HAMMAD KHAN MUSAKHEL

21801175

CS464

HW1

SECTION 1

1.1) In order to receive the first (A, heads), the previous seven events that had occurred must be either a A(tails), B(heads), or B(tails); the combined probability of those events must be: (1-P3*P1)^7. Thus, P(A, heads) in the eight trial: (1-P3*P1)^7 * P3 * P1.

1.2) the expected value is calculated by multiplying each of the possible outcomes by the likelihood each outcome will occur and then summing all of those values. Expected value of the score is P(X) * n, after the 10^{th} trial so n = 10. Thus:
P(score) = P(A) * P(H|A) + P(B)*P(H|B) = P3*P1 + (1-P3)*P2
Estimated score = (P3*P1 + (1-P3)*P2) * n [where n = 10]

1.3.a) the given probabilities for Oliver's guessing performances are measured by estimating how accurate he is. Oliver guesses that I will obtain heads in the next trial with a probability of 0.95; thus, it can be validated by assuming that if I had a head or tails before the trial where Oliver guessed. He uses his intuition to guess the occurrence of the event. Witnessing many throws landing on for heads must have propelled him to be more confident. His guesses recorded are then uses for further guesses.

1.3.b) P(Oliver predict heads) = 0.99
0.99^8 = 0.9228

1.3.c) P(Oliver predicting tails | Heads) = P(Oliver predicting tails)*P(Heads | Oliver predicting tails) / P(Heads) = 0.99 * 0.01 / 0.4 = 0.025

2.1) Euclidean Distance
One is most likely to use Euclidean distance when calculating the distance between two rows of data that have numerical values, such a floating point or integer values. We want to know the shortest distance between the two points, assumed to be on a plot. Manhattan Distance calculates the distance between two real-valued vectors; it is perhaps more useful to vectors that describe objects on a uniform grid, like a chessboard or city blocks. Other options also do not calculate the shortest distance.

2.2) Using all features in dataset result in increasing the time complexity of the algorithm. Not only this, there might be some features that might not affect the accuracy of the model; the removal of that feature from the model might result in an improved or same accuracy. Thus, making the model more efficient as less comparisons are needed in that case. In essence, the complexity of the model is reduced if less features are used as training and test features.

2.3 & 2.4)

```
Below is the iterative deletion from column 1 to 8:

removed column:  1  accuracy:  71.24183006535948  time taken:
0.7791996002197266
```

removed column: 2 accuracy: 60.130718954248366 time taken: 0.7740564346313477

removed column: 3 accuracy: 68.62745098039215 time taken: 0.7696323394775391

removed column: 4 accuracy: 69.93464052287581 time taken: 0.7670652866363525

removed column: 5 accuracy: 75.16339869281046 time taken: 0.7528347969055176

removed column: 6 accuracy: 71.24183006535948 time taken: 0.7597475051879883

removed column: 7 accuracy: 71.89542483660131 time taken: 0.7674460411071777

removed column: 8 accuracy: 74.50980392156863 time taken: 0.7576999664306641

Remove column 5 as its removal produces the highest accuracy 75.16339869281046

round two:

removed column: 1 accuracy: 76.47058823529412 time taken: 0.682835578918457

removed column: 2 accuracy: 60.130718954248366 time taken: 0.6843006610870361

removed column: 3 accuracy: 67.97385620915033 time taken: 0.6555564403533936

removed column: 4 accuracy: 75.81699346405229 time taken: 0.67620849609375

removed column: 5 accuracy: 71.89542483660131 time taken: 0.6711282730102539

removed column: 6 accuracy: 75.16339869281046 time taken: 0.6576104164123535

removed column: 7 accuracy: 74.50980392156863 time taken: 0.6872913837432861

Remove column 1 as its removal produces highest accuracy: 76.470588235294

round three:

```
removed column:  1  accuracy:  56.209150326797385 time taken:
0.5924801826477051

removed column:  2  accuracy:  68.62745098039215 time taken:
0.5794186592102051

removed column:  3  accuracy:  75.81699346405229 time taken:
0.5781137943267822

removed column:  4  accuracy:  73.8562091503268 time taken:
0.6008024215698242

removed column:  5  accuracy:  76.47058823529412 time taken:
0.5979933738708496

removed column:  6  accuracy:  75.81699346405229 time taken:
0.5823638439178467
```

Removed column 5 as its removal produced the highest accuracy 76.47058823529412

```
round four:

removed column:  1  accuracy:  56.209150326797385 time taken:
0.5049312114715576

removed column:  2  accuracy:  68.62745098039215 time taken:
0.49878621101379395

removed column:  3  accuracy:  75.81699346405229 time taken:
0.4997410774230957

removed column:  4  accuracy:  73.8562091503268 time taken:
0.5021054744720459

removed column:  5  accuracy:  75.81699346405229 time taken:
0.5205841064453125


No need to go to round five as we obtain lower accuracy for D-1 than D,
compared to previous round. Stop
```

In essence, what takes place is that as we remove the columns to compare the results of accuracies, we obtain better accuracies. If better accuracy is obtained, that column is gotten rid of as it is redundant or even causing noise and hampering better accuracy. Thus, we obtain better accuracy and also more efficient algorithm as time complexity is reduced. When we have eight columns, the time taken for the prediction is 0.**9157888889312744,** as one can see from the lines of results above. As we remove a column and test the training data with seven columns, the time is reduced to **0.7576999664306641 - 0.7696323394775391**. Ultimately, we remove three columns overall in order to obtain the best format for the Backward Elimination

method, and the time reaches a hall mark of **0.5823638439178467**. An overall 0.32 seconds, almost **35.5%** increase in efficiency is obtained. The above shows the step-by-step deletion of the columns.
We delete column number **5** in step 1, column number **1** in step 2, and column number **7** in step 3 - from the original dataset. The columns **2, 3, 4, 6, 8** from the original dataset remain as they are needed to predict the kNN algorithm with higher accuracy.

Column Removed	Accuracy (%)
5	**75.16339869281046**
1	**76.470588235294**
7	**76.47058823529412**

Table number 1.

No. of columns removed	Average validation time(s)
0	0.9157888889312744
1	0.7597475051879883
2	0.67620849609375
3	0.5823638439178467

Table number 2.

3.1)
Accuracy: 94.370522%
Confusion Matrix: FN = 41, TN = 824, FP = 14, TP = 98

3.2) To estimate this model, we need to estimate for each word in the vocabulary., so:

- $\theta j \mid y=spam$, the probability that a particular word in a spam sms will be the jth word of the vocabulary, $P(Xj|Y=1)$
- $\theta j \mid y=notspam$, the probability that a particular word in a nonspam sms will be the jth word of the vocabulary, $P(Xj|Y=0)$

The above are needed to calculate the estimation so that we can know which word appears more in which class, and tis way we are able to estimate the class of the new sms; we compare each word's occurrences in spam and non-spam and then estimate.
- Moreover, we need to know the $\pi_{y=normal/spam}$, we need to know the probability of nonspam and spam sms. Thus, in total we need 2 * V + 1 parameters: 6917.

3.3.a and 3.3.b)

No. of features employed in training and testing based on selection from nonspam sms	Accuracy (%)	Training time (s)
3458	94.98	9.35
600	73.08	2.03
500	71.35	1.69

400	68.27	1.33
300	63.43	1.01
200	55.78	0.63
100	41.96	0.33

Table 3.

Yes, there is a clear relationship between the three columns of table 3. As we can see, by reducing the number of features, we reduce the time complexity of the algorithm as fewer columns are referenced when determining the placement of a test row/sms, that is, fewer yet relevant words are determined.
In my program, I have decided to use the nonspam set of sms in order to deduce the relevant features. I have traversed through the array I had made, nonspamArrayB, whose second row has the probability for each word j/column. It has 2 rows and 3458 columns. I firstly chose those features that have highest probabilities for the nonspam sms. Since, the nonspam sms are bigger in number in the training data, training my model on those features did affect the accuracy. However, if we reduce the number of features, the accuracy falls to almost 41.96% with 100 features. However, we move from 3458 features to just 100 features, saving almost 96% of time as the time taken drops from 9.35 seconds to 0.30 seconds. It is not ironic that accuracy drops with less features as greater the number of features, greater the accuracy. The data is vast, so are the features. I have selected the features that have the highest probabilities compared to nonspam sms. I have selected the columns from nonspamarrayB that have the highest probabilities.
The accuracy was very low when feature selection was employed by using spam sms data. Thus, nonspam sms model was better than spam sms model for this part.

3.4) The Bernoulli classifier with all the features has almost the same accuracy as the multinomial classifier. Bernoulli models the presence/absence of a feature. Multinomial models the number of counts of a feature. In the Bernoulli classifier I obtained 97 True Positives, whereas in Multinomial I had 98 TP; likewise, in Bernoulli, I obtained 831 TN, whereas in Multinomial I obtained 824 TN. Multinomial had more FP and lesser FN.