Instituto Tecnológico y de Estudios Superiores de Monterrey

Campus Ciudad de México

Erick Axel Martinez Ríos

ID: A01331212

Applied Computing

December 1st, 2020

Content

Question 1

Link for GOOGLE COLAB: https://colab.research.google.com/drive/1qxsjAkKlC9pKK3qSXYsAWc8FT2kWi8M8

A supplier must prepare at least 500 gallons of an innovative beverage using 5 fruit drinks at the lowest possible cost. The new drink should contain at least 20% orange juice, 10% grapefruit juice, and 5% cranberry juice. The characteristics of the base drinks, their availability, and cost are given in the following table.

Base Drink	Orange Juice	Grapefruit juice	Cranberry juice	Inventory (gal)	Cost ($/gal)
Drink A	40%	40%	0	200	1.5
Drink B	5%	10%	20%	400	0.75
Drink C	100%	0	0	100	2.0
Drink D	0	100/	0	50	1.75
Drink E	0	0	0	800	0.25

Deliver a jupyter notebook file with the following answers:

1. Formulate the mathematical model (10 pts)
2. Use python (scypy.optimize/pulp libraries ONLY) to solve the model. Print objective function, variables, shadow price, and reduced cost results. (10 pts)
3. Give an interpretation of reduced cost and shadow price. (10 pts)

Note: To look at the procedure perform to make this analysis please refer to the attached Jupyter Notebook in the section for Question 1.

1. Formulate the mathematical model

Minimize Objective Function

The coefficients are obtained based on the cost established in the given table.

$Min\ z = \ 1.5A + 0.75B + 2.0C + 1.75D + 0.25E$

Constrains

1) $A + B + C + D + E \geq \ 500$ At least 500 gallons of an innovative beverage
2) $0.4A + 0.05B \ + \ C \geq \ (A + B + C + D + E)(0.2)$ At least 20% of orange juice
3) $0.4A + 0.1B \ + \ D \geq \ (A + B + C + D + E)(0.1)$ At least 10% of grapefruit juice
4) $0.2B \geq \ (A + B + C + D + E)(0.05)$ At least 5% of cranberry juice
5) $A \leq \ 200$ Inventory constrain of Drink A
6) $B \leq \ 400$ Inventory constrain of Drink B
7) $C \leq \ 100$ Inventory constrain of Drink C
8) $D \leq \ 50$ Inventory constrain of Drink D
9) $E \leq \ 800$ Inventory constrain of Drink E
10) $A, B, C, D, E \geq \ 0$ Constrain to assure that the variables are greater or equal to zero.

Explanation of the constrains.

- Since the problem states that **at least 500 gallons** of punch must be produced the sum of Drinks A, B, C, D, and E must satisfy that condition.
- With constrains 2,3, and 4 the problem states that the minimum amount of orange juice present in the punch must be at least 20%, 10% for the grapefruit juice, and 5% for cranberry juice. Therefore, the sum of the quantity of Drinks A, B, C, D, and C used to make the punch must satisfy those conditions.

- Conditions 5 to 9 established the availability of all the Drinks according to the given table.

2. Use python (scypy.optimize/pulp libraries ONLY) to solve the model. Print objective function, variables, shadow price, and reduced cost results. (10 pts)

Figure 1 shows the definition of the variables of the optimization problem using the Pulp library. In this case, the variables are related to the quantity of the Drinks use to produce the punch.

```
Define Minimization problem

[394] model_1 = LpProblem("Minimize", LpMinimize)

Define the variables

[395] A = LpVariable('A',0, None)
      B = LpVariable('B',0, None)
      C = LpVariable('C',0, None)
      D = LpVariable('D',0, None)
      E = LpVariable('E',0, None)
```

Figure 1. Model and variables definition using Pulp.

In Figures 2 and 3, the objective function and constrains are defined as they were established in the previous section.

```
#Define Objective Function
model_1 += 1.5*A + 0.75*B + 2*C+1.75*D+ 0.25*E
#Define constrains
model_1 += A + B + C + D + E >= 500  # Constrain for at least 500 gallons
model_1 += 0.4*A + 0.05*B + C >= (A+B+C+D+E)*0.2 # Constrain for at least 20% of orange juice
model_1 += 0.4*A + (0.1)*B + D >= (A+B+C+D+E)*0.1 # Constrain for at least 10% of grapefruit juice
model_1 += (0.2)*B >= (A+B+C+D+E)*0.05 # Constrain for at least 5% of cranberry juice
model_1 += A <= 200 # Constrain for Drink 1
model_1 += B <= 400 # Constrain for Drink 2
model_1 += C <= 100 # Constrain for Drink 3
model_1 += D <= 50 # Constrain for Drink 4
model_1 += E <= 800 # Constrain for Drink 5
```

Figure 2. Objective function and constraints.

```
Minimize:
MINIMIZE
1.5*A + 0.75*B + 2*C + 1.75*D + 0.25*E + 0.0
SUBJECT TO
_C1: A + B + C + D + E >= 500

_C2: 0.2 A - 0.15 B + 0.8 C - 0.2 D - 0.2 E >= 0

_C3: 0.3 A + 0 B - 0.1 C + 0.9 D - 0.1 E >= 0

_C4: - 0.05 A + 0.15 B - 0.05 C - 0.05 D - 0.05 E >= 0

_C5: A <= 200

_C6: B <= 400

_C7: C <= 100

_C8: D <= 50

_C9: E <= 800

VARIABLES
A Continuous
B Continuous
C Continuous
D Continuous
E Continuous
```

Figure 3. The objective function and constrain definition using Pulp.

The optimal values for each of the drinks and the minimum cost is displayed in Figure 4. It is possible to observe that for Drink D the amount to be used is equal to zero. This value is expected since the cost of that particular Drink is higher compared to other Drinks so its use could increase the optimal cost and the solver tries to minimize the cost of the given problem.

```
Use 93.75 gal of Drink A
Use 125.0 gal of Drink B
Use 56.25 gal of Drink C
Use 0.0 gal of Drink D
Use 225.0 gal of Drink E
Min cost: 403.125 $
```

Figure 4. Optimal values for each of the drinks and minimum cost.

The shadow price and reduced cost of the model can be appreciated in Figure 5.

```
A = 93.75       Reduced Cost = -4.1633363e-17
B = 125.0       Reduced Cost = 1.6653345e-16
C = 56.25       Reduced Cost = -4.1633363e-17
D = 0.0         Reduced Cost = 0.125
E = 225.0       Reduced Cost = 1.2490009e-16
  name  ShadowPrice        tSlack
0  _C1      0.80625 -0.000000e+00
1  _C2      1.75000 -1.421086e-14
2  _C3      1.37500 -0.000000e+00
3  _C4      1.37500 -3.552714e-15
4  _C5      0.00000  1.062500e+02
5  _C6      0.00000  2.750000e+02
6  _C7      0.00000  4.375000e+01
7  _C8      0.00000  5.000000e+01
8  _C9      0.00000  5.750000e+02
```

Figure 5. Reduced cost of each variable coefficient and shadow price of each constrain.

3. Give an interpretation of reduced cost and shadow price.

In this case, since the objective function it's been minimized, the reduced costs displayed in Figure 5 showed the amount by which the coefficients of the variables of the objective function should decrease to assume a positive value in the optimal solution or, in other words, the corresponding variable of that reduced cost would be cost-effective. For Drinks A, B, C, and E the reduced cost is very small in the order of 10^{-16} and 10^{-17}, so they can be neglected. Nevertheless, the Reduced cost of Drink D is 0.125, this value implies that it is necessary to reduce the value of the coefficient of the D variable to an amount of 0.125 to be cost-effective. The use of the reduced cost for Drink D can be appreciated in Figure 6. By looking at the results of applying the reduced cost in the original objective function for the D coefficient, it is possible to notice that the optimal solution did not change, nevertheless, the value of Drink D became positive in the optimal solution becoming cost-effective. One aspect to notice is that the value of the other variables in the optimal solution also changed, this is expected since the original objective function was modified. With this brief procedure, it is possible to observe that the reduced costs are working as expected.

```
#Define Objective Function
model_2 += 1.5*A + 0.75*B + 2*C+(1.75-0.125)*D+ 0.25*E
#Define constrains
model_2 += A + B + C + D + E >= 500 # Constrain for at least 500 gallons increase by one unit
model_2 += 0.4*A + 0.05*B + C >= (A+B+C+D+E)*0.2 # Constrain for at least 20% of orange juice
model_2 += 0.4*A + (0.1)*B + D >= (A+B+C+D+E)*0.1 # Constrain for at least 10% of grapefruit juice
model_2 += (0.2)*B >= (A+B+C+D+E)*0.05 # Constrain for at least 5% of cranberry juice
model_2 += A <= 200 # Constrain for Drink 1
model_2 += B <= 400 # Constrain for Drink 2
model_2 += C <= 100 # Constrain for Drink 3
model_2 += D <= 50 # Constrain for Drink 4
model_2 += E <= 800 # Constrain for Drink 5
```

```
Use 0.0 gal of Drink A
Use 125.0 gal of Drink B
Use 93.75 gal of Drink C
Use 37.5 gal of Drink D
Use 243.75 gal of Drink E
Min cost: 403.125 $
```

Figure 6. Reduction of the coefficient of Drink D according to the Reduce cost obtained.

In the case of constraints 5 to 9, the shadow prices are equal to zero. This implies that an increment of one unit on the right side of the constraint will not produce a change in the optimal value. On the other hand, the constraints 1 to 4 have a nonzero shadow price. With constrain 1, the increment of 1 unit on its right side will produce an increment in the optimal value of 0.80625. This increment is expected since the supplier will require more gallons of each drink to produce the required minimum amount. A similar case occurs for the constrains 2 to 4 that are related to the percentage of each juice required in the punch where an increment in the optimal value will occur if the required number of gallons of each drink increases.

To corroborate the behavior of the shadow prices, the following test was carried out for the first constrain. By looking at the figure below the right side of the first constrain was increase by one unit. So, it is expected that the optimal value increase by the amount of 0.80625.

```
#Define Objective Function
model_2 += 1.5*A + 0.75*B + 2*C+1.75*D+ 0.25*E
#Define constrains
model_2 += A + B + C + D + E >= 500 +1 # Constra
model_2 += 0.4*A + 0.05*B + C >= (A+B+C+D+E)*0.2
model_2 += 0.4*A + (0.1)*B + D >= (A+B+C+D+E)*0
model_2 += (0.2)*B >= (A+B+C+D+E)*0.05 # Constra
model_2 += A <= 200 # Constrain for Drink 1
model_2 += B <= 400 # Constrain for Drink 2
model_2 += C <= 100 # Constrain for Drink 3
model_2 += D <= 50 # Constrain for Drink 4
model_2 += E <= 800 # Constrain for Drink 5
```

Figure 7. Addition of unit on the right side of the first constrain.

By finding the optimal value, the optimal solution change by the amount expected as shown in Figure 8.

$$403.125 + 0.80625 = 403.93125$$

```
Use 93.9375 gal of Drink A
Use 125.25 gal of Drink B
Use 56.3625 gal of Drink C
Use 0.0 gal of Drink D
Use 225.45 gal of Drink E
Min cost: 403.93125000000003 $
```

Figure 8. Change in the optimal value due to the increase of unit on the right side of the first constrain.

This procedure can also be applied to other constrains to observe the expected increment. On the other hand, if a decrease of one unit occurs on the right side of the first constrain the optimal value will decrease by an amount of 0.80625. This is expected since the minimum quantity of punch is decreasing so fewer gallons of the drinks will be used reducing the optimal cost. The same behavior will be expected for constraints 2, 3, and 4.

```
#Define Objective Function
model_2 += 1.5*A + 0.75*B + 2*C+1.75*D+ 0.25*E
#Define constrains
model_2 += A + B + C + D + E >= 500 -1 # Constra
model_2 += 0.4*A + 0.05*B + C >= (A+B+C+D+E)*0.2
model_2 += 0.4*A + (0.1)*B + D >= (A+B+C+D+E)*0.
model_2 += (0.2)*B >= (A+B+C+D+E)*0.05 # Constra
model_2 += A <= 200 # Constrain for Drink 1
model_2 += B <= 400 # Constrain for Drink 2
model_2 += C <= 100 # Constrain for Drink 3
model_2 += D <= 50 # Constrain for Drink 4
model_2 += E <= 800 # Constrain for Drink 5
```

$$403.125 - 0.80625 = 402.319$$

```
Use 93.5625 gal of Drink A
Use 124.75 gal of Drink B
Use 56.1375 gal of Drink C
Use 0.0 gal of Drink D
Use 224.55 gal of Drink E
Min cost: 402.31874999999997 $
```

Figure 9. Change in the optimal value due to the decrease of a unit on the right side of the first restriction.

Question 2

- Perform a principal component analysis for the following file. pcanutrition.csv. Preview the document
- Define how many components are necessary to keep 60% of the variance of the data. (10 pts)
- What are the three most significant variables for the dataset? (10 pts)

Note: To look at the procedure perform to make this analysis please refer to the attached Jupyter Notebook in the section for Question 2.

The database has a total of 8618 samples and 42 columns. Nevertheless, some of the columns including the ID column were removed before performing the PCA analysis. The other columns that were removed and the number of different values that they have are listed below.

- Food group: 25
- ShortDescrip: 8614
- Descrip: 8618

Beef Products	946
Vegetables and Vegetable Products	828
Baked Products	797
Soups, Sauces, and Gravies	452
Lamb, Veal, and Game Products	438
Poultry Products	390
Legumes and Legume Products	389
Fast Foods	371
Breakfast Cereals	363
Baby Foods	362
Sweets	347
Fruits and Fruit Juices	346
Pork Products	343
Beverages	315
Finfish and Shellfish Products	267
Dairy and Egg Products	264
Sausages and Luncheon Meats	244
Fats and Oils	219
Cereal Grains and Pasta	183
Snacks	171
American Indian/Alaska Native Foods	165
Nut and Seed Products	133
Meals, Entrees, and Side Dishes	113
Restaurant Foods	108
Spices and Herbs	64

Figure 1. The number of samples for each of the categories contained in the FoodGroup column.

The description columns have different annotations (8618 different descriptions) for each of the samples of the database so treat them as categorical variables could complicate its analysis and the interpretation given by the PCA algorithm. For the FoodGroup it is necessary to perform an encoding process for each of the categories listed in Figure 1 that could be misinterpreted by the PCA analysis. If label encoding is performed it is not possible to know if the assigned label will represent the category. Another aspect to consider is that the use of label encoding will introduce a comparison between the categories of the variable or a hierarchical relationship between them that might not be the case for this particular variable. On the other hand, if the one-hot encoding is employed the size of the whole database will increase due to the number of dummy variables that are needed for each of the categories that this column has. If the procedure is applying the database will have 63 columns, which makes the analysis and interpretation more difficult. Therefore, these columns were removed from the database. With the removal of the mentioned columns, the size of the database is 8618 rows and 38 columns. Before applying the PCA analysis, it is important to standardize the variables for this purpose a z-score normalization was used, this process was done using the standard scaler function of the sklearn package. Finally, the PCA analysis was applied using the sklearn package. The red dots of Figure 2 indicate the accumulative variance percentage by using the first five and six principal components. By looking at Figure 2, it is possible to observe that the five principal components of the PCA analysis are keeping almost 60% of the variance of the data.

The specific percentage variance for the first 5 principal components is listed below.

- Variance percentage of the first component: 25.970166594146327%
- Variance percentage of the second component: 10.828205053166708%
- Variance percentage of the third component: 9.019010603018579%
- Variance percentage of the fourth component: 7.716865501709974%
- Variance percentage of the fifth component: 6.266698793844751%
- **Accumulative variance percentage of the first five principal components: 59.80094654588634%**

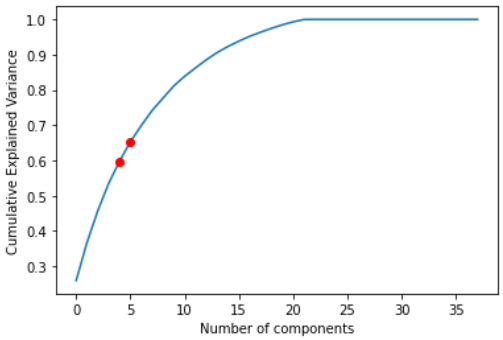


Figure 2. Accumulative explained variance.

As it was shown the first five principal components had an accumulative variance of 59.8009%. If six principal components are used, the variance percentage and accumulative variance are as follows:

- Variance percentage of the first component: 25.970166594146327%
- Variance percentage of the second component: 10.828205053166708%
- Variance percentage of the third component: 9.019010603018579%
- Variance percentage of the fourth component: 7.716865501709974%
- Variance percentage of the fifth component: 6.266698793844751%
- Variance percentage of the sixth component: 5.508855438676981%
- **Accumulative variance percentage of the first six principal components: 65.30980198456332%**

By using the first six components the accumulative variance is 65.30%. Since the problem asks to determine what are the number of principal components that keep 60% of the variance of the data, the first five principal components (59.80%) almost reach that 60% percent of the variance. On the other hand, the first six components reach 65.30% of variance so it will be **a better selection** to use the **first six principal components** to **assure that the sixty percent of the variance is kept.**

What are the three most significant variables for the dataset? (10 pts)

To determine what are the most significant variables of the PCA analysis, it is necessary to compute the loading scores of the first principal component that keeps the highest variance (25.97%) as shown in Figure 3.

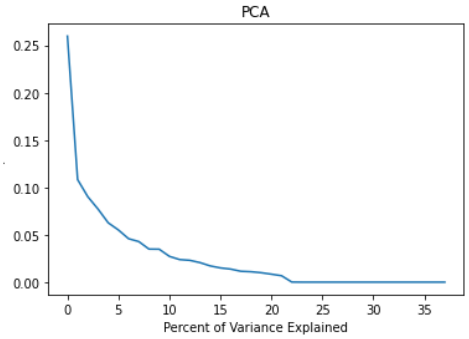


Figure 3. Percent of variance explained of each of the components in the PCA analysis.

The most significant variables are the ones with the highest absolute loading scores in the first principal component, for the PCA analysis that was done these variables can be observed in Figure 4. In this Figure, it is possible to notice that some variables have the same loading score. In fact, those variables are highly correlated, this correlation can be appreciated with more detail in the heatmap of Figure 5. This is expected since both variables contained the same information but with different units (mg and USRDA) according to the names of each of the columns. Taking this into account the most important variables according to the PCA analysis **are Riboflavin, Niacin, and VitB6 for both units mg and USRDA**.

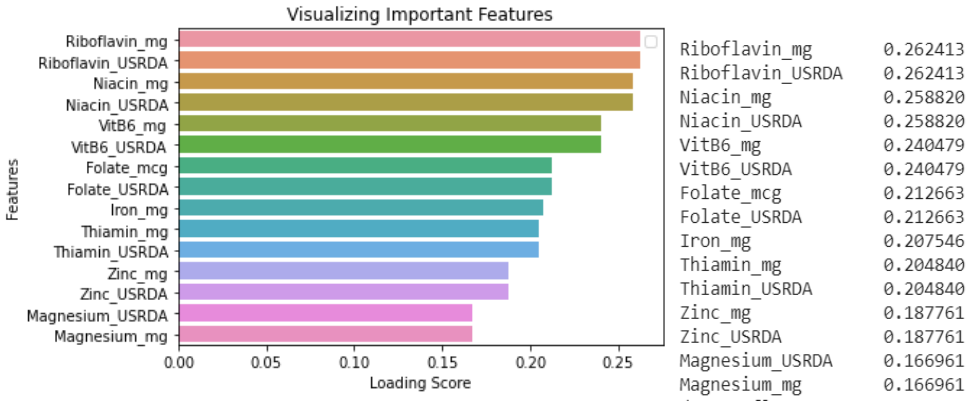


Figure 4. Highest Loading Scores in the first principal component.

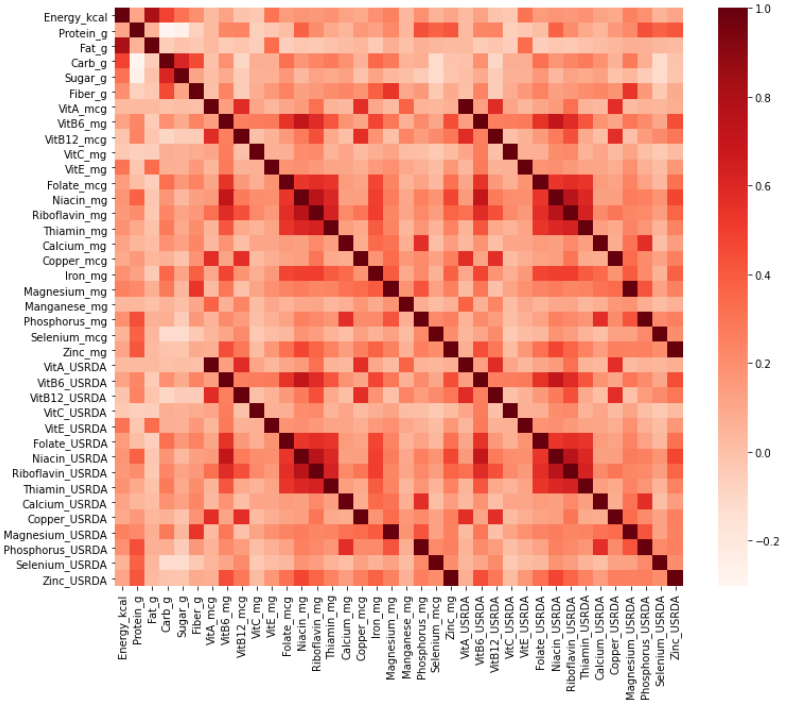


Figure 5. Heatmap of correlation between variables for the nutrition database.

Additional step.

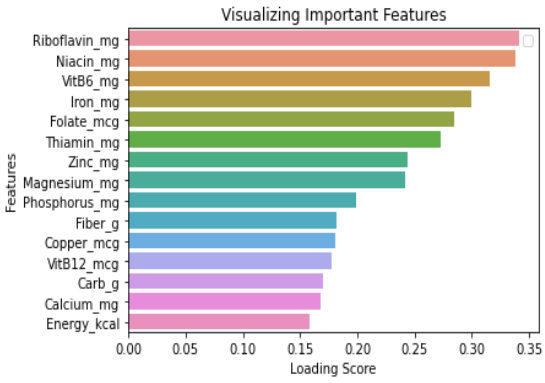
As an extra step, it is possible to remove the highly correlated variables that are appreciated in the heatmap of Figure 5 and compute the PCA analysis and loading scores again. According to the heatmap of Figure 5, 15 variables are highly correlated. By removing those variables, the size of the database is reduced to 23 columns.

In the case of removing the highly correlated variables, the accumulative variance is quite similar to the one obtained by using the highly correlated variables as was showed before. The values obtained are listed below.

- Variance percentage of the first component: 23.692546801354605 %
- Variance percentage of the second component: 11.384601533332067 %
- Variance percentage of the third component: 8.834337340257219 %
- Variance percentage of the fourth component: 8.170136690446628 %
- Variance percentage of the fifth component: 7.111614796968906 %
- Variance percentage of the sixth component: 4.958133319424717 %
- Accumulative variance percentage of the first five principal components: 59.19323716235941 %
- **Accumulative variance percentage of first sixth principal components: 64.15137048178413 %**

It is possible to observe that the first six components kept 60% of the variance of the original data as it was previously observed with all the variables of the database.

By making this procedure the loading scores and most important variables or the variables that hold the greatest variance in the first principal component are shown in the Figure below. **Riboflavin, Niacin, and VitB6 in mg units** are the variables that prevailed after removing the highly correlated features which are the same obtained by not removing the highly correlated variables that were presented in Figure 4.



Riboflavin_mg	0.341325
Niacin_mg	0.337779
VitB6_mg	0.315663
Iron_mg	0.299857
Folate_mcg	0.284102
Thiamin_mg	0.272453
Zinc_mg	0.243551
Magnesium_mg	0.241348
Phosphorus_mg	0.199403
Fiber_g	0.181570
Copper_mcg	0.180806
VitB12_mcg	0.177985
Carb_g	0.169685
Calcium_mg	0.168112
Energy_kcal	0.157814

Figure 6. Loading scores of the 15 most important variables by removing the highly correlated variables from the database.

Question 3

Perform a classification analysis of the following dataset. The target column is **res**. Sweden.csv.

1. Transform the data, split the dataset (80% training, 20% testing) (5 pts)

2. Perform a decision-tree classification and an SVC (10 pts)

3. Obtained the confusion matrix for both methods. Give an interpretation of the results. (20 pts)

NOTE: To look at the procedure perform to make the analysis please refer to the attached Jupyter Notebook in the section for Question 3.

Data Analysis and preprocessing.

The Sweden database has a total of 1936 rows and 19 columns. Nevertheless, some columns and rows were removed before creating the decision tree and the support vector classifiers. There were some columns with missing values as shown in Figure 1 so the rows that contained those missing values were removed since they were only two. This procedure leads to only have 1934 rows and 19 columns.

```
Country    0
League     0
Season     0
Date       0
Time       0
Home       0
Away       0
HG         0
AG         0
Res        0
PH         2
PD         2
PA         2
MaxH       0
MaxD       0
MaxA       0
AvgH       0
AvgD       0
AvgA       0
dtype: int64
```

Figure 1. Missing values in the database.

In the case of the columns, the ones labeled as Country and League were removed since they only have one category. The columns related to time information such as Season, Date, and Time were also removed. This was done since it is assumed that the results of the fight do not depend on the time or date in which the match took place since this information will always change every time a new match occurs in the future. Figure 2 shows the number of seasons, the number of dates in which a match took place, and the number of different times (hours) in which a match occurred.

```
Number of Seasons: 8
Number of dates: 766
Number of Times: 28
```

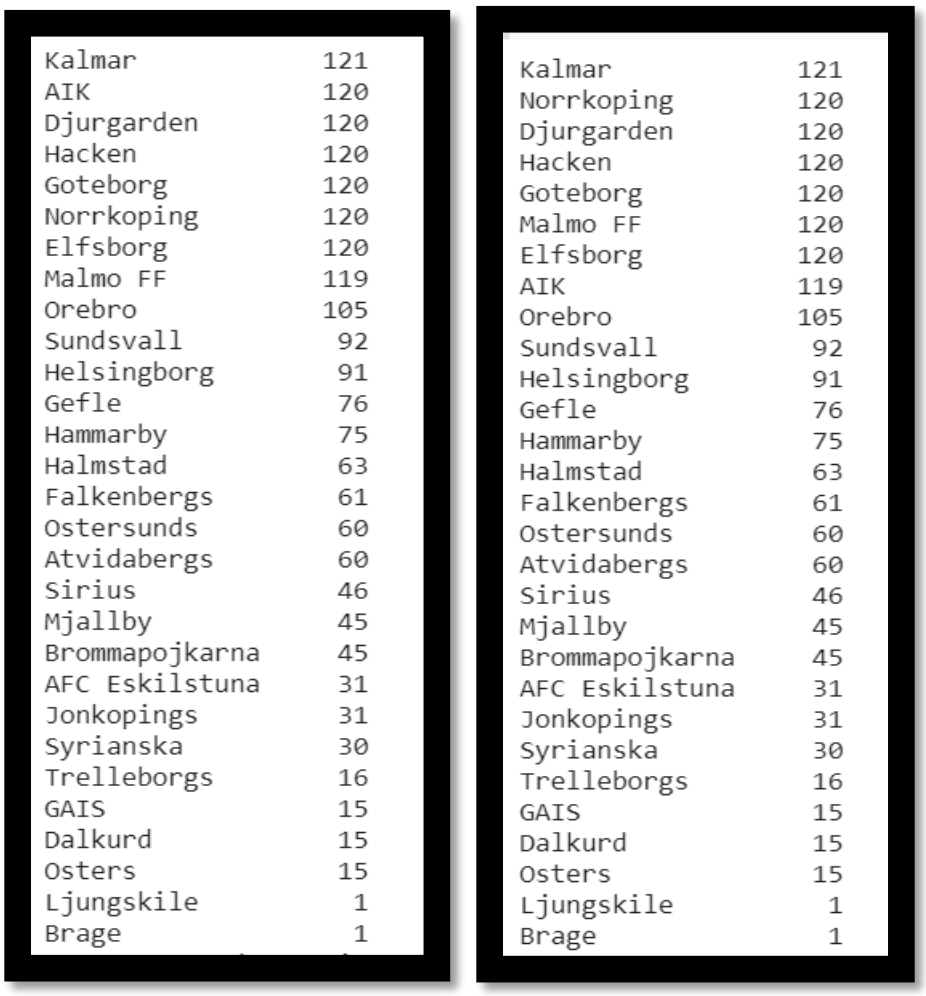
Figure 2. The number of seasons, dates, and times of the matches.

In the case of the name of the Home and Away team, both columns had a total of 29 teams. Nevertheless, to introduce these variables into the classifiers by encoding the name of those teams could be misinterpreted by the algorithm since a hierarchical relationship will be introduced to these variables. On the other hand, creating dummy variables for each of the categories will increase the size and complexity of the database and consequently of the classifier.

Figure 3.1 shows the number of teams in the Home and Away columns, it is possible to notice that all of the teams have played as the home and away team the same number of times.

```
Home: 29
Away: 29
```

Figure 3. The number of Home and Away Teams.



Kalmar	121
AIK	120
Djurgarden	120
Hacken	120
Goteborg	120
Norrkoping	120
Elfsborg	120
Malmo FF	119
Orebro	105
Sundsvall	92
Helsingborg	91
Gefle	76
Hammarby	75
Halmstad	63
Falkenbergs	61
Ostersunds	60
Atvidabergs	60
Sirius	46
Mjallby	45
Brommapojkarna	45
AFC Eskilstuna	31
Jonkopings	31
Syrianska	30
Trelleborgs	16
GAIS	15
Dalkurd	15
Osters	15
Ljungskile	1
Brage	1

Kalmar	121
Norrkoping	120
Djurgarden	120
Hacken	120
Goteborg	120
Malmo FF	120
Elfsborg	120
AIK	119
Orebro	105
Sundsvall	92
Helsingborg	91
Gefle	76
Hammarby	75
Halmstad	63
Falkenbergs	61
Ostersunds	60
Atvidabergs	60
Sirius	46
Mjallby	45
Brommapojkarna	45
AFC Eskilstuna	31
Jonkopings	31
Syrianska	30
Trelleborgs	16
GAIS	15
Dalkurd	15
Osters	15
Ljungskile	1
Brage	1

Home and Away

Figure 3.1. The number of Home teams and matches played by each team(Left) and the number of Away teams and matches played by each team(Right).

At the end, it was decided to only work with the statistics of the matches as input for the model. These variables can be appreciated below. To predict matches that have not occurred, it is easier to search for the statistics of each of the teams extract them and then compute the mean of these statistics and with that information make the prediction. This is done because it is impossible to know the statistics of matches that have not happened, but the input from the models can be obtained based on the mean of past values that have been registered and in this way make the predictions.

Variables to train and test the decision tree and support vector classifier.

PH	PD	PA	MaxH	MaxD	MaxA	AvgH	AvgD	AvgA

Since the target variable are the Results (Res) of the match, and this result is based on the number of goals obtained by either the Home or Away team, it was decided to remove the columns of HG and AG since they provided the same information than the Res column. After removing the mentioned columns, the database ended up with a total of 1934 rows and 10 columns. The Res column is an unbalanced target variable as appreciated in the percentage listed below where the Home team has a greater number of wins compared to the Away team and the number Draws.

- Home = 43.95%
- Away = 30.29%
- Draw = 25.74%

It is possible to notice that the Home team wins more frequently than the Away team and in the fewer cases, a Draw occurs.

Decision Tree

Considering that the target variable is unbalanced, it is necessary to split the data into two sets the training set and the testing set. The problem states that 80% percent of the data must be used for training the model and the last 20% percent for testing the model. For the training of the decision tree, the Gini importance was used to make the ramifications of the tree and a max depth of 4 was established to make the decision tree as simple as possible and increase its interpretability. With these characteristics, the decision tree was trained and tested to finally compute the confusion matrix as shown in Figure 4.

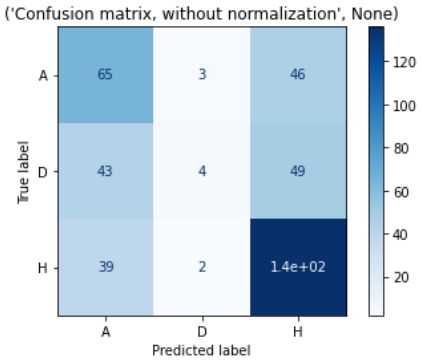


Figure 4. Confusion Matrix of the decision tree classifier.

From the confusion matrix presented in Figure 4, the classification report shown below was obtained. Since the database is unbalanced, it is better to analyze the performance of the model by its recall, precision, and F1-score value instead of only looking at the accuracy of the model.

```
Shape of input and output variables data X: (1934, 9) y: (1934, 1)
(1547, 9) (387, 9)
Accuracy with traning data set for Decision Tree: 0.5553
Accuracy with test data set for Desicion Tree: 0.5297
Confusion Matrix
[[ 65   3  46]
 [ 43   4  49]
 [ 39   2 136]]
              precision    recall  f1-score   support

           A       0.44      0.57      0.50       114
           D       0.44      0.04      0.08        96
           H       0.59      0.77      0.67       177

    accuracy                           0.53       387
   macro avg       0.49      0.46      0.41       387
weighted avg       0.51      0.53      0.47       387
```

Figure 5. Classification report for the decision tree.

Here is a brief description of the metrics that are reported from the confusion matrix based on the testing set.

- Accuracy: What percentage of predictions was correct?
- Recall: What percentage of positive cases were caught?
- Precision: What percentage of positive predictions were correct?
- F1-score: It is based on precision and accuracy, it ranges from 0 to 1, where 0 represents a bad accuracy and 1 represents a perfect accuracy.

By looking at Figure 5, it is possible to notice that the accuracy of the model is 0.5297, on the other hand, the F1 score value for the Away (A) class is lower compared to the value of the Home (H) class which indicates that the model predicts with higher confidence when the Home team wins compared to the Away team. Finally, the Draw class (D) has the lower F1 score value of all the three classes which indicates less performance when predicting when a Draw will occur. This low value is due to the low recall presented by the model for this class as is shown in Figure 5, in other words, this implies that the proportion of actual Draws that were identified correctly is very low.

With the Home (H) and Away (A) class, the recall values are higher in comparison with the precision of both classes. Moreover, it is possible to notice that both the precision and the recall are higher for the Home class in comparison with the Away class, this is expected due to the unbalanced proportion of classes in the database and this also contributed to the higher F1 score value that was obtained for the Home class. In general terms, there is a greater amount of false positives (FP) compare to the true positives (TP) which produce the low precision values observed for the Away and Home classes. On the other hand, there are fewer false negatives (FN) compare to the true positives (TP) in the testing process which generates higher values in the recall for the Home and Away classes.

In Figure 6 it is possible to notice the final model of the decision tree.

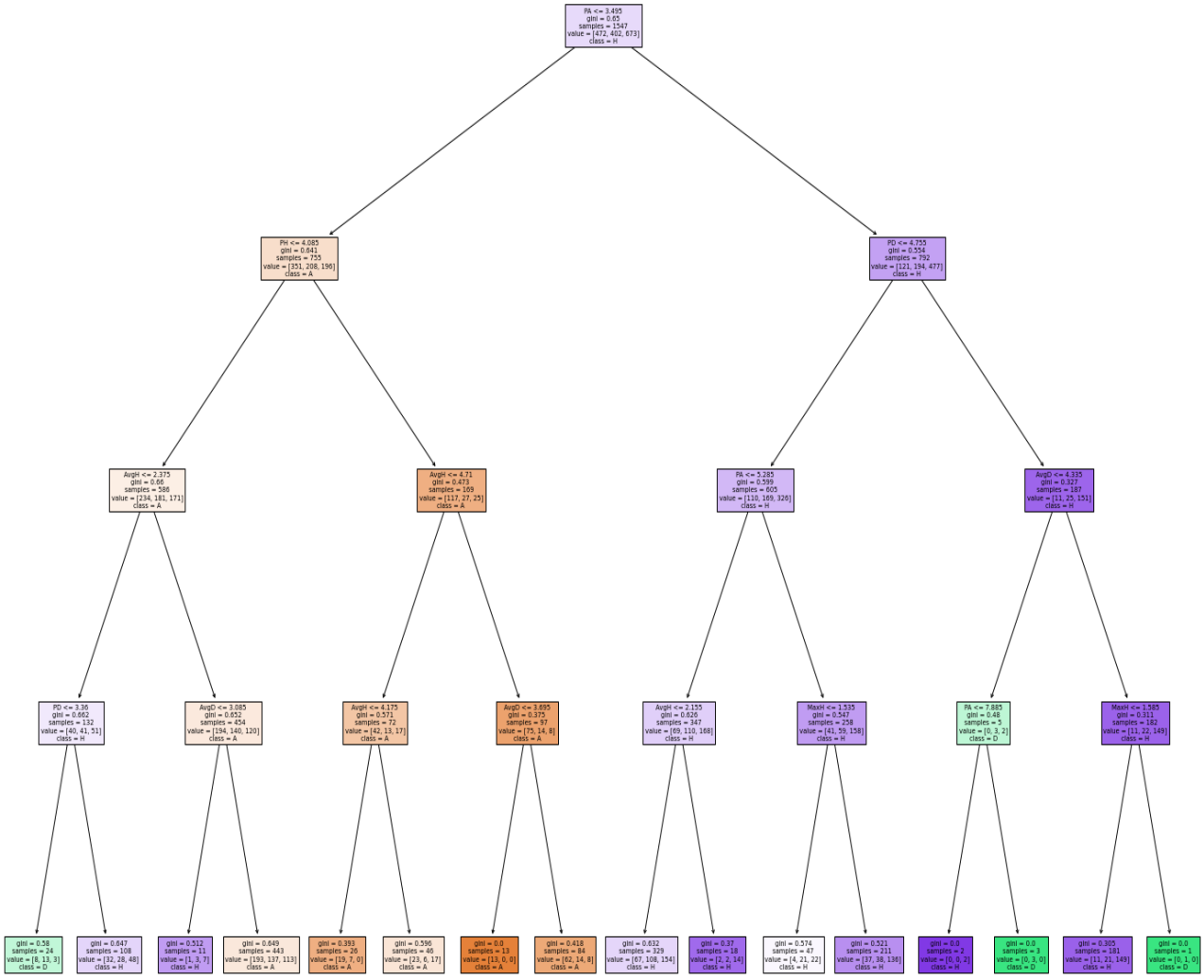


Figure 6. Decision tree model.

Support Vector Classifier

In the case of the support vector classifier, it is important to standardize the data since it computes margins for its training, the z- score normalization was used to standardize with the help of the Standard Scale Function of the sklearn package. In this case, a kernel was not used to train the model and the parameter C was set equal to 1. With these parameters, the support vector machine was trained, and the confusion matrix was extracted from the testing set leading to the result presented in Figure 7.

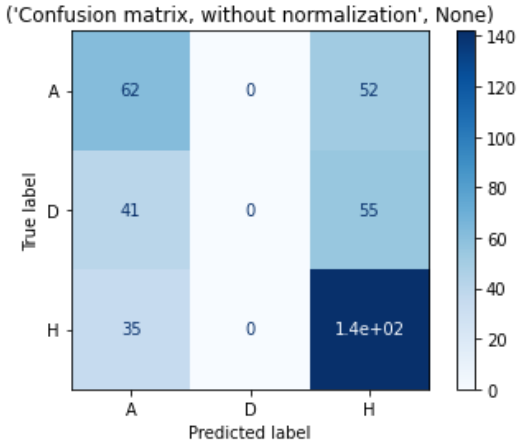


Figure 7. Confusion Matrix for the support vector classifier.

With the computation of the confusion matrix, it is possible to extract performance metrics such as the accuracy, recall, precision, and F1-score value. This leads to the next classification report shown in Figure 8.

```
Accuracy of SVM classifier on training set: 0.5294
Accuracy of SVM classifier on test set: 0.5271
Confusion Matrix
[[ 62   0  52]
 [ 41   0  55]
 [ 35   0 142]]
              precision    recall  f1-score   support

           A       0.45      0.54      0.49       114
           D       0.00      0.00      0.00        96
           H       0.57      0.80      0.67       177

    accuracy                           0.53       387
   macro avg       0.34      0.45      0.39       387
weighted avg       0.39      0.53      0.45       387
```

Figure 8. Classification report for the support vector machine classifier.

By looking at Figure 8, it is possible to notice the accuracy of the model for the testing set of 0.5271. On the other hand, the F1-score value for the Home class (H) was higher in comparison with the Away class (A) and the Draw class (D) this shows that the model can predict better the victory of the Home team similarly to the Decision Tree. One important aspect to notice is that the recall and precision for the Draw class are equal to zero what leads to an F1-score value of zero, this shows that the model using the support vector machine cannot predict when a Draw would occur. This was also expected to the unbalanced

proportion of the target variable. For this model, the recall values for the Away and Home class are higher in comparison with the precision similar to the behavior observed for the decision tree. In general terms, this implies that there are more false positives (FP) compare to the true positives (TP) for the Away and home classes at the time of testing which results in lower precision values. On the contrary, in the case of false negatives (FN), their quantity is lower compare to the true positives (TP), which generates higher results for the recall for the Away and Home class respectively.

As an extra step, the Rank Probability Score (RPS) of each of the models was calculated based on the testing set used in each of the models. The results are displayed below:

- Decision Tree RPS: 0.210249
- Support Vector Classifier RPS: 0.204828

It is important to remember that the Rank Probability is a measure of the quality of the prediction the lower the RPS the better. For this case, it is possible to observe that the RPS is lower for the Support Vector Classifier than for the Decision Tree. Nevertheless, the difference is not so high been of 0.005420 between each RPS value.

Comparison between the Decision Tree and Support Vector Classifier.

By comparing the two models both achieved similar results in terms of accuracy 0.5297 for the decision tree and 0.5271 for the support vector machine. Since the classes of the database are unbalanced, it is better to look at the F1-score values of each of the models to determine which one has a better performance. The F1-score value of the Away class (A) for both models are very similar 0.50 for the decision tree and 0.49 for the support vector machine, with the Home class (H) the F1-score value is 0.67 for both models. The highest difference between the models is that the decision tree could predict some Draws (D) based on the precision value obtained for that particular class while the support vector machine cannot do that. Also, the complexity for the decision tree is lower compared to the support vector machine which facilitates its interpretation as shown in Figure 6. The decision tree does not require that the data be standardized before training and testing while the support vector machine requires it. Taking these aspects into account it is possible to conclude that the Decision tree provides a better model that has a slightly better performance in terms of accuracy and F1-score values, but also it is less complex than the support vector machine and requires fewer preprocessing steps. This decision is also supported due to the low RPS value obtained by the Decision Tree that did not differ too much from the value obtained with the support vector classifier.

Question 4

Note: To look at the procedure perform to make this analysis please refer to the attached Jupiter Notebook in the section for Question 4.

For the following dataset, prepare the data and perform clustering analysis. wine.csv

1. Perform a DBSCAN clustering analysis

2. Perform a k-means and agglomerative cluster analysis. Find the best k using the elbow method. (10 pts)

3. Select the best model using the silhouette coefficient. (10 pts)

4. Give your conclusions of the results obtained (10 pts)

Perform a DBSCAN clustering analysis

The data has a total of 178 rows and 13 columns. Additionally, the data must be standardized since each of the columns has different ranges. Here the Standard Scaler function of Python was used to apply a z-score normalization on the database.

First DBSCAN was used for the clustering task. The problem of DBSCAN is how to determine the optimal value for epsilon and the min samples, moreover, the algorithm is also sensitive to the setting of both parameters. Moreover, it is important to consider that the DBSCAN algorithm does not work well with high dimensional data which also complicates the proposal of an epsilon value. Additionally, if the database has clusters with different densities the epsilon value that is proposed could produce an error in the clustering.

In this case, different values for min samples and epsilon were tried and the silhouette score was also computed to test the quality of the clustering. The final approach was by setting the min samples to 3 and testing different values of epsilon between the ranges of 2.3 to 2.5. By flowing this procedure, the epsilon value was set to 2.405, and the min samples parameter to 3, the silhouette score, and the number of clusters are as follows.

- **Silhouette score: 0.21701633523077773**
- **Number of clusters: 3**

By making this clustering of the data points, DBSCAN generated three clusters as shown in Figure 1. In this case, only two variables of the database were selected to visualize the results of the clustering technique since it is not possible to visualize the whole dimensionality of the database.

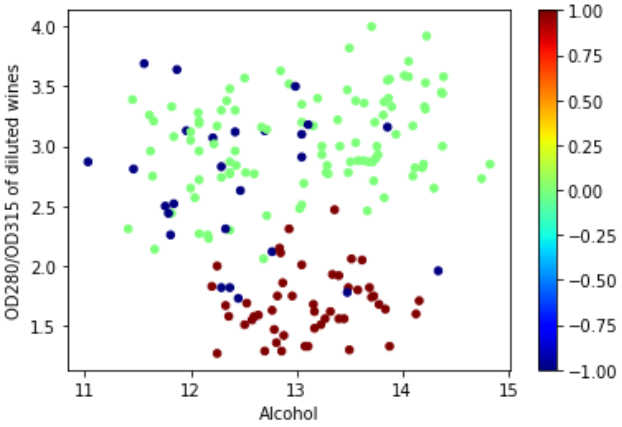


Figure 1. Clustering results by using DBSCAN and plotting the Alcohol column (x-axis) with the OD280/OD315 of diluted wines (y-axis).

As can be seen, clusters marked with brown dots are better separated than those with green or blue dots. The blue points and the green point are not well separated, which causes a greater penalty to the silhouette's score, which consequently results in a lower value. It is difficult to conclude whether the parameters selected for the epsilon and min samples are optimal. In the next section, we will test other methods for cluster analysis to determine if the clustering can be improved.

Perform a k-means and agglomerative cluster analysis. Find the best k using the elbow method.

Select the best model using the silhouette coefficient.

To perform the clustering analysis, it is important to standardize the data since K-Means, and Agglomerative clustering is based on the computation of distances and the variables of the database have different ranges. For this purpose, z-score normalization was performed in the variables. To select the best value of K for the K-Means algorithm and the Agglomerative clustering the elbow method was used for this purpose. The elbow method is one of the most common techniques to determine the optimal value for the clusters, the idea is to execute the K-means method for a range of different values of K, and for each value of K, the sum of the square distance from each point to its assigned center is computed, these distances are also known as distortions. By plotting the distortions it is possible to observe an arm in the curvature, therefore the elbow of that arm will be the optimal value for K. The results of applying the elbow method are shown in Figure 2. The elbow can be appreciated for K = 3 that is indicated with a red dot in Figure 2.

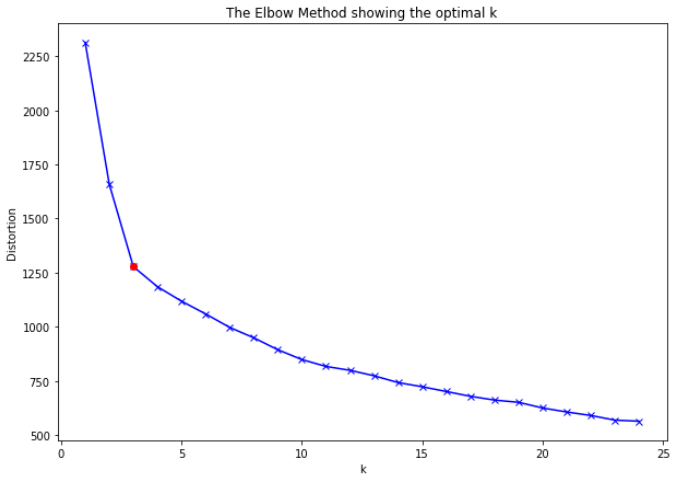


Figure 2. Elbow method for the optimal K.

With this value, the K-means and agglomerative clustering are computed, and later the silhouette score is used as a metric to determine the performance of using that specific value of K = 3. Below are listed the silhouette score for different values for K and are compared to the proposed optimal value according to the elbow method. It is possible to notice that the highest score is obtained with the value of K =3.

- **Silhouette score K = 3: 0.2848589191898987**
- Silhouette score K = 2: 0.2614352045273167
- Silhouette score K = 4: 0.24519129323772165

The clustering in three groups can be appreciated in the Figure below. Comparing Figure 3 with Figure 1, it is possible to observe that the K-means method provides a better separation in three clusters of the data compared to the DBSCAN. This is also reflected in the higher silhouette score.

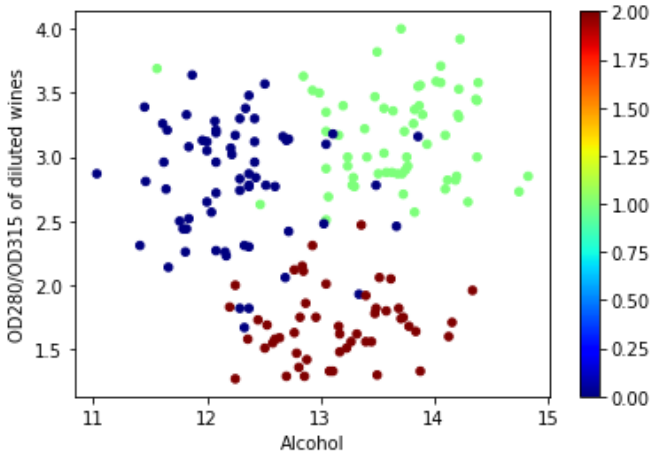


Figure 3. Clustering results for K =3 by using the K-Means method and plotting the Alcohol column (x-axis) with the OD280/OD315 of diluted wines (y-axis).

A similar result was obtained with the agglomerative clustering which shown a similar silhouette score with 3 clusters by comparing it with the K-Means method. The results of the agglomerative clustering analysis

can be observed by looking at Figure 4 where the clustering was performed with K = 3. Comparing Figure 3 and Figure 4, the results are very similar, this was expected since the silhouette was very similar for the K-means method and the agglomerative clustering.

- **Silhouette score K = 3 for Agglomerative clustering: 0.2774439826952265**
- Silhouette score K = 2 Agglomerative clustering: 0.2670131771272231
- Silhouette score K = 4 Agglomerative clustering: 0.225836659334758

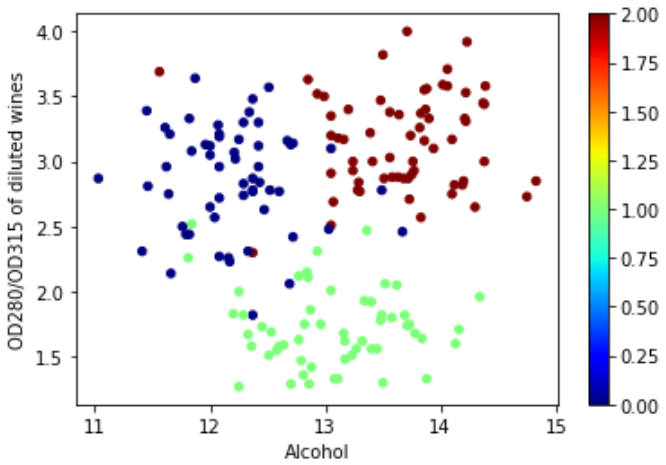


Figure 4. Clustering results for K =3 by using the agglomerative clustering method and plotting the Alcohol column (x-axis) with the OD280/OD315 of diluted wines (y-axis).

In this case, the K-means method provides slightly better results based on the silhouette score obtained compared to the agglomerative clustering by setting the number of clusters equal to 3 obtained from the elbow method.

Give your conclusions of the results obtained

By comparing the results of the different methods that were used for the clustering of the wine data, it is possible to observe that the best clustering was the K-Means method by using three clusters. Nonetheless, the Agglomerative clustering provides slightly different results in comparison with the K-means method. Additionally, both methods also had the advantage of setting only one parameter that is the number of clusters to be created which simplifies the testing process. Moreover, the elbow method provides a good approximation of the optimal value of K that must be set for both algorithms. On the other hand, in the DBSCAN method, it becomes more complicated to determine the setting of the epsilon value and the min parameters especially if it is considered the high dimensionality of the data and the normalization process that was performed. Therefore, it is possible to conclude that for this database the K-means method provides better results which less testing than the DBSCAN and slightly better results in terms of the silhouette score in comparison with the agglomerative clustering. Finally, another step that can be added to this procedure is to validate if the numbers of clusters obtained are the ones expected for this particular database. To make this, it is necessary to have more insights into the database and a description of it.