Homework #3: Model comparison

```
#install.packages("DALEX")
#install.packages("caret")
#install.packages("tidymodels")
#install.packages("rpart.plot")
#install.packages("ranger")
#install.packages("mlbench")
library(DALEX)
library(caret)
library(tidymodels)
library(rpart.plot)
library(ranger)
library(mlbench)
```

```
customer_data <- read.csv("customer_data.csv")
```

```
str(customer_data)
```

```
## 'data.frame':    1125 obs. of  13 variables:
##  $ label : int  1 0 0 1 0 0 1 1 0 0 ...
##  $ id    : int  54982665 59004779 58990862 58995168 54987320 59005995 59001917 54984789 58984557 549
##  $ fea_1 : int  5 4 7 7 7 6 4 5 5 4 ...
##  $ fea_2 : num  1246 1277 1298 1336 NA ...
##  $ fea_3 : int  3 1 1 1 2 3 3 3 3 2 ...
##  $ fea_4 : num  77000 113000 110000 151000 59000 56000 35000 78000 218000 35000 ...
##  $ fea_5 : int  2 2 2 2 2 2 2 2 2 2 ...
##  $ fea_6 : int  15 8 11 11 11 6 8 15 15 8 ...
##  $ fea_7 : int  5 -1 -1 5 5 -1 9 -1 5 5 ...
##  $ fea_8 : int  109 100 101 110 108 100 85 111 112 101 ...
##  $ fea_9 : int  5 3 5 3 4 3 5 3 4 3 ...
##  $ fea_10: int  151300 341759 72001 60084 450081 60091 60069 60030 151300 60029 ...
##  $ fea_11: num  245 207 1 1 197 ...
```

In the data set we have 13 features and 1125 observation. All observations are numeric and integer. Label features is; if it is 1, the customer is in high credit risk and if it is 0, the customer is in low credit risk. In this analysis our target is "label" which is credit risk.

```
sum(is.na(customer_data))
```

```
## [1] 149
```

Before the start analysing checking NA values. In the data set we have 149 NA at fea_2 column. In 1125 observation, 149 missing value is about %13 of whole observations. Clearing all the NA values could make analysis wrong. Thats why I decided to change all missing value to mean value of fea_2 column.

```
sum(customer_data$fea_2, na.rm = TRUE)
```

```
## [1] 1253098
```

sum of fea_2 column is 1253098 and mean of fea_2 is 1113.865.

```
customer_data$fea_2[is.na(customer_data$fea_2)] <- 1113.865
```

All NA values replaced with the mean value of fea_2.

```
sum(is.na(customer_data))
```

```
## [1] 0
```

Now there is no missing value. We can start analysis.

Logistic Regression Model

```
set.seed(1)
index <- sample(1 : nrow(customer_data), round(nrow(customer_data) * 0.80))
train <- customer_data[index, ]
test <- customer_data[-index, ]
```

Training the logistic regression model

```
lr_model <- glm(label ~ ., data = train, family = "binomial")
lr_model
```

```
##
## Call:  glm(formula = label ~ ., family = "binomial", data = train)
##
## Coefficients:
## (Intercept)           id        fea_1        fea_2        fea_3        fea_4
##  -1.906e+00    2.735e-08    1.631e-01   -1.283e-03    1.385e-01   -6.810e-06
##       fea_5        fea_6        fea_7        fea_8        fea_9       fea_10
##   1.879e-01   -2.286e-02   -2.300e-02   -3.572e-03    9.050e-02   -5.286e-07
##      fea_11
##   5.827e-04
##
## Degrees of Freedom: 899 Total (i.e. Null);  887 Residual
## Null Deviance:       889.5
## Residual Deviance: 853.9     AIC: 879.9
```

Decision Tree Model

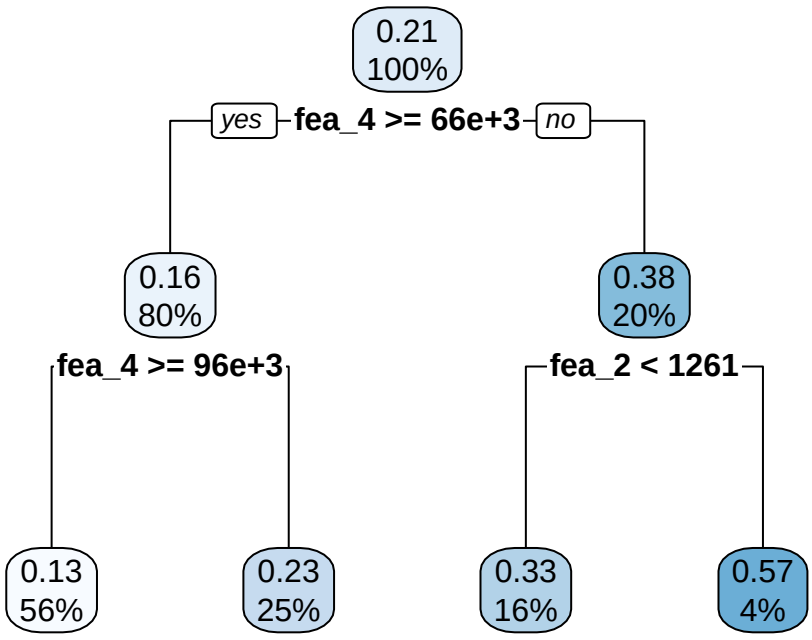
```
customer_split <- initial_split(data = customer_data,
                                prop = 0.80)

customer_train <- customer_split |> training()
customer_test  <- customer_split |> testing()
```

```
dt_model <- decision_tree() |>
  set_engine("rpart") |>
  set_mode("regression")
```

```
dt_customer <- dt_model |>
  fit(label ~., data = customer_train)
dt_customer
```

```
## parsnip model object
##
## n= 900
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
## 1) root 900 147.560000 0.2066667
##   2) fea_4>=65500 723  99.413550 0.1645920
##     4) fea_4>=95500 500  58.022000 0.1340000 *
##     5) fea_4< 95500 223  39.874440 0.2331839 *
##   3) fea_4< 65500 177  41.638420 0.3785311
##     6) fea_2< 1261.25 140  30.885710 0.3285714 *
##     7) fea_2>=1261.25 37   9.081081 0.5675676 *
```

```
rpart.plot(dt_customer$fit, roundint = FALSE)
```



Random Forest Model

```
set.seed(2)
trained_rf <- ranger(label ~ .,
                     data = customer_train)
```

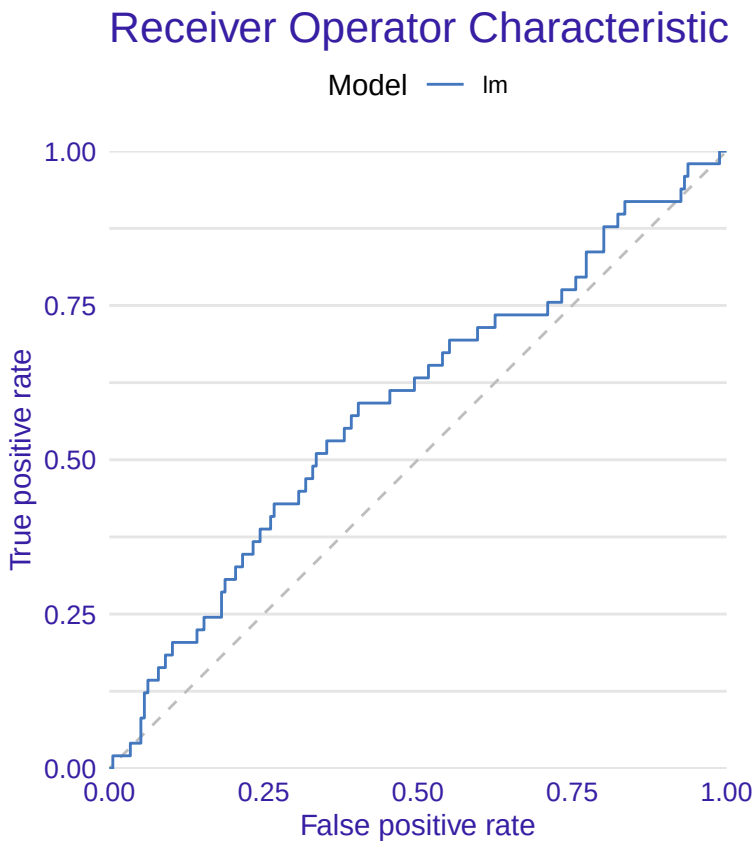
```
trained_rf
```

```
## Ranger result
##
## Call:
##  ranger(label ~ ., data = customer_train)
##
## Type:                             Regression
## Number of trees:                  500
## Sample size:                      900
## Number of independent variables:  12
## Mtry:                             3
## Target node size:                 5
## Variable importance mode:         none
## Splitrule:                        variance
## OOB prediction error (MSE):       0.1635193
## R squared (OOB):                  0.003768708
```

The Performance of The Models

1- Logistic Regression Model

```
explain_lr <- DALEX::explain(model = lr_model,
              data = test[, -1],
              y = test$label == "1",
              type = "classification",
              verbose = FALSE)
performance_lr <- model_performance(explain_lr)
plot(performance_lr, geom = "roc")
```




```
performance_lr
```

```
## Measures for:  classification
## recall     : 0
## precision  : NaN
## f1         : NaN
## accuracy   : 0.7822222
## auc        : 0.5867347
##
## Residuals:
##          0%         10%         20%         30%         40%         50%
## -0.39895357 -0.28481762 -0.23990566 -0.20774995 -0.18357901 -0.15185829
##         60%         70%         80%         90%        100%
## -0.12844349 -0.09097771  0.68057997  0.78871345  0.99441901
```

The reason of recall value 0 and precision and f1 NaN predicted props values are lower than 0.5. AUC value 0.58 is not bad but not reliable accuracy value 0.78 also not bad but still not reliable for model performance.

2- Decision Tree Model Performance

```
predictions_dt <- predict(dt_customer, customer_test)
r_squared <- cor(predictions_dt, customer_test$label)^2
r_squared
```

```
##             [,1]
## .pred 0.03746921
```

For the Decision Tree R^2 value is 0.03746921. Thats show us indicates model only explains 3.75% of the total variance. Need to be improve model.

3- Random Forest Model Performance

```
preds <- predict(trained_rf, customer_test)$predictions
accuracy <- mean(preds == customer_test$label)
accuracy
```

```
## [1] 0
```

If the accuracy is 0, this indicates that the estimates are incorrect. This means that the model does not classify correctly or produces incorrect predictions. Therefore, it is necessary to reconstruct the model and perform the analysis.

Cross Validation Method

```
set.seed(3)
control <- trainControl(method = "cv",
                        number = 10)
model <- train(as.factor(label) ~.,
               data = customer_data,
               trControl = control,
               method = "glm")
model
```

```
## Generalized Linear Model
##
## 1125 samples
##   12 predictor
##    2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 1012, 1012, 1013, 1012, 1013, 1012, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.8000158  0
```

According to this output, it can be said that the model has an accuracy rate of 80% and outperforms random classifications.