Kristy Mualim
260679030
Assignment 3

1. Dataset was formatted using the code in a3-preprocessing submitted as a jupyter notebook

2(a). Performance of random classifier and majority-class classifier

F1_measure for random classifier for yelp is: 0.204
F1_measure for majority classifier for yelp is: 0.351

2(b) Training was done in Naïve Bayes, Decision Trees and Linear SVM seen in jupyter notebook.

2(c)
List of parameters considered for Bernoulli Naïve Bayes:
Alpha parameters = [0, 0.0001, 0.001, 0.01, 0.1, 1, 2, 10, 12, 20, 50]
Best value chosen: 0.01
Bernoulli Naïve Bayes considered.

List of parameters considered for Decision Tree:
maxdepths = np.linspace(1,32,32,endpoint=True)
min_samples_split = np.linspace(0.1, 1.0, 10, endpoint=True)
min_samples_leaf = np.linspace(0.1, 0.5, 5, endpoint=True)
Best value chosen: 8.0 for maxdepth was best parameter, default values formin_samples_split and min_samples_leaf were selected

List of parameters considered for SVM:
parameters_lin = [0.0001, 0.001, 0.01, 0.1, 1, 2, 10, 20,100, 1000]
Best value chosen: 0.01

List of parameters tested for F1 score measure:
Average: “weighted”, “micro”, “macro”
All three values obtained similar thus “micro” was picked at random.

2(d) F1 for YELP BBOW

F1 SCORE FOR TRAINING, VALIDATION, TEST FOR NAÏVE BAYES:
Max F1 for Naive Bayes for training: 0.7458571428571429
seen at parameter: 0.01
Max F1 for Naive Bayes for validation: 0.426
seen at parameter: 0.01

Max F1 for Naive Bayes for testing: 0.443
seen at parameter: 0.01

F1 SCORE FOR TRAINING, VALIDATION, TEST FOR DECISION TREE:
Max F1 for Decision Tree for training: 0.48
seen at parameter: 8.0
Max F1 for Decision Tree for validation: 0.403
seen at parameter: 8.0
Max F1 for Decision Tree for testing: 0.3970000000000001
seen at parameter: 8.0

2(d)

F1 SCORE FOR TRAINING, VALIDATION, TEST FOR SVM:
Max F1 for SVM for training: 0.8418571428571429
seen at parameter: 0.01
Max F1 for SVM for validation: 0.499
seen at parameter: 0.01
Max F1 for SVM for testing: 0.5075
seen at parameter: 0.01

2(e)

Performance of classifiers for Binary bag of words:
Naïve Bayes, Decision tree and SVM all perform better than the random class, majority-class cla ssifier. SVM reported the best test performance at 0.51, Decision Tree registered the worst perfor mance at 0.397 while Naïve Bayes had medium performance at 0.443.
Naïve Bayes is a generative model that takes into consideration the distribution of data while SV M aims to maximize the distance between data points of different classes. Alternatively, Decisio n trees maximizes the information gain based on separability of data points. The dataset contains 5 different classes which could make classification via decision trees difficult to linearly separat e.

Why did SVM perform better? For the given parameters chosen, SVM performs slightly better than Naïve Bayes when C=0.01. SVM uses a kernel to project feature space into kernel space, m aking classes linearly separable thus allowing it to perform in this case. However, alternatively, Decision trees would try to separate data despite the data not being linearly separable as it tries to maximize information gain thus, may generate linear boundaries to satisfy this.

Role of hyperparameter that fetched best results?
The parameter that was tweaked for Naïve Bayes was alpha. Alpha is the Laplacian smoothing p arameter with an optimum value at the prior probability of the class and is based on the data distr ibution of each class.

The parameter that was tweaked in SVM was C: the penalty parameter C of the error term. This parameter tells the SVM optimization how much to avoid misclassification of each training samp le. Smaller C values causes the optimizer to look for larger-margin separating hyperplanes. For ti ny values of C, such as 0.01, there will be misclassified examples even if training data is linearly separable.

3(a) Training was done in Naïve Bayes, Decision Trees and Linear SVM seen in jupyter noteboo k.

3(b)
Gaussian Bayes considered

List of parameters considered for Decision Tree:
maxdepths = np.linspace(1,32,32,endpoint=True)
min_samples_split = np.linspace(0.1, 1.0, 10, endpoint=True)
min_samples_leaf = np.linspace(0.1, 0.5, 5, endpoint=True)
Best value chosen: 10.0 for maxdepth was best parameter, default values formin_samples_split and min_samples_leaf
3(b)

List of parameters considered for SVM:
parameters_lin = [0.0001, 0.001, 0.01, 0.1, 1, 2, 10, 20,100, 1000]
Best value chosen: 10

List of parameters for F1 score measure:
Average: "weighted", "micro", "macro"
All three values obtained similar thus "micro" was picked at random.

3(c) F1 for YELP FBOW

F1 SCORE FOR TRAINING, VALIDATION, TEST FOR GAUSSIAN BAYES:
Max F1 for Gaussian Bayes for training: 0.8045714285714286
Max F1 for Gaussian Bayes for validation: 0.295
Max F1 for Gaussian Bayes for testing: 0.31

F1 SCORE FOR TRAINING, VALIDATION, TEST FOR DECISION TREE:
Max F1 for Decision Tree for training: 0.5428571428571428
seen at parameter: 10.0
Max F1 for Decision Tree for validation: 0.427
seen at parameter: 10.0
Max F1 for Decision Tree for testing: 0.401
seen at parameter: 10.0

F1 SCORE FOR TRAINING, VALIDATION, TEST FOR SVM:
Max F1 for SVM for training: 0.6414285714285715
seen at parameter: 10
Max F1 for SVM for validation: 0.501
seen at parameter: 10
Max F1 for SVM for testing: 0.508

seen at parameter: 10

3(d) Performance of classifiers for frequency bag of words:
Naïve Bayes, Decision tree and SVM all perform better than the random class, majority-class cla ssifier. SVM reported the best test performance at 0.508, Decision Tree registered the medium pe rformance at 0.399 while Naïve Bayes had medium performance at 0.443. Naïve Bayes performe d better than gaussian Bayes classifier.

3(d)

Why did SVM perform better? For the given parameters chosen, SVM performs slightly better than Naïve Bayes when C=10.0. SVM uses a kernel to project feature space into kernel space, m aking classes linearly separable thus allowing it to perform in this case. However, alternatively, Decision trees would try to separate data despite the data not being linearly separable as it tries to maximize information gain thus, may generate linear boundaries to satisfy this.

Role of hyperparameter that fetched best results?
The parameter that was tweaked for Naïve Bayes was alpha. Alpha is the Laplacian smoothing p arameter with an optimum value at the prior probability of the class and is based on the data distr ibution of each class.

The parameter that was tweaked was C: the penalty parameter C of the errorterm. This parameter tells the SVM optimization how much to avoid misclassification of each training sample. Larger C values causes the optimizer to look for smaller-margin separating hyperplanes.

3(e) Performance between BBow and fBow was relatively similar, with the best performing class ifier being SVM. Naïve Bayes performed better in Yelp BBOW than in yelp FBOW. This observ ation could be attributed to GNB assuming a normal distribution while Bernoulli assumes an eev ent based model. In frequency bag of words, calculating the occurrence of words rarely assumes a normal distribution to allow GNB to model it perfectly.
It should be noted that FBow representation achieved a similar performance at a parameter of 10 as opposed to that of BBow at 0.01 in SVM. Frequency bag of words yelp data generates a small er data set with values that are close to zero. This increases the possibility of misclassication relat ive to binary bag of words. Thus, SVM tries to compensate for this misclassification, resulting in a bigger C value.

3(f) Representation seemed to have no effect relating to performance within similar classifiers.

4(a)
F1_measure for random classifier for IMDB is: 0.501

4(b) Training was done in Naïve Bayes, Decision Trees and Linear SVM seen in jupyter noteboo k.

4(c)
List of parameters considered for Naïve Bayes:

parameters = [0, 0.0001, 0.001, 0.01, 0.1, 1, 2, 10, 12, 20, 50]
Best value chosen: 0.1
Bernoulli Naïve Bayes considered. dual set to false.

List of parameters considered for Decision Tree:
maxdepths = np.linspace(1,32,32,endpoint=True)
min_samples_split = np.linspace(0.1, 1.0, 10, endpoint=True)

4(c)
min_samples_leaf = np.linspace(0.1, 0.5, 5, endpoint=True)
Best value chosen: 13.0 for maxdepth was best parameter, default values formin_samples_split and min_samples_leaf

List of parameters considered for SVM:
parameters_lin = [0.0001, 0.001, 0.01, 0.1, 1, 2, 10, 20, 100, 1000]
Best value chosen: 0.01

4(c)
List of parameters for F1 score measure:
Average: “weighted”, “micro”, “macro”
All three values obtained similar thus “micro” was picked at random.

4(d) F1 for IMDB BBOW

F1 SCORE FOR TRAINING, VALIDATION, TEST FOR NAÏVE BAYES:
Max F1 for Naive Bayes for training: 0.8709333333333333
seen at parameter: 0.1
Max F1 for Naive Bayes for validation: 0.8436
seen at parameter: 0.1
Max F1 for Naive Bayes for testing: 0.83176
seen at parameter: 0.1

F1 SCORE FOR TRAINING, VALIDATION, TEST FOR DECISION TREE:
Max F1 for Decision Tree for training: 0.7981999999999999
seen at parameter: 13.0
Max F1 for Decision Tree for validation: 0.7218
seen at parameter: 13.0
Max F1 for Decision Tree for testing: 0.72528
seen at parameter: 13.0

F1 SCORE FOR TRAINING, VALIDATION, TEST FOR SVM:
Max F1 for SVM for training: 0.9632666666666667

seen at parameter: 0.01
Max F1 for SVM for validation: 0.8746
seen at parameter: 0.01
Max F1 for SVM for testing: 0.86924
seen at parameter: 0.01

4(e) Performance of classifiers for frequency bag of words:

4(e)
Naïve Bayes, Decision tree and SVM all perform better than the random class classifier. SVM re ported the best test performance at 0.869, Decision Tree registered the worst performance at 0.72 8 while Naïve Bayes had medium performance at 0.832.

Naïve Bayes is a generative model that takes into consideration the distribution of data while SV M aims to maximize the distance between data points of different classes. Alternatively, Decisio n trees maximizes the information gain based on separability of data points.
The dataset contains 2 different classes which makes classification linearly separable hence prese nting relatively similar F1-scores.

Why did SVM perform better? For the given parameters chosen, SVM performs slightly better than Naïve Bayes when C=0.01. SVM uses a kernel to project feature space into kernel space, m aking classes linearly separable thus allowing it to perform in this case. However, alternatively,

4(e)
Decision trees would try to separate data despite the data not being linearly separable as it tries to maximize information gain thus, may generate linear boundaries to satisfy this.

Role of hyperparameter that fetched best results?
The parameter that was tweaked for Naïve Bayes was alpha. Alpha is the Laplacian smoothing p arameter with an optimum value at the prior probability of the class and is based on the data distr ibution of each class.

For SVM: Dual was set to false: given that the number of examples is greater than the number of features.

The parameter that was tweaked was C: the penalty parameter C of the error term. This paramete r tells the SVM optimization how much to avoid misclassification of each training sample. Small er C values causes the optimizer to look for larger-margin separating hyperplanes. For tiny value s of C, such as 0.01, there will be misclassified examples even if training data is linearly separabl e.

5(a) Training was done in Naïve Bayes, Decision Trees and Linear SVM seen in jupyter noteboo k.

5(b)
List of parameters considered for Naïve Bayes:
parameters = [0, 0.0001, 0.001, 0.01, 0.1, 1, 2, 10, 12, 20, 50]
Best value chosen: 0.1
Gaussian Bayes considered

List of parameters considered for Decision Tree:
maxdepths = np.linspace(1,32,32,endpoint=True)
min_samples_split = np.linspace(0.1, 1.0, 10, endpoint=True)
min_samples_leaf = np.linspace(0.1, 0.5, 5, endpoint=True)
Best value chosen: 10.0 for maxdepth was best parameter, default values for min_samples_split and min_samples_leaf

List of parameters considered for SVM:
parameters_lin = [0.0001, 0.001, 0.01, 0.1, 1, 2, 10, 20, 100, 1000]
Best value chosen: 100

List of parameters for F1 score measure:
Average: "weighted", "micro", "macro"
All three values obtained similar thus "micro" was picked at random.

5(c) F1 for IMDB FBOW

F1 SCORE FOR TRAINING, VALIDATION, TEST FOR GAUSSIAN BAYES:
5(c)
Max F1 for Gaussian Bayes for training: 0.8617333333333334
Max F1 for Gaussian Bayes for validation: 0.7508000000000001
Max F1 for Gaussian Bayes for testing: 0.68908

F1 SCORE FOR TRAINING, VALIDATION, TEST FOR DECISION TREE:
Max F1 for Decision Tree for training: 0.7663333333333332
seen at parameter: 10.0
Max F1 for Decision Tree for validation: 0.7115
seen at parameter: 10.0
Max F1 for Decision Tree for testing: 0.70976
seen at parameter: 10.0

F1 SCORE FOR TRAINING, VALIDATION, TEST FOR SVM:
Max F1 for Decision Tree for training: 0.938
seen at parameter: 100
Max F1 for Decision Tree for validation: 0.8778
seen at parameter: 100

Max F1 for Decision Tree for testing: 0.87416
seen at parameter: 100

5(d) Performance of classifiers for frequency bag of words:
Naïve Bayes, Decision tree and SVM all perform better than the random class, majority-class cla ssifier. SVM reported the best test performance at 0.858, Decision Tree registered the worst perf ormance at 0.711 while Naïve Bayes had medium performance at 0.832.

5(d)
Why did SVM perform better? For the given parameters chosen, SVM performs slightly better than Naïve Bayes when C=100. SVM uses a kernel to project feature space into kernel space, m aking classes linearly separable thus allowing it to perform in this case. However, alternatively, Decision trees would try to separate data despite the data not being linearly separable as it tries to maximize information gain thus, may generate linear boundaries to satisfy this.

Role of hyperparameter that fetched best results?
The parameter that was tweaked for Naïve Bayes was alpha. Alpha is the Laplacian smoothing p arameter with an optimum value at the prior probability of the class and is based on the data distr ibution of each class.

For SVM: Dual was set to false: given that the number of examples is greater than the number of features.

The parameter that was tweaked was C: the penalty parameter C of the error term. This paramete r tells the SVM optimization how much to avoid misclassification of each training sample. Large r C values causes the optimizer to look for smaller-margin separating hyperplanes.

5(e) SVM and Decision Tree Performance between BBow and fBow was relatively similar, with the best performing classifier being SVM and the worst being Decision Trees. However, Naïve B ayes performed better for BBOW than FBOW representation. This observation could be attribute d to GNB assuming a normal distribution while Bernoulli assumes an event-based model. In freq uency bag of words, calculating the occurrence of words rarely assumes a normal distribution to allow GNB to model it perfectly. It should be noted that FBow representation achieved a similar performance at a parameter of 100 as opposed to that of BBow at 0.01 in SVM. Frequency bag o f words yelp data generates a smaller data set with values that are close to zero. This increases th e possibility of misclassication relative to binary bag of words. Thus, SVM tries to compensate f or this misclassification, resulting in a bigger C value.

5(f) Representation seemed to have no effect relating to performance within similar classifiers.

5(g) All classifiers perform much better in the IMDB set than in the yelp set. This could be attributed to yelp having 5 classes and IMDB having just 2 classes. Thus, making classification for IMDB less complicated. SVM has the best performance for both IMDB and yelp datasets. In addition, the performance of both IMDB and yelp using the frequency bag of words
5(g)

representation via Gaussian Naïve Bayes illustrates it to be a poorer model than naïve Bayes. The lack of hyperparameters as well as the general underlying bias of requiring the data to be a uniform distribution could explain the poor performance in both datasets.