Bag of Words:
Enron1 Naive Bayes Test Results:
Accuracy: 0.9627
Precision: 0.9286
Recall: 0.9597
F1: 0.9439

Enron2 Naive Bayes Test Results:
Accuracy: 0.9665
Precision: 0.9191
Recall: 0.9615
F1: 0.9398

Enron4 Naive Bayes Test Results:
Accuracy: 0.9669
Precision: 0.9921
Recall: 0.9616
F1: 0.9766

Combined Naive Bayes Test Results:
Overall Accuracy: 0.9655
Overall Precision: 0.9626
Overall Recall: 0.9612
Overall F1: 0.9619

Enron1 Logistic Regression results:
Accuracy: 0.9342
Precision: 0.8362
Recall: 0.9933
F1 Score: 0.9080

Enron2 Logistic Regression results:
Accuracy: 0.9351
Precision: 0.8153
Recall: 0.9846
F1 Score: 0.8920

Enron4 Logistic Regression results:
Accuracy: 0.9779
Precision: 0.9749
Recall: 0.9949

F1 Score: 0.9848

Bernoulli

Enron1 Naive Bayes Test Results:
Accuracy: 0.7961
Precision: 0.9242
Recall: 0.4094
F1: 0.5674

Enron2 Naive Bayes Test Results:
Accuracy: 0.8285
Precision: 0.9138
Recall: 0.4077
F1: 0.5638

Enron4 Naive Bayes Test Results:
Accuracy: 0.5967
Precision: 1.0000
Recall: 0.4399
F1: 0.6110

Combined Test Accuracy: 0.7332

Enron1 Logistic Regression Results:
Accuracy: 0.9474
Precision: 0.8655
Recall: 0.9933
F1 Score: 0.9250

Enron2 Logistic Regression Results:
Accuracy: 0.9644
Precision: 0.8951
Recall: 0.9846
F1 Score: 0.9377

Enron4 Logistic Regression Results:
Accuracy: 0.9816
Precision: 0.9922
Recall: 0.9821
F1 Score: 0.9871

Hyperparameters were turned by having a list of lambda values initially to test. These values run through a training session with logistic regression. Then the f1 scores across all lambda values are collected and the one with the highest f1 score is selected for final training. For the max iteration limit, the number was modified manually to determine the one that yielded the highest performance x time. The iteration limits were 500,750,1000, and 1250.

1. Which combination of algorithm and data representation yielded the best performance? Why?

The best result was yielded with logistic regression combined with bernoulli. This is most likely because logistic regression relies on creating boundaries between labels and uses a sigmoid function to resolve probabilities. The input for this sigmoid function is simply 0/1 which is optimal input space and the reduction of features from many states to 2 states allows for better performance. Also, given that for a spam filter, the number of occurrences of a word doesn't matter as much and can be collapsed into a binary classification allows for better classification .

2. Did Multinomial Naive Bayes perform better than Logistic Regression on the Bag of Words representation? Explain.

The multinomial bayes performed better than the bow logistic regression. This makes sense since the bayes algorithm excels when given frequency data as input - which bag of words provides. The multinomial nature of the algorithm matches with the input provided with the bag of word representation. Logistic regression performs worse because it utilizes a decision boundary which is not optimal for complex relationships provided with bow and the input given in general doesn't work well with the binary-prefered regression model.

3. Did Discrete Naive Bayes perform better than Logistic Regression on the Bernoulli representation? Explain

No, logistic regression with Bernoulli representation performed better. The reasons again can be linked back to the answer to question 1 as to why it is optimal. Restated, logistic regression can handle binary data better because of the sigmoid function which provides output 0-1 for probability. Naive bayes doesn't perform as well because the features are learned as independent from each other and doesn't adapt as well to how important a given feature is to the ultimate output. While naive bayes output graph can be seen as uniform, logistic regression is more feature optimized and attempts to understand feature relation to output and give a weightage to each.

Outside libraries used:

- Sklearn countvectorizer - this creates the vocabulary from a given input text. This was used to generate the vocabulary columns from the input and perform operations with these columns.
- Sklearn test train split was used to split the data into 70% training and 30% validation