Data Mining

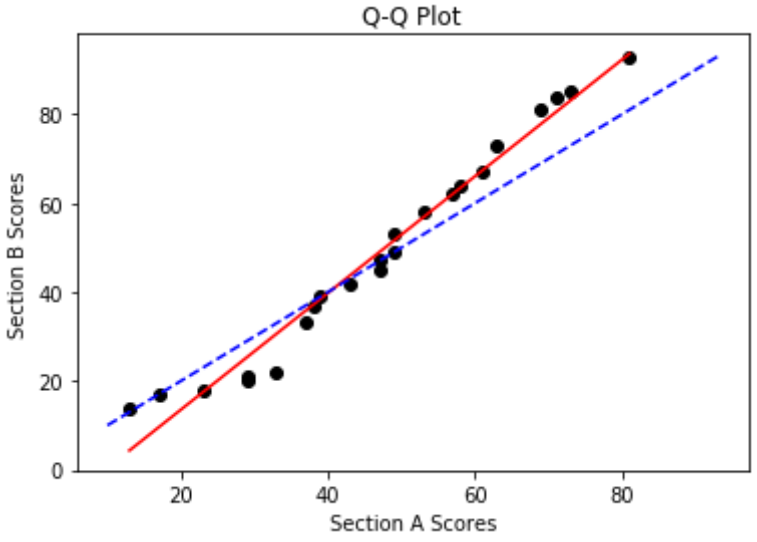
Home Work #2

Monica Dommaraju

014545336

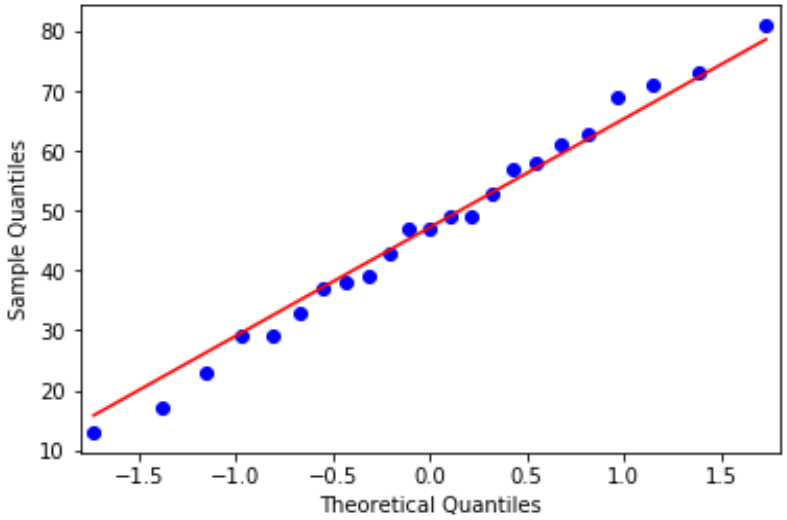
1. a)

- Based on the scores obtained by the scores of the two sections, I have generated a q-q plot using a scatter plot.
- Plotted the scores of Section A on x-axis and Section B on y-axis and have plotted a trend line that pass through all the points. I have used poly1d function to fit the linear line through all the points. I have also plotted the 45 degree reference line to understand
- As per the plot, we can say that Section-B scores (~13/23) are above the trend line and so, we can infer that Section-B performed better compared to Section-A
- Please Refer the python code in ipynb file

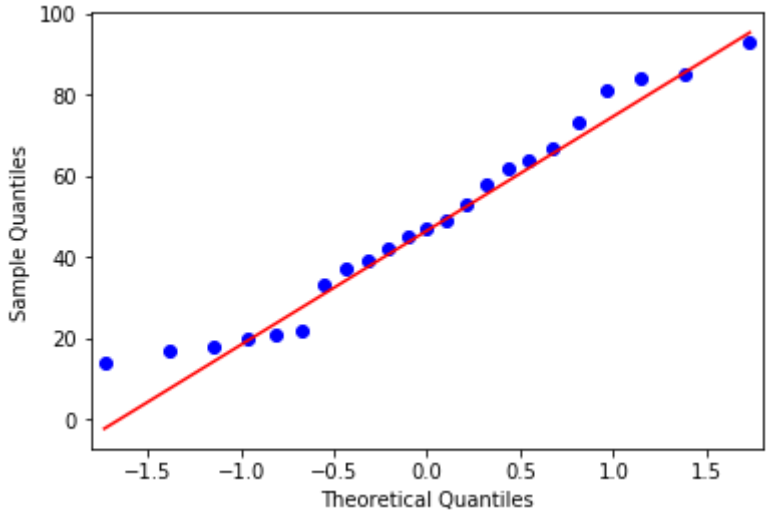


1 .b)

- In the first graph, I have considered scores obtained by Section-A and plotted them against the theoretical random samples from a normal distribution curve.



- For section-B, I have followed the same as above with samples of section B against theoretical random samples



- When we observe the graphs, we can easily conclude that Section-A is closer to normal distribution curve and the samples are more linear and aligned to theoretical quantiles.

2. a)

Equation for accuracy in terms of Specificity can be derived for these 3 assumptions.

1. **Number of positive and negative samples are equal**
   Let's consider, Total positive samples = P
   Total negative samples = N

   Positive samples are distributed between TP and FN
   FN = P - TP
   Similarly, negative samples are distributed between FP and TN
   FP = N - TN

   Sensitivity = TP/(TP + FN)
   = TP/(TP + P - TP)
   = TP/P
   Specificity = TN/(TN + FP)
   = TN/(TN + N - TN)
   = TN/N Accuracy
   = (TP + TN)/(P + N)

As the assumption is that positive and negative samples are equal

P = N

Accuracy = (TP + TN)/2P

= 1/2(TP/P + TN/N)

= 1/2(Sensitivity + Specificity)

Therefore,

Accuracy = 1/2(Sensitivity + Specificity)

2. **All samples are positive**

   As there are no negative samples in this case,

   (N = 0) => (TN + FP) = 0

   => (TN = 0 and FP = 0)

   Also P = TP + FN Accuracy = TP/(TP + FN)

   => TP/P Sensitivity = TP/(TP+FN) = TP/P

   Specificity = 0

   Therefore, **Accuracy = Sensitivity**

3. **All samples are negative**

   As there are no positive samples in this case,

   (P = 0) => (TP + FN) = 0

   => (TP = 0 and FN = 0)

   Also N = TN + FP

   Accuracy = TN/N

   Specificity = TN/(FP+TN) = TN/N

   Sensitivity = 0

   Therefore, **Accuracy = Specificity**

2. b)

I have implemented Naive Bayes and Logistic Regression algorithms on the Indian Liver Patient Records dataset.

Performance metrics for Logistic Regression

```
Accuracy Score:
0.7314285714285714

Confusion Matrix:
[[118   7]
 [ 40  10]]

Classification Report:
              precision    recall  f1-score   support
```

```
              1       0.75      0.94      0.83       125
              2       0.59      0.20      0.30        50

   micro avg       0.73      0.73      0.73       175
   macro avg       0.67      0.57      0.57       175
weighted avg       0.70      0.73      0.68       175
```

Performance metrics for Naive Bayes Classifier:

```
Accuracy Score:
0.5314285714285715

Confusion Matrix:
[[45 80]
 [ 2 48]]

Classification Report:
              precision    recall  f1-score   support

           1       0.96      0.36      0.52       125
           2       0.38      0.96      0.54        50

   micro avg       0.53      0.53      0.53       175
   macro avg       0.67      0.66      0.53       175
weighted avg       0.79      0.53      0.53       175
```

By comparing the above two algorithms, we can see that **Accuracy**, **Recall** and **F1-Score**s are better for **logistic regression** than Gaussian Naive Bayes. However precision is better for Naive Bayes.

Overall, we can say that **Linear Regression performed better than Naive Bayes.**
Please refer to ipynb file for implementations

3a)

Logistic Regression: Logistic Regression comes under supervised learning. It uses maximum likelihood estimate for training a logistic regression which is used to measure the relationship between dependent data variable and one or more independent variables by calculating probabilities using sigmoid function. It is mainly used for classification.

Applications:

- A medical use case where the probability of a person having cancer or not can be found out.

- In weather forecasting to predict if it's going to rain or not.

Advantages:

- Resistant to overfitting
- Very efficient and widely used
- Low Variance
- It can handle highly correlated overlapping features

Disadvantages:

- It can only handle linear data, and it cannot handle non-linear datasets.
- Variables should be independent of each other.
- We will have low accuracy when dataset is small

K-NN: K-NN is a supervised machine learning algorithm. It is lazy learning algorithm which means it builds the model when we give the test data. It classifies the data points based on its nearest neighbors. K-NN can be used for both classification and regression.

Applications:

- Used in Recommendation Systems
- In E-Commerce, to give customer suggestions
- K-NN can also be used to forecast the stock market, on the basis of company performance we can predict the price of the stock.

Advantages:

- K-NN uses a non parametric approach
- Works well when decision boundary is non-linear
- If there are any changes in the input it can quickly respond. This feature helps us to use it in real time.

Disadvantages:

- The main disadvantage is that it takes most of the Computation time during testing rather than training as it is a lazy learning algorithm.
- We cannot know which points are important while using K-NN
- Choosing K is the most difficult task.

SVM: Support Vector Machine is a supervised learning algorithm which can be used for classification problems. Support vectors are the data points which lie on the hyperplane. The main task of SVM is to maximize the margin of the hyperplane.

Applications:

- SVM can be used for the classification of images as it provides better accuracy
- Speech Recognition
- Text Analysis

Advantages:

- Non-linear decision boundaries can be modeled.
- SVM doesn't entirely depend on entire data.
- Missing values is not a problem in SVM as it mostly classifies the data based on the support vectors.

- High Accuracy
- SVM is effective when the number of dimensions is greater than the number of samples given.

Disadvantages:

- Not efficient when we take larger dataset because it takes more time for training the data.
- If the data has more noise ,SVM doesn't perform well as target classes will be overlapping.

Decision Tree: Decision tree is a supervised learning method used for classification and prediction. It is a tree like structure where each internal node denotes a test on the attribute and each branch represents an outcome of the test. Choosing the root node is the main task as it should give the maximum information for generating a tree.

Applications:

- In Customer Relationship Management(CRM) to understand customers
- In Detecting faults.

Advantages:

- Easy to understand and explain as it is a tree like structure.
- Can accept both numerical and categorical values.
- Quick to train.
- It can easily process the data with high dimension.
- There are no requirements of the domain knowledge in the construction of a decision tree.

Disadvantages:

- Decision tree uses a greedy algorithm , Sometimes by using this optimal tree is not formed which is NP-hard.
- When there is noise in the dataset, Overfitting occurs.
- High variance.
- Small variations in the data may result in completely different trees.

Naive-Bayes: Naive Bayes is a classification technique which uses Bayes theorem to classify the objects. Bayes theorem is stated as the probability of event B given A is equal to the probability of the event A given B multiplied by the probability of A upon Probability of B.

$$P(A|B) = P(B|A)*P(A)/P(B)$$

Applications:

- Spam Detection
- Sentimental Analysis
- Categorizing News

Advantages:

- Require less training data
- In most of the cases we get good results.
- High computational efficiency

Disadvantages:

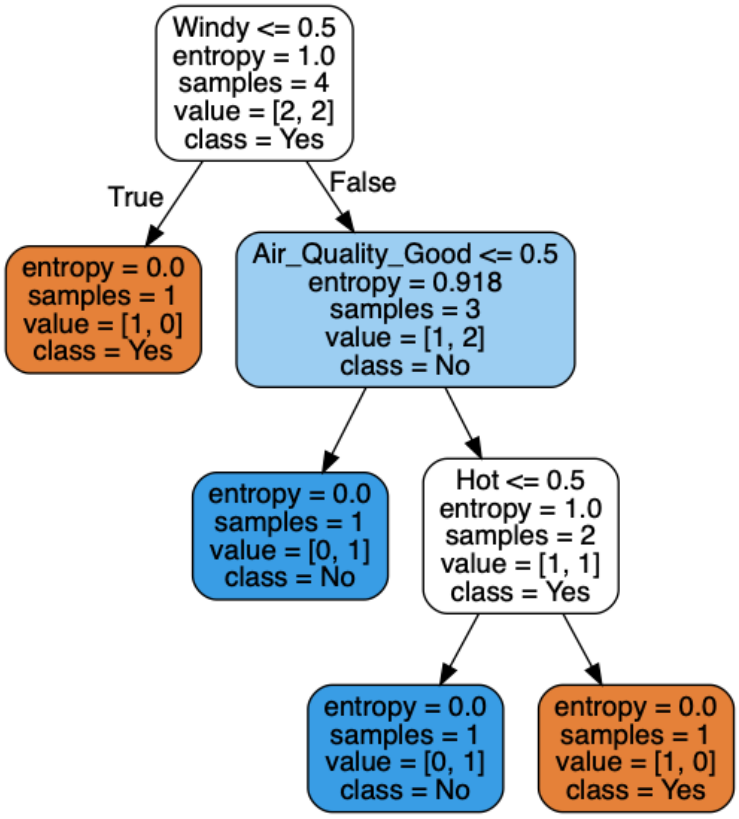
- The main disadvantage of naive bayes is its naive assumption that variables are independent.
- It cannot easily understand the relationship between features.

3.b) Please refer to ipynb file for implementations

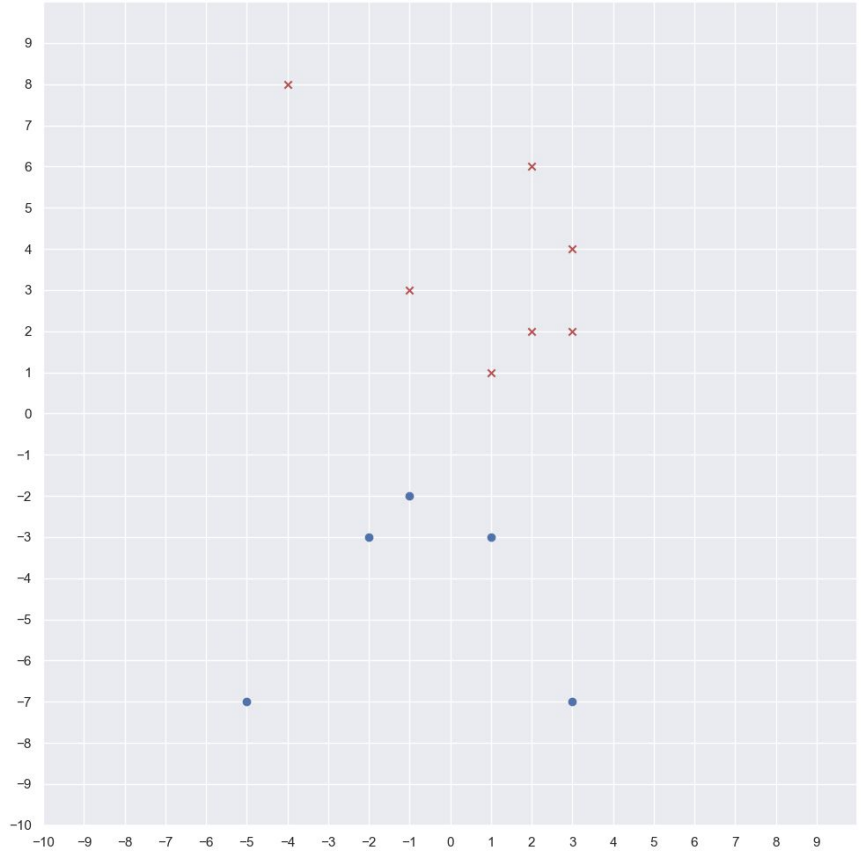
4.

Handwritten Manual decision tree implementation is also included in the submission.

Please refer to ipynb file for decision tree python implementation



5.a) Handwritten derivation is included as part of my submission.



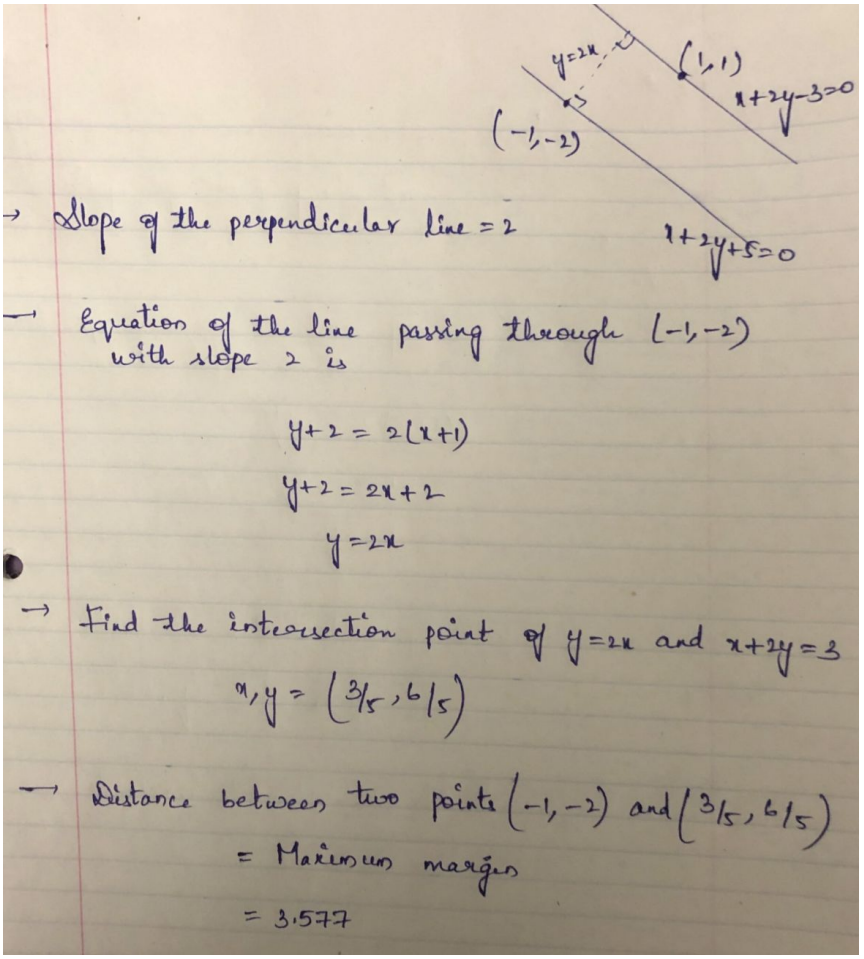
C1 -> x

C2 -> o

Observing from the scatter plot above, we can easily identify that support vectors.

1. number of support vectors from each class
   C1 -> 1
   C2 -> 2
2. support vectors
   c1 -> [1, 1]
   c2 -> [-1, -2] and [1, -3]
3. Equation of the hyperplanes (straight lines in this case) defining the sides of the margin
   Slope = -1/2
   Equation of the straight line passing through **[-1, -2] and [1, -3]** =>
   C2 => **x + 2y + 5 = 0**
   Equation of the straight line passing through [1, 1] and parallel to C2 is
   C1 => x + 2y - 3 = 0

4. Length of the maximum margin



Therefore , the length of the maximum margin = **3.577**

5. Equation of the maximum margin linear classifier.
   $w^T x+b=0$

   Where
   w = [0.24, 0.5]
   b = [0.25]

5.b) Results from the python implementation from ipynb file.

```
Support Vectors:
[[0.45951063 0.35475144 0.43749568 ... 0.30024055 0.61344372 0.13898728]
 [0.18784609 0.3936422  0.19425057 ... 0.48934708 0.2757737  0.26905418]
 [0.39703725 0.44132567 0.38981411 ... 0.50721649 0.19534792 0.08684245]
 ...
 [0.34260968 0.61345959 0.33694976 ... 0.28243986 0.06406466 0.15033451]
 [0.30806001 0.42576936 0.29797526 ... 0.44054983 0.25744136 0.09268005]
 [0.30664016 0.30571525 0.30163776 ... 0.49037801 0.3849793  0.20733307]]

Support Vector Indices
[ 21  32  33  36  37  38  52  53  58  71  74  87 103 108 123 131 133 145
```

```
 168 185 189 202 279 309 313 321 326 330 332 367 398 399 404 411 425 429
 439 441 444   1  13  20  30  48  49  56  75  88  96 110 115 116 156 157
 196 197 201 204 248 249 254 258 282 297 305 322 328 331 333 344 346 383
 393 402 416 428 438 450]

Number of support vectors per class
[39 39]
```

Please refer to ipynb file for implementation