I could not use the converted data format for making sparse matrix but I included all the data that were asked in **problem 1**. I used my own method to get required data to create sparse matrix in **problem 2 and on words.**

**Applied ML assignment 3 problem 1.1.ipynb** contains code to generate vocabulary and transformed data format for **IMDB data**

**Applied ML assignment 3 problem 1.2.ipynb** contains code to generate vocabulary and transformed data format for  **yelp data**

Created sparse matrix are uploaded in the file **(ex file name: IMDB\_train\_bag\_mat.npz** is saved sparse matrix generated frombinary bag of words, **IMDB\_train\_freq\_mat.npz** generated from frequency bag of words

**Problem 1:**

The required conversion for the reviews of Yelp and IMDB data sets have been done and converted Data has been saved as

IMDB\_train\_rev2num.txt

IMDB\_test\_rev2num.txt

IMDB\_valid\_rev2num.txt

yelp\_train\_rev2num.txt

yelp\_test\_rev2num.txt

yelp\_valid\_rev2num.txt

Top 10000 words from reviews of both datasets have been saved as

IMDB\_vocab.txt

yelp\_vocb.txt

where- word, word id, word frequency are mentioned column-wise.

**Problem 2:**

**2.a performance of the random and majority class classifier :**

performance of the random classifier :

F1 score for

1. training 0.267
2. validation 0.273
3. testing 0.278

performance of the majority classifier :

F1 score for

1. training 0.352
2. validation 0.351
3. testing 0.356

**2.b Classification performance of the naïve, decision tree, linear classifier using binary bag of words:**

**2.b.1. Naive bayes**

Bernoulli Naïve Bayes :For yelp data sets when used **Binary** bag of words I used **bernoullie naive bayes** using sklearn.naive\_bays.BernoulliNB

Hyperparameters:

1. alpha = [0, 1e-10, 1e-9, 1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1]

Best value of Hyperparameter:

1. alpha = 0.01

F1 score for:

1. Yelp validation data 0.427
2. Yelp Training set 0.749
3. Yelp Testing set 0.435

**2.b.2. Decision tree**

Hyperparameters:

1. criterion = [‘gini’,’entropy’]
2. Splitter = [‘best’,’random’]

Best value of Hyperparameter:

1. criterion = entropy
2. splitter = best

F1 score for:

1. Yelp validation data 0.37
2. Yelp training data 1.0
3. Yelp testing data 0.333

**2.b.3. Linear svc**

For linear svc, when we’re considering squared hinge loss, we need to do primitive optimization and it gives error in sklearn for this combination. So I have tuned other hyper-parameter(tolerance, C) twice keeping the loss, dual and penalty fixed.

Hyperparameters:

1. penalty = ['l1','l2']

2. loss = ['squared\_hinge']

3. dual = [False]

4. tolerance = [1e-10, 1e-9, 1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1]

5. C = [1e-10, 1e-9, 1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1]

Best value of Hyperparameter:

1. penalty = ['l2']

2. tolerance = 0.1

3. C = 0.01

F1 score for:

1. Yelp validation data 0.508
2. Yelp training data 0.81
3. Yelp testing data 0.502

Hyperparameters:

1. penalty = [''l2']

2. loss = ['hinge']

3. dual = [False]

4. tolerance = [1e-10, 1e-9, 1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1]

5. C = [1e-10, 1e-9, 1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1]

Best value of Hyperparameter:

1. penalty = ['l2']

2. tolerance = 0.1

3. C = 0.01

F1 score for:

1. Yelp validation data 0.508
2. Yelp training data 0.74
3. Yelp testing data 0.506

**Problem 3:**

**Performance comparison of binary and frequency bag of words:**

**3.a. Naive bayes**

For yelp data sets when used **Frequency** bag of words I tried out **Multinomial naive bayes and Gaussian navie bayes**.

3.a.1 Multinomial Naïve Bayes

Hyperparameters:

1. alpha = [0, 1e-10, 1e-9, 1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1,1]

Best value of Hyperparameter:

1. alpha = 1

F1 score for:

1. Yelp validation data 0.516
2. Yelp Training set 0.5974
3. Yelp Testing set 0.4085

3.a.2.Gaussian Naive Bayes : It requires the sparse matrix to convert to real form but that causes memory error.

**Comparison :** For binary and frequency bag of words I used different naïve classifier for the nature of the datasets. **Maximum F1 scores are similar** for both cases but for **different alpha**. Earlier I got optimised **alpha 0.01** and this time it’s **0.1**.

**3.b. Decision tree**

Hyperparameters:

1. criterion = [‘gini’,’entropy’]
2. Splitter = [‘best’,’random’]

Best value of Hyperparameter:

1. criterion = entropy
2. splitter = best

F1 score for:

1. Yelp validation data 0.3709
2. Yelp training data 1.0
3. Yelp testing data 0.3285

**Comparison :** **Maximum F1 scores are similar** and is achieved for **same optimised parameters**, where criterion is entropy (for information gain) and splitter is “best” is sklearn.

**3.c. Linear svc**

For linear svc, when we’re considering squared hinge loss, we need to do primitive optimization and it gives error in sklearn for this combination. So I have tuned other hyper-parameter(tolerance, C) twice keeping the loss, dual and penalty fixed.

Hyperparameters:

1. penalty = ['l1','l2']

2. loss = ['squared\_hinge']

3. dual = [False]

4. tolerance = [1e-10, 1e-9, 1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1]

5. C = [1e-10, 1e-9, 1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1]

Best value of Hyperparameter:

1. penalty = ['l2']

2. tolerance = 0.01

3. C = 0.01

F1 score for:

1. Yelp validation data 0.508
2. Yelp training data 0.855
3. Yelp testing data 0.508

Hyperparameters:

1. penalty = [''l2']

2. loss = ['hinge']

3. dual = [True]

4. tolerance = [1e-10, 1e-9, 1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1]

5. C = [1e-10, 1e-9, 1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1]

Best value of Hyperparameter:

1. penalty = ['l2']

2. tolerance = 0.01

3. C = 0.01

F1 score for:

1. Yelp validation data 0.507
2. Yelp training data 0.76
3. Yelp testing data 0.514

**Comparison :** For binary and frequency bag of words I used different naïve classifier for the nature of the datasets. **Maximum F1 scores are similar** for both cases we got **similar optimised hyper parameter except tolerance**. For frequency bag of word it’s 0.01 but for binary bag of word it’s 0.1

**Problem 4:**

**4.a performance of the random class classifier :**

performance of the random classifier :

F1 score for

1. training 0.5068
2. validation 0.5038
3. testing 0.5014

**4.b. Performance of the naïve, decision tree, linear classifier for binary bag of words:**

**Comment on classifier performance:**

Among the classifier used Naïve bayes and and linear svc classifies with higher accuracy than decision tree while using both **binary** and **frequency** bag of words. I got best performance for IMDB data set using Linear svc where both **binary** and **frequency** bag of words give similar performance, the hyper parameters are:

1. penalty = ['l2']
2. loss = ['squared\_hinge']
3. dual = False

2. tolerance = 0.01

3. C = 0.01

F1 score while using **frequency** bag of words:

1. Yelp validation data 0.8778
2. Yelp training data 0.9715
3. Yelp testing data 0.8692

Details results are given below

**4.b.1. Naive bayes**

Bernoulli Naïve Bayes :For IMDB data sets when used **Binary** bag of words I used **bernoullie naive bayes** using sklearn.naive\_bays.BernoulliNB

Hyperparameters:

1. alpha = [0, 1e-10, 1e-9, 1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1,1]

Best value of Hyperparameter:

1. alpha = 0.1

F1 score for:

1. IMDB validation data 0.8433
2. IMDB Training set 0.8707
3. IMDB Testing set 0.8318

**4.b.2. Decision tree**

Hyperparameters:

1. criterion = [‘gini’,’entropy’]
2. Splitter = [‘best’,’random’]

Best value of Hyperparameter:

1. criterion = gini
2. splitter = random

F1 score for:

1. Yelp validation data 0.6979
2. Yelp training data 1.0
3. Yelp testing data 0.6977

**4.b.3. Linear svc**

For linear svc, when we’re considering squared hinge loss, we need to do primitive optimization and it gives error in sklearn for this combination. So I have tuned other hyper-parameter(tolerance, C) twice keeping the loss, dual and penalty fixed.

Hyperparameters:

1. penalty = ['l1','l2']

2. loss = ['squared\_hinge']

3. dual = [False]

4. tolerance = [1e-10, 1e-9, 1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1]

5. C = [1e-10, 1e-9, 1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1]

Best value of Hyperparameter:

1. penalty = ['l2']

2. tolerance = 0.01

3. C = 0.01

F1 score for:

1. Yelp validation data 0.8744
2. Yelp training data 0.9630
3. Yelp testing data 0.8689

Hyperparameters:

1. penalty = [''l2']

2. loss = ['hinge']

3. dual = [False]

4. tolerance = [1e-10, 1e-9, 1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1]

5. C = [1e-10, 1e-9, 1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1]

Best value of Hyperparameter:

1. penalty = ['l2']

2. tolerance = 0.01

3. C = 0.01

F1 score for:

1. Yelp validation data 0.8736
2. Yelp training data 0.9286
3. Yelp testing data 0.8736

**4.c. Performance of the naïve, decision tree, linear classifier for frequency bag of words:**

**4.c.1. Naive bayes**

For yelp data sets when used **Frequency** bag of words I tried out **Multinomial naive bayes and Gaussian navie bayes**.

4.c.1.a Multinomial Naïve Bayes

Hyperparameters:

1. alpha = [0, 1e-10, 1e-9, 1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1,1]

Best value of Hyperparameter:

1. alpha = 1

F1 score for:

1. Yelp validation data 0.8288
2. Yelp Training set 0.8598
3. Yelp Testing set 0.8188

4.a.1.b.Gaussian Naive Bayes : It requires the sparse matrix to convert to real form but that causes memory error.

**Comparison :** I used different naïve classifier for the nature of the datasets. Maximum F1 scores are similar for both cases but for different alpha. Earlier I got optimised alpha 0.01 and this time it’s 0.1.

**4.c.2. Decision tree**

Hyperparameters:

1. criterion = [‘gini’,’entropy’]
2. Splitter = [‘best’,’random’]

Best value of Hyperparameter:

1. criterion = entropy
2. splitter = best

F1 score for:

1. Yelp validation data 0.7079
2. Yelp training data 1.0
3. Yelp testing data 0.6996

**Comparison :** Maximum F1 scores are similar and is achieved for same optimised parameters, where criterion is entropy (for information gain) and splitter is “best” is sklearn.

**4.c.3. Linear svc**

For linear svc, when we’re considering squared hinge loss, we need to do primitive optimization and it gives error in sklearn for this combination. So I have tuned other hyper-parameter(tolerance, C) twice keeping the loss, dual and penalty fixed.

Hyperparameters:

1. penalty = ['l1','l2']

2. loss = ['squared\_hinge']

3. dual = [False]

4. tolerance = [1e-10, 1e-9, 1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1]

5. C = [1e-10, 1e-9, 1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1]

Best value of Hyperparameter:

1. penalty = ['l2']

2. tolerance = 0.01

3. C = 0.01

F1 score for:

1. Yelp validation data 0.8796
2. Yelp training data 0.9415
3. Yelp testing data 0.8727

Hyperparameters:

1. penalty = [''l2']

2. loss = ['hinge']

3. dual = [False]

4. tolerance = [1e-10, 1e-9, 1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1]

5. C = [1e-10, 1e-9, 1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1]

Best value of Hyperparameter:

1. penalty = ['l2']

2. tolerance = 0.01

3. C = 0.01

F1 score for:

1. Yelp validation data 0.8778
2. Yelp training data 0.9715
3. Yelp testing data 0.8692