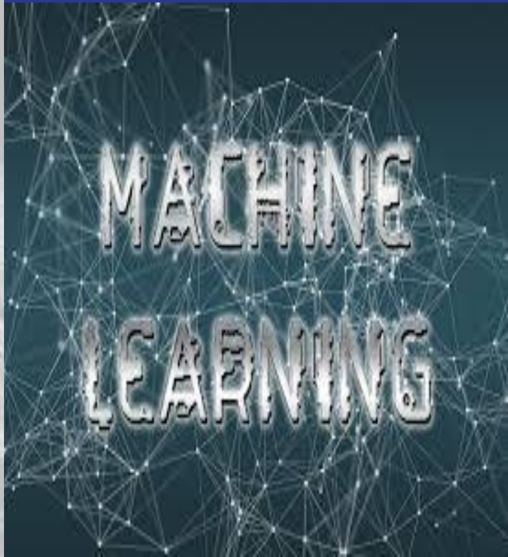
CSE 5160 Group 8 Project Presentation: Wine Data set



Erik Trujillo
Mario Valdez
Bat-Erdene Narangerel
Spring 2022
Dr. Qiao

Introduction:

Wine is a popular alcoholic drink made in many different countries and also sold in many different places. Wine is typically made from grapes that have been fermented. Fermentation is the process of turning the sugars that can be found in yeast into ethanol and carbon dioxide. There are many different variants of wine such as red, white, amber, and sparkling wine. One of the fascinating facts about wine is that one of its oldest producers, Italy, happens to be the largest producer of wine today. However, the origin processes of creating wine itself can vary from wine type to wine type. In order to truly decipher the origin of different types of wine, wine experts use a variety of chemical analysis methods to determine the origin of the wine. In this particular project, our group is interested in using chemical analysis to determine the origin of wines grown in the same region in Italy, but derived from three different cultivars (which are just plant varieties).

Background

Background of Original Study:

If we wish to find the answer to this question, we decided the best course of action was to have a look at a Wine data set that is provided in the UCI machine learning repository. The data set comes from a paper called “*An Extendible Package for Data Exploration, Classification, and Correlation. Institute Institute of Pharmaceutical and Food Analysis and Technologies*” from Professor Forina and Brigata Salerno at the University of Genoa in the nation of Italy. This dataset is basically the results of a chemical analysis of wine grown in the same region in Italy but derived from 3 different cultivars (type of plant reproduced for specific traits). An analysis was done to determine the quantities of 13 constituents/attributes found in each of the three types of wine. The list of the 13 attributes will be provided below, and a significant side note is that the **first attribute is the class identifier. Moreover, all the attributes are continuous** and some of the attributes include the physical characteristics of the wine itself such as magnesium content, ash, flavanoids, the specific alcalinity of the ash, etc. were all collected.

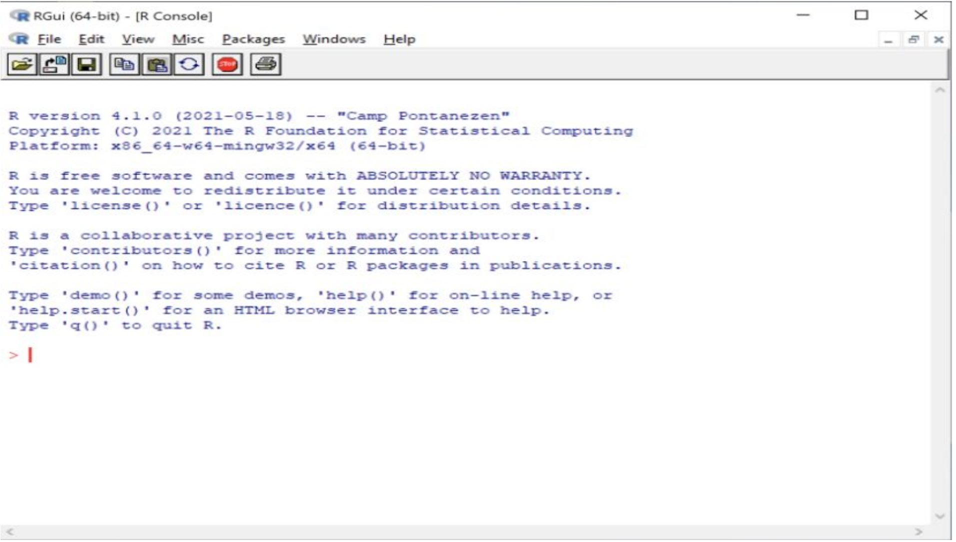
The Story Behind the Dataset

- Our data was set to analyze 3 types of wines that are not specified.
  - Instead of proper names, they were labeled as class 1, class 2 and class 3.
- It was stated that the original analysis contained 30 attributes but the author had said they “lost” that information.
  - Only includes 13.
- Associated tasks only include classification
- All attributes are continuous

Research Question

With the following wine dataset that is provided through the UCI Machine Learning Repository, we will be looking for attributes that are more commonly found amongst the three types of wines. Furthermore, what clues can the attributes give to deciphering what each of the 3 cultivars is in terms of the origin of the respective wines?

With the following wine dataset , what attributes are more commonly found in one wine type over the other? If enough clues are given with attributes, are we able to decipher what type of wine is respective to their class identifier? This dataset also has an emphasis on the "three types of wine" (which are in the 1st attribute class identifier (1-3))?



Tools and Methods:

Programming Languages and Tools included the R language, Rstudio, and R Gui

- R language is useful for machine learning due to its efficiency and versatility
- R studio uses the R language to create statistical programs and graphics
- R studio is the perfect integrated development environment for R programming and great for loading datasets

UCI Data Set: Wine (classes)

UCI MACHINE LEARNING REPOSITORY LINK:

https://archive.ics.uci.edu/ml/datasets/Wine

- Abstract Idea: Using chemical analysis to determine the origin of wines
- The Wine dataset involves data that is the result of wines grown in the same region in Italy, but they originate from three different cultivars (which are plant varieties)
- Each of the three different types of wine will be represented by a class
- The analysis focuses on determining and collecting different quantities of the 13 constituents found in each of the three types of wine which include ash, magnesium, etc.
- Tasks associated with this dataset are classification, and the attribute characteristics include integer and real
- The dataset is multivariate, with 13 attributes and 178 instances, and all the attributes are continuous

UCI

Machine Learning Repository

Center for Machine Learning and Intelligent Systems

About Citation Policy Donate a Data Set Contact

Search

Repository Web

View ALL Data Sets

Check out the beta version of the new UCI Machine Learning Repository we are currently testing! Contact us if you have any issues, questions, or concerns. Click here to try out the new site.

Wine Data Set

Download: Data Folder, Data Set Description

Abstract: Using chemical analysis determine the origin of wines

Data Set Characteristics:	Multivariate	**Number of Instances:**	178	**Area:**	Physical
Attribute Characteristics:	Integer, Real	**Number of Attributes:**	13	**Date Donated**	1991-07-01
Associated Tasks:	Classification	**Missing Values?**	No	**Number of Web Hits:**	1919108

Source:

Original Owners:

Forina, M. et al, PARVUS -
An Extendible Package for Data Exploration, Classification and Correlation.
Institute of Pharmaceutical and Food Analysis and Technologies, Via Brigata Salerno,
16147 Genoa, Italy.

Donor:

Stefan Aeberhard, email: stefan '@' coral.cs.jcu.edu.au

Data Set Information:

These data are the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. The analysis determined the quantities of 13 constituents found in each of the three types of wines.

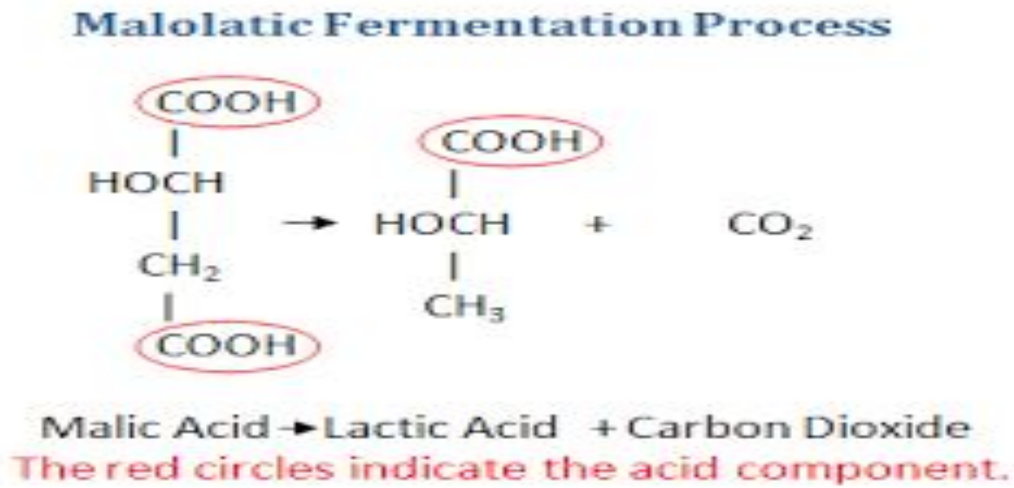
I think that the initial data set had around 30 variables, but for some reason I only have the 13 dimensional version. I had a list of what the 30 or so variables were, but a.) I lost it, and b.), I would not know which 13 variables are included in the set.

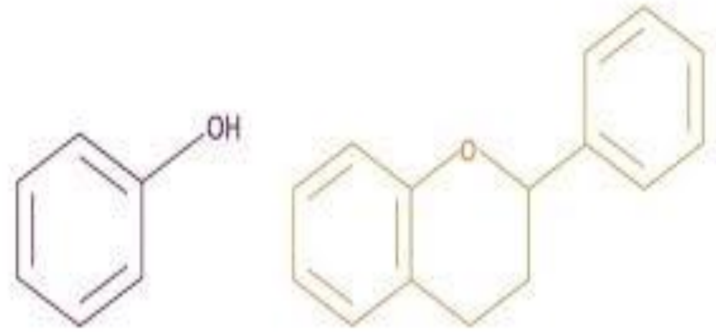
The attributes are (dontated by Riccardo Leardi, riclea '@' anchem.unige.it)
1) Alcohol
2) Malic acid
3) Ash
4) Alcalinity of ash
5) Magnesium
6) Total phenols

LIST OF ATTRIBUTES

Note: To emphasize, the 1st attribute is the class identifier (class 1, class 2, class 3)

- Alcohol
- Malic acid
- Ash
- Alcalinity of ash
- Magnesium
- Total phenols
- Flavanoids
- Nonflavanoid phenols
- Proanthocyanins
- Color intensity
- Hue
- OD280/OD315 of diluted wines
- Proline





Phenol Flavanoid

Possible Solution

- Using at least 2 machine learning techniques, it can be possible to determine what type of wine we are looking at just by looking at the amount of quantity that in the attributes

Implementation

Linear Discriminant Analysis:

One of the classifications methods used for predictions

LDA Includes

- Training a dataset
- Applying lda function in R
- Cross validation
- Prediction

Training

What exactly do we train?

```
#Splits wine data into 2 sets, 1 to train the other to test
spl = sample.split(wine_ds$class, SplitRatio = 0.8)
train <- wine_ds[spl==TRUE, ]
test <- wine_ds[spl == FALSE, ]
```

Implementation

Used all attributes, in order to determine which attributes are used among all of them.

```
#Uses LDA function with all variables
wine_ds.lda.fit <- lda(class ~ alcohol + malic_acid + ash + alcalinity_of_ash +
                         magnesium + total_phenols + flavanoids + nonflavanoid_phenols +
                         proanthocyanins + color_intensity + hue + OD280_OD315 + proline,
                       data = train)
#Displays object into console
wine_ds.lda.fit
```

Implementation Results

Prior Probabilities of Groups	
1	0.3262411
2	0.4042553
3	0.2695035

Group Means	Alcohol	Malic Acid	Ash	Alcalinity of Ash	Magnesium	Total Phenois	Flavanoids
1	13.66957	2.028913	2.432609	16.97826	105.41304	2.821087	2.9828261
2	12.27474	1.956842	2.267544	20.18772	96.07018	2.367895	2.1770175
3	13.17158	3.477368	2.437105	21.57895	99.13158	1.671842	0.7615789

Group Means	Nonflavanoid Phenois	Proanthocyanins	Color Intensity	Hue	OD280 OD315	Proline
1	0.2795652	1.906087	5.450652	1.0504348	3.166739	1087.8696
2	0.3612281	1.678421	3.110877	1.0492982	2.807368	522.9474
3	0.4576316	1.166316	7.590526	0.6718421	1.678421	625.3947

Implementation Results Cont...

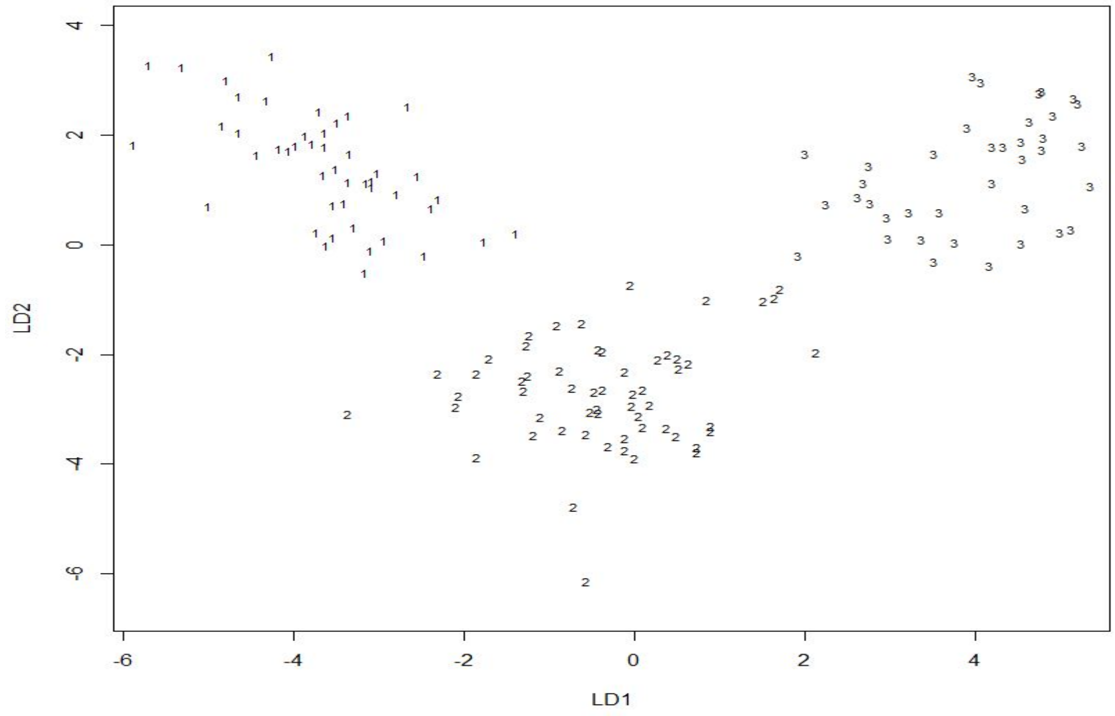
Proportion of Trace	
LD1	0.6904
LD2	0.3096

Coefficients of Linear Discriminants		
	LD1	LD2
Alcohol	-0.300574655	0.935208361
Malic Acid	0.228506698	0.364556348
Ash	0.128749154	1.835833249
Alcalinity of Ash	0.148774211	-0.147586674
Magnesium	-0.004415782	-0.006731251
Total Phenols	0.456787072	-0.295722861
Flavanoids	-1.743858041	-0.242434715
Nonflavanoids Phenols	-0.885072913	-1.156412271
Proanthocyanins	0.29893437	-0.334766653
Color Intensity	0.438744703	0.264543009
Hue	-0.557995425	-1.522650155
0D280_OD315	-1.084080155	0.32378188
Proline	-0.002660145	0.003794766

Graphing the results

Using the following:

```
#Provides a visual represntation
plot(wine_ds.lda.fit )
```



Cross Validation

Using cross validation technique, we can see that class has two attributes that are being misclassified as either class 1 and class 3.

```
#Applys cross validation check
wine_ds.lda.fit.2 <- lda(class ~ alcohol + malic_acid + ash + alcalinity_of_ash +
                          magnesium + total_phenols + flavanoids + nonflavanoid_phenols +
                          proanthocyanins + color_intensity + hue + OD280_OD315 + proline,
                        data = train, CV = TRUE)
#Displays Results into a table
table(wine_ds.lda.2$class, wine_ds[ ,1])
```

	1	2	3
1	58	1	0
2	0	69	0
3	0	1	48

Quadratic Discriminant Analysis

What is QDA ?

QDA is a generative model. QDA assumes that each class follow a Gaussian distribution. The class-specific prior is simply the proportion of data points that belong to the class. The class-specific mean vector is the average of the input variables that belong to the class.

QDA is an extension of Linear Discriminant Analysis (LDA). Unlike LDA, QDA considers each class has its own variance or covariance matrix rather than to have a common one.

Why Neural Networks?

Neural networks are very good at pattern recognition problems. A neural network with enough elements (called neurons) can classify any data with arbitrary accuracy. They are particularly well suited for complex decision boundary problems over many variables. Therefore, neural networks are a good candidate for solving the wine classification problem.

More about QDA

QDA algorithm is based on Bayes theorem and classification of an observation is done in following two steps.

Identify the distribution for input X for each of the class (or groups ex Y=k1, k2, k3 etc).
Flip the distribution using Bayes theorem to calculate the probability Pr(Y=k|X=x).

$$\Pr(Y=k \mid X=x) = \frac{\Pr(X=x|Y=k) * \Pr(Y=k)}{\sum_{p=1}^{p=k} \Pr(X=x|Y=p) * \Pr(Y=p)}$$

The above equation has following terms:

Pr(Y=k|X=x) – Probability that an observation belongs to response class Y=k, provided X=x.
Pr(X=x|Y=k) – Probability of X=x, for a particular response class Y=k.
The distribution of X=x needs to be calculated from the historical data for every response class Y=k. In LDA algorithm, the distribution is assumed to be Gaussian and exact distribution is plotted by calculating the mean and variance from the historical data.

Pr(Y=k) – a Prior probability that an observation is of particular class Y=k.
∑(Pr(X=x|Y=p)*Pr(Y=p)) – Sum of probability that an observation is of type X=x for all classes of Y.

QDA Implementation

```
> # Using wine dataset to get a more accurate esmation
  > table(qda.pred$class,wine[test,]$class)
> #setting range between 1 and 50 to test samples
  > for (i in 1:50) {
    + test=sample(178,45)
    + qda.fit=qda(class~.,data=wine[-test,])
    + qda.pred=predict(qda.fit,wine[test,])
```

The output contains the group means. But it does not contain the coefficients of the linear discriminants, because the QDA classifier involves a quadratic, rather than a linear, function of the predictors. The predict() function works in exactly the same fashion as for LDA.

```
> #Using QDA method to process and  to get a more accurate estimate of the performance

> qda.fit=qda(class~.,data=wine[-test,])

> qda.pred=predict(qda.fit,wine[test,])

> # Using wine dataset to get a more accurate esmation

> table(qda.pred$class,wine[test,]$class)
```

```
predictions_QDA = data.frame(predict(model_QDA, test))

predictions_QDA = cbind(test, predictions_QDA)

predictions_QDA %>%

  count(class, Direction)

predictions_QDA %>%

  summarize(score = mean(class == Direction))
```

```
> #Here it will show how it's more accurate
    + Accuracy[i]=mean(qda.pred$class==wine[test,]$class)
    + }
> #summing the all datasets to get a result of accuracy
    > #it will display the result of wine dataset
    > sum(Accuracy)/50
[1] 0.9866667
```

Discriminant analysis is statistical technique used to classify observations into non-overlapping groups, based on scores on one or more quantitative predictor variables. For example, a doctor could perform a discriminant analysis to identify patients at high or low risk for stroke.

KNN Analysis & Implementation

K-Nearest Neighbors

LDA Includes

- Training a dataset
- Applying KNN in Rstudio
- KNN is simply short for K-nearest Neighbors
  - One of the most important concepts during this semester

What Exactly is KNN?

- KNN stands for k-nearest neighbors, and can be seen as a type of algorithm which is a non-parametric supervised learning method useful for classification and regression. Therefore, KNN is an excellent fit for this particular data set and project which is classification-heavy.
- K Nearest Neighbor(KNN) algorithm is a very simple and versatile machine learning algorithms where the output is a class membership in classification.
- An object is classified by a plurality vote of its neighbours, with the object allocated to the class most frequent among its k nearest neighbours (k is a positive integer). If k = 1, 2, 3,4 ,etc., then the object is simply reserved to the class of that single nearest neighbour.

KNN Implementation prologue *summary

```
> #Since we have three types of wine on our hands, let's start with class 1 which is the 1st type of wine, the one with 59 instances
> onewine<-subset(wine_ds,wine_ds$class == 1)
> summary(onewine)
     class        alcohol          malic_acid          ash          alcalinity_of_ash   magnesium       total_phenols     flavonoids     nonflavanoid_phenols
 Min.   :1    Min.   :12.85   Min.   :1.350   Min.   :2.040   Min.   :11.20    Min.   : 89.0   Min.   :2.20   Min.   :2.190   Min.   :0.170
 1st Qu.:1    1st Qu.:13.40   1st Qu.:1.665   1st Qu.:2.295   1st Qu.:16.00    1st Qu.: 98.0   1st Qu.:2.60   1st Qu.:2.680   1st Qu.:0.255
 Median :1    Median :13.75   Median :1.770   Median :2.440   Median :16.80    Median :104.0   Median :2.80   Median :2.980   Median :0.290
 Mean   :1    Mean   :13.74   Mean   :2.011   Mean   :2.456   Mean   :17.04    Mean   :106.3   Mean   :2.84   Mean   :2.982   Mean   :0.290
 3rd Qu.:1    3rd Qu.:14.10   3rd Qu.:1.935   3rd Qu.:2.615   3rd Qu.:18.70    3rd Qu.:114.0   3rd Qu.:3.00   3rd Qu.:3.245   3rd Qu.:0.320
 Max.   :1    Max.   :14.83   Max.   :4.040   Max.   :3.220   Max.   :25.00    Max.   :132.0   Max.   :3.88   Max.   :3.930   Max.   :0.500
 proanthocyanins color_intensity       hue           OD280_OD315       proline
 Min.   :1.250   Min.   :3.520   Min.   :0.820   Min.   :2.510   Min.   : 680.0
 1st Qu.:1.640   1st Qu.:4.550   1st Qu.:0.995   1st Qu.:2.870   1st Qu.: 987.5
 Median :1.870   Median :5.400   Median :1.070   Median :3.170   Median :1095.0
 Mean   :1.899   Mean   :5.528   Mean   :1.062   Mean   :3.158   Mean   :1115.7
 3rd Qu.:2.090   3rd Qu.:6.225   3rd Qu.:1.130   3rd Qu.:3.420   3rd Qu.:1280.0
 Max.   :2.960   Max.   :8.900   Max.   :1.280   Max.   :4.000   Max.   :1680.0
> dim(onewine)
[1] 59 14
> #everything seems to check out, now we can move on to class 2, which has 71 instances
> twowine<-subset(wine_ds, wine_ds$class == 2)
> summary(twowine)
     class        alcohol          malic_acid          ash          alcalinity_of_ash   magnesium        total_phenols      flavonoids      nonflavanoid_phenols
 Min.   :2    Min.   :11.03   Min.   :0.740   Min.   :1.360   Min.   :10.60    Min.   : 70.00   Min.   :1.100   Min.   :0.570   Min.   :0.1300
 1st Qu.:2    1st Qu.:11.91   1st Qu.:1.270   1st Qu.:2.000   1st Qu.:18.00    1st Qu.: 85.50   1st Qu.:1.895   1st Qu.:1.605   1st Qu.:0.2700
 Median :2    Median :12.29   Median :1.610   Median :2.240   Median :20.00    Median : 88.00   Median :2.200   Median :2.030   Median :0.3700
 Mean   :2    Mean   :12.28   Mean   :1.933   Mean   :2.245   Mean   :20.24    Mean   : 94.55   Mean   :2.259   Mean   :2.081   Mean   :0.3637
 3rd Qu.:2    3rd Qu.:12.52   3rd Qu.:2.145   3rd Qu.:2.420   3rd Qu.:22.00    3rd Qu.: 99.50   3rd Qu.:2.560   3rd Qu.:2.475   3rd Qu.:0.4300
 Max.   :2    Max.   :13.86   Max.   :5.800   Max.   :3.230   Max.   :30.00    Max.   :162.00   Max.   :3.520   Max.   :5.080   Max.   :0.6600
 proanthocyanins color_intensity       hue           OD280_OD315       proline
 Min.   :0.410   Min.   :1.280   Min.   :0.690   Min.   :1.590   Min.   :278.0
 1st Qu.:1.350   1st Qu.:2.535   1st Qu.:0.925   1st Qu.:2.440   1st Qu.:406.5
 Median :1.610   Median :2.900   Median :1.040   Median :2.830   Median :495.0
 Mean   :1.630   Mean   :3.087   Mean   :1.056   Mean   :2.785   Mean   :519.5
 3rd Qu.:1.885   3rd Qu.:3.400   3rd Qu.:1.205   3rd Qu.:3.160   3rd Qu.:625.0
 Max.   :3.580   Max.   :6.000   Max.   :1.710   Max.   :3.690   Max.   :985.0
> dim(twowine)
[1] 71 14
> #before we move on, we should make same quick observations
> #the max of alcohol in class 1 is bigger than class2, max malic acid content is higher in class2, max ash is higher slighlty in class 2, but what does this all
mean? I will elaborate on this further in the project report.
> #finally, we conclude with class 3
> threewine<-subset(wine_ds, wine_ds$class == 3)
> summary(threewine)
     class        alcohol          malic_acid          ash          alcalinity_of_ash   magnesium         total_phenols      flavonoids        nonflavanoid_phenols
 Min.   :3    Min.   :12.20   Min.   :1.240   Min.   :2.100   Min.   :17.50    Min.   : 80.00   Min.   :0.980   Min.   :0.3400   Min.   :0.1700
 1st Qu.:3    1st Qu.:12.80   1st Qu.:2.587   1st Qu.:2.300   1st Qu.:20.00    1st Qu.: 89.75   1st Qu.:1.407   1st Qu.:0.5800   1st Qu.:0.3975
 Median :3    Median :13.16   Median :3.265   Median :2.380   Median :21.00    Median : 97.00   Median :1.635   Median :0.6850   Median :0.4700
 Mean   :3    Mean   :13.15   Mean   :3.334   Mean   :2.437   Mean   :21.42    Mean   : 99.31   Mean   :1.679   Mean   :0.7815   Mean   :0.4475
 3rd Qu.:3    3rd Qu.:13.51   3rd Qu.:3.958   3rd Qu.:2.603   3rd Qu.:23.00    3rd Qu.:106.00   3rd Qu.:1.808   3rd Qu.:0.9200   3rd Qu.:0.5300
 Max.   :3    Max.   :14.34   Max.   :5.650   Max.   :2.860   Max.   :27.00    Max.   :123.00   Max.   :2.800   Max.   :1.5700   Max.   :0.6300
 proanthocyanins color_intensity       hue            OD280_OD315       proline
 Min.   :0.550   Min.   : 3.850   Min.   :0.4800   Min.   :1.270   Min.   :415.0
 1st Qu.:0.855   1st Qu.: 5.438   1st Qu.:0.5875   1st Qu.:1.510   1st Qu.:545.0
 Median :1.105   Median : 7.550   Median :0.6650   Median :1.660   Median :627.5
 Mean   :1.154   Mean   : 7.396   Mean   :0.6827   Mean   :1.684   Mean   :629.9
 3rd Qu.:1.350   3rd Qu.: 9.225   3rd Qu.:0.7525   3rd Qu.:1.820   3rd Qu.:695.0
 Max.   :2.700   Max.   :13.000   Max.   :0.9600   Max.   :2.470   Max.   :880.0
> dim(threewine)
[1] 48 14
> #59 + 71 + 48 = 178 instances, check confirmed
```

KNN Implementation

```
> library(class)
> knn.pred = knn(wine_ds[-test, 2:14], wine_ds[test, 2:14],wine_ds[-test, ]$class, k = 1)
> table(knn.pred,wine_ds[test, ]$class)

knn.pred  1  2  3
       1 12  0  0
       2  5 17  4
       3  0  0  7
> mean(knn.pred == wine_ds[test, ]$class)
[1] 0.8
```

```
> for (i in 1:50){
+ test = sample(178,59)
+ knn.pred = knn(wine_ds[-test, 2:14], wine_ds[test, 2:14],wine_ds[-test, ]$class, k = 1)
+ Percentage[i] = mean(knn.pred==wine_ds[test,]$class)
+ }
> sum(Percentage)/50
[1] 0.7352542
> |
```

KNN implementation : To begin coding, we need library(class) to successfully move forward with the KNN code. A confusion matrix is generated and it examines segments of class 1, class 2 and class 3. The mean turns out to be 0.8, the median is 0.78 (not shown on this slide).

The picture on the left is KNN method, but not quite rescaled. It is always important to scale as rescaling (e.g. convert char to factor, normalize,etc.)gives you a better idea of the data should actually look like.

I went from K = 1, to K = 2, K= 3, K = ?, etc., for maximum accuracy.(further elaborated on project report)

KNN Implementation

```
> #KNN implementation
> Percentage = rep(0,50)
> test = sample(178,59) #1st class with 59 instances
> for (i in 1:50){
+ test = sample(178,59)
+ knn.pred = knn(scale(wine_ds[-test, 2:14]), scale(wine_ds[test, 2:14]),wine_ds[-test, ]$class,
k = 5)
+ Percentage[i] = mean(knn.pred==wine_ds[test,]$class)
+ }
> sum(Percentage)/50
[1] 0.9501695
```

```
> for (i in 1:50){
+ test = sample(178,71)
+ knn.pred = knn(scale(wine_ds[-test, 2:14]), scale(wine_ds[test, 2:14]),wine_ds[-test, ]$class,
k = 5)
+ Percentage[i] = mean(knn.pred==wine_ds[test,]$class)
+ }
> sum(Percentage)/50
[1] 0.9546479
> #class 2 with 71 instances
```

```
> #class 3 with 48 instances
> for (i in 1:50){
+ test = sample(178,48)
+ knn.pred = knn(scale(wine_ds[-test, 2:14]), scale(wine_ds[test, 2:14]),wine_ds[-test, ]$class,
k = 5)
+ Percentage[i] = mean(knn.pred==wine_ds[test,]$class)
+ }
> sum(Percentage)/50
[1] 0.95125
```

In order for the R studio code to work properly in the context of the implementation, data must be pre-processed appropriately for K-nearest neighbors, and the rescaling of the data is taking place in the above code. This code has now been rescaled, and this is contrasted with the previous slide. Class 1:59 instances, class 2: 71 instances, class 3: 48 instances

K = 1 to K = 3

```
> # from K = 1 to K = 3
> for (i in 1:50){
+ test = sample(178,59)
+ knn.pred = knn(scale(wine_ds[-test, 2:14]), scale(wine_ds[test, 2:14]),wine_ds[-test, ]$class, k = 1)
+ Percentage[i] = mean(knn.pred==wine_ds[test,]$class)
+ }
> sum(Percentage)/50
[1] 0.9440678
> for (i in 1:50){
+ test = sample(178,59)
+ knn.pred = knn(scale(wine_ds[-test, 2:14]), scale(wine_ds[test, 2:14]),wine_ds[-test, ]$class, k = 2)
+ Percentage[i] = mean(knn.pred==wine_ds[test,]$class)
+ }
> sum(Percentage)/50
[1] 0.9311864
> for (i in 1:50){
+ test = sample(178,59)
+ knn.pred = knn(scale(wine_ds[-test, 2:14]), scale(wine_ds[test, 2:14]),wine_ds[-test, ]$class, k = 3)
+ Percentage[i] = mean(knn.pred==wine_ds[test,]$class)
+ }
> sum(Percentage)/50
[1] 0.9416949
>
```

K = 1 to K=3 of K Nearest Neighbors. This is using wine type number 1 as an example, but the other two classes were used extensively. We get a clear output and percentage rate, and it will be the same situation for K = 4 , to K = 7

KNN Implementation K=4 to K = 7

```
> #from K = 4 to K = 6
> for (i in 1:50){
+ test = sample(178,59)
+ knn.pred = knn(scale(wine_ds[-test, 2:14]), scale(wine_ds[test, 2:14]),wine_ds[-test, ]$class, k = 4)
+ Percentage[i] = mean(knn.pred==wine_ds[test,]$class)
+ }
> sum(Percentage)/50
[1] 0.9525424
> for (i in 1:50){
+ test = sample(178,59)
+ knn.pred = knn(scale(wine_ds[-test, 2:14]), scale(wine_ds[test, 2:14]),wine_ds[-test, ]$class, k = 5)
+ Percentage[i] = mean(knn.pred==wine_ds[test,]$class)
+ }
> sum(Percentage)/50
[1] 0.9461017
> for (i in 1:50){
+ test = sample(178,59)
+ knn.pred = knn(scale(wine_ds[-test, 2:14]), scale(wine_ds[test, 2:14]),wine_ds[-test, ]$class, k = 6)
+ Percentage[i] = mean(knn.pred==wine_ds[test,]$class)
+ }
> sum(Percentage)/50
[1] 0.9474576
> for (i in 1:50){
+ test = sample(178,59)
+ knn.pred = knn(scale(wine_ds[-test, 2:14]), scale(wine_ds[test, 2:14]),wine_ds[-test, ]$class, k = 7)
+ Percentage[i] = mean(knn.pred==wine_ds[test,]$class)
+ }
> sum(Percentage)/50
[1] 0.9538983
> 
```

K = 4 to K=7 of K Nearest Neighbors. This is using wine type number 1 as an example still, but the other two classes were used extensively. We get a clear output and percentage rate.

Comparison of Data/ Summary

(contrast the KNN,LDA, and QDA results/outputs, revert back to research question)

- QDA and LDA (99 - 96%)most accurate of the ML methods for
- KNN (90 - 95%) is slightly behind both in terms of efficiency, accuracy,etc.

Challenges deep-dive

LDA

- What to look for as far as implementing this method (prediction)

QDA

KNN

- Rstudio cloud is not as good as regular rstudio as encountering issues with saving and crashes was a regular occurrence

Limitations/Lessons Learned

Limitations

- Initial Dataset had around 30 variables, but author states information was “lost”
- Regularized discriminant analysis (RDA) was not covered
- Having trouble organizing project meetings due to our busy schedules outside of class

Adjustments Made

- Effective usage of discord and when2meet or organize good progress
- Divided work equally as much as possible to leave time for other classes

Lessons Learned

- This wine dataset was great for first testing of a new classifier
- Data set is good for classification tasks and good for anyone looking to improve in that subject

REFERENCES

- Wine Data Set

  https://archive.ics.uci.edu/ml/datasets/Wine

- Forina, M. et al, PARVUS - An Extendible Package for Data Exploration, Classification and Correlation. Institute of Pharmaceutical and Food Analysis and Technologies, Via Brigata Salerno, 16147 Genoa, Italy.

CONCLUSION

- Thank you for listening to our presentation!