)), ntree = 100)
train_rf$bestTune
```

```
##    mtry
## 10   10
```

```r
ggplot(train_rf)
```




```r
#and get the best rmse
min(train_rf[["results"]][["RMSE"]])
```

```
## [1] 0.02854399
```

As can be seen, a mtry = 10 is chosen to minimise the loss function, which yields an RMSE in the training set of 2.85bps.

3 Validation and Results

After carefully estimate the models and calibrate them, it is possible to start the validation process using the test set. In this manner, the models can be compared in term of RMSE and a model can be chosen.

For the first model (baseline model), predictions can be derived from the following manner:

```r
#correlation between all IRS
y_hat_lm_1 <- predict(train_lm_1, test_set, type = "raw")
```

Then, an RMSE is calculated for this baseline model using the test set.

```r
#this would be for the validation
base_model_rmse <- sqrt(sum((test_set$irs10yr - y_hat_lm_1)^2)/nrow(test_set))
base_model_rmse
```

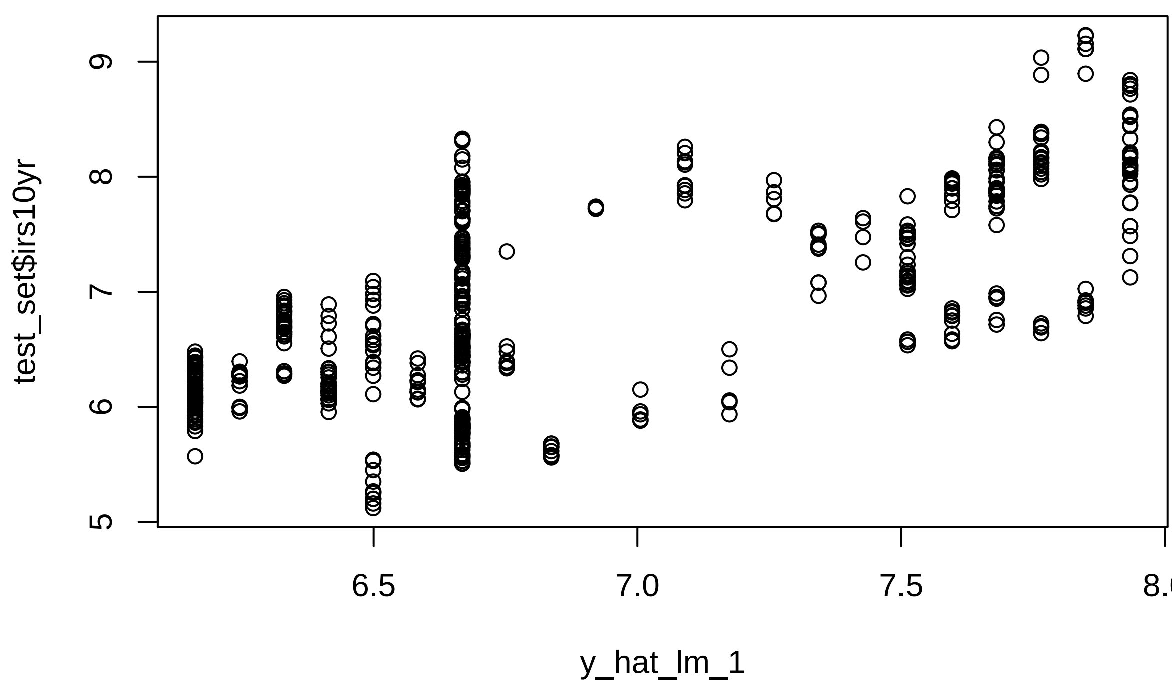
```
## [1] 0.6309902
```

The results are stored for comparison in the next table:

```r
rmse_results <- tibble(method = "Baseline Model", RMSE = base_model_rmse)
```

It can be observed that the model does not do a good job fitting the data:

```r
#this would be for the validation
plot(y_hat_lm_1, test_set$irs10yr)
```

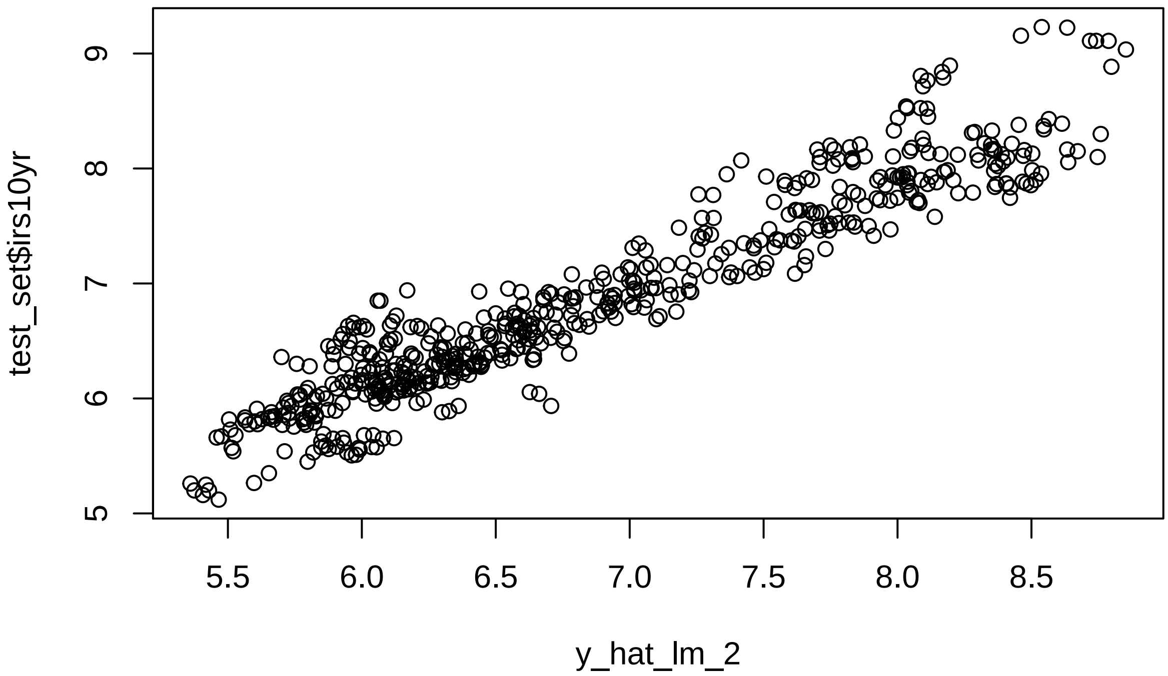


For the second model (economic model) a similar process can be achieved. First, the predictions are estimated using the trained model and then an RMSE is estimated for the test set. A plot is also presented. Now, it can be noted that this model does a better job than the base model.

```
y_hat_lm_2 <- predict(train_lm_2, test_set, type = "raw")
second_model_rmse <- sqrt(sum((test_set$irs10yr - y_hat_lm_2)^2)/nrow(test_set))
second_model_rmse
```

```
## [1] 0.2844804
```

```
plot(y_hat_lm_2, test_set$irs10yr)
```



The results are stored:

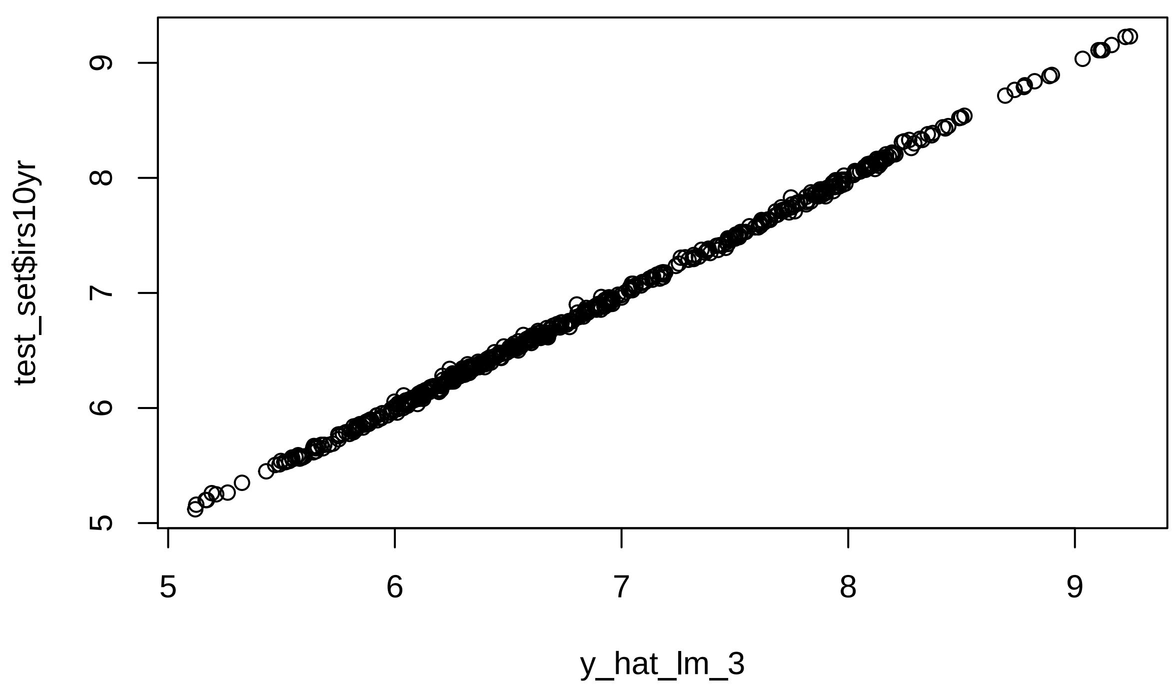
```
rmse_results <- bind_rows(rmse_results,
                          tibble(method="Economic Model", RMSE = second_model_rmse))
```

For the third model (linear model including all variables) a similar process can be achieved. First, the predictions are estimated using the trained model and then an RMSE is estimated for the test set. A plot is also presented.

```
y_hat_lm_3 <- predict(train_lm_3, test_set, type = "raw")
third_model_rmse <- sqrt(sum((test_set$irs10yr - y_hat_lm_3)^2)/nrow(test_set))
third_model_rmse
```

```
## [1] 0.02388543
```

```r
plot(y_hat_lm_3, test_set$irs10yr)
```



The results are stored:

```r
rmse_results <- bind_rows(rmse_results,
                          tibble(method="Linear Model",
                                 RMSE = third_model_rmse))
```

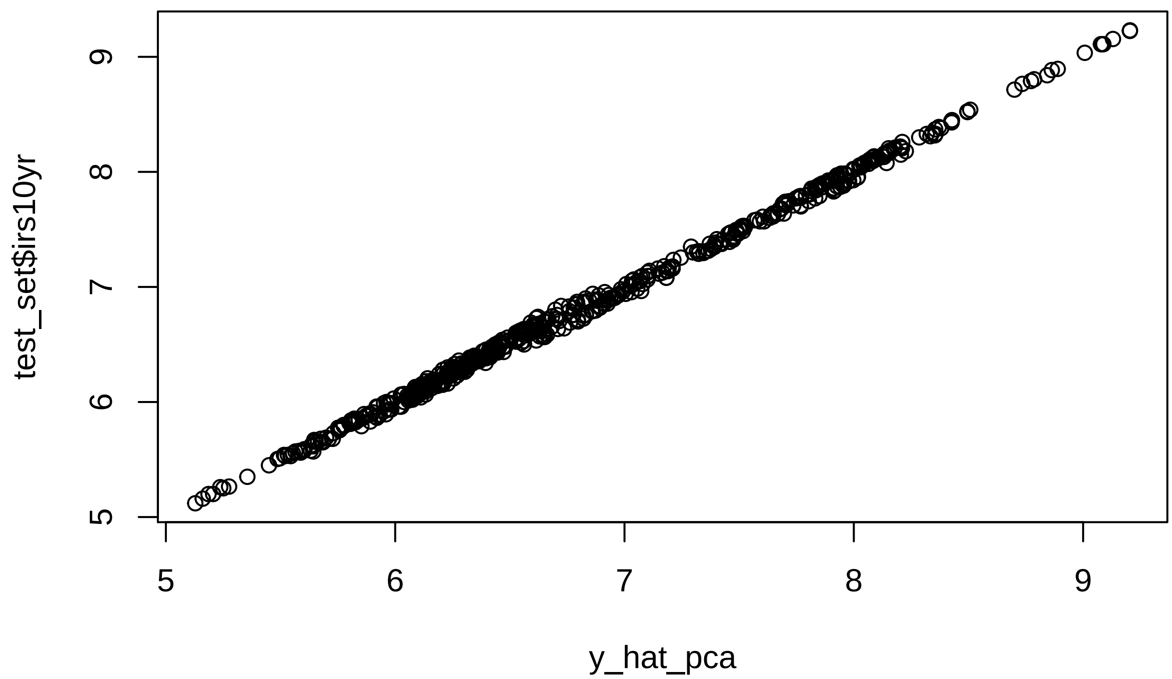
For the fourth model (PCA model) a similar process can be achieved. First, the predictions are estimated using the trained model and then an RMSE is estimated for the test set. A plot is also presented.

```r
#compute pca in test set
x <- test_set[,2:11] %>% as.matrix()
pca <- prcomp(x)

#computing residuals
# we will extract the pcas and the rotation matrix
pc1to3 <- pca$x[,1:3]
pc_rot <- pca$rotation[,1:3]

# create a matrix of the curve
curva_con_pca <-  data.frame(t(sapply(seq(1:nrow(test_set)), curvapca)))

#then we can compute the squared residuals
pca_rmse <- sqrt(colSums((x - curva_con_pca)^2)/nrow(test_set))
```

```
fourth_model_rmse <- pca_rmse[10]
fourth_model_rmse
```

```
##    irs10yr
## 0.03962744
```

```
y_hat_pca <- as.numeric(unlist(curva_con_pca[10]))
plot(y_hat_pca, test_set$irs10yr)
```



The results are stored:

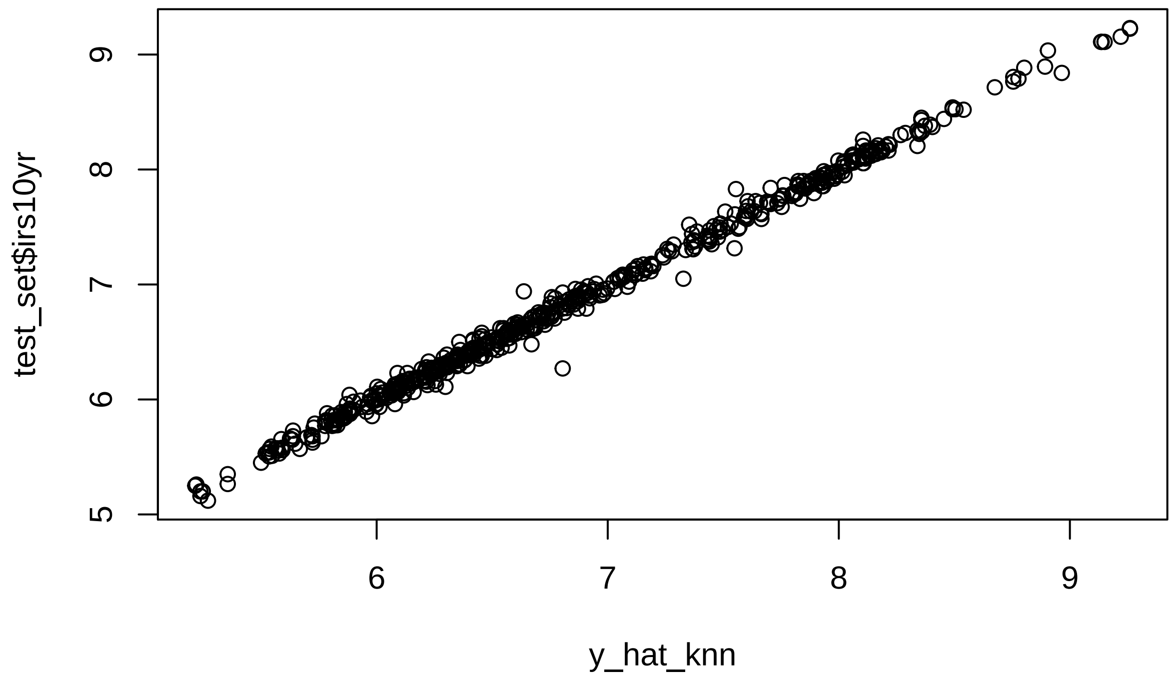
```
rmse_results <- bind_rows(rmse_results,
                          tibble(method="PCA Model",
                                 RMSE = fourth_model_rmse))
```

For the fifth model (KNN) a similar process can be achieved. First, the predictions are estimated using the trained model and then an RMSE is estimated for the test set. A plot is also presented.

```
y_hat_knn <- predict(train_knn, test_set, type = "raw")
fifth_model_rmse <- sqrt(sum((test_set$irs10yr - y_hat_knn)^2)/nrow(test_set))
fifth_model_rmse
```

```
## [1] 0.05899655
```

```r
plot(y_hat_knn, test_set$irs10yr)
```



The results are stored:

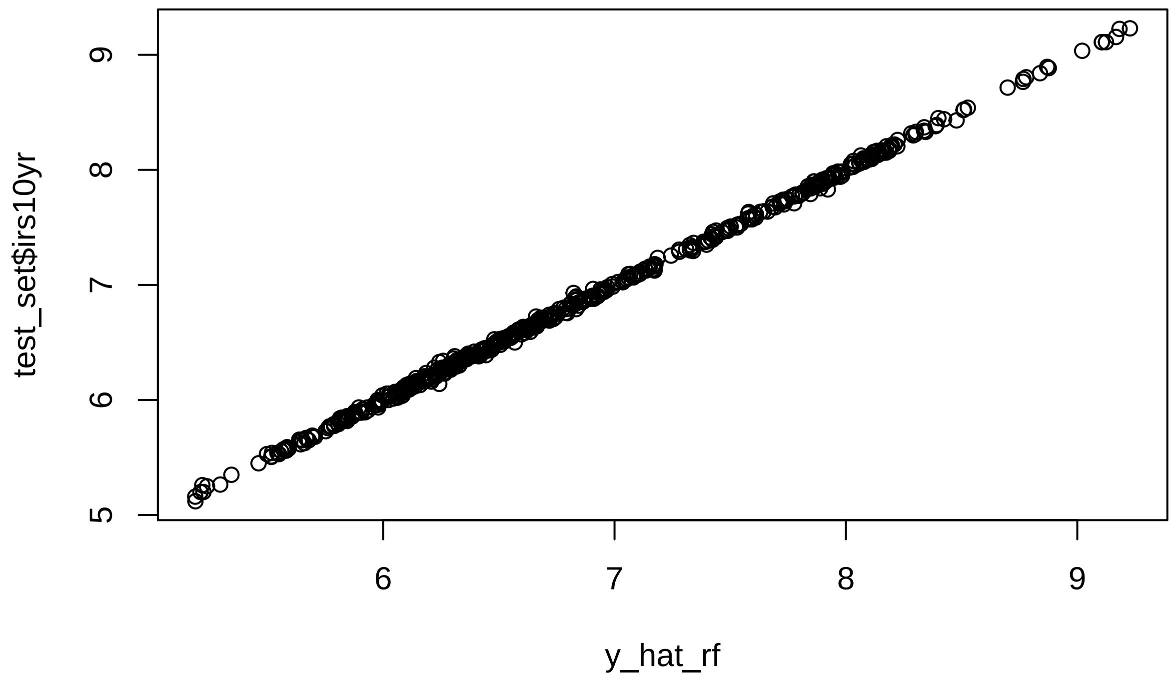
```r
rmse_results <- bind_rows(rmse_results,
                          tibble(method="K Nearest Neighbours Model",
                                 RMSE = fifth_model_rmse))
```

For the sixth model (RF) a similar process can be achieved. First, the predictions are estimated using the trained model and then an RMSE is estimated for the test set. A plot is also presented.

```r
y_hat_rf <- predict(train_rf, test_set, type = "raw")
sixth_model_rmse <- sqrt(sum((test_set$irs10yr - y_hat_rf)^2)/nrow(test_set))
sixth_model_rmse
```

```
## [1] 0.02239061
```

```r
plot(y_hat_rf, test_set$irs10yr)
```



The results are stored:

```
rmse_results <- bind_rows(rmse_results,
                          tibble(method="Random Forest Model",
                                 RMSE = sixth_model_rmse))
```

This is the final summary of the results

```
## # A tibble: 6 x 2
##   method                      RMSE
##   <chr>                      <dbl>
## 1 Baseline Model            0.631
## 2 Economic Model            0.284
## 3 Linear Model              0.0239
## 4 PCA Model                 0.0396
## 5 K Nearest Neighbours Model 0.0590
## 6 Random Forest Model       0.0224
```

method	RMSE
Baseline Model	0.6309902
Economic Model	0.2844804
Linear Model	0.0238854
PCA Model	0.0396274
K Nearest Neighbours Model	0.0589965
Random Forest Model	0.0223906

4 Conclusion

This paper has implemented six different models to estimate the 10 year IRS rate. Three of those models area linear models and three of them are machine learning models.

With respect to the linear models:

- The baseline model performs rather poorly (RMSE: 63bps) but it makes sense to estimate the 10 year rate with nothing but the overnight rate.
- The economic model uses theoretical relationships between interest rates, exchange rate, foreign interest rates and maturity. It yields better results but if implemented it would be used as a long-term model due to its short-term biases that tend to prevail during short term windows (RMSE: 28.4bps).
- The linear model that includes all the variables performs rather good (RMSE:2.4bps). Indeed it is the second best model. However, it requires a lot of variables as inputs. This is actually a problem for all the other models but the PCA. In terms of time of computing it is also efficient and in practice it would make an ideal model for short term estimates.
- The PCA model performs a little worst than the linear model (RMSE: 4bps) but it could be handy in practice as it only requires the IRS curve to provide a good estimate of the 10 year rate.
- The KNN model has an RMSE of 5.9bps. It does not outperform the linear model and it does require all the variables for its estimation. In consequence, it would not be the first choice when estimating rates.
- Finally, the Random Forest model outperforms all the other models with an RMSE of 2.2bps. However its time consuming and it is only slightly better than the linear model. Furthermore, it uses all the variables to achieve this RMSE. In practice it would be better to habe a slighly worst model in terms of RMSE but quite faster, such as the PCA or the linear model.

Even if the Random Forest Model outperforms the other ones, in practice it would be useful to use more cost efficient models such as PCA or linear models. The models developed provide a statistical basis for estimation of the 10 year rate. It could further expand in the direction of a more theoretically sound model of the 10yr rate, including other variables such as inflation, equity index and so on.