Counterfeits in Swiss Bank Notes

Yunyi Huang

Introduction

In this report, the Swiss bank notes dataset is analyzed. I wish to know whether or not we can predict whether a note is false or counterfeit using supervised learning. I will create logistic regression model and linear discriminant analysis model for helping me to answer the research question.

The dataset was originally extracted from “Multivariate Statistics: A practical approach”, by Bernhard Flury and Hans Riedwyl, Chapman and Hall, 1988, and contains six variables measured on 100 genuine and 100 counterfeit old Swiss 1000-franc bank notes:

1. Length: Length of the note
2. Left: Width of the Left-Hand side of the note
3. Right: Width of the Right-Hand side of the note
4. Bottom: Width of the Bottom Margin
5. Top: Width of the Top Margin
6. Diagonal: Diagonal Length of Printed Area

Analysis

Load & Read Data

```
notes <- read.table("data/SBN.txt")
head(notes)
```

```
##      Length  Left Right Bottom  Top Diagonal
## BN1   214.8 131.0 131.1    9.0  9.7    141.0
## BN2   214.6 129.7 129.7    8.1  9.5    141.7
## BN3   214.8 129.7 129.7    8.7  9.6    142.2
## BN4   214.8 129.7 129.6    7.5 10.4    142.0
## BN5   215.0 129.6 129.7   10.4  7.7    141.8
## BN6   215.7 130.8 130.5    9.0 10.1    141.4
```

I began by reading the data and take a look. There are 6 variables, besides index, of 200 observations of Swiss bank note.

Add a new column for counterfeits

Since the original data set does not contain a column for whether a Swiss bank note is counterfeit or not (the outcome variable), I added a column called “ctf” as the binary variable, and set the first 100 bank notes to 0 (for genuine), and set the last 100 bank notes to 1 (for counterfeit).

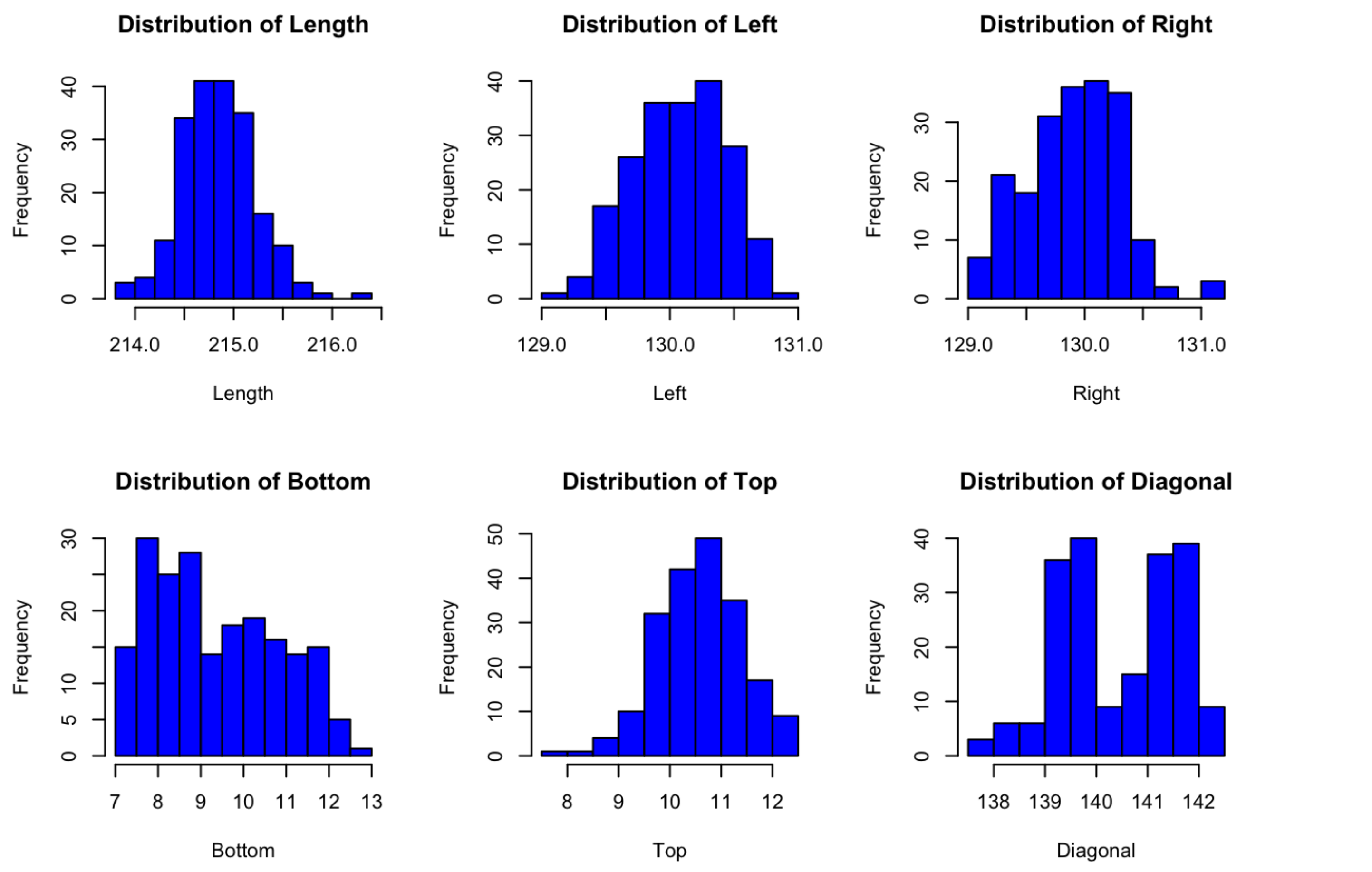
```
notes$num <- seq(1,200)
ctf <- ifelse(notes$num > 100, 1, 0)
notes <- cbind(notes, ctf)
notes <- subset(notes, select = -c(num))
head(notes)
```

```
##     Length  Left Right Bottom  Top Diagonal ctf
## BN1  214.8 131.0 131.1    9.0  9.7    141.0   0
## BN2  214.6 129.7 129.7    8.1  9.5    141.7   0
## BN3  214.8 129.7 129.7    8.7  9.6    142.2   0
## BN4  214.8 129.7 129.6    7.5 10.4    142.0   0
## BN5  215.0 129.6 129.7   10.4  7.7    141.8   0
## BN6  215.7 130.8 130.5    9.0 10.1    141.4   0
```

Visualization

Before formulating our test hypothesis, I first created some visualizations to get a brief picture of the data set.

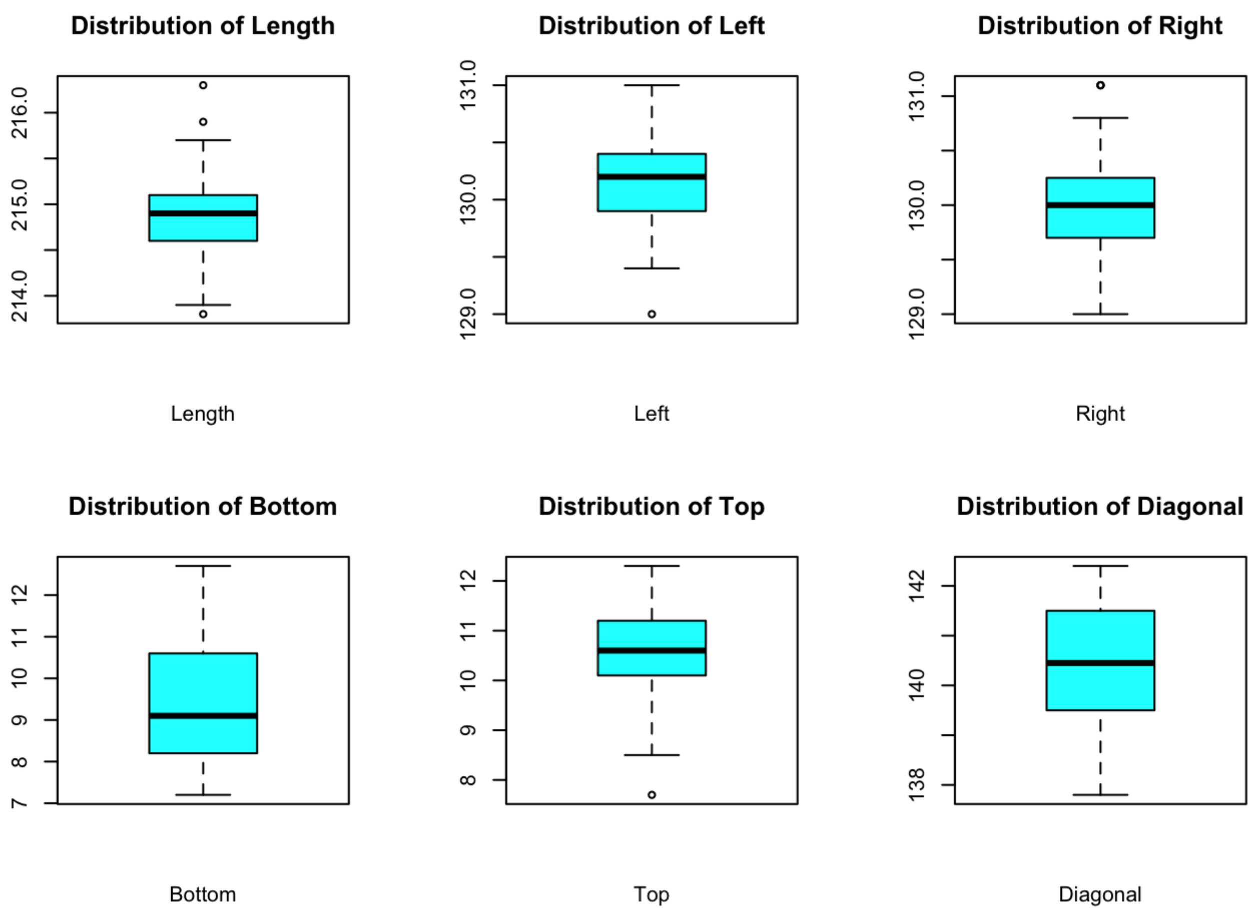
```
par(mfrow=c(2,3))
hist(notes$Length, main = "Distribution of Length", xlab = "Length", col = "blue")
hist(notes$Left, main = "Distribution of Left", xlab = "Left", col = "blue")
hist(notes$Right, main = "Distribution of Right", xlab = "Right", col = "blue")
hist(notes$Bottom, main = "Distribution of Bottom", xlab = "Bottom", col = "blue")
hist(notes$Top, main = "Distribution of Top", xlab = "Top", col = "blue")
hist(notes$Diagonal, main = "Distribution of Diagonal", xlab = "Diagonal", col = "blue")
```



From the plots above, I can see that Length, Right, and Bottom have a slight right-skewed distribution, and Left and Top have a slight left-skewed distribution. Interestingly, Diagonal has a bimodal distribution, which mens that there are two peaks in the histogram.

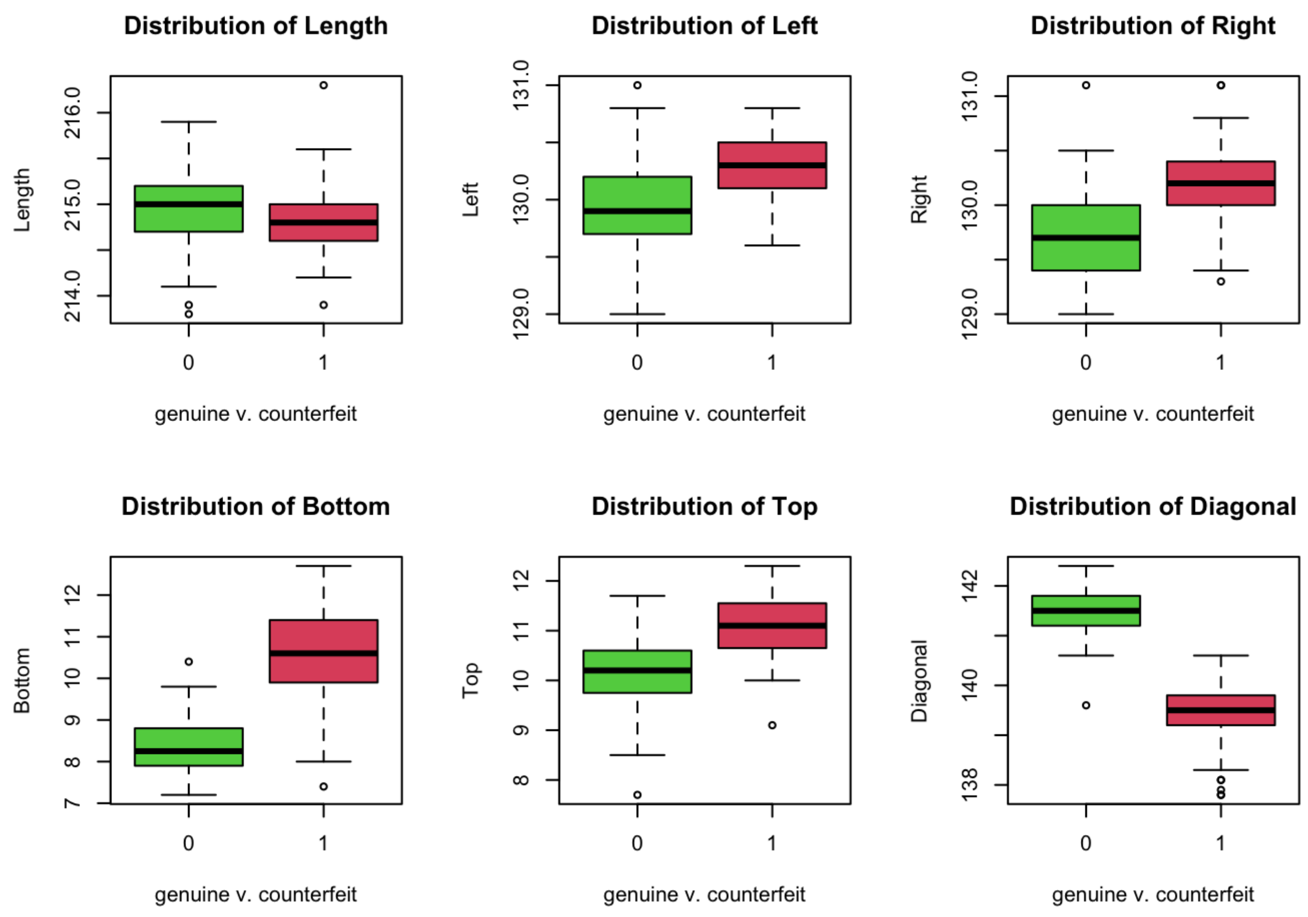
Since a Swiss bank note is rectangular, I was expecting that the distributions of Left and Right, also Bottom and Top, are identical to some extent. Noted that there are some differences among these variables.

```
par(mfrow=c(2,3))
boxplot(notes$Length, main = "Distribution of Length", xlab = "Length", col = "cyan")
boxplot(notes$Left, main = "Distribution of Left", xlab = "Left", col = "cyan")
boxplot(notes$Right, main = "Distribution of Right", xlab = "Right", col = "cyan")
boxplot(notes$Bottom, main = "Distribution of Bottom", xlab = "Bottom", col = "cyan")
boxplot(notes$Top, main = "Distribution of Top", xlab = "Top", col = "cyan")
boxplot(notes$Diagonal, main = "Distribution of Diagonal", xlab = "Diagonal", col = "cya
n")
```



However, the histograms and boxplots above for the data distributions are not enough to tell the details of the genuine and counterfeit bank notes. For better visualization, I divided the data into two parts: genuine and counterfeit. I will compare the two groups by different variables using boxplots.

```
par(mfrow = c(2,3))
boxplot(Length ~ ctf, data = notes, col= c(3,2), main = "Distribution of Length", xlab =
"genuine v. counterfeit")
boxplot(Left ~ ctf, data = notes, col= c(3,2), main = "Distribution of Left", xlab = "ge
nuine v. counterfeit")
boxplot(Right ~ ctf, data = notes, col= c(3,2), main = "Distribution of Right", xlab =
"genuine v. counterfeit")
boxplot(Bottom ~ ctf, data = notes, col= c(3,2), main = "Distribution of Bottom", xlab =
"genuine v. counterfeit")
boxplot(Top ~ ctf, data = notes, col= c(3,2), main = "Distribution of Top", xlab = "genu
ine v. counterfeit")
boxplot(Diagonal ~ ctf, data = notes, col= c(3,2), main = "Distribution of Diagonal", xl
ab = "genuine v. counterfeit")
```



By creating boxplots separately for genuine group (green) and counterfeit group (red), I can easily see the difference in data distributions between these two groups. The genuine group and counterfeit group nearly do not have overlaps in data distribution. For Length, their distribution is overlapped and their median is close. However, the other variables data show two clusters clearly. Noted that this may affect the model performance later.

```
# detailed information of genuine group
summary(notes[1:100,1:6])
```

```
##      Length           Left            Right           Bottom
##  Min.   :213.8   Min.   :129.0   Min.   :129.0   Min.   : 7.200
##  1st Qu.:214.7   1st Qu.:129.7   1st Qu.:129.4   1st Qu.: 7.900
##  Median :215.0   Median :129.9   Median :129.7   Median : 8.250
##  Mean   :215.0   Mean   :129.9   Mean   :129.7   Mean   : 8.305
##  3rd Qu.:215.2   3rd Qu.:130.2   3rd Qu.:130.0   3rd Qu.: 8.800
##  Max.   :215.9   Max.   :131.0   Max.   :131.1   Max.   :10.400
##       Top            Diagonal
##  Min.   : 7.700   Min.   :139.6
##  1st Qu.: 9.775   1st Qu.:141.2
##  Median :10.200   Median :141.5
##  Mean   :10.168   Mean   :141.5
##  3rd Qu.:10.600   3rd Qu.:141.8
##  Max.   :11.700   Max.   :142.4
```

```
# variance matrix
var(notes[1:100,1:6])
```

```
##                Length        Left       Right        Bottom         Top
## Length   0.150241414  0.05801313  0.05729293  0.0571262626  0.01445253
## Left     0.058013131  0.13257677  0.08589899  0.0566515152  0.04906667
## Right    0.057292929  0.08589899  0.12626263  0.0581818182  0.03064646
## Bottom   0.057126263  0.05665152  0.05818182  0.4132070707 -0.26347475
## Top      0.014452525  0.04906667  0.03064646 -0.2634747475  0.42118788
## Diagonal 0.005481818 -0.04306162 -0.02377778 -0.0001868687 -0.07530909
##                Diagonal
## Length    0.0054818182
## Left     -0.0430616162
## Right    -0.0237777778
## Bottom   -0.0001868687
## Top      -0.0753090909
## Diagonal  0.1998090909
```

```
# covariance matrix
cov(notes[1:100,1:6])
```

```
##                Length        Left       Right        Bottom         Top
## Length   0.150241414  0.05801313  0.05729293  0.0571262626  0.01445253
## Left     0.058013131  0.13257677  0.08589899  0.0566515152  0.04906667
## Right    0.057292929  0.08589899  0.12626263  0.0581818182  0.03064646
## Bottom   0.057126263  0.05665152  0.05818182  0.4132070707 -0.26347475
## Top      0.014452525  0.04906667  0.03064646 -0.2634747475  0.42118788
## Diagonal 0.005481818 -0.04306162 -0.02377778 -0.0001868687 -0.07530909
##                Diagonal
## Length    0.0054818182
## Left     -0.0430616162
## Right    -0.0237777778
## Bottom   -0.0001868687
## Top      -0.0753090909
## Diagonal  0.1998090909
```

```
# detailed information of counterfeit group
summary(notes[100:200,1:6])
```

```
##      Length          Left           Right          Bottom           Top       
##  Min.   :213.9   Min.   :129.6   Min.   :129.3   Min.   : 7.4   Min.   : 9.10  
##  1st Qu.:214.6   1st Qu.:130.1   1st Qu.:130.0   1st Qu.: 9.9   1st Qu.:10.60  
##  Median :214.8   Median :130.3   Median :130.2   Median :10.6   Median :11.10  
##  Mean   :214.8   Mean   :130.3   Mean   :130.2   Mean   :10.5   Mean   :11.12  
##  3rd Qu.:215.0   3rd Qu.:130.5   3rd Qu.:130.4   3rd Qu.:11.4   3rd Qu.:11.50  
##  Max.   :216.3   Max.   :130.8   Max.   :131.1   Max.   :12.7   Max.   :12.30  
##     Diagonal    
##  Min.   :137.8  
##  1st Qu.:139.2  
##  Median :139.5  
##  Mean   :139.5  
##  3rd Qu.:139.8  
##  Max.   :141.2
```

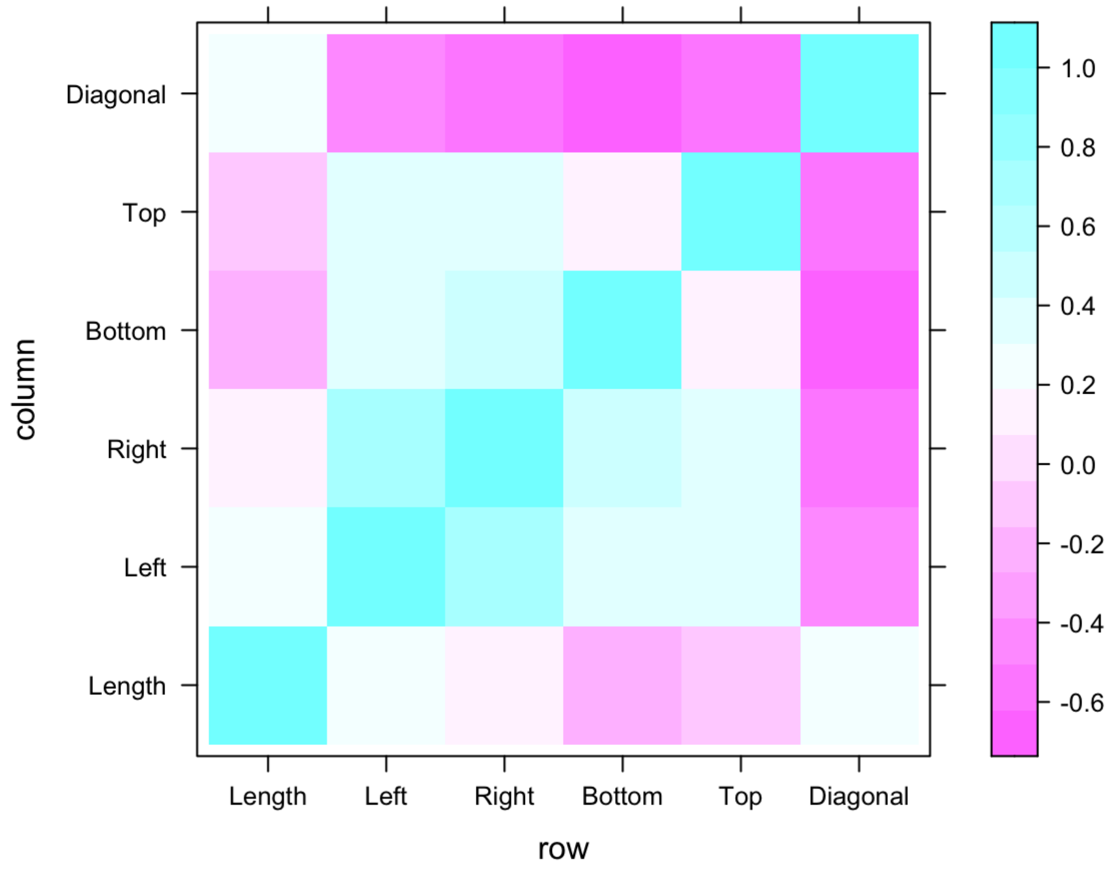
```
# variance matrix
var(notes[100:200,1:6])
```

```
##                Length         Left       Right       Bottom          Top
## Length    0.122920792  0.031565347 0.024726733 -0.096265347  0.020620792
## Left      0.031565347  0.065291089 0.048655446 -0.015691089 -0.008434653
## Right     0.024726733  0.048655446 0.094277228  0.003044554  0.009026733
## Bottom   -0.096265347 -0.015691089 0.003044554  1.342291089 -0.454665347
## Top       0.020620792 -0.008434653 0.009026733 -0.454665347  0.413120792
## Diagonal  0.009318812 -0.010198020 0.020109901  0.188798020 -0.041481188
##              Diagonal
## Length    0.009318812
## Left     -0.010198020
## Right     0.020109901
## Bottom    0.188798020
## Top      -0.041481188
## Diagonal  0.338421782
```

```
# covariance matrix
cov(notes[100:200,1:6])
```

```
##                Length         Left       Right       Bottom          Top
## Length    0.122920792  0.031565347 0.024726733 -0.096265347  0.020620792
## Left      0.031565347  0.065291089 0.048655446 -0.015691089 -0.008434653
## Right     0.024726733  0.048655446 0.094277228  0.003044554  0.009026733
## Bottom   -0.096265347 -0.015691089 0.003044554  1.342291089 -0.454665347
## Top       0.020620792 -0.008434653 0.009026733 -0.454665347  0.413120792
## Diagonal  0.009318812 -0.010198020 0.020109901  0.188798020 -0.041481188
##              Diagonal
## Length    0.009318812
## Left     -0.010198020
## Right     0.020109901
## Bottom    0.188798020
## Top      -0.041481188
## Diagonal  0.338421782
```

```
library("lattice")
levelplot(cor(notes[,1:6]))
```



In the plot above, the light blue color represents strong correlation between two variables, and purple represents weak correlation between two variables. I can see that most of the variables have some correlation. However, the variable of Diagonal seems to have really low correlation with other variables. Also, the variable of Length has low correlation with other variables as well. For Top, Bottom, Right, and Left, these four variable seem to have considerable correlation with each other. It is useful for the assumptions section later.

```
library(lattice)
library(ellipse)
```

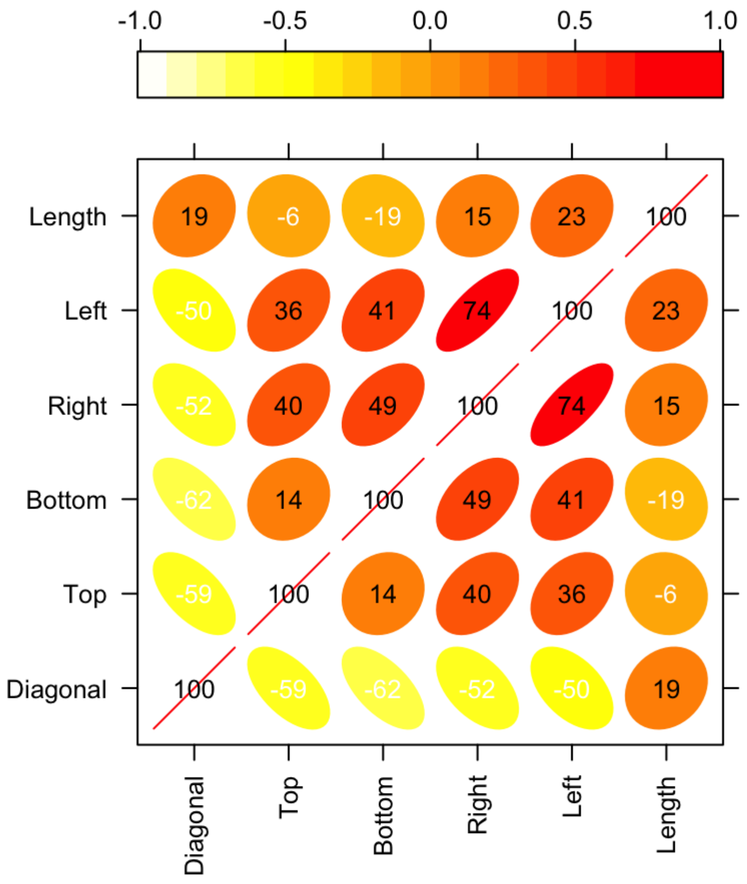
```
##
## Attaching package: 'ellipse'
```

```
## The following object is masked from 'package:graphics':
##
##     pairs
```

```r
cor_df <- cor(notes[,1:6])

#Function to generate correlation plot
panel.corrgram <- function(x,y,z,subscripts,at,level=0.9,label=FALSE,...){
require("ellipse", quietly=TRUE)
  x<-as.numeric(x)[subscripts]
  y<-as.numeric(y)[subscripts]
  z<-as.numeric(z)[subscripts]
  zcol<-level.colors(z, at=at, ...)
  for (i in seq(along=z)) {
    ell=ellipse(z[i], level=level, npoints=50,
                scale=c(.2,.2), centre=c(x[i], y[i]))
    panel.polygon(ell, col=zcol[i], border=zcol[i], ...)
  }
  if (label)
    panel.text(x=x, y=y, lab=100*round(z,2),cex=0.8,
               col=ifelse(z<0, "White", "Black"))
}

##generate correlation plot
print(levelplot(cor_df[seq(6,1), seq(6,1)], at=do.breaks(c(-1.01,1.01), 20), xlab=NULL,
 ylab=NULL, colorkey=list(space="top"), col.regions=rev(heat.colors(100)),
                scale=list(x=list(rot=90)),
                panel=panel.corrgram, label=TRUE))
```



LDA & Logistic Regression

For this part of the analysis, I chose to use the caret package to implement the models.

```
# import package
library(caret)
```

```
## Loading required package: ggplot2
```

```
# shuffle the original data and set random seed
set.seed(1192)
rows <- sample(nrow(notes))
notes <- notes[rows,]
head(notes)
```

```
##       Length  Left Right Bottom  Top Diagonal ctf
## BN92   215.4 130.0 129.9    8.5  9.7    141.4   0
## BN131  214.3 130.2 130.0   10.7 10.5    139.8   1
## BN5    215.0 129.6 129.7   10.4  7.7    141.8   0
## BN164  214.7 130.1 130.2   11.6 10.9    139.1   1
## BN37   215.5 130.3 130.0    8.4  9.7    141.8   0
## BN171  213.9 130.7 130.5    8.7 11.5    137.8   1
```

The data has been shuffled above, and its purpose is to make sure the training and validation sets have some of the genuine and counterfeit notes.

Dividing Data with K-fold Cross-Validation

```
# Relabel values of outcome (0 = No, 1 = Yes)
notes$ctf[notes$ctf==0] <- "No"
notes$ctf[notes$ctf==1] <- "Yes"

# Convert outcome variable to type factor
notes$ctf <- as.factor(notes$ctf)

# Specify the type of training method used
# K-fold Cross-Validation Implementation
ctrlspecs <- trainControl(method = "cv", number = 10,
                          savePredictions = "all",
                          classProbs = TRUE)
```

Logistic Regression Model

```
# Set random seed
set.seed(788)
# Specify Logistic Regression Model
model1 <- train(ctf ~ Length + Left + Right + Bottom + Top + Diagonal,
                data = notes,
                method = "glm",
                family = binomial,
                trControl = ctrlspecs)
model1
```

```
## Generalized Linear Model
##
## 200 samples
##   6 predictor
##   2 classes: 'No', 'Yes'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 180, 180, 180, 180, 180, 180, ...
## Resampling results:
##
##   Accuracy  Kappa
##   0.985     0.97
```

According to the results above, the logistic regression model has an overall accuracy of 98.5% with a Kappa score of 0.97, indicating that it is an almost perfect model.

```
# summary of this model
summary(model1)
```

```
##
## Call:
## NULL
##
## Deviance Residuals:
##        Min          1Q      Median          3Q         Max
## -8.216e-05  -2.100e-08   0.000e+00   2.100e-08   7.237e-05
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.013e+03  3.063e+07   0.000    1.000
## Length       3.084e+01  1.666e+05   0.000    1.000
## Left        -1.075e+01  3.254e+05   0.000    1.000
## Right        1.056e+01  2.541e+05   0.000    1.000
## Bottom       5.876e+01  4.354e+04   0.001    0.999
## Top          4.983e+01  3.730e+04   0.001    0.999
## Diagonal    -4.040e+01  3.655e+04  -0.001    0.999
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2.7726e+02  on 199  degrees of freedom
## Residual deviance: 2.0593e-08  on 193  degrees of freedom
## AIC: 14
##
## Number of Fisher Scoring iterations: 25
```

```
# variable importance
varImp(model1)
```

```
## glm variable importance
##
##           Overall
## Bottom   100.0000
## Top       98.9740
## Diagonal  81.4394
## Length    11.5513
## Right      0.6491
## Left       0.0000
```

From the variance importance score above, I can see that the Bottom variable is the most important for variance, and Left variable is the least importance.

```
# check the accuracy for each fold
glm_fold_accuracy1 <- model1$resample
glm_fold_accuracy1
```

```
##    Accuracy Kappa Resample
## 1      1.00   1.0   Fold01
## 2      1.00   1.0   Fold02
## 3      1.00   1.0   Fold03
## 4      1.00   1.0   Fold04
## 5      0.95   0.9   Fold05
## 6      1.00   1.0   Fold06
## 7      0.95   0.9   Fold07
## 8      1.00   1.0   Fold08
## 9      1.00   1.0   Fold09
## 10     0.95   0.9   Fold10
```

The table above shows this model’s accuracy for each fold through the 10-fold cross validation. I saved this result to a variable for later comparison and analysis.

Linear Discriminant Analysis Model

```
# Set random seed
set.seed(788)
# Specify Linear Discriminant Analysis Model
model2 <- train(ctf ~ Length + Left + Right + Bottom + Top + Diagonal,
                data = notes,
                method = "lda",
                family = binomial,
                trControl = ctrlspecs)
model2
```

```
## Linear Discriminant Analysis
##
## 200 samples
##   6 predictor
##   2 classes: 'No', 'Yes'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 180, 180, 180, 180, 180, 180, ...
## Resampling results:
##
##   Accuracy  Kappa
##   0.995     0.99
```

According to the results above, the LDA model has an overall accuracy of 99.5% with a Kappa score of 0.99, indicating that it is an almost perfect model.

```
# summary of this model
summary(model2)
```

```
##             Length Class      Mode
## prior        2     -none-     numeric
## counts       2     -none-     numeric
## means       12     -none-     numeric
## scaling      6     -none-     numeric
## lev          2     -none-     character
## svd          1     -none-     numeric
## N            1     -none-     numeric
## call         4     -none-     call
## xNames       6     -none-     character
## problemType  1     -none-     character
## tuneValue    1     data.frame list
## obsLevels    2     -none-     character
## param        1     -none-     list
```

```
varImp(model2)
```

```
## ROC curve variable importance
##
##          Importance
## Diagonal     100.00
## Bottom        85.24
## Top           62.29
## Right         61.02
## Left          43.98
## Length         0.00
```

From the variance importance score above, I can see that the Diagonal variable is the most important for variance, and Length variable is the least importance.

```
# check the accuracy for each fold
lda_fold_accuracy1 <- model2$resample
lda_fold_accuracy1
```

```
##    Accuracy Kappa Resample
## 1      1.00   1.0   Fold01
## 2      1.00   1.0   Fold02
## 3      1.00   1.0   Fold03
## 4      1.00   1.0   Fold04
## 5      0.95   0.9   Fold05
## 6      1.00   1.0   Fold06
## 7      1.00   1.0   Fold07
## 8      1.00   1.0   Fold08
## 9      1.00   1.0   Fold09
## 10     1.00   1.0   Fold10
```

The table above shows this model’s accuracy for each fold through the 10-fold cross validation. I saved this result to a variable for later comparison and analysis.

Using Factor Model & Reduce Dimensions

In the previous session, I used all six of the variables as the input variables in training data, which is high dimensional. It is always useful to reduce the dimension of data to get a better sense of it. To accomplish this, I chose to do a principle component analysis and a maximum likelihood estimation.

Principle Component Analysis

```
library(factoextra)
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WB
a
```

```
# drop the outcome column for convenience
new_notes <- notes[,1:6]

# compute PCA
notes_pca <- prcomp(new_notes, scale = TRUE)
notes_pca
```

```
## Standard deviations (1, .., p=6):
## [1] 1.7162629 1.1305237 0.9322192 0.6706480 0.5183405 0.4346031
##
## Rotation (n x k) = (6 x 6):
##                   PC1         PC2         PC3        PC4        PC5         PC6
## Length   -0.006987029  0.81549497  0.01768066  0.5746173 -0.0587961  0.03105698
## Left      0.467758161  0.34196711 -0.10338286 -0.3949225  0.6394961 -0.29774768
## Right     0.486678705  0.25245860 -0.12347472 -0.4302783 -0.6140972  0.34915294
## Bottom    0.406758327 -0.26622878 -0.58353831  0.4036735 -0.2154756 -0.46235361
## Top       0.367891118 -0.09148667  0.78757147  0.1102267 -0.2198494 -0.41896754
## Diagonal -0.493458317  0.27394074 -0.11387536 -0.3919305 -0.3401601 -0.63179849
```

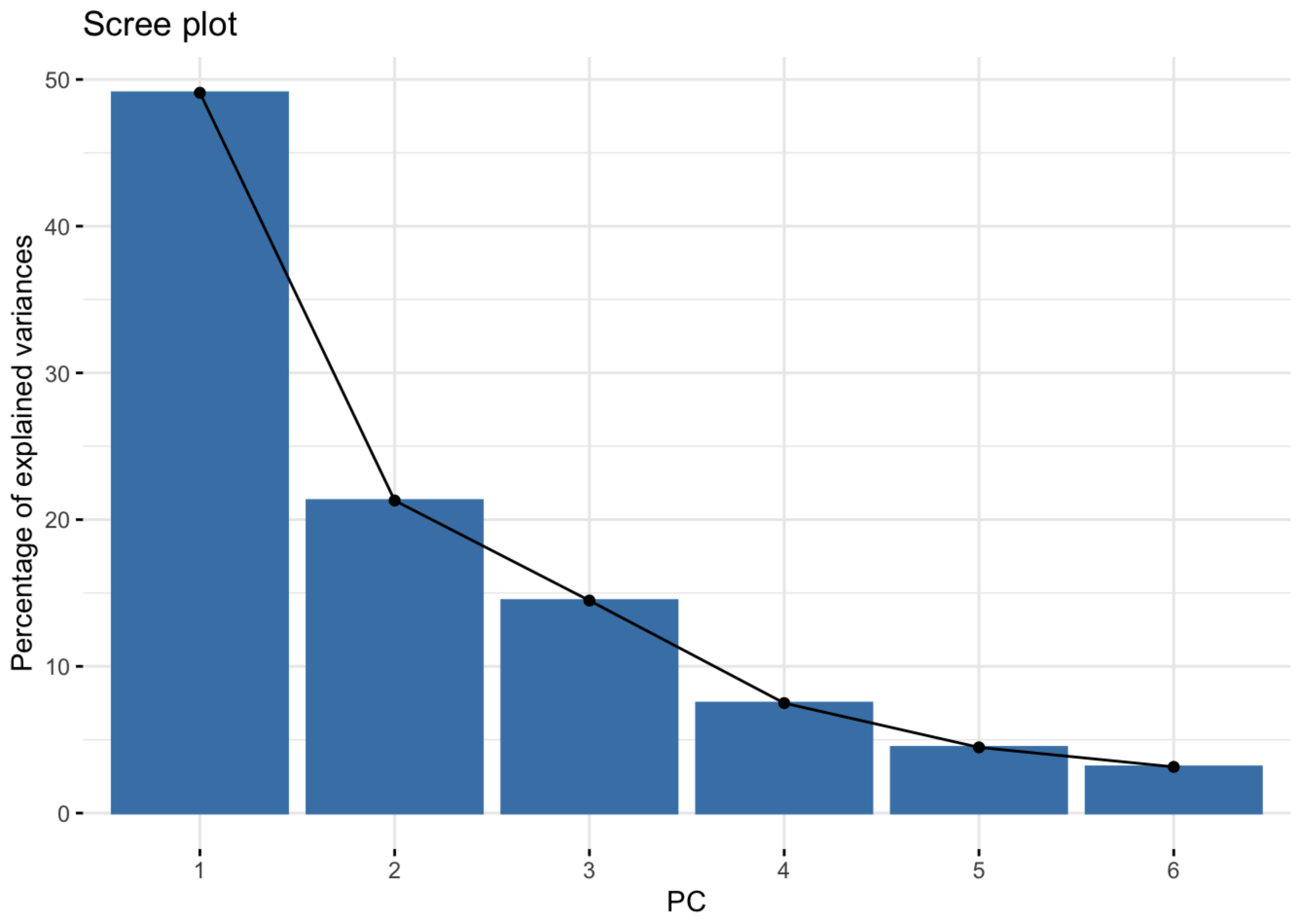
The results above show how each principle components affects different variables, and I made a scree plot for visualization for better understanding.

```
get_eig(notes_pca)
```

```
##       eigenvalue variance.percent cumulative.variance.percent
## Dim.1  2.9455582        49.092637                    49.09264
## Dim.2  1.2780838        21.301396                    70.39403
## Dim.3  0.8690326        14.483876                    84.87791
## Dim.4  0.4497687         7.496145                    92.37405
## Dim.5  0.2686769         4.477948                    96.85200
## Dim.6  0.1888799         3.147998                   100.00000
```

The table above provides the eigenvalue, variance percent, and cumulative variance percent.

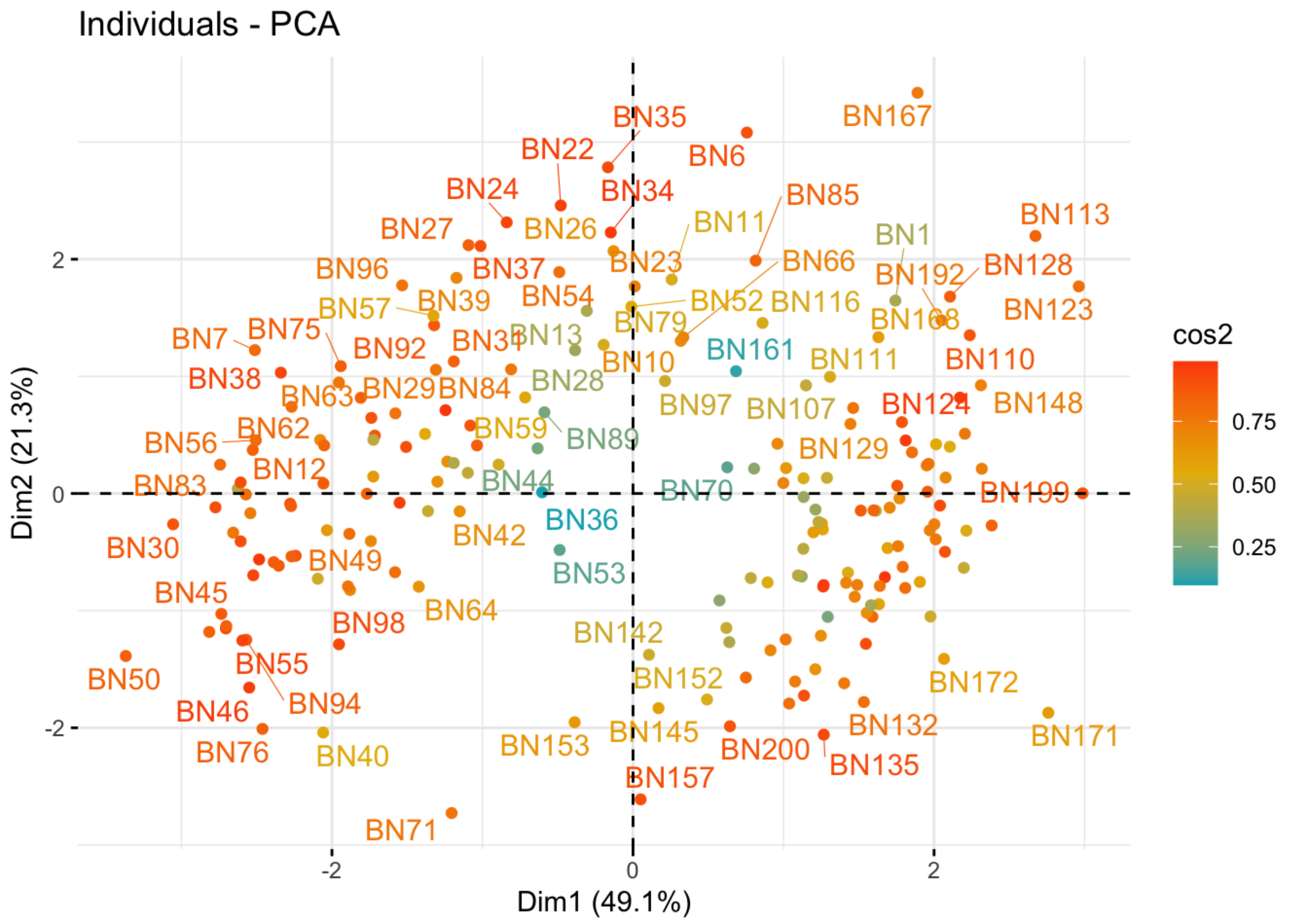
```
# scree plot
# show the percentage of variances explained by each principal component
fviz_eig(notes_pca, xlab = "PC")
```



From the scree plot above, I can see that there is a “elbow” occurred on PC2’s data point. Specifically, PC1 accounts for nearly 50% of the variance, and for PC2 it is little above 20% of the variance.

```
# graph of individuals
fviz_pca_ind(notes_pca,
             col.ind = "cos2", # Color by the quality of representation
             gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
             repel = TRUE     # Avoid text overlapping
             )
```

```
## Warning: ggrepel: 121 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



From the graph above, I can see that individuals with a similar profile are grouped together based on the first two principle components.

Maximum Likelihood Estimation

Using the implication from PCA above, I would set the number of factors to 2.

```
# Maximum likelihood
n.factors <- 2
fa_fit <- factanal(new_notes, n.factors, scores = "regression", rotation = "varimax")
fa_fit
```

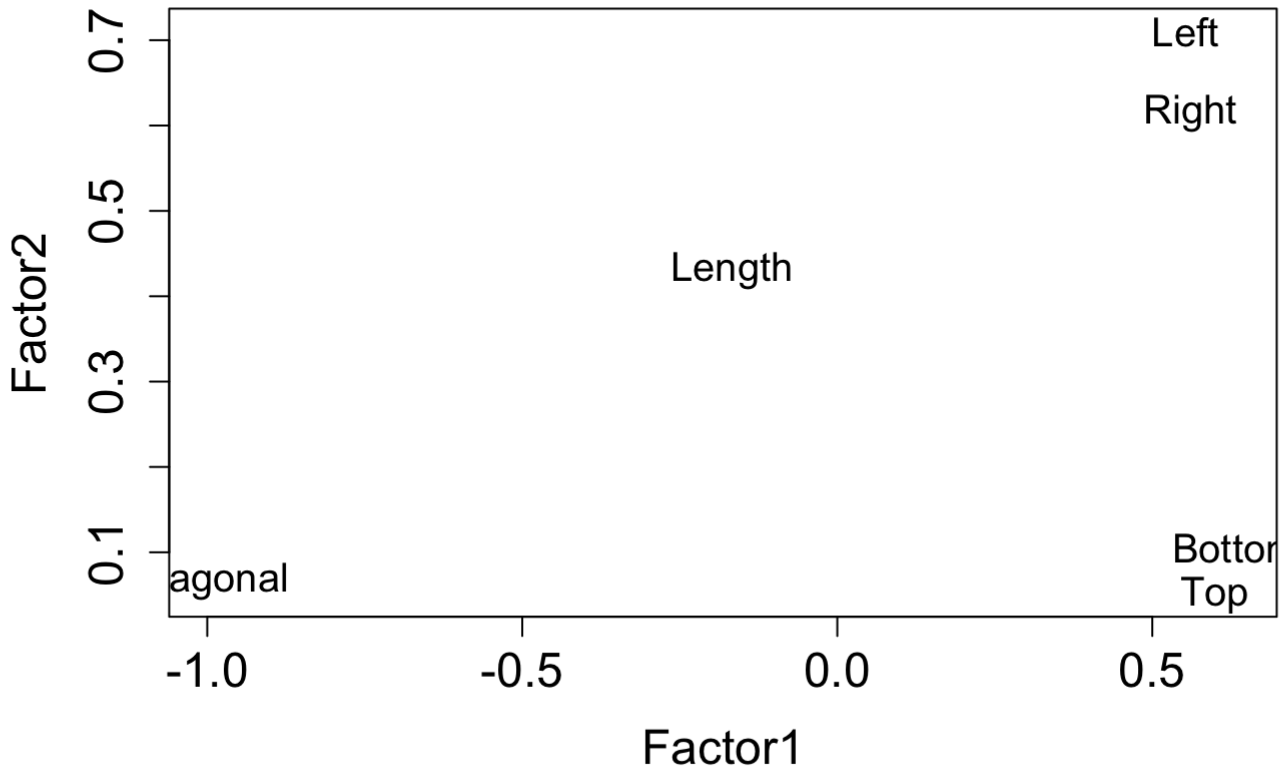
```
##
## Call:
## factanal(x = new_notes, factors = n.factors, scores = "regression",     rotation = "v
arimax")
##
## Uniquenesses:
##   Length     Left    Right   Bottom      Top Diagonal
##    0.787    0.190    0.309    0.589    0.638    0.005
##
## Loadings:
##          Factor1 Factor2
## Length   -0.167   0.431
## Left      0.553   0.711
## Right     0.560   0.614
## Bottom    0.632   0.105
## Top       0.599
## Diagonal -0.995
##
##                Factor1 Factor2
## SS loadings      2.397   1.085
## Proportion Var   0.399   0.181
## Cumulative Var   0.399   0.580
##
## Test of the hypothesis that 2 factors are sufficient.
## The chi square statistic is 51.41 on 4 degrees of freedom.
## The p-value is 1.83e-10
```

```
# check the factor loading
loading <- fa_fit$loadings[, 1:2]
t(loading)
```

```
##             Length      Left     Right    Bottom        Top    Diagonal
## Factor1 -0.1670846 0.5527990 0.5603285 0.6323679 0.59914467 -0.99529478
## Factor2  0.4305069 0.7105278 0.6142669 0.1047625 0.05087066  0.06627958
```

As suggested by the results, Factor 1 contributed to 39.9% of the proportion variance, and Factor 2 has 18%.

```
# visualize the loading
plot(loading, type = 'n', cex.axis=1.5, cex.lab=1.5)
text(loading, labels=names(new_notes), cex=1.25)
```



The visualization above shows the variable relationships with two factors. I can see that Left and Right load heavily on both factors, Bottom and Top load heavily on Factor 1, Length loads more on Factor 2, and Diagonal load more on Factor 2 as well.

Re-run analysis on Factor Scores

In this part, I will re-run both models on the factor scores of Factor 1 and Factor 2 I got from the previous section.

Modified the dataframe

```
# Add the factor scores into the dataframe
scores <- fa_fit$scores
notes<- cbind(notes, scores)

# shuffle the original data
set.seed(1192)
rows <- sample(nrow(notes))
notes <- notes[rows,]
head(notes)
```

```
##       Length  Left Right Bottom  Top Diagonal ctf    Factor1    Factor2
## BN134  214.6 130.2 130.4   10.5 11.8    139.7 Yes  0.6969301  0.1209735
## BN118  214.7 130.4 130.1   12.1 10.4    139.9 Yes  0.5379739  0.3739718
## BN37   215.5 130.3 130.0    8.4  9.7    141.8  No -1.0636706  1.2122022
## BN11   215.3 130.4 130.3    7.9 11.7    141.8  No -1.0201000  1.6762754
## BN198  214.8 130.3 130.4   10.6 11.1    140.0 Yes  0.4607311  0.5154474
## BN163  214.6 130.1 130.0   11.5 10.6    139.5 Yes  0.8176676 -0.5606407
```

Dividing Data with K-fold Cross-Validation

```
# Relabel values of outcome (0 = No, 1 = Yes)
notes$ctf[notes$ctf==0] <- "No"
notes$ctf[notes$ctf==1] <- "Yes"

# Convert outcome variable to type factor
notes$ctf <- as.factor(notes$ctf)

# Specify the type of training method used
# K-fold Cross-Validation Implementation
ctrlspecs <- trainControl(method = "cv", number = 10,
                          savePredictions = "all",
                          classProbs = TRUE)
```

New Logistic Regression Model

```
# Set random seed
set.seed(788)
# Specify Logistic Regression Model
model1 <- train(ctf ~ Factor1+Factor2,
                data = notes,
                method = "glm",
                family = binomial,
                trControl = ctrlspecs)
model1
```

```
## Generalized Linear Model
##
## 200 samples
##   2 predictor
##   2 classes: 'No', 'Yes'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 180, 180, 180, 180, 180, 180, ...
## Resampling results:
##
##   Accuracy  Kappa
##   0.99      0.98
```

According to the results above, the logistic regression model has an overall accuracy of 99% with a Kappa score of 0.98, indicating that it is an almost perfect model.

```
# summary of this model
summary(model1)
```

```
##
## Call:
## NULL
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.3443  -0.0071  -0.0001   0.0668   0.8952
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -1.4104     1.3110  -1.076 0.282019
## Factor1       9.4684     2.6515   3.571 0.000356 ***
## Factor2       0.9077     0.8290   1.095 0.273572
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 277.26  on 199  degrees of freedom
## Residual deviance:  16.48  on 197  degrees of freedom
## AIC: 22.48
##
## Number of Fisher Scoring iterations: 10
```

```
# variable importance
varImp(model1)
```

```
## glm variable importance
##
##         Overall
## Factor1     100
## Factor2       0
```

From the variance importance score above, I can see that Factor 1 is the most important for variance, and Factor 2 is the least importance.

```
# check the accuracy for each fold
glm_fold_accuracy2 <- model1$resample
glm_fold_accuracy2
```

```
##    Accuracy Kappa Resample
## 1      1.00   1.0   Fold01
## 2      1.00   1.0   Fold02
## 3      1.00   1.0   Fold03
## 4      1.00   1.0   Fold04
## 5      1.00   1.0   Fold05
## 6      0.95   0.9   Fold06
## 7      1.00   1.0   Fold07
## 8      0.95   0.9   Fold08
## 9      1.00   1.0   Fold09
## 10     1.00   1.0   Fold10
```

The table above shows this model’s accuracy for each fold through the 10-fold cross validation. I saved this result to a variable for later comparison and analysis.

New Linear Discriminant Analysis Model

```
# Set random seed
set.seed(788)
# Specify Linear Discriminant Analysis Model
model2 <- train(ctf ~ Factor1+Factor2,
                data = notes,
                method = "lda",
                trControl = ctrlspecs)
model2
```

```
## Linear Discriminant Analysis
##
## 200 samples
##   2 predictor
##   2 classes: 'No', 'Yes'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 180, 180, 180, 180, 180, 180, ...
## Resampling results:
##
##   Accuracy  Kappa
##   0.995     0.99
```

According to the results above, the LDA model has an overall accuracy of 99.5% with a Kappa score of 0.99, indicating that it is an almost perfect model.

```
# summary of this model
summary(model2)
```

```
##             Length Class      Mode
## prior       2      -none-     numeric
## counts      2      -none-     numeric
## means       4      -none-     numeric
## scaling     2      -none-     numeric
## lev         2      -none-     character
## svd         1      -none-     numeric
## N           1      -none-     numeric
## call        3      -none-     call
## xNames      2      -none-     character
## problemType 1      -none-     character
## tuneValue   1      data.frame list
## obsLevels   2      -none-     character
## param       0      -none-     list
```

```
# variable importance
varImp(model2)
```

```
## ROC curve variable importance
##
##         Importance
## Factor1        100
## Factor2          0
```

From the variance importance score above, I can see that Factor 1 is the most important for variance, and Factor 2 is the least importance.

```
# check the accuracy for each fold
lda_fold_accuracy2 <- model2$resample
lda_fold_accuracy2
```

```
##    Accuracy Kappa Resample
## 1      1.00   1.0   Fold01
## 2      1.00   1.0   Fold02
## 3      1.00   1.0   Fold03
## 4      1.00   1.0   Fold04
## 5      1.00   1.0   Fold05
## 6      0.95   0.9   Fold06
## 7      1.00   1.0   Fold07
## 8      1.00   1.0   Fold08
## 9      1.00   1.0   Fold09
## 10     1.00   1.0   Fold10
```

The table above shows this model’s accuracy for each fold through the 10-fold cross validation. I saved this result to a variable for later comparison and analysis.

Discuss Assumptions

(For convenience, most of the graphs and tables mentioned in justifications are already done in the Visualization Section)

For Logistic Regression

The assumptions for logistic regression are listed below, as well as the justifications:

1. The outcome is a binary or dichotomous variable like 1 v. 0.

Justification: the outcome variable is whether Swiss bank note is genuine or counterfeit, taking values of 0 (genuine) and 1 (counterfeit), which is binary. This assumption is satisfied.

2. The observations are independent of each other.

Justification: in the original data set, the index names are like “BN1”, “BN2”, and “BN3”, etc. which indicates that each observations are unique and independent. This assumption is satisfied.

3. Little to no multicollinearity among the independent variables.

Justification: according the levelplot I have made in the project’s visualization section, the variables have low correlations with each other. This assumption is satisfied.

4. Linearity of independent variables and log odds.

Justification: I do know the exact relationship between each predictor variable and the log odds of the outcome, but for perform an analysis, I can assume that there is a linear relationship between independent variables and log odds.

5. A large sample size.

Justification: there are 200 observations in the dataset, which is large enough. This assumption is satisfied.

For Linear Discriminant Analysis

The assumptions for LDA are listed below, as well as the justifications:

1. The dependent variable Y is discrete.

Justification: the dependent variable Y in this analysis is whether Swiss bank note is genuine or counterfeit, taking values of 0 (genuine) and 1 (counterfeit), which is discrete. This assumption is satisfied.

2. The independent variable(s) X come from Gaussian distributions.

Justification: Gaussian distributions means that the values for each variables have a normal distribution. In the visualization section of this report, the histograms show that most of the variables have a normal distribution except Diagonal. I can still say that this assumption is satified, but there might be some potential errors since not all variables have normal distributions.

3. Same Variance

Justification: according to the variance and covariance matrix in this project’s visualization section, there is same variance among the variables. This assumption is satisfied.

For Principle Component Analysis

The assumptions for PCA are listed below, as well as the justifications:

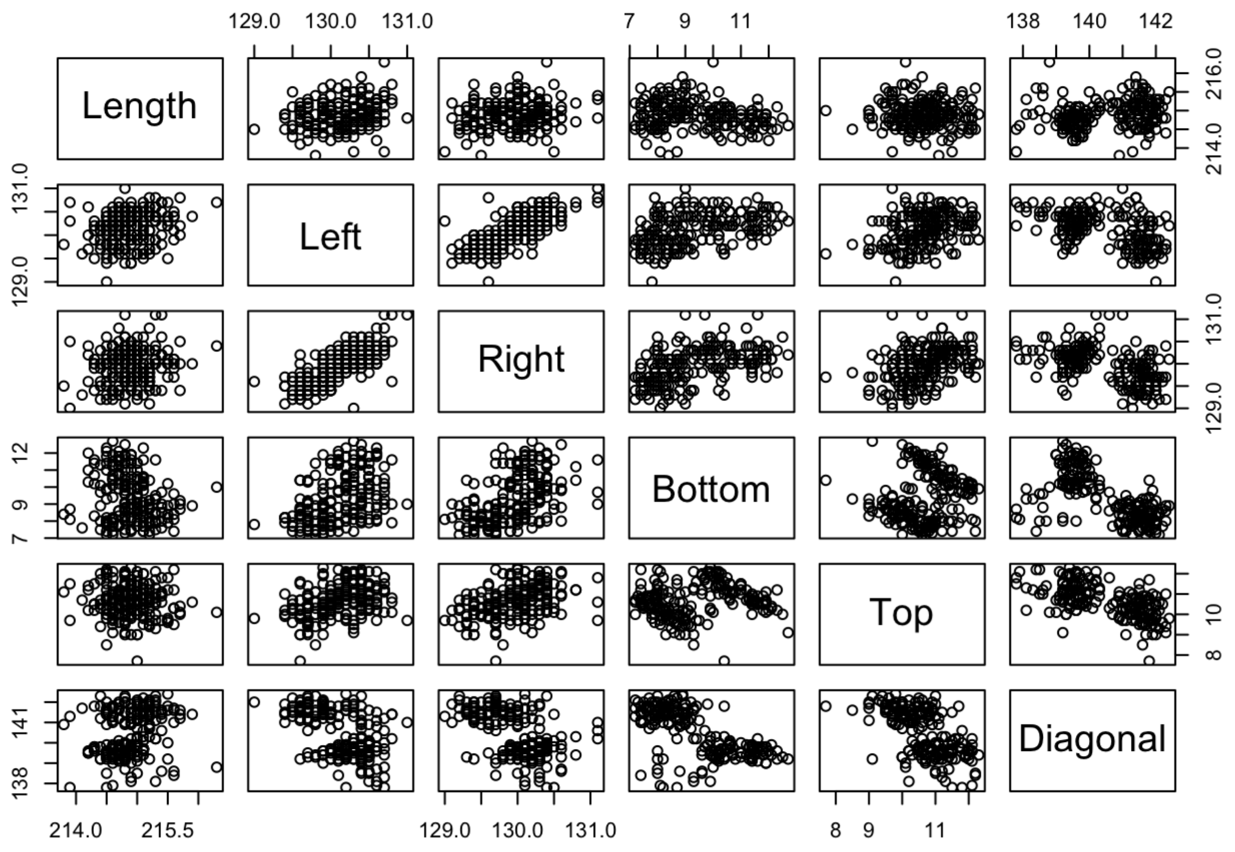
1. There are multiple variables that should be measured at the continuous level.

Justification: since the variables are all measured in length unit, whic are continuous. This assumption is satisfied.

2. There needs to be a linear relationship between all vairables.

Justification: according to the plots below, I can assume that there is a linear relationship between all vairables except for the variable of Diagonal. However, I can still say that this assumption is satified, but there might be some potential errors since not all variables have a linear relationship with other variables.

```
pairs(notes[,1:6])
```



3. There is a sampling accuracy.

Justification: there are 200 observations in the dataset, which is large enough. This assumption is satisfied.

4. The data set should be suitable for data reduction.

Justification: according to the levelplot in the project's visualization section, I can see that there are adequate correlations between the variables to be reduced to a samaller number of components. This assumption is satisfied.

5. There should be no significant outliers.

Justification: according to the boxplots in the project's visualization section, I can see that most of the variables have no significant outliers but some milds ones. However, these might cause some potential errors in the analysis.

For Maximum Likelihood Estimation

The underlying assumption of MLE is that the data are independently sampled from a multivariate normal distribution with mean vector $\underline{\mu}$ and variance-covariance matrix $\mathbf{\Sigma}$ of the form:

$$\mathbf{\Sigma} = \mathbf{L}\,\mathbf{L}' + \mathbf{\Psi}.$$

Justification: in the original data set, the index names are like “BN1”, “BN2”, and “BN3”, etc. which indicates that each observations are unique and independent. This also means that the data are independently distributed, which was shown by the histograms in the project’s visualization section. This assumption is satisfied.

Does the factor analysis help, or is it a waste of time?

From the results from the models before and after the factor analysis, it seems to be that the factor analysis does help the logistic regression model, but does not help the LDA model. I would say that the factor analysis helps even though the accuracies are already considerably high for the models before the dimension reduction. In the process of factor analysis, I am able to find out which variables are the most essential to the outcome, and therefore I can reduce the number of input variables. This process has improved the interpretation of the parameters of the model, making it easier to visualize the data by lowering the dimension, thereby reducing redundancy and space complexity.

To arrive at a final model for each fold, the following section will complete this task.

Combine Results

```
# combine all fold accuracy for each model into one dataframe
folds_accuracy <- cbind(glm_fold_accuracy1[,1],glm_fold_accuracy2[,1],lda_fold_accuracy1
[,1],lda_fold_accuracy2[,1])
colnames(folds_accuracy) <- c("glm1", "glm2", "lda1", "lda2")
rownames(folds_accuracy) <- c("Fold 1", "Fold 2", "Fold 3", "Fold 4", "Fold 5", "Fold 6"
, "Fold 7", "Fold 8", "Fold 9", "Fold 10")
folds_accuracy <- as.data.frame(folds_accuracy)
folds_accuracy
```

```
##         glm1 glm2 lda1 lda2
## Fold 1  1.00 1.00 1.00 1.00
## Fold 2  1.00 1.00 1.00 1.00
## Fold 3  1.00 1.00 1.00 1.00
## Fold 4  1.00 1.00 1.00 1.00
## Fold 5  0.95 1.00 0.95 1.00
## Fold 6  1.00 0.95 1.00 0.95
## Fold 7  0.95 1.00 1.00 1.00
## Fold 8  1.00 0.95 1.00 1.00
## Fold 9  1.00 1.00 1.00 1.00
## Fold 10 0.95 1.00 1.00 1.00
```

The column names above stand for: 1. glm1: original logistic regression model 2. glm2: modified logistic regression model 3. lda1: original LDA model 4. lda2: modified LDA model

The table above sums up the accuracy of each fold under four models (original and modified logistic regression model, original and modified LDA model). As the table suggests, glm 1 has 100% accuracy for most of the folds, and 95% for Fold 5, 7, 10; glm 2 has 100% accuracy for most of the folds, and 95% for Fold 6 and 8. I can say that the new logistic regression model has an improved performance, since it has higher overall accuracy.

For the LDA model, lda 1 has 100% accuracy for most of the folds, and 95% for Fold 5; lda 2 has 100% accuracy for most of the folds, and 95% for Fold 6. There is no difference between the original LDA model’s and modified LDA model’s overall accuracy.

To put all these 4 models into comparison, I would say that both of the LDA model (lda 1 & lda 2) are the best model, since they has 100% accuracy for all folds except for one fold the accuracy is 95%. However, I would use the modified LDA model (lda 2) as the final model, since it has reduced dimension and redundancy, becoming more efficient in application.

Conclusion (with displays)

In this project, I try to find out whether or not we can predict whether a note is false or counterfeit using supervised learning. To begin with, I loaded the dataset and I have add one more column for authenticity as “ctf”, where 0 is genuine and 1 is counterfeit. Then, I created some visualizations to get a brief picture of the dataset. To have a better understanding, I plotted the graphs for the dataset as a whole and for two groups (genuine and counterfeit) separately. At this point, I have noticed that there are some considerable difference between these two groups, which might affect the analysis later. I have shuffled the data for better modeling.

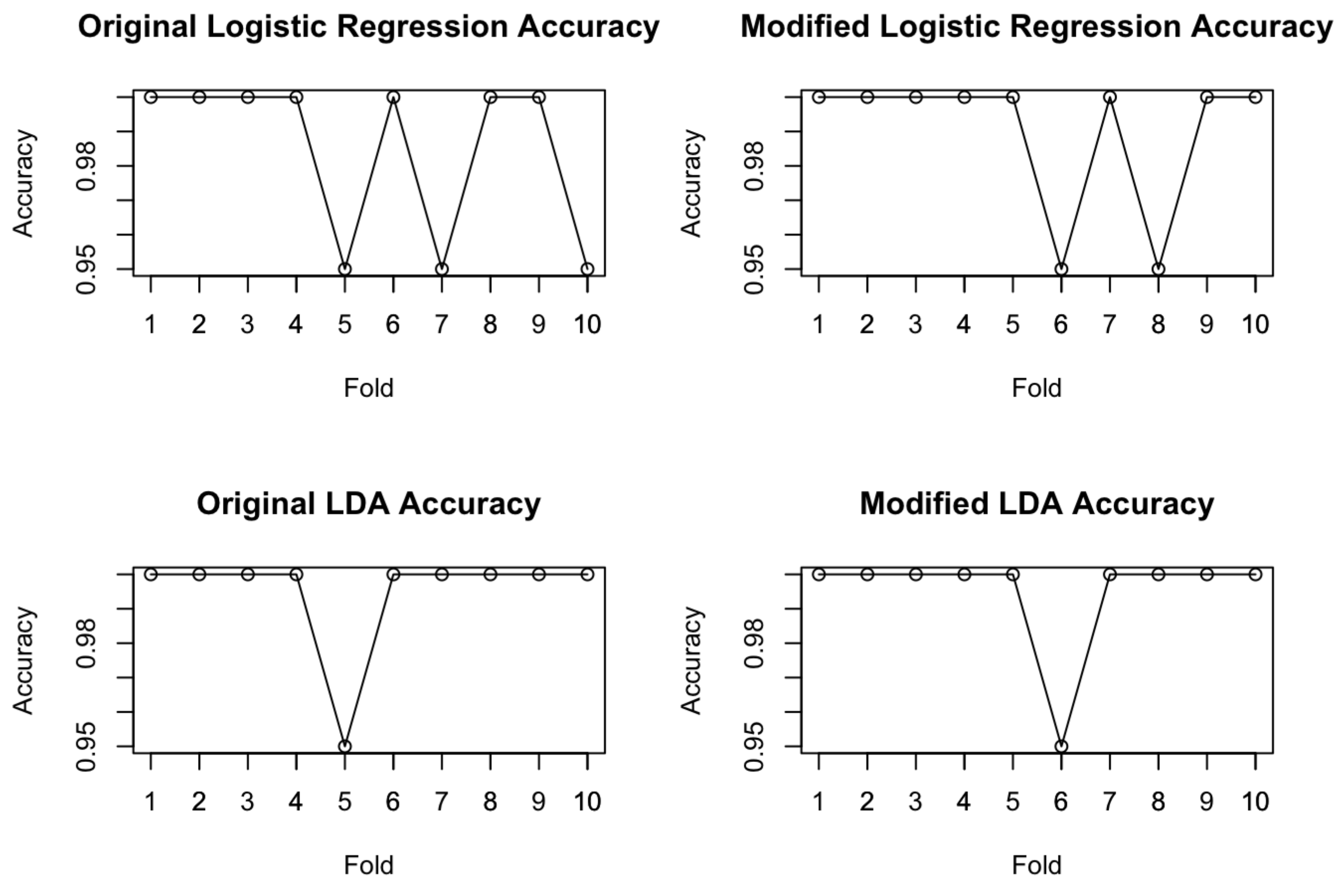
In the first part of the analysis, I built linear regression model and LDA model using caret, predicting the outcome variable by all six of the variables. The overall accuracy for linear regression model is 98.5% with a Kappa score of 0.97, indicating that it is an almost perfect model. For LDA model, the overall accuracy is 99.5% with a Kappa score of 0.99, indicating that it is an almost perfect model as well. I saved the accuracy for each fold for later interpretation.

After first modeling, I have conducted factor analysis, including PCA and MLE, trying to refine the six covariates using a factor model to reduce the dimension and remove any redundancy. As a result, I have refined the variables into 2 factors. By using these two factors scores as predictor variables, I re-run the linear regression model and LDA model. The new logistic regression model has an overall accuracy of 99% with a Kappa score of 0.98, indicating that it is an almost perfect model. Also, the new LDA model has an overall accuracy of 99.5% with a Kappa score of 0.99, indicating that it is an almost perfect model. I also saved the accuracy for each fold for later interpretation.

Comparing the model performance before and after, the logistic regression model has an improved accuracy, while the performance is the same for the LDA model. Noted that the accuracies for both original model were considerably high before the factor analysis. However, I would still say that the factor analysis is somewhat helpful, since it improved some accuracy and reduce the dimension and redundancy.

Combining the results across fold, I have made visualizations by using line plots, since it can clearly show the accuracy for each fold in this situation. The plots are below:

```
par(mfrow = c(2,2))
plot(folds_accuracy$glm1, main = "Original Logistic Regression Accuracy", ylab = "Accura
cy", xlab = "Fold")
lines(folds_accuracy$glm1)
axis(side = 1, at = c(1:10))
plot(folds_accuracy$glm2, main = "Modified Logistic Regression Accuracy", ylab = "Accura
cy", xlab = "Fold")
lines(folds_accuracy$glm2)
axis(side = 1, at = c(1:10))
plot(folds_accuracy$lda1, main = "Original LDA Accuracy", ylab = "Accuracy", xlab = "Fol
d")
lines(folds_accuracy$lda1)
axis(side = 1, at = c(1:10))
plot(folds_accuracy$lda2, main = "Modified LDA Accuracy", ylab = "Accuracy", xlab = "Fol
d")
lines(folds_accuracy$lda2)
axis(side = 1, at = c(1:10))
```



From the plots above, I can see that all of these four models have really high accuracies for each fold. Recall the boxplots in the project’s visualization section, there are kind of difference between the genuine group and counterfeit group, which might be the reason why all of the models have an almost perfect accuracy. Among all these models, both of the LDA model have the best performance by having 100% accuracy for 9 fold and 95% for 1 fold. To pick a final model, I would use the modified LDA model (lda 2), because it has reduced dimension and redundancy, becoming more efficient in application. After all these analysis, I believe that we can predict whether a note is false or counterfeit using supervised learning.