Chenghan Yue

Professor Barbara Kuzmak

STAT 4893W

2020/05/03

Final Data Analysis Independent Report

Introduction and Background

The research objective is to find the most accurate model among Linear Discriminant Analysis (LDA), Classification and Regression Trees (CART) and k-Nearest Neighbors (KNN). And, according to the statistical literature, I've been looking for, “this is a good mixture of simple linear (LDA) and nonlinear (CART, KNN) methods”(Jason Brownlee, November 25, 2016). In my report, I referred to four materials to help me. From *an Introduction to Statistical Learning with Applications in R*, I learned what Linear Discriminant Analysis and K-Nearest Neighbors algorithms were. From Jason Brownlee’s essay, *Your First Machine Learning Project in R Step by Step*, I learned how to analyze data and find the most accurate model by machine learning. And in her another essay, *Classification and Regression Trees for Machine Learning*, Jason points out what Classification and Regression Trees algorithm are. In the Kumar Skand’s essay, *KNN (k-Nearest Neighbor) Algorithm in R*, he taught me what KNN is and how to build KNN model. Without the help from these materials, I can’t finish my project, so I was grateful to these authors.

Materials and Methods

My dataset was from the Kaggle website. Tthe data is called “Banknote”, which “were extracted from images that were taken from genuine and forged banknote-like specimens. For digitization, an industrial camera usually used for print inspection was used. The final images have 400x 400 pixels. Due to the object lens and distance to the investigated object gray-scale pictures with a resolution of about 660 dpi were gained. Wavelet Transform tool were used to extract features from images" (Kaggle website, 2018).

This dataset contains five variables, which are variance of Wavelet Transformed image (continuous), skewness of Wavelet Transformed image (continuous), curtosis of Wavelet Transformed image (continuous), entropy of image (continuous) and class (authentic, inauthentic). The following Table 1 shows the first 5 rows of the data:

variance	skewness	curtosis	entropy	class
3.62160	8.6661	-2.8073	-0.44699	authentic
4.54590	8.1674	-2.4586	-1.46210	authentic
3.86600	-2.6383	1.9242	0.1645	authentic
3.45660	9.5228	-4.0112	-3.59440	Authentic
0.32924	9.6718	-3.9606	-3.16250	authentic

Table 1: data preview

To analyze the data, I created a training dataset, which was 80% of the original dataset, then the remaining 20% of the data was used for testing the model. I will use the 80% dataset to train the KNN, IDA and CART models. In the training process, K-fold method will be used to get the best values of parameters. After the models are trained, I will the rest 20% to predict the data and compare the performance of these models. ( ) More details of analysis will be introduced in the following analysis part.

Analysis

During the research, my project can be summarized as five steps. I collected the data, summarized the dataset, visualized the dataset, evaluated some algorithms and finally concluded my research.

In the above part, I had described my dataset and show how to get the validation variable. With the help of R studio, I summarized the dataset (table 2). From table 2, we see that all of the numerical values have the ranges [-14, 18]. This table contains the mean, the min and max values as well as some percentiles (25^{th}, 50^{th} or media and 75^{th} values at this point if we ordered all the values for an attribute).

	variance of Wavelet Transformed image	skewness of Wavelet Transformed image	curtosis of Wavelet Transformed image	entropy of image
min	-7.0421	-13.773	-5.2861	-8.5482
1st Qu	-1.784	-1.906	-1.6345	-2.352
Median	0.4009	2.257	0.6668	-0.6054
Mean	0.4189	1.873	1.4143	-1.1972
3rd Qu	2.8075	6.797	3.1638	0.3948
Max	5.9374	12.73	18	2.4495

Table 2: table summary

Methodology

In this part, I decided to build two kinds of plots which are Univariate plots and Multivariate plots. According to the Jason's words, she points out "the univariate plots is to better understand each attribute and Multivariate plots is to better understand the relationships between attributes”. I built boxplot by R studio which can give me a clearer idea of the distribution of the input attributes. (Figure 1)

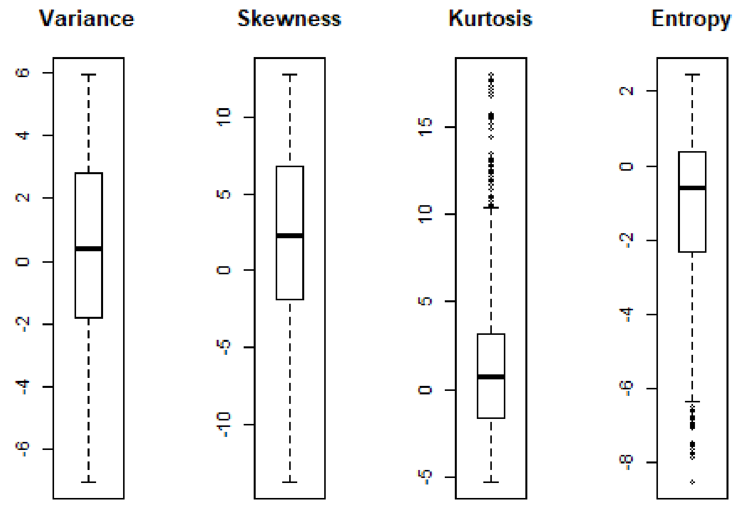


Figure 1: boxplot of variables

After that I also built multivariate plots which can help to tease out obvious linear separations between the classes and it also useful to find that there are much clearer different distributions of the attributes for each value. (Figure 2)

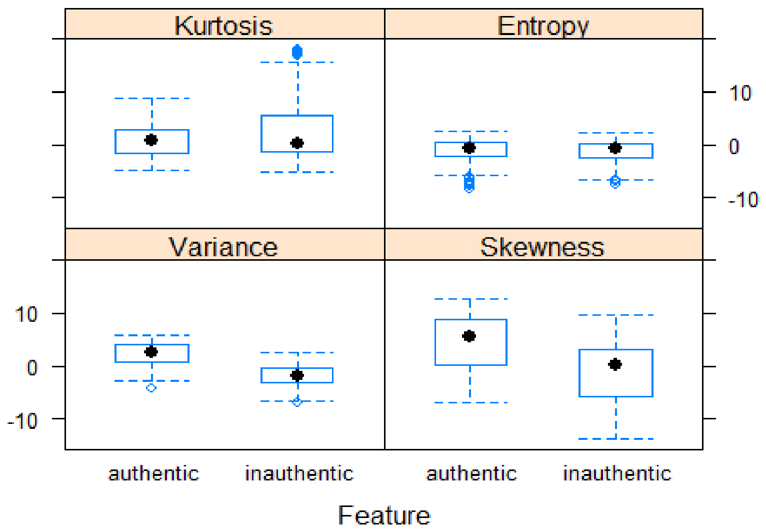


Figure 2

Besides, I built probability density plots, which can give us nice smooth lines for each distribution, as Jason did in his article. These plots can help me to see the difference in distribution of each attribute by class value. (Figure 3)

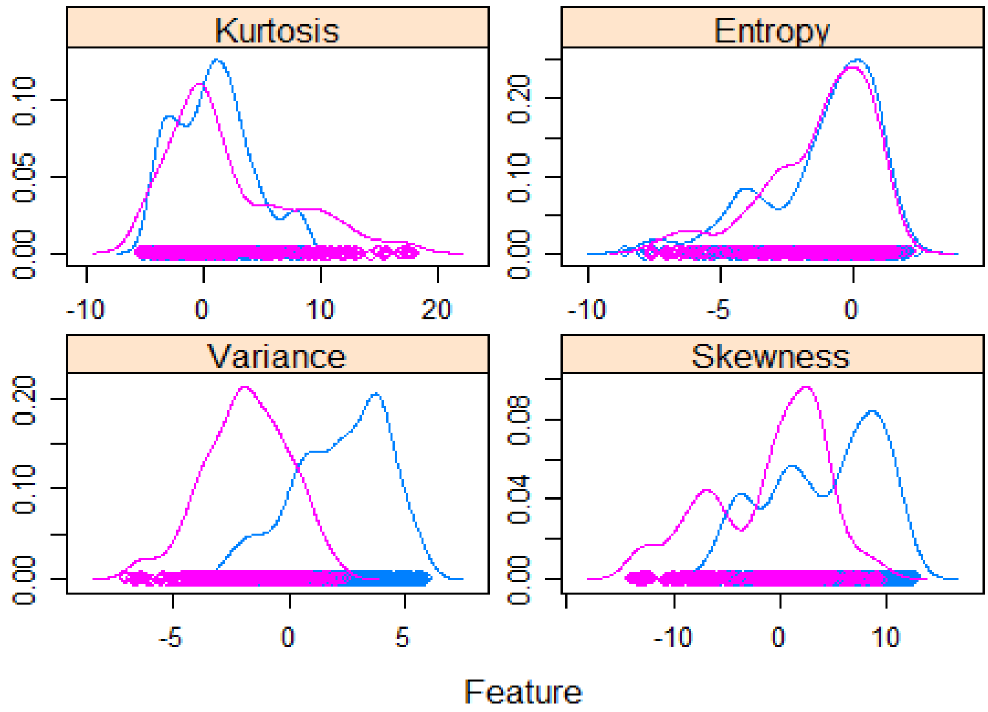


Figure 3

With the help of above parts, I'm going to create the KNN, IDA and CART to compare their performances. First of all, I use 10-fold cross validation to train the models, which will split the training dataset into 10 parts, 9 parts are training data and 1 part is testing data. It also releases for all combinations of train-test splits. For getting more accurate estimates, I will repeat the process 3 times for each algorithm with different splits of the data into 10 parts. Since I don't know which algorithms would build the best model on this problem, I have to compare these three models to each other and select the most accurate one.

Conclusion

To conclude my research, I build a table to compare these three models' summary outputs which can help us to find the best one. According to the table 3, we can find that when k = 10, the accuracy and kappa of KNN model will in the range [0.9984793, 0.9987823] and [0.9969255, 0.9975376]. Because I separate the KNN model from the lda model and cart model, I have to compare lad model and cart model first. According to the figure 4, we can find that the LDA model is better than CART model. Then, I compare the LDA model with KNN model by their summary output. From their summary output, I find their accuracy and kappa value are very close. To examine which one is the best, I decide to make predictions by both of them. From their confusionMatrix outputs, I find that KNN model has higher accuracy value. The accuracy is 100% which is better than LDA's accuracy value 99.64%. Thus, I conclude my research that the most accurate model is the KNN model.

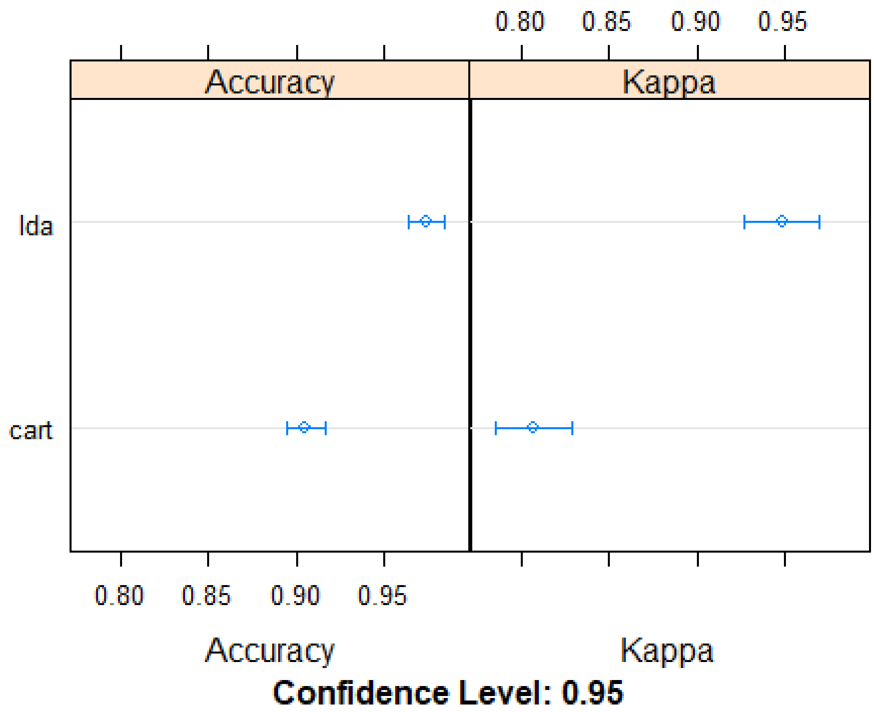


Figure 4

Discussion and Summary

According to the part of result, I find the most accurate model is the KNN model which means that in this data, if I decided to make predictions, I need to build a KNN model. But, if change the dataset, I still need to analyze which is the best model. In my opinion, machine learning is a very useful technique which taught me how to build the best model and make predictions. But, during the researching, I still confused about how to compare the KNN model with LDA model and CART model together. If I could put all three of them in a table, it will become more convenience for us to find the best model. That's my further research direction.

Reference

Gareth James (2013), *An Introduction to Statistical Learning with Applications in R*

Kumar Skand (Oct 8th, 2017), *kNN (k-Nearest Neighbor) Algorithm in R*

Jason Brownlee (April 8th, 2016), *Classification and Regression Trees for Machine Learning*

Jason Brownlee (Feb 3rd, 2016), *Your First Machine Learning Project in R Step by Step*

Appendix

```r
library(caret)

## Loading required package: lattice

## Loading required package: ggplot2

dat = read.csv("C:/Users/yuech/Downloads/stat 4052/data_banknote_authentication.csv", header = T)
```

##Create a validation Dataset

```r
### create a list of 80% of the rows in the original dataset we can use for training
validation_index = createDataPartition(dat$Class, p = 0.80, list = F)

### select 20% of the data for validation
validation = dat[-validation_index,]

### use the remaining 80% of data to training and testing the models
dat = dat[validation_index,]
```

##Summarize Dataset

```r
## dimensions of dataset
dim(dat)

## [1] 1098    5
```

According to the output, I need to see 1098 instances and 5 attributes.

##Type of Attributes

```
## list types for each attribute
sapply(dat, class)
```

```
##  Variance  Skewness  Kurtosis   Entropy     Class
## "numeric" "numeric" "numeric" "numeric"  "factor"
```

##Peak at the Data

```
## take a peak at the first 5 rows of the data
head(dat)
```

```
##   Variance Skewness Kurtosis  Entropy      Class
## 3  3.86600  -2.6383  1.92420  0.10645  authentic
## 4  3.45660   9.5228 -4.01120 -3.59440  authentic
## 5  0.32924  -4.4552  4.57180 -0.98880  authentic
## 6  4.36840   9.6718 -3.96060 -3.16250  authentic
## 7  3.59120   3.0129  0.72888  0.56421  authentic
## 9  3.20320   5.7588 -0.75345 -0.61251  authentic
```

levels of the Class

```
## list the levels for the class
levels(dat$Class)
```

```
## [1] " authentic"   " inauthentic"
```

Class Distribution

```
## summarize the class distribution
percentage = prop.table(table(dat$Class))*100
cbind(freq = table(dat$Class), percentage = percentage)
```

```
##              freq  percentage
##   authentic   610    55.55556
## inauthentic   488    44.44444
```

statistical Summary

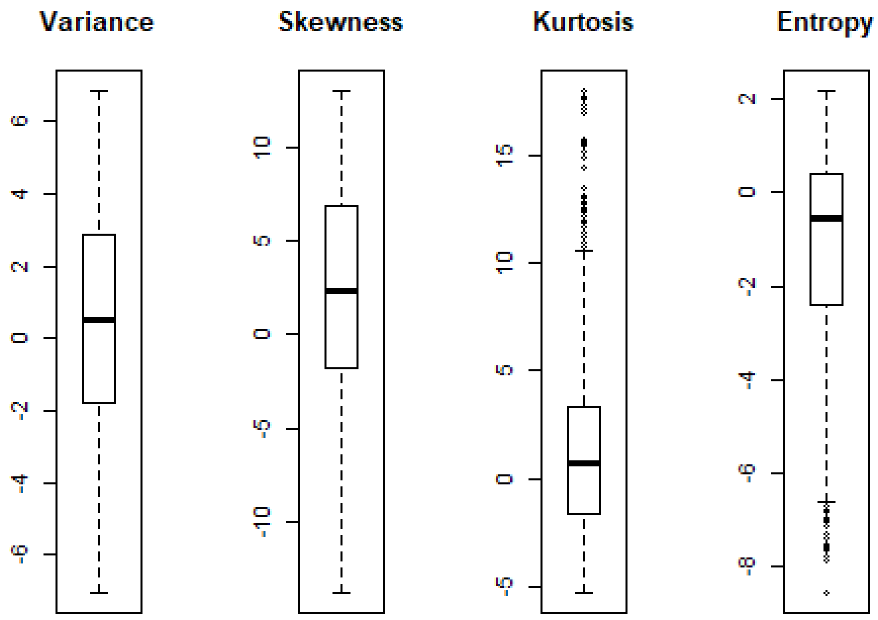
```
## summarize attribute distributions
summary(dat)
```

```
##     Variance          Skewness           Kurtosis          Entropy
##  Min.   :-7.0421   Min.   :-13.773   Min.   :-5.2861   Min.   :-8.5482
##  1st Qu.:-1.7840   1st Qu.: -1.782   1st Qu.:-1.6315   1st Qu.:-2.4205
##  Median : 0.4923   Median :  2.295   Median : 0.6805   Median :-0.5660
##  Mean   : 0.4442   Mean   :  1.895   Mean   : 1.4097   Mean   :-1.1830
##  3rd Qu.: 2.8465   3rd Qu.:  6.808   3rd Qu.: 3.2911   3rd Qu.: 0.4022
##  Max.   : 6.8248   Max.   : 12.952   Max.   :17.9274   Max.   : 2.1625
##          Class
##  authentic  :610
##  inauthentic:488
##
##
##
##
```

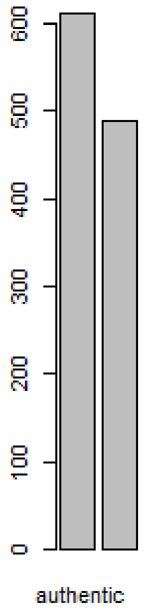
Univariate Plots

```
x = dat[,1:4]
y = dat[,5]
par(mfrow = c(1,4))
```

```
for (i in 1:4) {
  boxplot(x[,i],main = names(dat)[i])
}
```

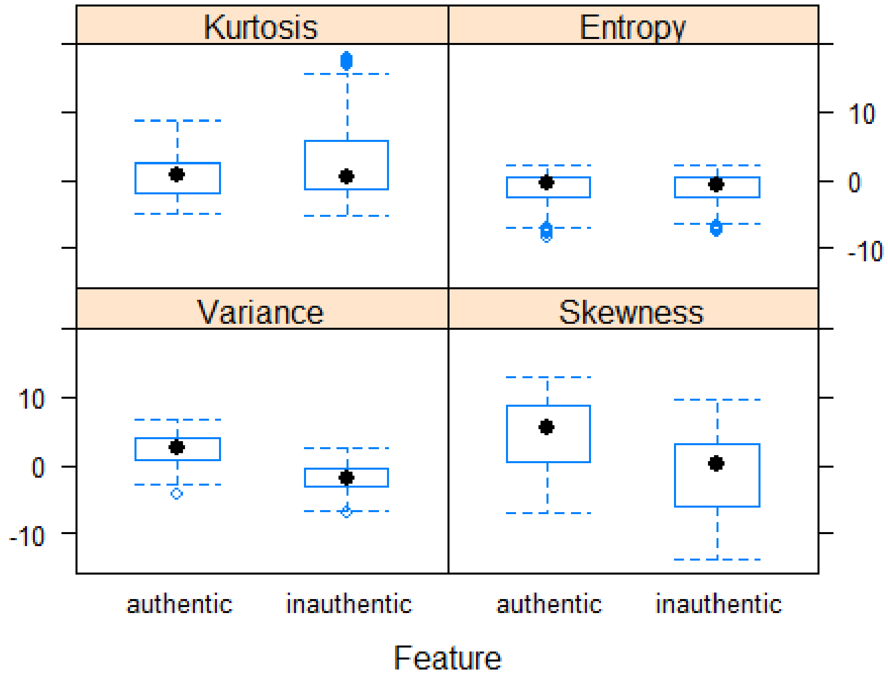



```
## barplot for class breakdown
plot(y)
```

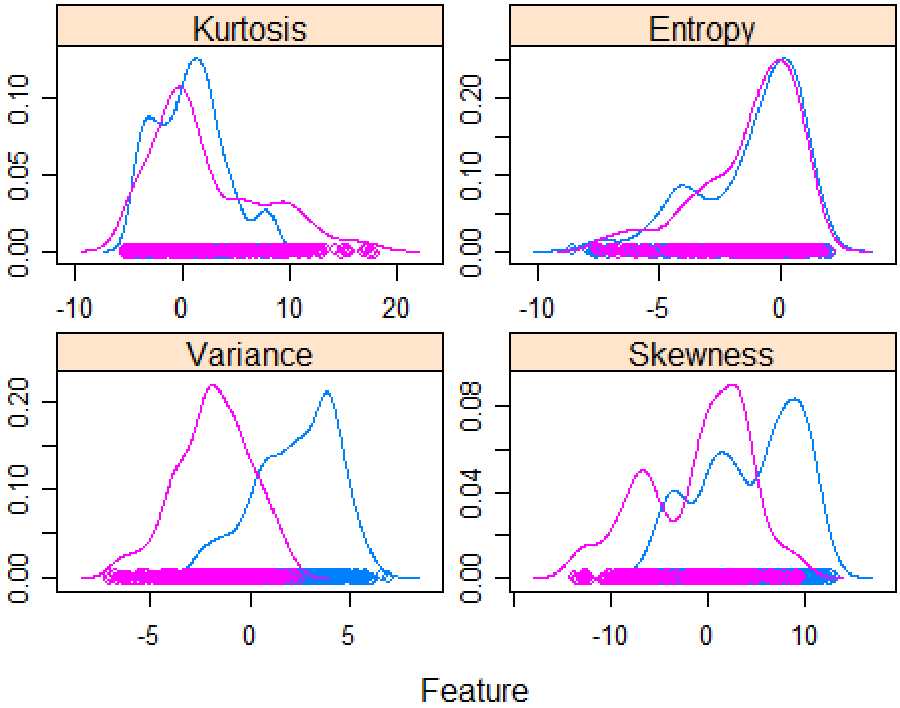


multivariate Plots

```
##scatterplot matrix
featurePlot(x=x,y=y, plot = "box")
```



```
## density plots for each attribute by class value
scales = list(x=list(relation = "free"), y = list(relation = "free"))
featurePlot(x=x,y=y, plot = "density", scales = scales)
```



Evaluate Some Algorithms

```
## Run algorithms using 10-fold croos validation
control = trainControl(method = "cv", number = 10)
metric = "Accuracy"
```

Build Model

```
## LDA
set.seed(4893)
fit.lda = train(Class~., data = dat, method = "lda", metric = metric, trControl = control)
## CART
set.seed(4893)
```

```
fit.cart = train(Class~., data = dat, method = "rpart", metric = metric, trControl = control)

## knn
set.seed(4893)
trctrl = trainControl(method = "repeatedcv", number = 10, repeats = 3)
fit.knn = train(Class~., data = dat, method = "knn", trControl = trctrl, preProcess = c("center","scale"), tuneLength = 10)
fit.knn

## k-Nearest Neighbors
##
## 1098 samples
## 4 predictor
## 2 classes: ' authentic', ' inauthentic'
##
## Pre-processing: centered (4), scaled (4)
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 988, 988, 988, 989, 988, 988, ...
## Resampling results across tuning parameters:
##
## k Accuracy Kappa
## 5 0.9981790 0.9963203
## 7 0.9981790 0.9963203
## 9 0.9984821 0.9969324
## 11 0.9987879 0.9975515
## 13 0.9966611 0.9932625
```

```
##    15    0.9927106    0.9852944
##    17    0.9905838    0.9809989
##    19    0.9908924    0.9816240
##    21    0.9899750    0.9797750
##    23    0.9899750    0.9797750
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 11.
```

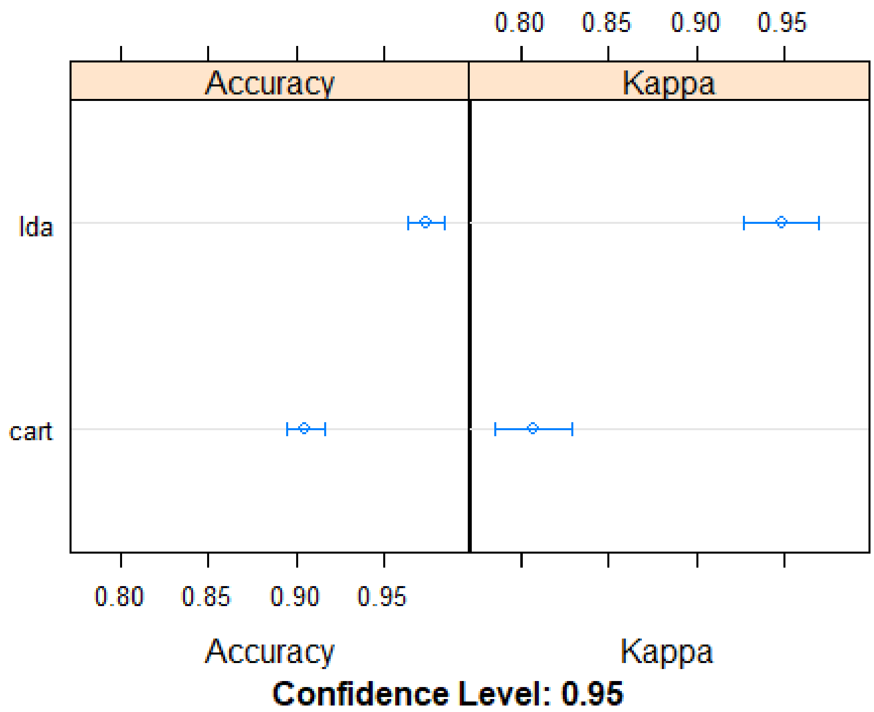
Select Best Models

```
## summarize accuracy of models
results = resamples(list(lda = fit.lda, cart = fit.cart))
summary(results)
```

```
##
## Call:
## summary.resamples(object = results)
##
## Models: lda, cart
## Number of resamples: 10
##
## Accuracy
##           Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## lda  0.9454545 0.9658465 0.9772727 0.9744871 0.9818182 1.0000000    0
## cart 0.8818182 0.8929525 0.9045455 0.9052627 0.9181818 0.9272727    0
##
## Kappa
```

```
##          Min.   1st Qu.    Median      Mean   3rd Qu.     Max. NA's
## lda  0.8909091 0.9313696 0.9542368 0.9487133 0.9633456 1.000000    0
## cart 0.7612688 0.7806125 0.8061784 0.8069803 0.8337232 0.851602    0

## compare accuracy of models
dotplot(results)
```




```
print(fit.lda)

## Linear Discriminant Analysis
##
## 1098 samples
## 4 predictor
## 2 classes: ' authentic', ' inauthentic'
##
## No pre-processing
```

```
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 988, 988, 988, 989, 988, 988, ...
## Resampling results:
##
## Accuracy Kappa
## 0.9744871 0.9487133
```

Make predictions

```
set.seed(4893)
## estimate skill of knn on the validation dataset
predictions = predict(fit.knn, validation)
confusionMatrix(predictions, validation$Class)
```

```
## Confusion Matrix and Statistics
##
## Reference
## Prediction authentic inauthentic
## authentic 152 0
## inauthentic 0 122
##
## Accuracy : 1
## 95% CI : (0.9866, 1)
## No Information Rate : 0.5547
## P-Value [Acc > NIR] : < 2.2e-16
##
## Kappa : 1
##
```

```
## Mcnemar's Test P-Value : NA
##
## Sensitivity : 1.0000
## Specificity : 1.0000
## Pos Pred Value : 1.0000
## Neg Pred Value : 1.0000
## Prevalence : 0.5547
## Detection Rate : 0.5547
## Detection Prevalence : 0.5547
## Balanced Accuracy : 1.0000
##
## 'Positive' Class : authentic
##
```

```
set.seed(4893)
predictions_2 = predict(fit.lda, validation)
confusionMatrix(predictions_2, validation$Class)
```

```
## Confusion Matrix and Statistics
##
## Reference
## Prediction authentic inauthentic
## authentic 148 0
## inauthentic 4 122
##
## Accuracy : 0.9854
## 95% CI : (0.963, 0.996)
```

```
##                   No Information Rate : 0.5547
##                   P-Value [Acc > NIR] : <2e-16
##
##                                 Kappa : 0.9705
##
##                Mcnemar's Test P-Value : 0.1336
##
##                           Sensitivity : 0.9737
##                           Specificity : 1.0000
##                        Pos Pred Value : 1.0000
##                        Neg Pred Value : 0.9683
##                            Prevalence : 0.5547
##                        Detection Rate : 0.5401
##                  Detection Prevalence : 0.5401
##                     Balanced Accuracy : 0.9868
##
##                      'Positive' Class : authentic
##
```