Report on forcast

Loading the libraries

```
library(readxl)
library(MASS)
library(caret)
library(rpart)
library(rpart.plot)
library(randomForest)
library(glmnet)
library(xgboost)
```

The data

Cleaning the data

```
df <- read_xlsx("ML_ready/data_ML.xlsx")

main <- read_xls("economy.xls", sheet = '2011-2019 NACE 2')

colnames(main) <- main[3,]

main <- main[4,]
main <- main[,c(16:103)]

df = as.data.frame(df)
rownames(df) = df$months
df$months = NULL

df = df[, colSums(df != 0) > 0]
df = df[, (df[86,] != 0) > 0]

df_sum <- colSums(df)
df_sum <- sort(df_sum,decreasing = T)

top_20 = head(df_sum,n = 20)
top_20_names = colnames(t(as.data.frame(top_20)))
top_20_loc = which(colnames(df) %in% top_20_names)

try <- df[, top_20_loc]
try <- try[-c(1:33),]

main <- main[,c(34:88)]
main <- t(main)

names <- colnames(try)
colnames(try) <- paste0("name",c(1:20))

try <- try[-c(56,57),]
```

```r
try$main <- as.numeric(main)
try$main <- try$main * 100000000 ### Multiply by hundred million
```

Printing the final form of the data

```r
head(try)
```

```
##                name1   name2 name3 name4      name5   name6   name7
## 2014_OCT    811759.0 2244268     0     0 8568607.00 2990509 4130414
## 2014_NOV    835944.9 2249903     0     0 5739873.00 3258572 4121925
## 2014_DEC    959725.1 2229910     0     0 2058646.00 2711211 3549545
## 2015_JAN    849626.8 2156033     0     0   43373.25 4171518 3013833
## 2015_FEB    741612.9 1871933     0     0   16650.15 2855290 3855382
## 2015_MARCH 1161081.3 2114494     0     0       0.00 3693042 3970585
##               name8  name9  name10   name11  name12 name13  name14  name15
## 2014_OCT    2192361 342413 4464967  9847851 3008645      0       0 2180016
## 2014_NOV    2214470 412123 3765709  9464512 2672553      0 1010326 1248139
## 2014_DEC    2371809 901545 4150028 10270711 2662845      0 2853744       0
## 2015_JAN     506449 244103 3801753 10267426 2539542      0 2842016 4019259
## 2015_FEB    1193319 559894 3455077  8458058 3543618      0 2702967 3680741
## 2015_MARCH 1801947 530894 3793357  8802300 2429327      0 2796296 4621621
##            name16  name17 name18 name19  name20         main
## 2014_OCT        0 2363502      0 895053 3184475 1.211289e+13
## 2014_NOV        0 2302617      0 810880 3082643 1.203592e+13
## 2014_DEC        0 2644527      0 919017 3761799 1.276962e+13
## 2015_JAN        0 2550053      0 186685 3744077 8.866020e+12
## 2015_FEB        0 2023233      0 353564 3617373 9.936690e+12
## 2015_MARCH      0 2200895      0 382379 3645492 1.068529e+13
```

Spliting the data into train.*85 and test.*15

```r
index <- sample(1:nrow(try),round(0.85*nrow(try)))

train <- try[index,]
test <- try[-index,]

n <- names(train)
```

Our formula

```r
f <- as.formula(paste("main ~",paste(n[!n %in% "main"], collapse = " + ")))
```

Prediction Models

Linear Regression

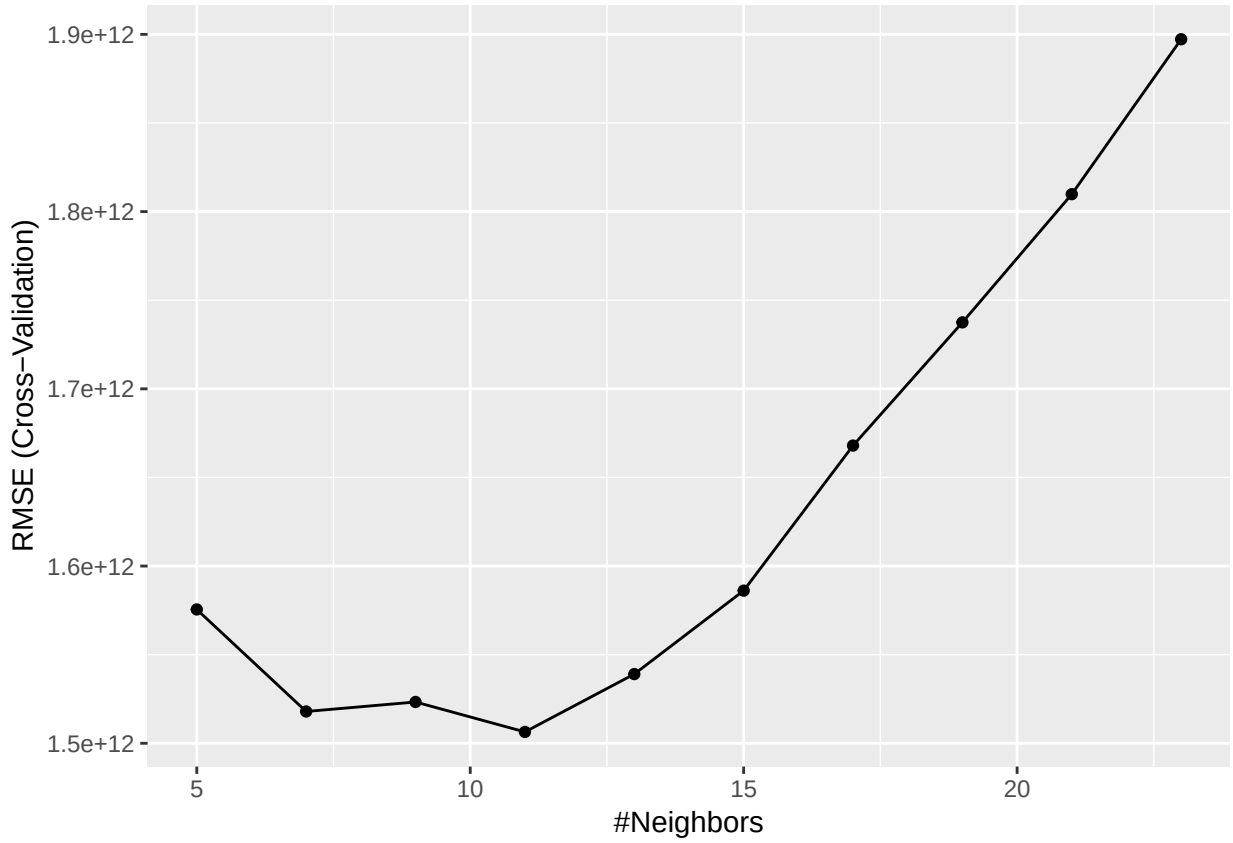
```
fit <- lm(main~., train)

pred1<-predict(fit, newdata = test)

RMSE1<-RMSE(test$main, pred1)

MAE1 <- MAE(test$main, pred1)
```

```
## [1] "RMSE for Linear Regression: 1287777678920.08"
```

```
## [1] "MAE for Linear Regression: 1079859319574.57"
```

K- Nearest Neighbors

```
ctrl <- trainControl(method = "cv", number = 10)

knn_c <- train(f, data = train, method = "knn",
               trControl = ctrl, preProcess = c("center", "scale"), tuneLength = 10)
```



```
model_predict_test = predict(knn_c, newdata = test)
which.min(c(sqrt(mean(abs(model_predict_test - test$main)^2)),sqrt(mean((test$main- predict(fit, test))
```

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

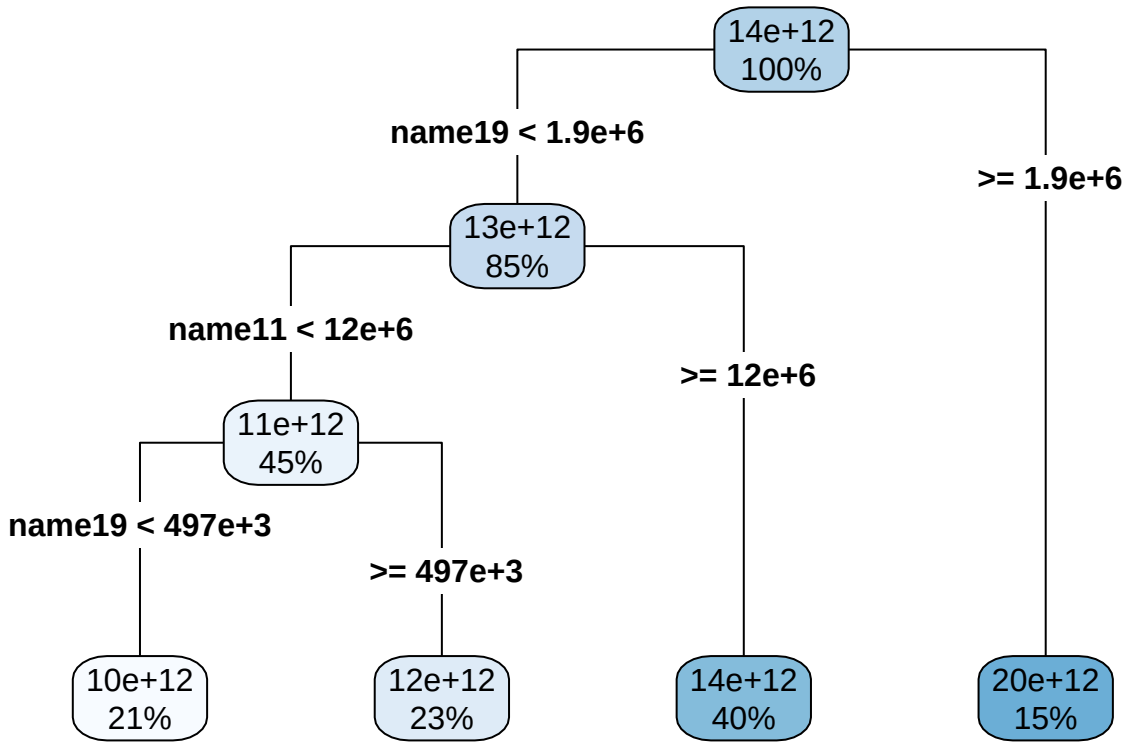
```
RMSE_KNN <- RMSE(test$main, model_predict_test)

MAE_KNN <- MAE(test$main, model_predict_test)
```

```
## [1] "RMSE for KNN: 883379772988.028"
```

```
## [1] "MAE for KNN: 729448863636.363"
```

Tree

```
my_model <- rpart(f,subset = index, data= try)
```




```
predictions<-predict(my_model,newdata=test)

RMSE_Tree <- RMSE(predictions, test$main)

MAE_Tree <- MAE(predictions, test$main)
```

```
## [1] "RMSE for Tree: 1237594660062.14"
```

```
## [1] "MAE for Tree: 823796345693.78"
```

forrest

```
set.seed(1)
bag.black <- randomForest(f,data=try, subset=index,importance =TRUE)

prediction_forest = predict(bag.black, newdata=test[1,])
```

```
getTree(bag.black,1,labelVar=TRUE)
```

```
##    left daughter right daughter split var split point status   prediction
## 1              2              3    name19   1896354.0     -3 1.364855e+13
## 2              4              5    name18    450338.0     -3 1.290054e+13
## 3              6              7     name9   2144449.5     -3 1.875995e+13
## 4              8              9     name8    831905.5     -3 1.078578e+13
## 5             10             11     name3   1321678.5     -3 1.349532e+13
## 6              0              0      <NA>         0.0     -1 1.803596e+13
## 7              0              0      <NA>         0.0     -1 2.237989e+13
## 8              0              0      <NA>         0.0     -1 8.866020e+12
## 9             12             13     name9    704149.0     -3 1.133428e+13
## 10            14             15     name2   1775472.5     -3 1.287417e+13
## 11            16             17     name6   4419834.5     -3 1.508269e+13
## 12            18             19     name7   3952765.5     -3 1.109505e+13
## 13             0              0      <NA>         0.0     -1 1.276962e+13
## 14            20             21    name19    598654.5     -3 1.248858e+13
## 15            22             23    name12   5401188.5     -3 1.359716e+13
## 16             0              0      <NA>         0.0     -1 1.331479e+13
## 17            24             25    name11  15642877.0     -3 1.530368e+13
## 18             0              0      <NA>         0.0     -1 1.048763e+13
## 19             0              0      <NA>         0.0     -1 1.139877e+13
## 20             0              0      <NA>         0.0     -1 1.134413e+13
## 21            26             27    name11  15457833.5     -3 1.266465e+13
## 22             0              0      <NA>         0.0     -1 1.422645e+13
## 23             0              0      <NA>         0.0     -1 1.296788e+13
## 24             0              0      <NA>         0.0     -1 1.456715e+13
## 25            28             29     name4   2894730.5     -3 1.540890e+13
## 26            30             31     name8   1557298.5     -3 1.255740e+13
## 27             0              0      <NA>         0.0     -1 1.395159e+13
## 28             0              0      <NA>         0.0     -1 1.511224e+13
## 29             0              0      <NA>         0.0     -1 1.552756e+13
## 30            32             33    name19   1034047.5     -3 1.235654e+13
## 31             0              0      <NA>         0.0     -1 1.295913e+13
## 32             0              0      <NA>         0.0     -1 1.212681e+13
## 33            34             35    name13   1771262.5     -3 1.243311e+13
## 34             0              0      <NA>         0.0     -1 1.242180e+13
## 35             0              0      <NA>         0.0     -1 1.244442e+13
```

```
MAE_forest <- MAE(prediction_forest, test$main)
RMSE_forest <- RMSE(prediction_forest, test$main)
```

```
## [1] "RMSE for Random Forrest: 1577588439372.11"
```

```
## [1] "MAE for Random Forrest: 1301383928666.67"
```

Ridge Regression

```
x = model.matrix(f, data = try)[,c(-1,-2)]
y = try$main

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

ridge.mod = glmnet(x[index,], y[index], alpha = 0, lambda = grid,
                   thresh = 1e-12)
ridge.pred =predict(ridge.mod, s = 4, newx = x[-index,])

RMSE_ridge <- RMSE(ridge.pred, y[-index])

MAE_ridge <- MAE(ridge.pred, y[-index])
```

```
## [1] "RMSE for Ridge Regression: 1148903643001.52"
```

```
## [1] "MAE for Ridge Regression: 927072840973.144"
```

Lasso Regression

```
set.seed(1)

lasso.mod=glmnet(x[index,],y[index],alpha=1,lambda=grid)
cv.out=cv.glmnet(x[index,],y[index],alpha=1)

bestlam=cv.out$lambda.min

lasso.pred=predict(lasso.mod,s=bestlam,newx=x[-index,])
RMSE_lasso <- RMSE(lasso.pred, y[-index])

MAE_lasso <- MAE(lasso.pred, y[-index])
```

```
## [1] "RMSE for Lasso Regression: 1110923782965.66"
```

```
## [1] "MAE for Lasso Regression: 922032541404.381"
```

Extreme Gradient Boosting

```
set.seed(1)
dtrain2 <- xgb.DMatrix(data = x[index,], label = y[index])
dtest2 <- xgb.DMatrix(data = x[-index,], label = y[-index])
watchlist <- list(train= dtrain2, test= dtest2)
set.seed(1)
bst2 <- xgb.train(data= dtrain2, max.depth=20, eta=0.09, nrounds=120,watchlist=watchlist,
                  base_score = 0.1)

xgb_test <- predict(bst2, data.matrix(test[,-c(1,21)]))

RMSE_xgboost <- RMSE(test$main, xgb_test)

MAE_xgboost <- MAE(test$main,xgb_test)
```

```
## [1] "RMSE for XGB: 766695227097.678"

## [1] "MAE for XGB: 628982514144"
```

```
head(rmse)
```

```
##              [,1]
## RMSE1        "1287777678920.08"
## RMSE_forest  "1577588439372.11"
## RMSE_lasso   "1110923782965.66"
## RMSE_ridge   "1148903643001.52"
## RMSE_Tree    "1237594660062.14"
## RMSE_xgboost "766695227097.678"

## [1] "the Algorithm with the least error is: 766695227097.678"
```

```
## Ridge Regression
mean_error(difference(ridge.pred,test$main))
```

```
## [1] 0.07648632
```

```
## Extreme Gradient Boosting
mean_error(difference(xgb_test,test$main))
```

```
## [1] 0.04703214
```

```
## Lasso Regression
mean_error(difference(lasso.pred,test$main))
```

```
## [1] 0.07493814
```

```
##Forest
mean_error(difference(prediction_forest,test$main))
```

```
## [1] 0.101472
```

```r
# Linear Regression
mean_error(difference(pred1,test$main))
```

```
## [1] 0.08775921
```

```r
## knn
mean_error(difference(model_predict_test,test$main))
```

```
## [1] 0.05498922
```

```r
## Tree
mean_error(difference(model_predict_test,test$main))
```

```
## [1] 0.05498922
```