Multinomial Naive Bayes:

Training on enron1 dataset

Results for enron1
Accuracy: 0.9320
Precision: 0.9275
Recall: 0.8591
F1-score: 0.8920

Training on enron2 dataset

Results for enron2
Accuracy: 0.9351
Precision: 0.9091
Recall: 0.8462
F1-score: 0.8765

Training on enron4 dataset

Results for enron4
Accuracy: 0.9687
Precision: 0.9698
Recall: 0.9872
F1-score: 0.9785

Discrete Naive Bayes:

Test Set Performance:
Accuracy: 0.9013
Precision: 0.9906
Recall: 0.7047
F1-score: 0.8235

Logistic Regression:

Processing dataset: enron1

Training with λ = 0.01
Iteration 0: Log-Likelihood = -243.1827
Iteration 100: Log-Likelihood = -4.1844
Iteration 200: Log-Likelihood = -2.5338
Iteration 300: Log-Likelihood = -1.9531
Iteration 400: Log-Likelihood = -1.6655
F1-score for λ=0.01: 0.8941

Training with λ = 0.1
Iteration 0: Log-Likelihood = -244.9118
Iteration 100: Log-Likelihood = -7.4283
Iteration 200: Log-Likelihood = -6.2140
Iteration 300: Log-Likelihood = -5.8135
Iteration 400: Log-Likelihood = -5.6027
F1-score for λ=0.1: 0.8941

Training with λ = 1
Iteration 0: Log-Likelihood = -262.2027
Iteration 100: Log-Likelihood = -22.1016
Iteration 200: Log-Likelihood = -20.6634
Iteration 300: Log-Likelihood = -20.5250
Iteration 400: Log-Likelihood = -20.5185
F1-score for λ=1: 0.8571

Training with λ = 10
Iteration 0: Log-Likelihood = -435.1118
Iteration 100: Log-Likelihood = -60.6955
Iteration 200: Log-Likelihood = -60.6955
Iteration 300: Log-Likelihood = -60.6955
Iteration 400: Log-Likelihood = -60.6955
F1-score for λ=10: 0.8889

Best λ: 0.01 with F1-score: 0.8941
Iteration 0: Log-Likelihood = -478.6928
Iteration 100: Log-Likelihood = -5.4579
Iteration 200: Log-Likelihood = -3.3377
Iteration 300: Log-Likelihood = -2.5907
Iteration 400: Log-Likelihood = -2.2203

Results for enron1 (bow representation):
Accuracy: 0.9583
Precision: 0.9062
Recall: 0.9732
F1-score: 0.9385

Training with λ = 0.01
Iteration 0: Log-Likelihood = -123.2335
Iteration 100: Log-Likelihood = -5.8988
Iteration 200: Log-Likelihood = -3.4078
Iteration 300: Log-Likelihood = -2.5564
Iteration 400: Log-Likelihood = -2.1417
F1-score for λ=0.01: 0.9630

Training with λ = 0.1
Iteration 0: Log-Likelihood = -123.4203
Iteration 100: Log-Likelihood = -8.8273
Iteration 200: Log-Likelihood = -7.2598
Iteration 300: Log-Likelihood = -6.9001
Iteration 400: Log-Likelihood = -6.7813

F1-score for λ=0.1: 0.9630

Training with λ = 1
Iteration 0: Log-Likelihood = -125.2889
Iteration 100: Log-Likelihood = -27.2217
Iteration 200: Log-Likelihood = -27.1360
Iteration 300: Log-Likelihood = -27.1321
Iteration 400: Log-Likelihood = -27.1318
F1-score for λ=1: 0.9630

Training with λ = 10
Iteration 0: Log-Likelihood = -143.9748
Iteration 100: Log-Likelihood = -78.3121
Iteration 200: Log-Likelihood = -78.3121
Iteration 300: Log-Likelihood = -78.3121
Iteration 400: Log-Likelihood = -78.3121
F1-score for λ=10: 0.9091

Best λ: 0.01 with F1-score: 0.9630
Iteration 0: Log-Likelihood = -208.2585
Iteration 100: Log-Likelihood = -7.0405
Iteration 200: Log-Likelihood = -4.0790
Iteration 300: Log-Likelihood = -3.0730
Iteration 400: Log-Likelihood = -2.5844

Results for enron1 (bernoulli representation):
Accuracy: 0.9583
Precision: 0.9221
Recall: 0.9530
F1-score: 0.9373

Processing dataset: enron2

Training with λ = 0.01
Iteration 0: Log-Likelihood = -167.5701
Iteration 100: Log-Likelihood = -6.2887
Iteration 200: Log-Likelihood = -3.9561
Iteration 300: Log-Likelihood = -3.0781
Iteration 400: Log-Likelihood = -2.6249
F1-score for λ=0.01: 0.9538

Training with λ = 0.1
Iteration 0: Log-Likelihood = -169.3095
Iteration 100: Log-Likelihood = -9.7804
Iteration 200: Log-Likelihood = -8.0364
Iteration 300: Log-Likelihood = -7.4520
Iteration 400: Log-Likelihood = -7.1601
F1-score for λ=0.1: 0.9394

Training with λ = 1
Iteration 0: Log-Likelihood = -186.7039
Iteration 100: Log-Likelihood = -25.5403
Iteration 200: Log-Likelihood = -23.8745
Iteration 300: Log-Likelihood = -23.6900
Iteration 400: Log-Likelihood = -23.6752
F1-score for λ=1: 0.9231

Training with λ = 10
Iteration 0: Log-Likelihood = -360.6479
Iteration 100: Log-Likelihood = -65.9558
Iteration 200: Log-Likelihood = -65.9558
Iteration 300: Log-Likelihood = -65.9558
Iteration 400: Log-Likelihood = -65.9558
F1-score for λ=10: 0.8667

Best λ: 0.01 with F1-score: 0.9538
Iteration 0: Log-Likelihood = -396.6470
Iteration 100: Log-Likelihood = -7.7300
Iteration 200: Log-Likelihood = -4.9763
Iteration 300: Log-Likelihood = -3.9291
Iteration 400: Log-Likelihood = -3.3855

Results for enron2 (bow representation):
Accuracy: 0.9519
Precision: 0.8963
Recall: 0.9308
F1-score: 0.9132

Training with λ = 0.01
Iteration 0: Log-Likelihood = -120.7644
Iteration 100: Log-Likelihood = -6.7985
Iteration 200: Log-Likelihood = -4.2010
Iteration 300: Log-Likelihood = -3.2582
Iteration 400: Log-Likelihood = -2.7766
F1-score for λ=0.01: 0.9333

Training with λ = 0.1
Iteration 0: Log-Likelihood = -120.9552
Iteration 100: Log-Likelihood = -9.6300
Iteration 200: Log-Likelihood = -7.9616
Iteration 300: Log-Likelihood = -7.5379
Iteration 400: Log-Likelihood = -7.3840
F1-score for λ=0.1: 0.9189

Training with λ = 1
Iteration 0: Log-Likelihood = -122.8636
Iteration 100: Log-Likelihood = -27.4583
Iteration 200: Log-Likelihood = -27.3644

Iteration 300: Log-Likelihood = -27.3603
Iteration 400: Log-Likelihood = -27.3600
F1-score for λ=1: 0.9167

Training with λ = 10
Iteration 0: Log-Likelihood = -141.9474
Iteration 100: Log-Likelihood = -77.9907
Iteration 200: Log-Likelihood = -77.9907
Iteration 300: Log-Likelihood = -77.9907
Iteration 400: Log-Likelihood = -77.9907
F1-score for λ=10: 0.8657

Best λ: 0.01 with F1-score: 0.9333
Iteration 0: Log-Likelihood = -212.9513
Iteration 100: Log-Likelihood = -9.1320
Iteration 200: Log-Likelihood = -5.6136
Iteration 300: Log-Likelihood = -4.3237
Iteration 400: Log-Likelihood = -3.6672

Results for enron2 (bernoulli representation):
Accuracy: 0.9498
Precision: 0.9077
Recall: 0.9077
F1-score: 0.9077

Processing dataset: enron4

Training with λ = 0.01
Iteration 0: Log-Likelihood = -225.9457
Iteration 100: Log-Likelihood = -3.5305
Iteration 200: Log-Likelihood = -2.0956
Iteration 300: Log-Likelihood = -1.6194
Iteration 400: Log-Likelihood = -1.3903
F1-score for λ=0.01: 0.9798

Training with λ = 0.1
Iteration 0: Log-Likelihood = -227.0083
Iteration 100: Log-Likelihood = -6.5138
Iteration 200: Log-Likelihood = -5.5225
Iteration 300: Log-Likelihood = -5.2305
Iteration 400: Log-Likelihood = -5.0839
F1-score for λ=0.1: 0.9798

Training with λ = 1
Iteration 0: Log-Likelihood = -237.6343
Iteration 100: Log-Likelihood = -21.7123
Iteration 200: Log-Likelihood = -20.7580
Iteration 300: Log-Likelihood = -20.6659
Iteration 400: Log-Likelihood = -20.6578
F1-score for λ=1: 0.9758

Training with λ = 10
Iteration 0: Log-Likelihood = -343.8945
Iteration 100: Log-Likelihood = -67.5273
Iteration 200: Log-Likelihood = -67.5273
Iteration 300: Log-Likelihood = -67.5273
Iteration 400: Log-Likelihood = -67.5273
F1-score for λ=10: 0.9680

Best λ: 0.01 with F1-score: 0.9798
Iteration 0: Log-Likelihood = -513.1457
Iteration 100: Log-Likelihood = -4.1940
Iteration 200: Log-Likelihood = -2.5208
Iteration 300: Log-Likelihood = -1.9624
Iteration 400: Log-Likelihood = -1.6926

Results for enron4 (bow representation):
Accuracy: 0.9595
Precision: 0.9467
Recall: 1.0000
F1-score: 0.9726

Training with λ = 0.01
Iteration 0: Log-Likelihood = -169.0172
Iteration 100: Log-Likelihood = -4.3566
Iteration 200: Log-Likelihood = -2.5046
Iteration 300: Log-Likelihood = -1.9049
Iteration 400: Log-Likelihood = -1.6199
F1-score for λ=0.01: 0.9808

Training with λ = 0.1
Iteration 0: Log-Likelihood = -169.3462
Iteration 100: Log-Likelihood = -7.2094
Iteration 200: Log-Likelihood = -6.0944
Iteration 300: Log-Likelihood = -5.8622
Iteration 400: Log-Likelihood = -5.7882
F1-score for λ=0.1: 0.9808

Training with λ = 1
Iteration 0: Log-Likelihood = -172.6365
Iteration 100: Log-Likelihood = -25.1857
Iteration 200: Log-Likelihood = -25.1145
Iteration 300: Log-Likelihood = -25.1121
Iteration 400: Log-Likelihood = -25.1120
F1-score for λ=1: 0.9808

Training with λ = 10
Iteration 0: Log-Likelihood = -205.5387

Iteration 100: Log-Likelihood = -80.1764
Iteration 200: Log-Likelihood = -80.1764
Iteration 300: Log-Likelihood = -80.1764
Iteration 400: Log-Likelihood = -80.1764
F1-score for λ=10: 0.9808

Best λ: 0.01 with F1-score: 0.9808
Iteration 0: Log-Likelihood = -285.3286
Iteration 100: Log-Likelihood = -5.5128
Iteration 200: Log-Likelihood = -3.1834
Iteration 300: Log-Likelihood = -2.4233
Iteration 400: Log-Likelihood = -2.0613

Results for enron4 (bernoulli representation):
Accuracy: 0.9705
Precision: 0.9607
Recall: 1.0000
F1-score: 0.9799

Each classifier's performance was optimized through hyperparameter tuning; for Naive Bayes classifiers, the main hyperparameter, alpha, regulates Laplace smoothing; for Logistic Regression, the key hyperparameter is the regularization strength (C), which was varied across a logarithmic scale from 0.01 to 10; the optimization solver used was 'lbfgs'; and the maximum number of iterations was set to 1000 to ensure convergence.

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

The combination of the Bag of Words representation with Logistic Regression performed the best. This method produced the best accuracy and F1-score, most likely as a result of the model's capacity to identify subtle word associations in the dataset. Logistic regression can model more intricate interactions, improving predictive performance, whereas Naive Bayes assumes that features are independent.

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

No, using the Bag of Words representation, Multinomial Naive Bayes did not perform better than Logistic Regression. Multinomial Naive Bayes assumes that word frequencies are conditionally independent given the class, which makes it ideal for text classification tasks. In contrast, logistic regression is more effective for the given dataset because it does not rely on this assumption and is capable of learning more intricate decision boundaries.

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

No, Discrete Naive Bayes (Bernoulli Naive Bayes) did not outperform Logistic Regression on the Bernoulli representation. Although Bernoulli Naive Bayes is designed for binary feature representations, it struggles with correlated features, which are common in natural language processing tasks. Logistic Regression is more robust in handling such dependencies and thus demonstrated superior performance in terms of accuracy and F1-score.