CS 6375 Homework 1

Parker Whitehead
Pmw180000

Results for Each Dataset:

Enron1 Dataset:

Multinomial Naive Bayes Results:
Accuracy: 0.9583333333333334
Precision: 0.9012345679012346
Recall: 0.9798657718120806
F1 Score: 0.9182743837084673

Discrete Naive Bayes Results:
Accuracy: 0.956140350877193
Precision: 0.8862275449101796
Recall: 0.9932885906040269
F1 Score: 0.9016766600044415

Bag of Words MCAP Logistic Regression Results:
Accuracy: 0.9451754385964912
Precision: 0.9428571428571428
Recall: 0.8859060402684564
F1 Score: 0.9746099678261397

Bernoulli MCAP Logistic Regression Results:
Accuracy: 0.956140350877193
Precision: 0.9708029197080292
Recall: 0.8926174496644296
F1 Score: 0.9962581278370751

Bag of Words SGDClassifier Results:
Accuracy: 0.9385964912280702
Precision: 0.9548872180451128
Recall: 0.8523489932885906
F1 Score: 0.9918501928580316

Bernoulli SGDClassifier Results:
Accuracy: 0.9692982456140351
Precision: 0.9411764705882353
Recall: 0.9664429530201343
F1 Score: 0.9564598582549086

Enron4 Dataset:

Multinomial Naive Bayes Results:
Accuracy: 0.9576427255985267
Precision: 0.9554455445544554
Recall: 0.9872122762148338
F1 Score: 0.9419831590164124

Discrete Naive Bayes Results:
Accuracy: 0.9686924493554327
Precision: 0.9722222222222222
Recall: 0.9846547314578005
F1 Score: 0.9625381135907858

Bag of Words MCAP Logistic Regression Results:
Accuracy: 0.9594843462246777
Precision: 0.9533169533169533
Recall: 0.9923273657289002
F1 Score: 0.9402465648467394

Bernoulli MCAP Logistic Regression Results:
Accuracy: 0.9558011049723757
Precision: 0.9421686746987952
Recall: 1.0
F1 Score: 0.9273405261642652

Bag of Words SGDClassifier Results:
Accuracy: 0.9686924493554327
Precision: 0.9698492462311558
Recall: 0.9872122762148338
F1 Score: 0.960098219766728

Bernoulli SGDClassifier Results:
Accuracy: 0.9760589318600368
Precision: 0.9725
Recall: 0.9948849104859335
F1 Score: 0.9649533308654219

HW1 Dataset:

Multinomial Naive Bayes Results:
Accuracy: 0.9539748953974896
Precision: 0.8648648648648649
Recall: 0.9846153846153847
F1 Score: 0.8922067367026881

Discrete Naive Bayes Results:
Accuracy: 0.9205020920502092
Precision: 0.7771084337349398
Recall: 0.9923076923076923
F1 Score: 0.8085491349089675

Bag of Words MCAP Logistic Regression Results:
Accuracy: 0.9267782426778243
Precision: 0.8682170542635659
Recall: 0.8615384615384616
F1 Score: 0.9303565370510977

Bernoulli MCAP Logistic Regression Results:
Accuracy: 0.9476987447698745
Precision: 0.9487179487179487
Recall: 0.8538461538461538
F1 Score: 0.9975776260735522

Bag of Words SGDClassifier Results:
Accuracy: 0.9225941422594143
Precision: 0.8604651162790697
Recall: 0.8538461538461538
F1 Score: 0.9261562818048175

Bernoulli SGDClassifier Results:
Accuracy: 0.9602510460251046
Precision: 0.9236641221374046
Recall: 0.9307692307692308
F1 Score: 0.9565719232433992

Hyper-Parameter Tuning:

Logistic Regression:

The initial train dataset was sent off to a function within the Logistic Regression class called 'findOptimalLambda.' This function takes a dataframe, splits it into a 70-30 train-test, and then builds a model, each time using a different value of lambda. It then returns the value of lambda that produced the greatest sum of accuracy, precision, recall, and F1. An iteration limit of 50 was set for determining the best lambda, while an iteration limit of 200 was set for normal training.

SGDClassifier:

An initial dictionary of hyper-parameter values to choose from was created. This included the loss function, alpha values, and penalties. SKLearn's GridSearchCV was used to find the best hyper-parameters given from the dictionary.

Questions:

Q1:

Across almost all metrics, Bernoulli SGDClassifier had the best results. However, Discrete Naive Bayes provides a very consistent & phenomenal Recall, meaning that it's true positives are consistently correct. My reasoning for this is that SGDClassifier employs a much stronger cross validation for parameter tuning. As such, it creates a more effective model. Additionally, the Bernoulli implementation of the model avoids the need to normalize the data; the bag model was not normalized, which could have hindered the effectiveness of those that utilized it. Another thing to consider is that we did not use any sort of redundant-word filtering. As such, the Bag-of-words model could become filled with irrelevant noise that could lead to lesser results.

Q2:

In 2 of the 3 cases (Enron1 & HW1), Multinomial Naive Bayes outperforms both LR and SGDClassifier in accuracy and recall. I believe this to be a result of the benefit that normalization of data provides to both LR and SGDClassifier. Had normalization been applied, we could have seen greater results. However, it is less consistent with its F1 score and precision. This is indicative of it over-predicting spam. In this use case, however, this can be seen as better than the alternative, as a false positive means one might need to dig through a spam folder, but a false negative could lead someone to being scammed.

Q3:

In all cases, Discrete Naive Bayes is outperformed by SGDClassifier (except for recall, which is most likely due to NB frequently guessing positive over negative, due to the decreased amount of information it is given) and follows closely to the performance of LR. This is most likely due to Naive Bayes simplistic approach suffering to capture the complexity of emails/vocabulary importance with the limited binary data model of Bernoulli. In contrast, both linear regression and SGDClassifier tune parameters & perform numerous iterations to better fit the model that they are given. As such, they are able to pull more information from the limited dataset. The outperformance of SGDClassifier in this instance is most likely due to the more advanced parameter tuning compared to the LR implementation.

Q4:

LR & SGDClassifier perform very similarly when using a Bag-of-Words data model. However, SGDClassifier pulls ahead by a slight margin across all metrics when using Bernoulli. This is most likely due to the more advanced parameter tuning method (cross validation) that SGD employs, enabling it to pull more information from the binary data model.