IS507 Assignment 7

1. **Develop a Linear Discriminant Analysis model to classify the variable LeaveOrNot from the other variables in Employee.csv**

Solutions:

a. **What is the performance of the classifier using cross validation?**

Answer:

```
##############################################################################
### LDA USING LEAVE ONE OUT CROSS VALIDATION ###
##############################################################################
# DEPENDENT VAR: LeaveOrNot
library(MASS)

employeeLDA_CV <- lda(LeaveOrNot ~ ., data=df, CV=TRUE)
employeeLDA_CV

# percent correct for each category of LeaveOrNot
ct <- table(df$LeaveOrNot, employeeLDA_CV$class)
diag(prop.table(ct, 1))

# total percent correct
sum(diag(prop.table(ct)))

#Accuracy is around 71.05%
```

- Using the LDA function in the MASS package, we build a linear discriminant analysis model using Leave one out cross validation (LOOCV). We do this by setting CV=True in that function.
- The dependent variable is LeaveOrNot and all the other variables are the independent variables. We build an LDA model to classify the LeaveOrNot event based on the other independent variables.
- We see that the overall accuracy is ~71.05 %, which is not very high.
- The individual accuracies for each class comes to 87.7 % for LeaveOrNot == 0 and 39.18% to LeaveOrNot == 1. This could be caused due to the class imbalance problem. There's twice the number of samples for class 0 when compared to class 1.
- From this we can say the model is overfitting to class 0 and doesn't generalize well.

Output:

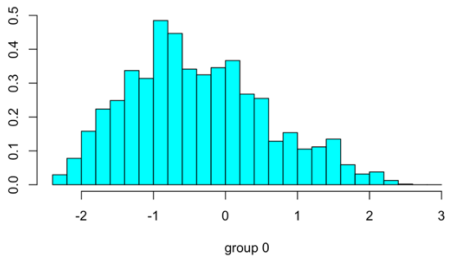
```
> # percent correct for each category of LeaveOrNot
> ct <- table(df$LeaveOrNot, employeeLDA_CV$class)
> diag(prop.table(ct, 1))
        0         1
0.8774975 0.3918750
> # total percent correct
> sum(diag(prop.table(ct)))
[1] 0.7105093
```

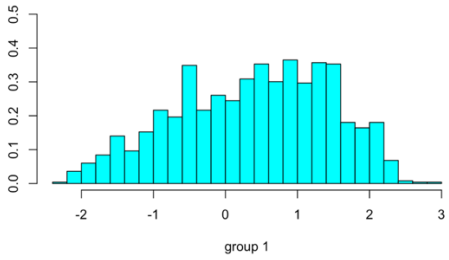
b. What is the performance of the classifier using train/test datasets?

Answer:

```
#############################################################################
### LDA USING TRAIN AND TEST ###
#############################################################################
#Creating Training and Testing Samples

library(caTools)
set.seed(42)

sample = sample.split(df,SplitRatio = 0.80)
train =subset(df,sample ==TRUE)
test=subset(df, sample==FALSE)

employeeLDA_train = lda(LeaveOrNot ~ ., data=train)
employeeLDA_train

plot(employeeLDA_train)

predicted<-predict(employeeLDA_train, newdata=test[,1:8])$class

# Compare the results of the prediction
table(predicted, test$LeaveOrNot)

mean(predicted== test$LeaveOrNot)
# Accuracy =72.43%
```

- For this, I split the entire dataset using the 80-20 rule: 80% for training and 20% for testing. Out of the entire sample size of 4653 entries: 3619 random samples were selected for training and 1034 for testing.
- We train the LDA model using the train set and visualize it.





- We then make predictions for our test set to see how well the model performs on unseen data. We exclude the LeaveOrNot column when performing predictions.

```
> # Compare the results of the prediction
> table(predicted, test$LeaveOrNot)

predicted   0   1
        0 613 217
        1  68 136
> mean(predicted== test$LeaveOrNot)
[1] 0.7243714
```

- We see from the confusion matrix that 613 samples were correctly classified as class 0 and 136 samples were correctly classified as class 1. Totally, 285 out of the 1034 test samples were misclassified.
- We measure the accuracy of the model, and we get a value of 72.43%, which marginally outperforms the LOOCV model. The same problem could be caused here of class imbalance.

c. Would certain misclassification errors be worse than others? If so, how would you measure this?

Answer:

- There are two types of errors: type 1 and type 2 errors. Type 1 errors talk about false positives (alpha) and type 2 errors talk about the false negatives (beta).
- If we look at the confusion matrix for the train/test split, 1 represents "Leaving the company" and 0 represents "Not leaving the company". The confusion matrix is constructed as follows:

Predicted | Actual ->	0 (Positive)	1 (Negative)
0 (Positive)	True +ve	False +ve. (Type 1 error)
1 (Negative)	False -ve (Type 2 error)	True -ve

- 1 can be considered negative since they leave the company and 0 can be considered positive. 217 negative samples were wrongly misclassified as positive (0), meaning the classifier misclassified them as "not leaving the company" when the actual predicted value should've been "leaving the company".
- 68 samples were falsely classified as negative (1) meaning they "leave the company" when in fact the ground truth value indicates that they do not leave the company (0).
- This could be an issue on the application side of things, because for example, misclassifying samples with ground truth of "not leaving the company" being predicted as "leaving the company" could lead to a false sense of people not liking the work environment or the job itself.

2. Select one of the techniques and apply it to some aspect of your final project dataset.

Answer:

- For the breast cancer detection dataset, the aim is to classify tumors into malignant or benign, based on several factors such as radius, smoothness etc.
- I've chosen to perform LDA and classify the "diagnosis" column, treating the other variables as independent variables.
- The missing values were filled with 0s. The "diagnosis" string column (M-malignant, B-benign) were converted and represented as numbers.
- Two types of LDA were performed: LDA with LOOCV and LDA with train and test.
- The ID and "X" columns were removed.
- Total number of samples: 569

LDA + LOOCV:

```
#remove ID
data_new = data[2:32]
names(data_new)

library(MASS)

diagnosisLDA_CV <- lda(diagnosis ~ ., data=data_new, CV=TRUE)
diagnosisLDA_CV

# percent correct for each category of LeaveOrNot
ct <- table(data_new$diagnosis, diagnosisLDA_CV$class)
diag(prop.table(ct, 1))

# total percent correct
sum(diag(prop.table(ct)))

#Accuracy comes to ~ 95.78%
```

Output:

```
> # percent correct for each category of LeaveOrNot
> ct <- table(data_new$diagnosis, diagnosisLDA_CV$class)
> diag(prop.table(ct, 1))
        1         2
0.9943978 0.8962264
> # total percent correct
> sum(diag(prop.table(ct)))
[1] 0.9578207
```

- We can see that the percentage of correctly classified samples for Class 1 (Malignant) is 99.4% and for class 2 (Benign) is 89.66%
- The overall classification accuracy comes to 95.78%, which is a good result for LOOCV built models.

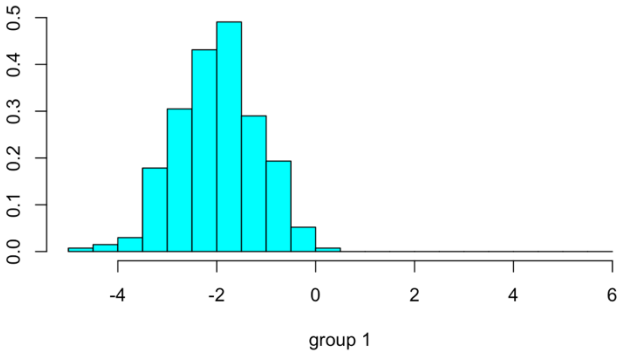
LDA + Train/Test Split

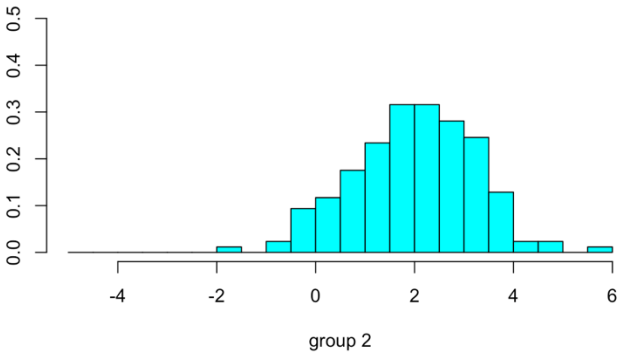
- I use an 80-20 split on the data to get the train and test sets.
    - Train samples: 440
    - Test samples: 129

```r
#Creating Training and Testing Samples

library(caTools)
set.seed(123)

sample = sample.split(data_new,SplitRatio = 0.80)
train =subset(data_new,sample ==TRUE)
test=subset(data_new, sample==FALSE)

diagnosisLDA_train = lda(diagnosis ~ ., data=train)
diagnosisLDA_train

plot(diagnosisLDA_train)

predicted<-predict(diagnosisLDA_train, newdata=test[,2:31])$class

# Compare the results of the prediction
table(predicted, test$diagnosis)

mean(predicted== test$diagnosis)
#Accuracy  = 98.44%
```

- The LDA model is trained using the train set and tested on the test set. The “diagnosis” column is excluded when testing.

```
> # Compare the results of the prediction
> table(predicted, test$diagnosis)

predicted  1  2
        1 87  5
        2  1 36
> mean(predicted== test$diagnosis)
[1] 0.9534884
>
```





- The model was visualized, and results obtained.
- From the confusion matrix, we can see that 87 samples in class 1 and 36 samples in class 2 were correctly classified and only 6 out of the 129 samples were misclassified, which is a good sign.
- The accuracy of the model comes to 95% which is similar to the LOOCV model. This accuracy can change depending on the samples in the train/test splits.

3. Using google scholar, locate a journal article which uses cluster analysis in your field of interest. Write a summary of the article in two to three paragraphs. Cite the paper in APA format.

Answer:

The paper I chose talks about analyzing game data, particularly the data collected from a massively online multiplayer, first person shooter game, Destiny. The aim of the paper is to create behavioral profiles through cluster analysis. The data they uses consists of 4800 samples (of players at the highest level across the game platform) and 41 features. Since the game has multiple modes, they focus their cluster analysis on two game modes: Player vs. Environment (PvE) and Player vs. Player (PvP).

Before building the cluster model, there was some cleaning and feature selection done on the dataset. The goal of doing this was to reduce noise, which might negatively affect the end results. They were able to select 41 features from 1400 features. They also did some exploratory analysis on the dataset prior to building the cluster models. They build 4 types of cluster models: K-means, Gaussian Mixture Models (GMM), Archetype analysis and K-Maxoids. The k-means algorithm resulted in 4 meaningful clusters (the solution should have high between-cluster variance and low within-cluster variance). GMMs resulted in 4 equally sized ellipsoidal clusters for PvP and 6 equally sized ellipsoidal clusters for PvE. Archetype analysis, which identifies points by convex combinations to represent a dataset, gave rise to 4 clusters for PvP data and 5 for PvE data. The number of clusters for archetype was chosen using scree plot and residual sum of squares. Finally, they used K-maxoids which finds cluster prototypes that represents extremes of a dataset rather than the modes. Using this technique, 4 clusters were chosen for the PvP data and 5 for PvE.

Using these clustering algorithms, they concluded that there were 4 main types of players in the PvP dataset from the archetype analysis: Aggressive close range, Marksmen, Objective killers and Casual players. For the PvE dataset and GMM they concluded there were 5 types of player profiles: High DPS, Guerilla Warriors, Close Combatants, Sitting duck snipers and Mobile Marksmen.

Citation:

Drachen, A., Green, J., Gray, C., Harik, E., Lu, P., Sifa, R., & Klabjan, D. (2016). Guns and guardians: Comparative cluster analysis and behavioral profiling in destiny. *2016 IEEE Conference on Computational Intelligence and Games (CIG)*. https://doi.org/10.1109/cig.2016.7860423