CS 373 Homework 4
By: Siddharth Shah

Perceptron

1) $f(x) = \begin{cases} 1 & b + \sum w_i x_i \geq 0 \\ 0 & b + \sum w_i x_i < 0 \end{cases}$

The bias term shifts the estimation function of the classifier to the left or right depending on the sign of the bias.

2) a) i) Since the data is not linearly separable, having a bias or not doesn't affect the accuracy much. The accuracy will be low (< 0.95)

ii) Since the data is not linearly separable, having a bias or not doesn't affect the accuracy much. The accuracy will be low (< 0.95), but this will have a slightly higher accuracy than perceptron without bias.

b) i) Since the data is not linearly separable, having a bias or not doesn't affect the accuracy much. The accuracy will be low (< 0.95)

ii) Since the data is not linearly separable, having a bias or not doesn't affect the accuracy much. The accuracy will be low (< 0.95) but since the absolute magnitude of the data points is higher, bias will contribute to a better accuracy than one without.

c) i) Since the data is linearly separable, I think the perceptron without bias will yield an accuracy >0.95.

ii) Since the data is linearly separable, I think the perceptron with bias will yield an accuracy >0.95.

d) i) Since the boundary between the two classes is more ambiguous than c) I think perceptron without bias will yield accuracy <0.95.

ii) Since the boundary between the two classes is more ambiguous than c) I think perceptron with bias will yield accuracy >0.95.

3) I'd perform PCA on the given data set to capture some sort of related behavior between existing features. Since PCA translates all the current features onto a different vector space, a combination of dominant features will capture more data definition and trends as compared to just raw features.

4) Weights and bias are the w_i and b terms of the f(x) as defined in question 1. The weights are the coefficients of the linear estimation function and the bias is the constant of the estimation function, which creates the boundary between classes for the classifier. The goal of perceptron classifier is to best estimate the decision boundary, and we update the weights and bias after calculating error at every step of the perceptron training to better fit the data. Lines 8 and 9 are only executed when the prediction and actual labels differ. Since in such a case we need to improve the classification we update the error to the bias and the weights with the error.

5) a) Pseudocode for Average Perceptron

```
function Train(D, MaxIter):
        $w_i \leftarrow 0$ for all i = 1,..D
        $a_i \leftarrow 0$ for all i = 1,…D

        step ← len(D) x MaxIter
        b ← 0
        beta ← 0
        for iter = 1,..MaxIter do
           for all (x, y) ∈ D do
              error ← y - f(x)
                if error then
                   b ← b + error
                   $w_i \leftarrow w_i$ + (error $\times x_i$), for all i = 1,..D
                   $a_i \leftarrow a_i + (error \times x_i \times \frac{step}{len(D) \times MaxIter}$ for all i = 1,….D
                   beta ← beta + error * step / len(D) x MaxIter
        return beta, a
```

b) The advantage of averaged perceptron is that we only keep running sum of the averaged weight vector and the averaged bias term (less memory foot print) with minimal difference in testing accuracy.

Naive Bayes

1) $P(Y|X) = P(Y)\prod_{i=1}^{n} P(x_i|Y)$

2) $y = argmax_{(k \in \{1...K\})} p(Y) \cdot \prod_{i=1}^{n} p(x_i|Y)$

3) We can assume that X and Y are conditionally independent events, when P(X | Y) = P($x_1 | x_2 \ldots . x_n, Y$) . P($x_2, \ldots x_n, Y$).
Using the chain rule, we can expand this to the simplest probability as follows:

$$P(X|Y) = P(x_1|x_2 \ldots x_n, Y) \cdot P(x_2|x_3 \ldots x_n, Y) \ldots P(x_n|Y) \cdot p(Y)$$

where P(Y) is the simplest known probability.

4) The class prior is P(Y) where Y corresponds to the class variable. 0.6 for goodForGroups = 1 and 0.4 for goodForGroups = 0. Smoothing doesn't affect the maximum likelihood much for the Yelp dataset, but it accounts for missing data nonetheless as a "fail-safe" probability calculator to prevent a division by zero.

5) 290 parameters need to be estimated for the NBC. These will be unique collection of different possible feature values for each columns in the full data set.

6) P(alcohol | goodForGroups)

7) With Smoothing

Attribute	Value	CFD	goodForGroups
Stars	3.5	0.02025173662649172	1
Stars	3.5	0.01595185020062416З	0
Stars	4.0	0.01715559518502006	0
Stars	4.0	0.018351837558629697	1
Stars	4.5	0.011805617476593847	0
Stars	4.5	0.018351837558629697	1
Stars	3.0	0.013815828534109125	1
Stars	3.0	0.011377619259919751	0
Stars	2.5	0.0067677218011591616	0
Stars	2.5	0.006887134120999822	1
Stars	2.0	0.003334819438252341	0
Stars	2.0	0.0025767381107878644	1
Stars	1.5	0.0015247436469014713	0
Stars	1.5	0.0008133942884284272	1
Attire	casual	0.0279001337494	0
Attire	casual	0.0621445110728	1
Attire	BLANK	0.203217954046	1
Attire	BLANK	0.407543468569	0
waiterService	TRUE	0.0962951968177	1
waiterService	TRUE	0.0463753901025	0
waiterService	FALSE	0.137404262899	1
waiterService	FALSE	0.0649754792688	0
waiterService	BLANK	0.203217954046	1
waiterService	BLANK	0.407543468569	0
caters	BLANK	0.018351837558629697	1
caters	BLANK	0.0067677218011591616	0
caters	FALSE	0.02025173662649172	0
caters	FALSE	0.015951850200624163	1
caters	TRUE	0.02025173662649172	1
caters	TRUE	0.015951850200624163	0

Without Smoothing

Attribute	Value	CFD	goodForGroups
caters	BLANK	0.01598426604684427	0
caters	BLANK	0.020280718448911623	1
caters	TRUE	0.011827284105131414	0
caters	TRUE	0.020280718448911623	1
caters	FALSE	0.01719113177185768	0
caters	FALSE	0.013833710003568455	1

waiterService	BLANK	0.203562507434	1
waiterService	BLANK	0.408591096013	0
waiterService	TRUE	0.09645533484	1
waiterService	TRUE	0.0464866797783	0
waiterService	FALSE	0.137635303913	1
waiterService	FALSE	0.0651349901663	0
Stars	1.5	0.001519756838905775	0
Stars	1.5	0.0008088497680504342	1
Stars	2.0	0.0025752349232782204	1
Stars	2.0	0.003334525299481495	0
Stars	2.5	0.006893065302723921	1
Stars	2.5	0.0067763275522975145	0
Stars	3.0	0.011398176291793313	0
Stars	3.0	0.013833710003568455	1
Stars	3.5	0.020280718448911623	1
Stars	3.5	0.01598426604684427	0
Stars	4.0	0.01837754252408707	1
Stars	4.0	0.01719113177185768	0
Stars	4.5	0.011827284105131414	0
Stars	4.5	0.007440228381110979	1
Attire	casual	0.0279635258359	0
Attire	casual	0.0622457475913	1
Attire	BLANK	0.408591096013	0
Attire	BLANK	0.203562507434	1

Analysis

1) a) Avg. Perceptron

1% - 0.24848387186552046
10% - 0.205178445525193
50% - 0.20497267572652939
70% - 0.20354790091366234

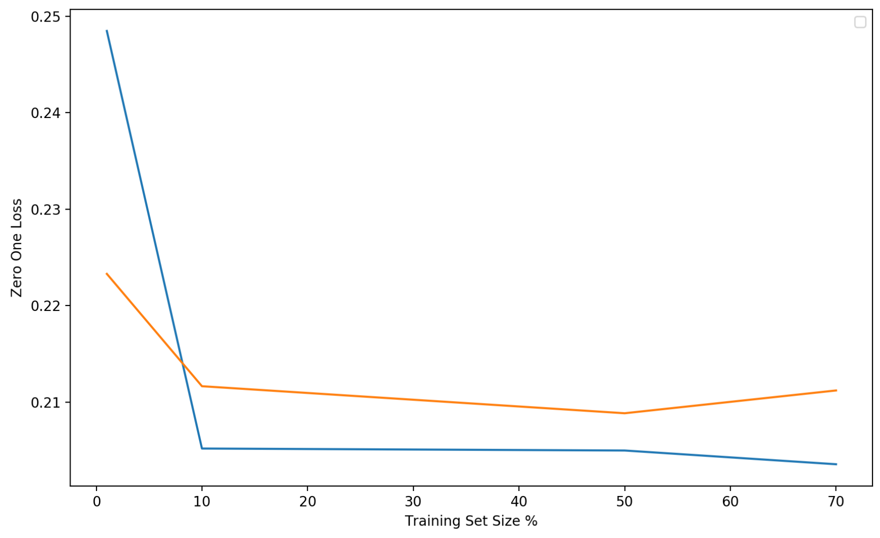
NBC

1% - 0.22329282178863546
10% - 0.21164068323738422
50% - 0.2088362413814773
70% - 0.21119981917238712

b) Squared Error for NBC

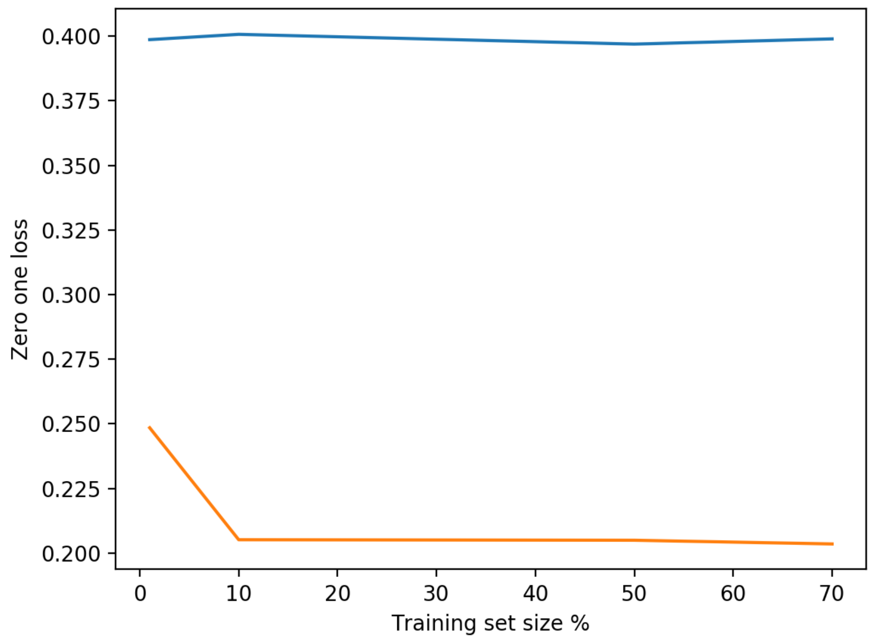
1% - 0.18297099293155722
10% - 0.17821606244576138
50% - 0.17835277127576082
70% - 0.1773361755448243

2) a)



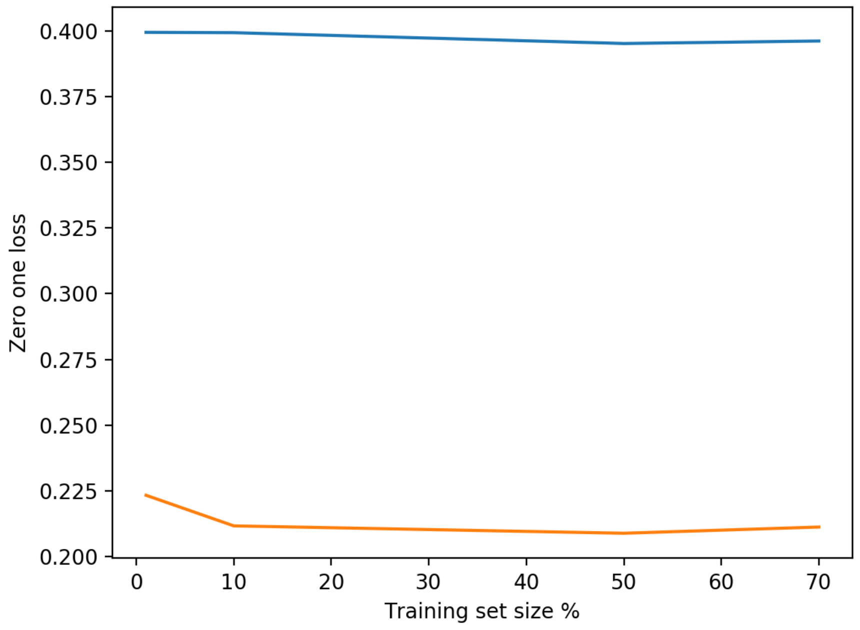
Key : Orange - NBC Average Loss
Blue - Average Perceptron Average Loss

b)



Averaged Perceptron with baseline(Blue)

Average Perceptron with baseline (blue)

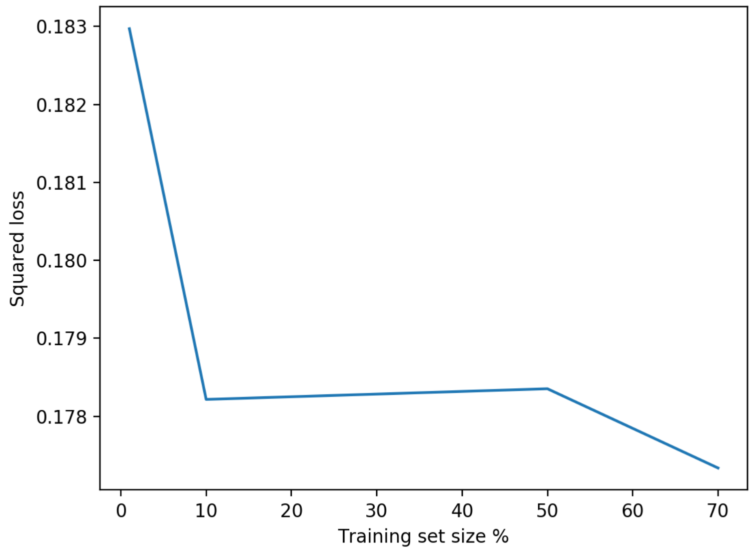


Naive Bayes with baseline (blue)

c) - Both the classifiers perform better than the baseline, however Perceptron performs slightly better when trained with larger training dataset.

- The zero-one loss looks like an elbow plot with increasing training set sizes. After a point training on new data points results only in marginal increase in the accuracy for both classifiers.

3) a)



b) I think zero-one loss has better performance as compared to squared loss, since squared loss penalized outliers while zero-one loss penalized misclassification. This is a mode estimation, since naive bayes predicts the most probable value, it is suitable for naive bayes.