Assignment 5

Net ID: ss46
Name: Sidhartha Satapathy

1a. Based on the training data, we want to construct a Naive Bayes classifier. (No smoothing is required.) Please estimate the following terms:

1a(i). Pr(Popularity = ‘P')
Ans. 0.7

1a(ii). Pr(Popularity = ‘NP')
Ans. 0.3

1a(iii). Pr(Price = ‘$’, Delivery = ‘Yes’, Cuisine = ‘Korean’ | Popularity = ‘P’)
Ans. 0.093294406 (32/ (7*7*7))

1a(iv). Pr(Price = ‘$’, Delivery = ‘Yes’, Cuisine = ‘Korean’ | Popularity = ‘NP’)
Ans = 0.0740740741

1b.[6] Suppose a restaurant has the values: Price = ‘$’, Delivery = ‘Yes’ , Cuisine = ‘Korean’. Based on the calculation in part (1a.), is this restaurant classified as popular?
Ans. Yes, it is popular (Value for popular is 0.065, Value for other is 0.022)

1c. [4] Design an ensemble method for Naive Bayes to further improve the accuracy and briefly describe the steps.

Ans. Bagging is an ensemble method that can be used to improve the Naive Bayes accuracy.
The algorithm is as follows:
Parameters are: **Number of models**, **Learning Algorithm**
Inputs include: **Training Examples**

To create the different models:

Do the following iteratively the number of model times.

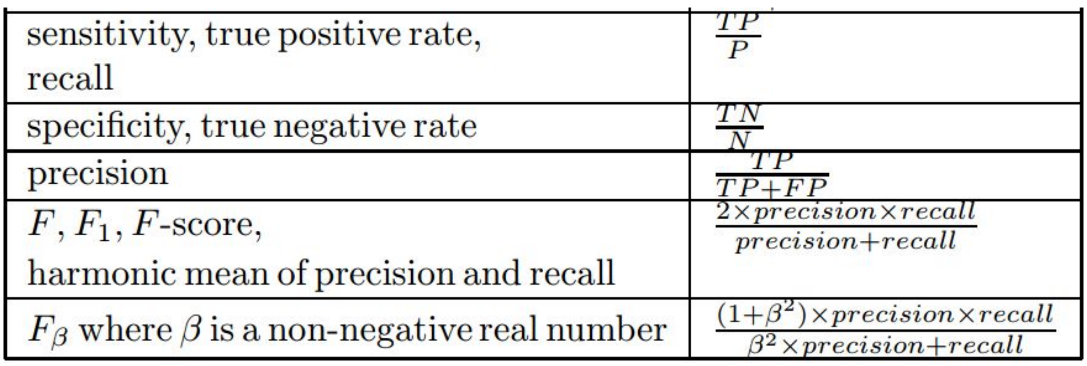
1. Bootstrap the examples by sampling d with replacement
2. Use these training examples to create the Naive Bayes model

At test time, we classify any test example by a majority vote.

1d. [4] Describe the metrics that can effectively evaluate the classification of data with rare positive examples.

Ans.
Evaluation metrics such as F, Fβ, Recall, Specificity, Precision can effectively evaluate the classification of data with rare positive examples.

These are the definitions: (Source book)



sensitivity, true positive rate, recall	$\frac{TP}{P}$
specificity, true negative rate	$\frac{TN}{N}$
precision	$\frac{TP}{TP+FP}$
F, F_1, F-score, harmonic mean of precision and recall	$\frac{2 \times precision \times recall}{precision + recall}$
F_β where β is a non-negative real number	$\frac{(1+\beta^2) \times precision \times recall}{\beta^2 \times precision + recall}$

In this algorithm we choose k nearest neighbours according to a distance metric and choose the majority vote amongst those.

2a. Testing Error = 1 - 0.75 = 0.25 = 25%
In this case we use the euclidean distance as the distance metric.

1 (2.7, 2.7), +1 2.3, 3.0, Final pred:-1 (Misclassified)
2 (2.5, 1.0), +1   2.0, 1.2, Final pred:+1
3 (1.5, 2.5), -1 1.5, 2.0, Final pred:-1
4 (1.2, 1.0), -1 0.8, 1.0, Final pred:-1

2b. Testing Error = 1 - 0.75 = 0.25 = 25%

1 (2.7, 2.7), +1 2.3, 3.0, -1 2.5, 2.0, +1 3.0, 2.0, +1 Final pred: +1

2 (2.5, 1.0), +1 2.0, 1.2, +1 2.5, 2.0, +1 3.0, 2.0, +1 Final pred:+1

3 (1.5, 2.5), -1 1.5, 2.0, -1 1.2, 1.9, -1 2.3, 3.0, -1 Final pred: -1

4 (1.2, 1.0), -1 0.8, 1.0, -1 1.0, 0.5, +1 2.0, 1.2, +1 Final pred: +1 (Misclassified)

2c.

So I choose a = 1, b= -1 and c =0 (F(x) = x1-x2)

From the figure we can clearly see that this will produce a training accuracy of 100%. As a result I choose this as my linear classifier.

Training Error = 1 - 0 = 0%

1 (2.7, 2.7), 2.7-2.7 = 0 predict as + 1

2 (2.5, 1.0), 2.5-1.0 = 1.5 predict as + 1

3 (1.5, 2.5), 1.5-2.5 = -1 predict as - 1

4 (1.2, 1.0), 1.2-1.0 = 0.2 predict as + 1 (Misclassified)

Testing Error = 1 - 0.75 = 0.25 = 25%

2d. The differences between the two algorithms, SVM and KNN are the way we train and test the models.

KNN:

There is no training time, but the testing time is high.

It takes up a lot of time to test as we have to compute the distance of the test point with each training point. It also takes a lot of memory as we have to store all the points.

SVM:

With SVM, training takes time, but testing is faster.

In this case we pick the kernel and use our parameters that we store to predict the test examples.
For the specific data we get better results with SVM as the data is clearly linearly separable and SVM works well for such type of data.

3a.

K means algorithm (Source Slides):

1. **Select K points as initial centroids**
2. **Repeat Until convergence criterion is satisfied**
    a. **Form K clusters by assigning each point to its closest centroid**
    b. **Re-compute the centroids (i.e., mean point) of each cluster**

Point [1,3], [1,2], [2,1], [2,2], [2,3], [3,2] is in cluster 0.

Point [5,3], [4,3], [4,5], [5,4], [5,5], [6,4], [6,5] is in cluster 1.

The cluster centroids are:

Cluster 0: [1.83333333, 2.16666667]

Cluster 1: [5, 4.14285714]

3b.

Algorithm is based on the slides: (Source slides)

1. Arbitrarily select a point p.

2. Retrieve all points density-reachable from p wrt Eps and MinPts. If p is a core point, a cluster is formed and If p is a border point, no points are density-reachable from p, and DBSCAN visits the next point of the database.

3. Continue the process until all of the points have been processed.

For our problem with the given epsilon and min points all the points form a single cluster.

[1,3], [1,2], [2,1], [2,2], [2,3], [3,2], [5,3], [4,3], [4,5], [5,4], [5,5], [6,4], [6,5] they all form a single cluster, that is they are all assigned to cluster 0.

3c. Algorithm: Continuously merge nodes that have the least dissimilarity and eventually all nodes belong to the same cluster. (Source Slides)

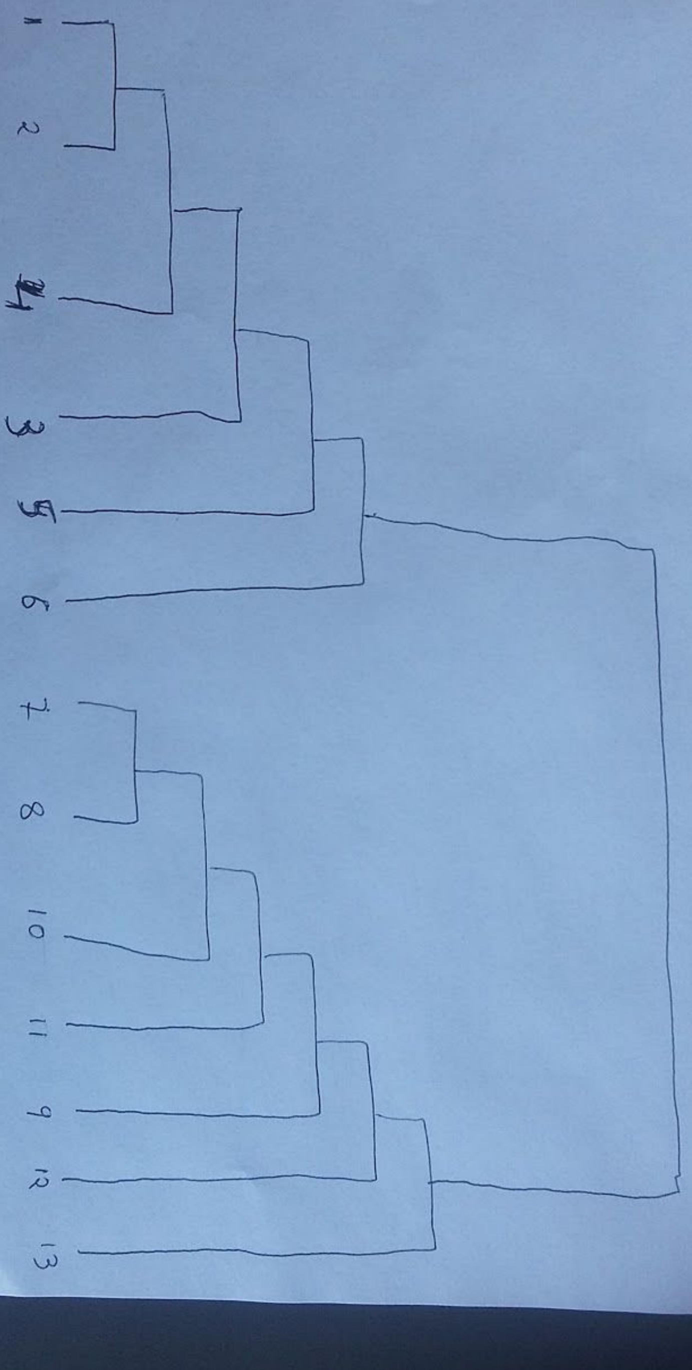
We use the single link method and use euclidean distance as the dissimilarity measure.

First half: [1,2] are merged first, it is then merged with point 4, then point 3, then 5 and then point 6.

Second half: First [7,8] are merged, then point 10, then point 11, then point 9, then point 12 and then point 13.

Finally these two are merged.

And the following dendrogram is formed: (It is on the next page)


1
2
4
3
5
6
7
8
10
11
9
12
13