Machine Learning Analysis Report

Ryota Nishida

Executive Summary

This machine learning analysis report presents the results of my investigations into various data analysis and modeling techniques. The report covers hierarchical clustering, regression analysis, neural network modeling, and principal component analysis (PCA) applied to different datasets.

1. Introduction

Machine learning is a critical field in data science, and this report focuses on my efforts to apply machine learning techniques to analyze and model diverse datasets.

2. Hierarchical Clustering for Bank Data

2.1. Data Exploration

In this section, I will explore the bank dataset from the gclus library, emphasizing features relevant to hierarchical clustering. I will remove Bottom and find an appropriate distance measure to use, then I will perform the clustering method. The Status variable will also be removed as it's unsupervised.

2.2 Hierarchical Clustering

I think that an appropriate distance measure would be the euclidean distance because I am interested in calculating the distance between two rows of data that have numerical values, such as floating point or integer values. Furthermore, hierarchial clustering will be used which makes the euclidean distance a good option to apply to the clustering.

2.3 Linkage Method Selection

This discussion centers on the choice of linkage methods in hierarchical clustering and their impact on the clustering results. I will apply hierarchical clustering with three linkage types,"Complete Linkage","Average Linkage", and "Single Linkage". I will also provide the dendrograms for each. Then I will decide on which method is best.

2.3.1. Results

```
##   average    single  complete
## 0.8215476 0.6854183 0.9185301
```

By using the agnes function to measure the agglomerative coefficient,it seems that the best method to use would be complete. This is because the value of 0.9185301 means that the complete linkage has the strongest clustering structure. Single has the lowest, this is most likely due to the fact that single linkage is susceptible to chaining.

Table 1: Bank Data

Status	Length	Left	Right	Bottom	Top	Diagonal
0	214.8	131.0	131.1	9.0	9.7	141.0
0	214.6	129.7	129.7	8.1	9.5	141.7
0	214.8	129.7	129.7	8.7	9.6	142.2
0	214.8	129.7	129.6	7.5	10.4	142.0
0	215.0	129.6	129.7	10.4	7.7	141.8
0	215.7	130.8	130.5	9.0	10.1	141.4
0	215.5	129.5	129.7	7.9	9.6	141.6
0	214.5	129.6	129.2	7.2	10.7	141.7
0	214.9	129.4	129.7	8.2	11.0	141.9
0	215.2	130.4	130.3	9.2	10.0	140.7
0	215.3	130.4	130.3	7.9	11.7	141.8
0	215.1	129.5	129.6	7.7	10.5	142.2
0	215.2	130.8	129.6	7.9	10.8	141.4
0	214.7	129.7	129.7	7.7	10.9	141.7
0	215.1	129.9	129.7	7.7	10.8	141.8
0	214.5	129.8	129.8	9.3	8.5	141.6
0	214.6	129.9	130.1	8.2	9.8	141.7
0	215.0	129.9	129.7	9.0	9.0	141.9
0	215.2	129.6	129.6	7.4	11.5	141.5
0	214.7	130.2	129.9	8.6	10.0	141.9
0	215.0	129.9	129.3	8.4	10.0	141.4
0	215.6	130.5	130.0	8.1	10.3	141.6
0	215.3	130.6	130.0	8.4	10.8	141.5
0	215.7	130.2	130.0	8.7	10.0	141.6
0	215.1	129.7	129.9	7.4	10.8	141.1
0	215.3	130.4	130.4	8.0	11.0	142.3
0	215.5	130.2	130.1	8.9	9.8	142.4
0	215.1	130.3	130.3	9.8	9.5	141.9
0	215.1	130.0	130.0	7.4	10.5	141.8
0	214.8	129.7	129.3	8.3	9.0	142.0
0	215.2	130.1	129.8	7.9	10.7	141.8
0	214.8	129.7	129.7	8.6	9.1	142.3
0	215.0	130.0	129.6	7.7	10.5	140.7
0	215.6	130.4	130.1	8.4	10.3	141.0
0	215.9	130.4	130.0	8.9	10.6	141.4
0	214.6	130.2	130.2	9.4	9.7	141.8
0	215.5	130.3	130.0	8.4	9.7	141.8
0	215.3	129.9	129.4	7.9	10.0	142.0
0	215.3	130.3	130.1	8.5	9.3	142.1
0	213.9	130.3	129.0	8.1	9.7	141.3
0	214.4	129.8	129.2	8.9	9.4	142.3
0	214.8	130.1	129.6	8.8	9.9	140.9
0	214.9	129.6	129.4	9.3	9.0	141.7
0	214.9	130.4	129.7	9.0	9.8	140.9
0	214.8	129.4	129.1	8.2	10.2	141.0
0	214.3	129.5	129.4	8.3	10.2	141.8
0	214.8	129.9	129.7	8.3	10.2	141.5
0	214.8	129.9	129.7	7.3	10.9	142.0
0	214.6	129.7	129.8	7.9	10.3	141.1
0	214.5	129.0	129.6	7.8	9.8	142.0
0	214.6	129.8	129.4	7.2	10.0	141.3
0	215.3	130.6	130.0	9.5	9.7	141.1
0	214.5	130.1	130.0	7.8	10.9	140.9
0	215.4	130.2	130.2	7.6	10.9	141.6
0	214.5	129.4	129.5	7.9	10.0	141.4

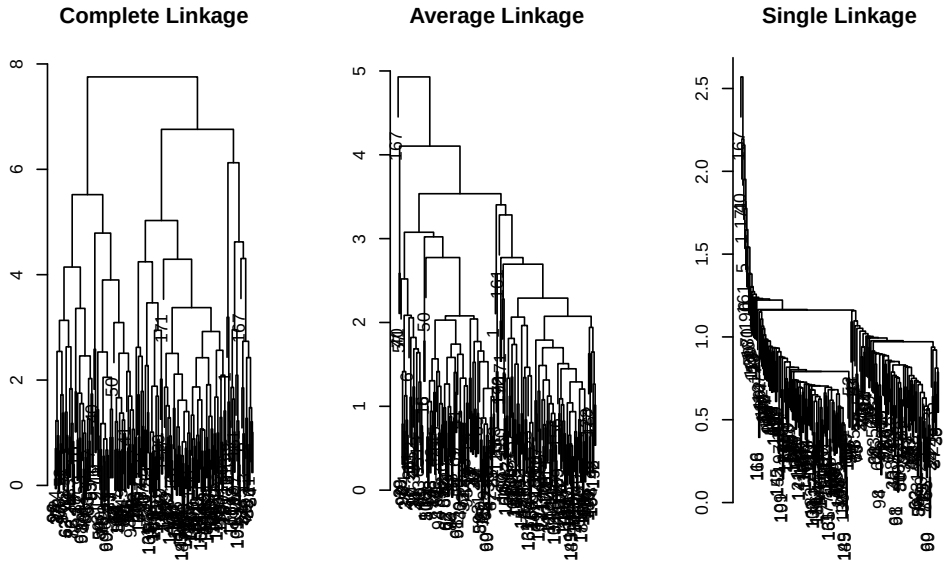


Figure 1: Dendogram

2.4. K-Means Clustering

I will apply K-means clustering to compare the results with hierarchical clustering. I will use K = 2 and set.seed(632) prior to the analysis (for consistency) on the scaled data and raw data. Then I will Provide a classification table and the misclassification rate.

```
##
##       1  0
##   0 19 81
##   1 97  3
```

```
##
##       1   0
##   0   1  99
##   1 100   0
```

2.4.1. Results

The classification table has column and row labels of 0 and 1. The 0 represents a model that had zero incorrect predicitions, and the 1 represents one with completely incorrect predictions. The missclassification rate is found by dividing the number of incorrect predicitions by the number of total predicitions. The classification tables brought fourth a misclassification rate of 0.11 for the scaled data, and 0.005 for the raw data. The value of 0.005 means that the raw data performed better. The reason why this may have performed better than the scaled data may be because the unscaled data doesn't take into account how the data is distributed.

3. Regression Analysis for Professor Salaries

3.1. Linear Regression Tree

In this section, I will introduce a machine learning model for predicting professor salaries based on relevant variables.I will load the data Salaries.csv(located in the data folder) which contains information on professors and their salaries. The data description elements can be found in the R Studio help section. I will be looking for answers to questions such as: "Would an early career professor (Assistant or Associate) make more money in an applied or theoretical department?"

3.1.1. Regression Tree

I will fit a regression tree (using the 'tree' library) for a professor's salary given the remainder of the variables in the data set. Then I will provide the tree, including labels.Using the command text(treename, pretty=0), I will also provide more understandable split labeling for the questions that follow.

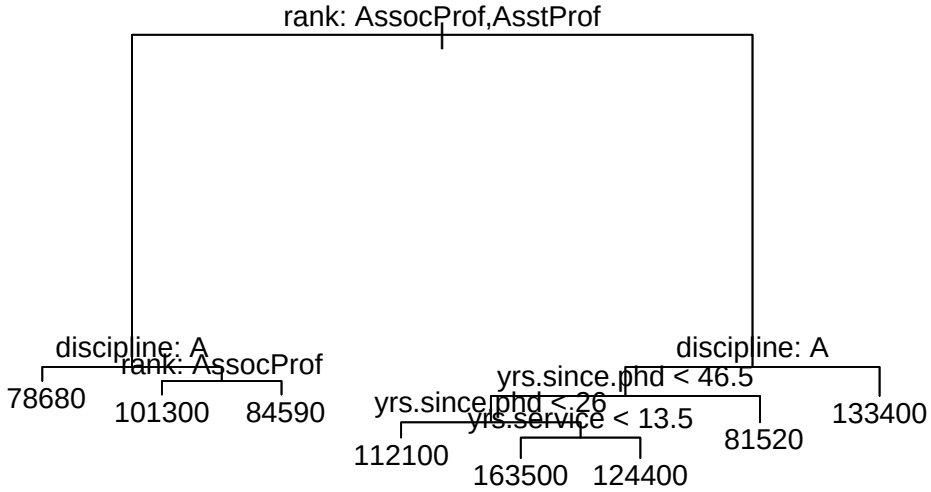


Figure 2: Regression Tree

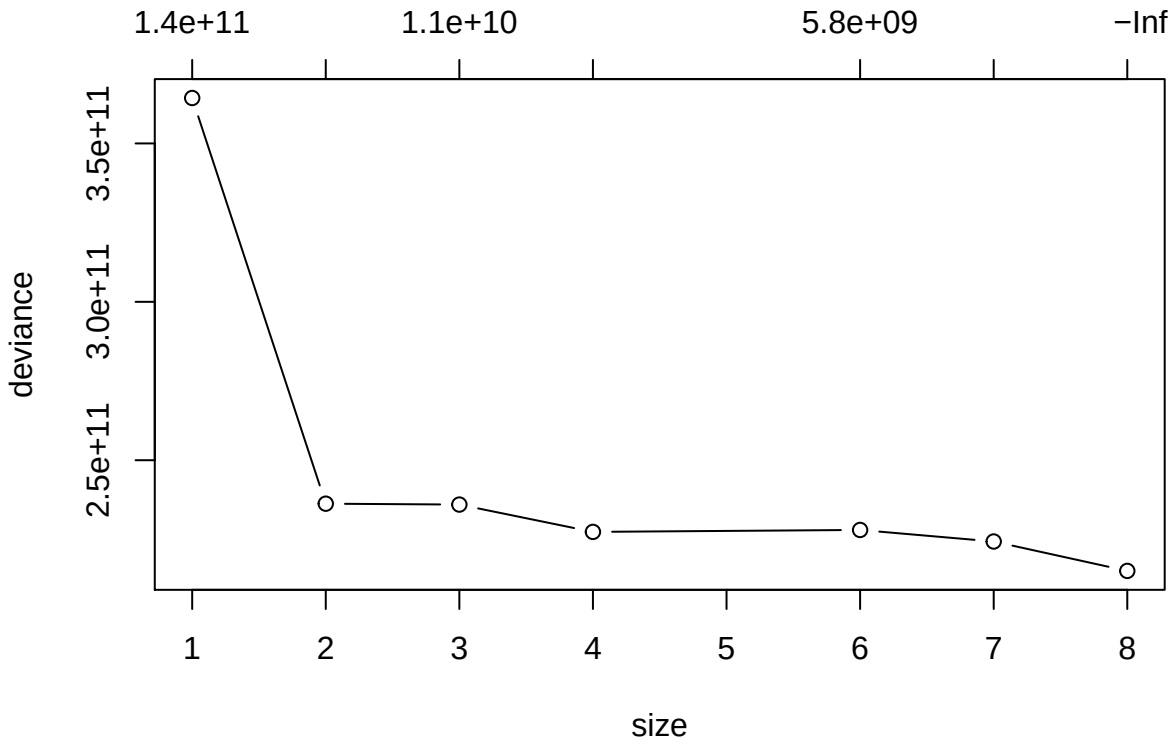
Based on this tree, as an early career professor, it seems that those in an applied field(B), make more money than those in an theoretical field(A).However, associate professors in field B make more than assistant professors in field B.

3.2. Cross-Validation

I will now discuss cross-validation to ensure the reliability of the regression model.

3.2.1. LOOCV (Leave One Out Cross Validation)

I will now perform LOO cross-validation using cv.tree and plot the resulting object in order to see how many terminal nodes the cross-validation suggests in order to get the best size of the tree.



3.2.2. Results

The cross-validation suggests 8 terminal nodes, as it has the lowest level of deviance.

3.3. Model Performance Evaluation

I will evaluate the model's performance and assess its accuracy in predicting professor salaries.

3.3.1. Linear Model Data Prediction

I will now predict what is the salary for an Assistant Professor, in an applied department, who got their PhD in 2012, has 5 years of service, and is identified as female. I will enter the data into R and use the predict() function.

```
##        1
## 83275.86
```

3.3.2. Results

The results show that it is predicted that they would make $83275.86 dollars.

4. Neural Network Modeling for Fish Toxicity Data

I will fit a neural network (using nnet) on the fish_toxicity data set(located in the data folder) with the 7th variable as the response, with one hidden layer and 5 hidden layer variables in order to predict toxicity using all other variables. Note: The original source for this can be seen here: https://archive.ics.uci.edu/ml/datasets/QSAR+fish+toxicity.

In order to provide a proper fit, the variables will need be normalized. With an R object called fishdata, the following command can quickly normalize the variables:

fishdat <- read.csv("fish_toxicity.csv", sep=";", header=FALSE)

```
## # weights:  41
## initial  value 21.021470
## iter  10 value 8.698053
## iter  20 value 8.386183
## iter  30 value 7.771483
## iter  40 value 7.520154
## iter  50 value 7.362388
## iter  60 value 7.335942
## iter  70 value 7.297150
## iter  80 value 7.231848
## iter  90 value 7.175887
## iter 100 value 7.135093
## final  value 7.135093
## stopped after 100 iterations
```

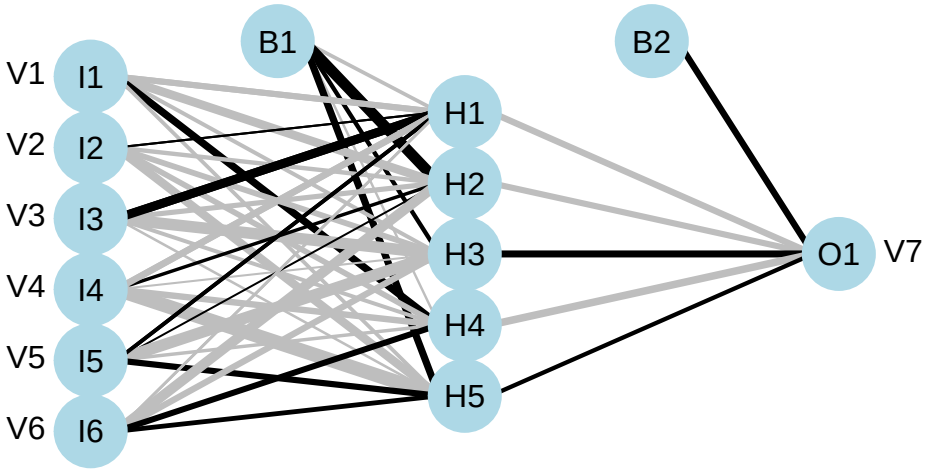


Figure 3: CV Tree

4.1 MSE Calculation

I will calculate the Mean Squared Error (MSE) to gauge the neural network model's performance.

```
## [1] 908   7
```

```
## [1] 0.007858032
```

4.1.1. Results

The scaled MSE is equal to 0.007858032. However, this could be overfitting, so I will set up a training and testing scenario.

4.2 Training and Testing Split

The dataset is split for training and testing, and model performance is evaluated on the test set.

```
## # weights:  41
## initial  value 74.420749
## iter  10 value 4.679182
## iter  20 value 4.521400
## iter  30 value 4.233496
## iter  40 value 3.908401
```

```
## iter  50 value 3.757015
## iter  60 value 3.610422
## iter  70 value 3.515805
## iter  80 value 3.443341
## iter  90 value 3.420330
## iter 100 value 3.400435
## final  value 3.400435
## stopped after 100 iterations
```

```
## [1] 0.0103977
```

4.2.1. Results

The MSE for the test set gave a scaled MSE of 0.0103977.

4.2.2 Original Scale MSE

The MSE above is on the normalized scale. I will find the MSE for the test set on the original scale of the response variable. The model will be on the scaled data, so I will scale the MSE back to the original data's scale.

```
## [1] 0.9500849
```

This brought back a original scale MSE of 0.9500849.

4.3 Model Comparison

I will compare the neural network model's performance with that of a linear model and discuss the trade-offs.

4.3.1. Linear Model and Neural Net comparison

I will fit a linear model on the same training set (but ensure it is on the unscaled data) and compare the MSE from the test set to that which was found from a neural net.Then I will determine which model is better in terms of prediction.

```
## [1] 0.8939699
```

4.3.2. Results

When comparing the uncaled MSE to that of the linear model, the linear model is better with an MSE of 0.8939699 compared to 0.9500849.

4.4. Optimal Neural Network Configuration

This exploration aims to identify the optimal neural network structure for the given data.

4.4.1. Optimal Neural Network Configuration Exploration

I will explore some other neural net structures as allowable within nnet (so effectively just # hidden layer variables and switching between logistic and linear activation functions), and see what structure appears to be best for the 50-50 training/testing split that I have already made. Then I will also find what the associated MSE on the original scale is.

```
## [1] "Number of hidden layer variables for linear activation: 1"
## [1] 0.8642176
## [1] "Number of hidden layer variables for linear activation: 2"
## [1] 0.8294055
## [1] "Number of hidden layer variables for linear activation: 3"
```

```
## [1] 0.9150119
## [1] "Number of hidden layer variables for linear activation: 4"
## [1] 0.953569
## [1] "Number of hidden layer variables for linear activation: 5"
## [1] 0.9499171
## [1] "Number of hidden layer variables for linear activation: 6"
## [1] 0.9582515
## [1] "Number of hidden layer variables for linear activation: 7"
## [1] 0.9455187
## [1] "Number of hidden layer variables for linear activation: 8"
## [1] 1.004026
## [1] "Number of hidden layer variables for linear activation: 9"
## [1] 1.04651
## [1] "Number of hidden layer variables for linear activation: 10"
## [1] 1.231521
```

```
## [1] "Number of hidden layer variables for logistic activation : 1"
## [1] 0.8627276
## [1] "Number of hidden layer variables for logistic activation : 2"
## [1] 0.8340326
## [1] "Number of hidden layer variables for logistic activation : 3"
## [1] 0.9106819
## [1] "Number of hidden layer variables for logistic activation : 4"
## [1] 0.8896531
## [1] "Number of hidden layer variables for logistic activation : 5"
## [1] 0.9283874
## [1] "Number of hidden layer variables for logistic activation : 6"
## [1] 0.9110632
## [1] "Number of hidden layer variables for logistic activation : 7"
## [1] 1.019936
## [1] "Number of hidden layer variables for logistic activation : 8"
## [1] 1.027484
## [1] "Number of hidden layer variables for logistic activation : 9"
## [1] 0.9377698
## [1] "Number of hidden layer variables for logistic activation : 10"
## [1] 0.993025
```

```
## [1] 0.9697948
```

```
## [1] 0.9314761
```

4.4.2.Results

According to the data, the best structure for the 50-50 training/testing split is a structure that has 2 hidden variables within the hidden layer. This is because for both activation functions, the model featuring 2 hidden variables had the lowest MSE. Also, Logistic activation is the best function according to the data. This is because it featured the lowest average MSE. The associated MSE on the original scale for the best structure is 0.8340326.

5. Principal Component Analysis (PCA) on Wine Data

I will install/load the gclus library and run data(wine) to load the 13-variable Italian red wine data set.

5.1. PCA

PCA is applied to the wine dataset to reduce dimensionality and identify critical features. I will perform principal component analysis using the prcomp function on the scaled numeric predictors (everything but Class), then provide a data summary of the prcomp object, biplot, and scree Plot.

5.2. Principal Component Selection

I will discuss the selection of principal components and their importance in explaining variance.

```
## Standard deviations (1, .., p=13):
##  [1] 2.1692801 1.5801996 1.2025231 0.9586868 0.9236863 0.8010317 0.7423184
##  [8] 0.5903918 0.5374585 0.5008359 0.4751700 0.4107890 0.3215140
##
## Rotation (n x k) = (13 x 13):
##                           PC1          PC2         PC3         PC4         PC5
## Alcohol         -0.144326211  0.483648385 -0.20739872  0.01781373 -0.26563591
## Malic            0.245196693  0.224942839  0.08900284 -0.53678723  0.03545487
## Ash              0.002056414  0.316067878  0.62621546  0.21414563 -0.14311322
## Alcalinity       0.239324235 -0.010588567  0.61208187 -0.06080291  0.06609613
## Magnesium       -0.141996869  0.299616842  0.13076470  0.35200289  0.72696995
## Phenols         -0.394655158  0.065062879  0.14618093 -0.19816859 -0.14926756
## Flavanoids      -0.422934929 -0.003347063  0.15069113 -0.15234224 -0.10900952
## Nonflavanoid     0.298539475  0.028777435  0.17034777  0.20317159 -0.50076653
## Proanthocyanins -0.313433863  0.039306718  0.14947245 -0.39892406  0.13694364
## Intensity        0.088623467  0.529995887 -0.13732184 -0.06593107 -0.07639601
## Hue             -0.296690253 -0.279281608  0.08517565  0.42783328 -0.17362338
## OD280           -0.376166559 -0.164477553  0.16601481 -0.18419094 -0.10112608
## Proline         -0.286760071  0.364882753 -0.12674656  0.23209522 -0.15796405
##                          PC6         PC7         PC8         PC9         PC10
## Alcohol          0.21354986 -0.05635897  0.39579331 -0.50928188  0.21079621
## Malic            0.53678293  0.42061037  0.06595816  0.07573444 -0.30900511
## Ash              0.15450332 -0.14916568 -0.17006366  0.30788889 -0.02638908
## Alcalinity      -0.10081371 -0.28691985  0.42790507 -0.20086630  0.05220344
## Magnesium        0.03807568  0.32277194 -0.15666579 -0.27141732  0.06746554
## Phenols         -0.08412421 -0.02797494 -0.40604422 -0.28532209 -0.32091668
## Flavanoids      -0.01890293 -0.06067708 -0.18718662 -0.04925174 -0.16342863
## Nonflavanoid    -0.25858079  0.59540750 -0.23350998 -0.19577688  0.21509037
## Proanthocyanins -0.53380396  0.37221491  0.36834278  0.20870151  0.13467639
## Intensity       -0.41865995 -0.22773645 -0.03374735 -0.05552801 -0.29058602
## Hue              0.10594057  0.23207847  0.43653403 -0.08526588 -0.52232380
## OD280            0.26587118 -0.04474898 -0.07822262 -0.13786189  0.52371887
## Proline          0.11976715  0.07687881  0.12039492  0.57533225  0.16268297
##                         PC11        PC12        PC13
## Alcohol          0.22591140 -0.26620294  0.01499766
## Malic           -0.07631939  0.12184544  0.02588269
## Ash              0.49867120 -0.04963472 -0.14120724
## Alcalinity      -0.47929429 -0.05576510  0.09169239
## Magnesium       -0.07131525  0.06223205  0.05676776
## Phenols         -0.30411269 -0.30377800 -0.46389704
## Flavanoids       0.02581911 -0.04261856  0.83223814
## Nonflavanoid    -0.11700075  0.04233008  0.11404990
## Proanthocyanins  0.23728638 -0.09566443 -0.11687878
## Intensity       -0.03175508  0.60435992 -0.01216038
## Hue              0.04852012  0.25948145 -0.09006869
```

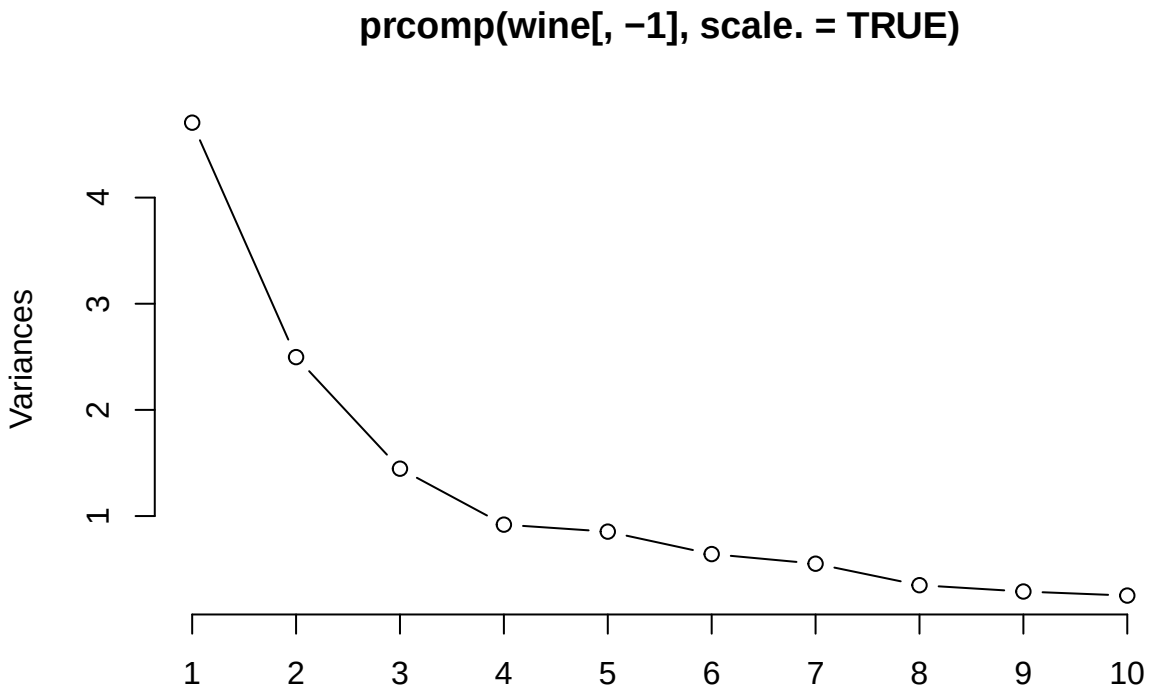
```
## OD280           -0.04680231  0.60072485 -0.15676499
## Proline         -0.53940647 -0.07969117  0.01453577
```

```
## Importance of components:
##                           PC1    PC2    PC3    PC4     PC5     PC6     PC7
## Standard deviation     2.169 1.5802 1.2025 0.9587 0.92369 0.80103 0.74232
## Proportion of Variance 0.362 0.1921 0.1112 0.0707 0.06563 0.04936 0.04239
## Cumulative Proportion  0.362 0.5541 0.6653 0.7360 0.80163 0.85098 0.89337
##                           PC8     PC9   PC10    PC11    PC12    PC13
## Standard deviation     0.59039 0.53746 0.5008 0.47517 0.41079 0.32151
## Proportion of Variance 0.02681 0.02222 0.0193 0.01737 0.01298 0.00795
## Cumulative Proportion  0.92018 0.94240 0.9617 0.97907 0.99205 1.00000
```

5.2.1. Principal Component Selection according to the Kaiser criterion

According to the Kaiser criterion,PC1 through PC3 principal components should be kept. If at least 90% of the variance in the data is to be retained then PC1 through PC7 principal components should be kept.

5.2.2. Principal Component Selection according to the Scree Plot



According to the Scree Plot, PC1 through PC4 principal components should be kept as there is no longer any drastic changes in variances after the fourth component.

5.3. Biplot

5.3.1.Visualization

A biplot visualization has been created to elucidate the relationships between variables and observations.

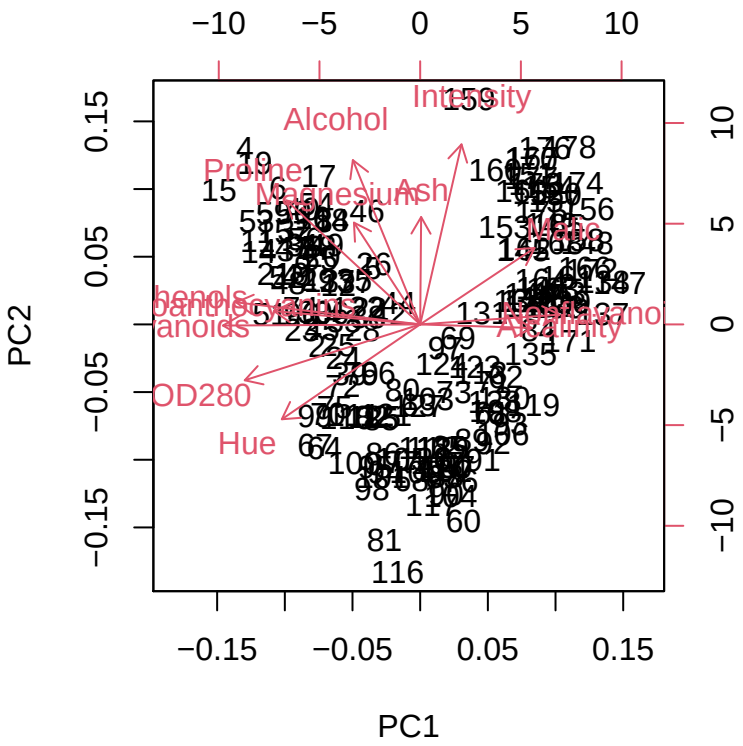


Figure 4: BiPlot

5.4. Cross-Validated Logloss

5.4.1. LDA (with built-in leave-one-out cross-validation)

I will retain the components suggested by the Kaiser criterion, and perform an LDA for the wine class using the retained scores as the predictors, and find the cross-validated logloss of the model.

```
##
##      1  2  3
##   1 57  2  0
##   2  2 66  3
##   3  0  0 48
```

```
## [1] 0.1098738
```

5.4.2. Results

The results show that this model has a cross-validated log-loss of 0.1098738.

6. Conclusion

This machine learning report summarizes my findings, highlighting insights gained from hierarchical clustering, regression analysis, neural network modeling, and PCA. These findings provide valuable information for data-driven decision-making and further machine learning research.

7. Appendices

```
knitr::opts_chunk$set(echo = TRUE)
library(kableExtra)
library(gclus)
library(purrr)
data("bank")

knitr::kable(bank, caption = 'Bank Data')

bank1 <- subset(bank, select = -c(Status,Bottom) )

sd_bank1 =scale(bank1)
par(mfrow = c(1,3))
bank1_dist = dist(sd_bank1,method = "euclidean")
plot(hclust(bank1_dist,method = "complete"), main = "Complete Linkage", xlab = "", sub = "",ylab = "")
plot(hclust(bank1_dist, method = "average"), main = "Average Linkage", xlab = "", sub = "",ylab = "")
plot(hclust(bank1_dist, method = "single"),  main = "Single Linkage", xlab = "", sub = "",ylab = "")
```

```
linkages<- c( "average", "single", "complete")
names(linkages) <- c( "average", "single", "complete")

# function to compute coefficient
ac <- function(x) {
  agnes(bank1, method = x)$ac
}
map_dbl(linkages, ac)

bank2 <- subset(bank, select = -c(Bottom) )
bank4 <- subset(bank, select = -c(Bottom) )
bank4$Status<-factor(bank4$Status)

set.seed(632)
k2 <- kmeans(scale(bank2[,-1]), centers = 2, nstart = 100)

y<-table(as.factor(bank2[,1]),k2$cluster)
colnames(y) = c("1", "0")
y

set.seed(632)
k3 <- kmeans((bank2[,-1]), centers = 2, nstart = 100)
z<-table(bank2[,1],k3$cluster)
colnames(z) = c("1", "0")
z

S<-read.csv("/Users/ryotanishida/R/Data/Salaries.csv",stringsAsFactors = T)

library(tree)
stree <- tree(salary~., data=S)
plot(stree)
text(stree, pretty=0)

plot(cv.tree(stree),type = 'b')
```

```r
model <- lm(salary~., data=S)

new <- data.frame(rank=c('AsstProf' ),discipline=c('B'), yrs.since.phd=c(10),yrs.service=c(5),sex=c('Fe

predict(model, newdata=new)

library(nnet)
fishdat <- read.csv("/Users/ryotanishida/R/Data/fish_toxicity.csv", sep=";", header=FALSE)
sfishdat<-apply(fishdat, 2, function(v) (v-min(v))/(max(v)-min(v)))
set.seed(4521)
nnfishdat <- nnet(V7~., data=sfishdat, size=5)
library(NeuralNetTools)
plotnet(nnfishdat)
```

```r
dim(sfishdat)
sum(resid(nnfishdat)^2)/908

set.seed(217)
ind <- sample(1:nrow(fishdat), 454)
train <- sfishdat[ind,]
test <- sfishdat[-ind,]
nnfishtr <- nnet(V7~., data=train, size=5)
prfishdat <-predict(nnfishtr, newdata=test[,-7])


MSEfishdat <- sum((test[,7]-prfishdat)^2)/nrow(test)
MSEfishdat

prfishdat1 <- prfishdat*(max(fishdat$V7)-min(fishdat$V7))+min(fishdat$V7)
test1 <- (test[,7])*(max(fishdat$V7)-min(fishdat$V7))+min(fishdat$V7)

MSEfishdat1 <- sum((test1-prfishdat1)^2)/nrow(test)
MSEfishdat1

set.seed(217)
Train <-fishdat[ind,]
Test <- fishdat[-ind,]

lmfishdat <- lm(V7~., data=Train)
m<-summary(lmfishdat)


prlmfishdat <- predict(lmfishdat,Test)
MSE.lmfishdat <- sum((prlmfishdat - Test[,7])^2)/nrow(test)
MSE.lmfishdat


for(i in 1:10){
nnfishtr3 <- nnet(V7~.,data=train,size=i,
trace=FALSE,linout= TRUE)
prnnfishtr3 <-predict(nnfishtr3, newdata=test[,-7])
prnnfishtr3.1 <-predict(nnfishtr3, newdata=test[,-7])*(max(fishdat$V7)-min(fishdat$V7))+min(fishdat$V7)
MSEfishtr3 <- sum((test1-prnnfishtr3.1)^2)/nrow(test)
print(paste("Number of hidden layer variables for linear activation:", i))
```

```
print(MSEfishtr3)
}

set.seed(217)
for(i in 1:10){
nnfishtr4 <- nnet(V7~.,
data=train,size=i,trace=FALSE,linout= FALSE)
prnnfishtr4 <-predict(nnfishtr4, newdata=test[,-7])
prnnfishtr4.1 <-predict(nnfishtr4, newdata=test[,-7])*(max(fishdat$V7)-min(fishdat$V7))+min(fishdat$V7)
MSEfishtr4 <- sum((test1-prnnfishtr4.1)^2)/nrow(test)
print(paste("Number of hidden layer variables for logistic activation :", i))
print(MSEfishtr4)
}

totalmse<-0.8642176+0.8294055+0.9150119+
  0.953569+0.9499171+0.9582515+
  0.9455187+1.004026+1.04651+1.231521
avgtotalmse<-totalmse/10
avgtotalmse
totalmse1<-0.8627276+0.8340326+0.9106819+0.8896531+0.9283874+0.9110632+1.019936+1.027484+0.9377698+
0.993025
avgtotalmse1<-totalmse1/10
avgtotalmse1

library(gclus)
data(wine)

prcomp(wine[,-1],scale.=TRUE)
summary(prcomp(wine[,-1],scale.=TRUE))

plot(prcomp(wine[,-1],scale.=TRUE), type="lines")

biplot(prcomp(wine[,-1],scale.=TRUE))

set.seed(217)
library(MASS)
library(MLmetrics)
pcwine<-prcomp(wine[,-1],scale.=TRUE)
pck <- pcwine$x[,1:3]
lda.fit=lda(wine$Class~pck,CV=T)
table(wine$Class, lda.fit$class)
MultiLogLoss(lda.fit$posterior,wine$Class)
```

8. References

[1] James, G & Witten, D & Hastie, T & Tibshirani, R and Taylor, J. (2021) *An Introduction to Statistical Learning with Applications in R* New York: Springer.