DATA 622 - Homework 1

OMER OZEREN

Table of Contents

Load the Data

```
df <- read.table("C:/Users/OMERO/Documents/GitHub/DATA622/data.txt",header =
T,sep=',')
df$label <- ifelse(df$label =="BLACK",1,0)
df$y <- as.numeric(df$y)
df$X <- as.factor(df$X)
```

Split Data into Train(60%) and Test data(30%)

```
set.seed(998)
split_df <- createDataPartition(df$label, p = .60, list = FALSE)
df_train <- df[split_df,]
df_test <- df[-split_df,]
```

Knn k=3

```
knnK3.model <- knn(df_train,df_test,cl=df_train$label,k=3)
knnK3.cm <- table(knnK3.model,df_test$label)
knnK3.pred <- prediction(as.numeric(knnK3.model) ,df_test$label)
knnK3.perf <- performance(knnK3.pred,measure="tpr",x.measure="fpr")
auc_roc(knnK3.model, df_test$label)

## [1] 0.5833333

# Store the results

KNN_K3_RESULTS <- data.frame("ALGO"="KNN(K=3)","AUC" =
performance(knnK3.pred,"auc")@y.values[[1]],
                  "ACC" = sum(diag(knnK3.cm)/(sum(rowSums(knnK3.cm)))),
                  "TPR" = knnK3.cm[2,2] / sum(knnK3.cm[2,]),
                  "FPR"= knnK3.cm[1,2] / sum(knnK3.cm[1,]),
```

```
                 "TNR" =  knnK3.cm[1,1] / sum(knnK3.cm[1,]),
                 "FNR"=  knnK3.cm[2,1] / sum(knnK3.cm[2,]))
KNN_K3_RESULTS
```

```
##       ALGO       AUC       ACC       TPR FPR TNR       FNR
## 1 KNN(K=3) 0.5833333 0.6428571 0.8888889 0.8 0.2 0.1111111
```

Knn k=5

```
knnK5.model <- knn(df_train,df_test,cl=df_train$label,k=5)
knnK5.cm <- table(knnK5.model,df_test$label)
knnK5.pred <- prediction(as.numeric(knnK5.model) ,df_test$label)
knnK5.perf <- performance(knnK5.pred,measure="tpr",x.measure="fpr")
auc_roc(knnK5.model, df_test$label)
```

```
## [1] 0.5833333
```

```
# Store the results

KNN_K5_RESULTS <- data.frame("ALGO"="KNN(K=5)","AUC" =
performance(knnK5.pred,"auc")@y.values[[1]],
                 "ACC" = sum(diag(knnK5.cm)/(sum(rowSums(knnK5.cm)))),
                 "TPR" = knnK5.cm[2,2] / sum(knnK5.cm[2,]),
                 "FPR"= knnK5.cm[1,2] / sum(knnK5.cm[1,]),
                 "TNR" =  knnK5.cm[1,1] / sum(knnK5.cm[1,]),
                 "FNR"=  knnK5.cm[2,1] / sum(knnK5.cm[2,]))
KNN_K5_RESULTS
```

```
##       ALGO       AUC       ACC       TPR FPR TNR       FNR
## 1 KNN(K=5) 0.5833333 0.6428571 0.8888889 0.8 0.2 0.1111111
```

LR

```
lr.model <- glm(label ~ ., data=df_train,family = "binomial")
lr.test = predict(lr.model, newdata=df_test,type="response")
lr.cm <- table(lr.test > 0.5,df_test$label)
lr.pred <- prediction(as.numeric(lr.test > 0.5),df_test$label)
lr.perf <- performance(lr.pred,measure="tpr",x.measure="fpr")
# Store the results
LR_RESULTS <- data.frame("ALGO"="LR","AUC" =
performance(lr.pred,"auc")@y.values[[1]],
                 "ACC" = sum(diag(lr.cm)/(sum(rowSums(lr.cm)))),
                 "TPR" = lr.cm[2,2] / sum(lr.cm[2,]),
                 "FPR"= lr.cm[1,2] / sum(lr.cm[1,]),
                 "TNR" =  lr.cm[1,1] / sum(lr.cm[1,]),
                 "FNR"=  lr.cm[2,1] / sum(lr.cm[2,]))
LR_RESULTS
```

```
##   ALGO       AUC       ACC       TPR       FPR       TNR        FNR
## 1   LR 0.6666667 0.7857143 0.9090909 0.6666667 0.3333333 0.09090909
```

Naive Bayes

```
nb.model = naive_bayes(as.character(label)~., data=df_train)

## Warning: naive_bayes(): Feature X - zero probabilities are present.
Consider
## Laplace smoothing.

nb.test= predict(nb.model, newdata=df_test)

## Warning: predict.naive_bayes(): more features in the newdata are provided
as
## there are probability tables in the object. Calculation is performed based
on
## features to be found in the tables.

nb.cm <- table(nb.test,df_test$label)
nb.pred <- prediction(as.numeric(nb.test) ,df_test$label)
nb.perf <- performance(nb.pred,measure="tpr",x.measure="fpr")
# Store the results
NB_RESULTS <- data.frame("ALGO"="NB","AUC" =
performance(nb.pred,"auc")@y.values[[1]],
                 "ACC" = sum(diag(nb.cm)/(sum(rowSums(nb.cm)))),
                 "TPR" = nb.cm[2,2] / sum(nb.cm[2,]),
                 "FPR"= nb.cm[1,2] / sum(nb.cm[1,]),
                 "TNR" =  nb.cm[1,1] / sum(nb.cm[1,]),
                 "FNR"=  nb.cm[2,1] / sum(nb.cm[2,]))
NB_RESULTS

##   ALGO       AUC       ACC       TPR FPR TNR       FNR
## 1   NB 0.5833333 0.6428571 0.8888889 0.8 0.2 0.1111111
```

SUMMARY

```
##       ALGO       AUC       ACC       TPR       FPR       TNR        FNR
## 1 KNN(K=3) 0.5833333 0.6428571 0.8888889 0.8000000 0.2000000 0.11111111
## 2 KNN(K=5) 0.5833333 0.6428571 0.8888889 0.8000000 0.2000000 0.11111111
## 3       NB 0.5833333 0.6428571 0.8888889 0.8000000 0.2000000 0.11111111
## 4       LR 0.6666667 0.7857143 0.9090909 0.6666667 0.3333333 0.09090909
```

- The table above shows each model capacity to learn and capacity to generalize results.
- The data set is too small for machine learning models.Due to lack of data ,Machine Learning models will suffer the perform good results.I splited the dat in to train(60%) and test(40%) to check each model's performance on unseen data.
- When we review at the each models' ability or capacity to learn. LR model gives the best AUC (66%) and best Accuracy (78.5%).
  - The AUC provides an aggregate measure of performance across all possible classification threshold.The seperation between BLACK and BLUE for all models above 0.58% which indicates that models are able to get okay results in

class separation.KNN with k=3,KNN with k-5 and Naive Bayes models give identical results in ability to learn.This results might be caused because lack of data.

  - All of the models accuracy rate (ability orcapacity to learn) over 64% indicating that they all have an slightly better random change of correctly classifying Black (1) vs. Blue(0)

- When we review at the each model's ability to generalize,LR gives better results among the other models.

  - True Positive Rate **(Sensitivity)** : Sensitivity measures how the model is to detecting events in the positive (Black) class,in this case When it's actuall value is **Black (1)**, LR model predicted output **Black (1)** with 0.90%,NB,and KNN models predicted outout with 0.88%.

  - False Positive Rate **(Specificity)**: Specificity measures how exact the assignment to the positive class is, in this case, When it's actuall value is **Blue (0)**, LR model predicted output **Black (1)** with 0.66%,NB,and KNN models predicted outout with 0.80%.

  - True Negative Rate: When it's actuall value is **Blue (0)**, LR model predicted output **Blue (0)** with 0.33%,NB,and KNN models predicted outout with 0.220%

  - True Negative Rate When it's actuall value is **Black (1)**, LR model predicted output **Blue (0)** with 0.09%,NB,and KNN models predicted outout with 0.11%.

- Logistic Regression gives better result in terms of ability or capacity to learn and ability to generalize.

- One of the most important aspects of an algorithm is how fast it is. It is often easy to come up with an algorithm to solve a problem, but if the algorithm is too slow, it's back to the drawing board.In terms of ascpects of Algorithm, LR is also a quick and reasonably method.

- In machine learning, the more data is usually better than better algorithms. There are multiple aspects of data quality that comprise fitness for modeling: **relevance, accuracy, completeness, recency and cleanliness**.
  - The data needs to accurately reflect or correspond to what we're measuring, to the required level of measurement.
  - The Complete data measures or describes all the relevant aspects of the problem you're trying to solve.
  - The Recent data reflects the current state of a measurement.
  - The Clean data is free of duplicate values. The data is organized, standardized, structured and labelled or documented to the extent possible