CS464 Introduction to Machine Learning
Fall 2021
HOMEWORK #1
REPORT

Section 1
Lara Fenercioğlu
21802536

Question 1

1.1 $(1\text{-}P3*P1)^7 * (P3*P1)$

1.2 (P3*P1) * ((1-P3) * P2) * 10

1.3 Calculating the probability of getting head:
where P(head|obtain head) = 0.95
P(not head|obtain head) = 0.05
P(obtain head) = 0.99
P(not head|not obtain head) = 0.99
P(head|not obtain head) = 0.01
P(not obtain head) = 0.01

P(head) = P(head|obtain head) * P(obtain head) + P(head|not obtain head) * P(not obtain head) = 0.9406

1.3.a Oliver has tossed the coin N times and recorded each time when head and tail is observed. He may be tossed 100 times and each time he said head will come, he recorded the times when head is observed and created a ratio out of it. Likewise, he tossed 100 times and each time he said head won't be obtained, he records the times when head is not observed. However since we know that Oliver makes a prediction that you will not obtain head once in 100 trials, my approach won't work correctly. If we count this as an error then we can assume that after 100 trials of tossing a coin and after Oliver predicting each trial as head will be obtained, he obtains 95 heads which corresponds to the probability that is given.

1.3.b (0.9406)^8 = 0.61268

1.3.c

P(not obtain head|head) = P(head|not obtain head)*P(not obtain head) / ((P(head|not obtain head)*P(not obtain head) + P(head|obtain head)*P(obtain head))) = (0.01*0.01)/(0.01*0.01+0.95*0.99) = 0.0001

Question 2

2.1 We are determining a distance metric to find the distance between consecutive points in KNN. I used Euclidean Distance because it is generally used to find the distance between real values like integer or float. In this problem we have input variables that are of similar type. All eight features are numeric, real valued.

2.2 Less features means lower complexity so the model is less prone to overfitting. Also, too many features requires too much time and space. Getting rid of irrelevant features reduces confusion which eventually results in better predictive models. Moreover, a small set of features help to build a better relationship between the features and labels. So, in this problem we are going to use Backward Elimination because it starts with the full set of features and greedily removes the one that most improves performance, or degrades performance slightly.

2.3 code

2.4

First step of the backward elimination (without removing any feature):

Train time (s)	0.0 seconds
Validation time (s)	0.4589567184448242 seconds

Second step of the backward elimination (removing each feature one by one):

Removed feature	Insulin	Age	SkinThickness	BloodPressure	Glucose	Pregnancies	DiabetesPedigreeFunction	BMI
Train time (s)	0.0 seconds	0.0 seconds	0.0 seconds	0.0 seconds	0.0 seconds	0.0 seconds	0.0 seconds	0.0 seconds
Validation time (s)	0.4060072898864746 seconds	0.40903282165527344 seconds	0.39896512031555176 seconds	0.4019598960876465 seconds	0.4100475311279297 seconds	0.39705729484558105 seconds	0.39695119857788086 seconds	0.40695738792419434 seconds

Third step after removing the feature “Insulin” and removing the rest of the features one by one:

Removed feature	Age	SkinThickness	BloodPressure	Glucose	Pregnancies	DiabetesPedigreeFunction	BMI
Validation time (s)	0.0 seconds	0.0 seconds	0.0 seconds	0.0 seconds	0.0 seconds	0.0 seconds	0.0 seconds
Test time (s)	0.38900089263916016 seconds	0.41598033905029297 seconds	0.4210011959075928 seconds	0.40299463272094727 seconds	0.3740386962890625 seconds	0.37300848960876465 seconds	0.471024751663208 seconds

Fourth step after removing the features “Pregnancies” and “Insulin” and removing the rest of the features one by one:

Removed feature	Age	SkinThickness	BloodPressure	Glucose	DiabetesPedigreeFunction	BMI
Train time (s)	0.0 seconds	0.0 seconds	0.0 seconds	0.0 seconds	0.0 seconds	0.0 seconds
Validation time (s)	0.3600013256072998 seconds	0.4049983024597168 seconds	0.41297149658203125 seconds	0.37499499320983887 seconds	0.36803245544433594 seconds	0.367046594619751 seconds

Training takes O(1) time and can be seen on the above tables as 0 seconds which is constant time. Since all computations done during the prediction, test time differs each step but mostly around 0.36 to 0.47 seconds. In theory, prediction complexity is calculated as the multiplication of k, number of points in training data and the dimensionality, O(k * n * d).

Question 3

3.1 Accuracy: 94.3762781186094
Confusion Matrix:
[[99 41]
 [14 824]]

3.2 We need to calculate 2N+2 parameters where N is the number of features in our dataset because we need to estimate 2 priors for messages being spam or ham. Also, we need to estimate likelihoods of each feature, word, which counts as 2*N where multiplication of 2 comes from having 2 classes. Each feature’s likelihood is calculated for both classes.

3.3.a

	first step	second step	third step	fourth step	fifth step	sixth step
Training time	0.0636701583862304 seconds	0.12107038497924805 seconds	0.17743635177612305 seconds	0.2396702766418457 seconds	0.29857563972473145 seconds	0.38512206077575684 seconds
Accuracy	96.6257668 7116564	96.21676891615542	96.0122699	96.21676891615542	96.31901840490798	96.2167689

			386503			1615542

3.3.b In each step, we take 100 more features which results in more training time complexity. So, there is a linear relationship between the number of features that we consider and the training time.

3.4 Multinomial Bayes classifier gives an accuracy rate as 94.274028629856. Bernoulli Bayes classifier gives an accuracy rate as 96.62576687116564 with 100 features. So, Bernoulli Bayes classifier is better at accuracy rate because it performs a feature selection and take the best features that has important information about the data which gives more insight about the data while understanding the output.