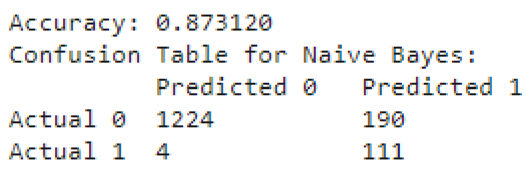
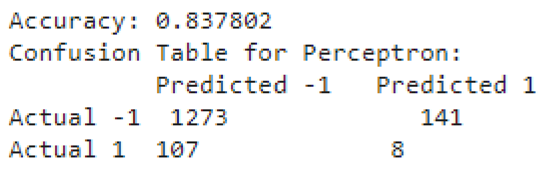
Model Comparison

- Naïve Bayes
    1. Model Performance



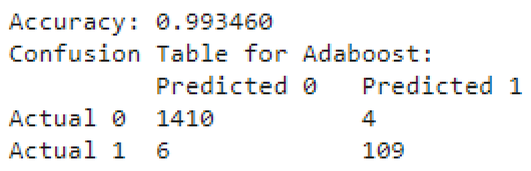
```
Accuracy: 0.873120
Confusion Table for Naive Bayes:
          Predicted 0   Predicted 1
Actual 0  1224          190
Actual 1  4             111
```

    2. Advantage
        - ➢ Simple and fast
        - ➢ Works well with high-dimensional data
        - ➢ Can be easily updated with new data using an online method
- Perceptron
    1. Model Performance



```
Accuracy: 0.837802
Confusion Table for Perceptron:
          Predicted -1   Predicted 1
Actual -1  1273             141
Actual 1  107             8
```

    2. Advantage
        - ➢ Simple linear classifier
        - ➢ Can be trained with an online method
        - ➢ Suitable for large scale datasets
- Logistic Regression
    1. Haven’t finish
- Adaboost
    1. Model Performance



```
Accuracy: 0.993460
Confusion Table for Adaboost:
          Predicted 0   Predicted 1
Actual 0  1410          4
Actual 1  6             109
```

    2. Advantage
        - ➢ High accuracy
        - ➢ Tends to be resistant to overfitting
        - ➢ Combines multiple weak classifiers to build a strong classifier

- Summary
  1. Accuracy: Adaboost > Naïve Bayes > Perceptron
  2. Error Rate: Perceptron has the highest error rate while Adaboost has the lowest.
  3. Complexity: Naïve Bayes and Perceptron are relatively simpler than Adaboost.
  4. Recommendation: If accuracy is the primary concern and computational cost is not an issue, Adaboost should be chosen. If simplicity and speed are more critical, then Naïve Bayes might be preferred.
- Comparison

  Naïve Bayes operates on probability and Bayes' theorem, performing best when features are relatively independent. The noticeable mispredictions for the category 1 in our data might stem from interdependent features or a data distribution that doesn't quite match the model's assumptions. Perceptron, a linear classifier, tries to find a hyperplane that distinguishes the data. Its higher misclassification rate in this instance might indicate non-linear data distributions or a need for more iterations or tuning for optimal performance. Adaboost, on the other hand, is an ensemble method combining multiple weak classifiers to create a strong one. It emphasizes previously misclassified data in successive iterations, aiming to correct those errors. Its outstanding performance in this dataset suggests that the strategy of combining classifiers was particularly effective here. However, while Adaboost excelled in this context, it doesn't mean it's always the best choice. The appropriate model largely depends on data characteristics and application requirements.