Atharv Karkar

Foundation of Machine Learning

Course Instructor : Arun Rajkumar

Report on Spam Classification Model Using Naive Bayes with Laplace Smoothing

1. Overview of Data

The code for training is given in SpamHamCV.ipynb

The dataset used for this analysis is an email spam classification dataset with 5,172 rows and 3,002 columns, where each row represents an email. The last column (`Prediction`) is the target variable indicating whether an email is spam (1) or ham (0). The remaining columns are numerical counts of specific words/features within each email.

2. Dataset Preparation and Splitting

1. **Data Splitting**: The dataset was split into an 80% training set and a 20% test set:
2. Training data: 4,137 rows and 3,002 columns.
3. Test data: 1,035 rows and 3,002 columns.

3. Probability Calculation

$$P(\hat{y}^{test} = 1|\mathbf{x}^{test}) \geq P(\hat{y}^{test} = 0|\mathbf{x}^{test})$$

If the above inequality holds, then $\hat{y}^{test} = 1$, otherwise $\hat{y}^{test} = 0$.

Using Bayes' rule, we can express $P(\hat{y}^{test} = 1|\mathbf{x}^{test})$ and $P(\hat{y}^{test} = 0|\mathbf{x}^{test})$ as follows:

$$P(\hat{y}^{test} = 1|\mathbf{x}^{test}) = \frac{P(\mathbf{x}^{test}|\hat{y}^{test} = 1) * P(\hat{y}^{test} = 1)}{P(\mathbf{x}^{test})}$$
$$P(\hat{y}^{test} = 0|\mathbf{x}^{test}) = \frac{P(\mathbf{x}^{test}|\hat{y}^{test} = 0) * P(\hat{y}^{test} = 0)}{P(\mathbf{x}^{test})}$$

1. **Class Priors**: Probability of spam and ham emails in the training set:
2. Probability of spam: (P(\text{spam}) = 0.287)
3. Probability of ham: (P(\text{ham}) = 0.713)
4. **Word Counts and Laplace Smoothing**:
5. To avoid zero probabilities, Laplace smoothing (with (\alpha = 1)) was applied to word counts, initializing each word count as 1. This technique helps in handling previously unseen words in new emails.
6. Vocabulary size (3,000) was derived by excluding the `Email No.` and `Prediction` columns.

4. Probability of Words Given Spam or Ham

1. Probabilities for each word given that an email is spam or ham were computed using Laplace smoothing:

2. **Spam**: (P(\text{word}_i|\text{spam}) = \frac{\text{word count in spam}}{\text{total word count in spam}})
3. **Ham**: (P(\text{word}_i|\text{ham}) = \frac{\text{word count in ham}}{\text{total word count in ham}})

$$\hat{p}_j^y = \frac{\sum_{i=1}^{n} \mathbb{1}(f_j^i = 1, y_i = y)}{\sum_{i=1}^{n} \mathbb{1}(y_i = y)} \quad \text{for all } j \in \{1, 2, \ldots, d\}, \text{ and } \forall y \in \{0, 1\}$$

4. This resulted in log probabilities for each word in both categories to facilitate calculations.

$$\sum_{i=1}^{d} \left(f_i \log \left(\frac{\hat{p}_i^1 (1 - \hat{p}_i^0)}{\hat{p}_i^0 (1 - \hat{p}_i^1)} \right) \right) + \sum_{i=1}^{d} \log \left(\frac{1 - \hat{p}_i^1}{1 - \hat{p}_i^0} \right) + \log \left(\frac{\hat{p}}{1 - \hat{p}} \right) \geq 0$$

5. Prediction on Test Set

For each email in the test set:

1. The conditional probabilities (P(X | \text{spam})) and (P(X | \text{ham})) were calculated.
2. For words that do not appear, the complementary probability (1 - P(\text{word})) was used.

3. The final spam or ham probability for each email was computed by adding log probabilities for words, adjusted by the class priors (P(\text{spam})) and (P(\text{ham})).
4. Classification was made by comparing (P(X|\text{spam})) and (P(X|\text{ham})), assigning the label with the higher probability.

6. Model Performance Metrics

The model's performance on the test set was evaluated using several metrics:

1. **Accuracy**: (93.62%)
2. **Precision**: (85.63%) for spam
3. **Recall**: (94.90%) for spam
4. **F1-Score**: (90.03%) for spam

The confusion matrix is as follows:

Predicted\Truth	Ham (0)	Spam (1)
Ham (0)	671	50
Spam (1)	16	298

These metrics indicate that the model is effective, with a high recall for spam (ensuring most spam emails are correctly identified) and a strong precision score.

7. Confusion Matrix Visualization

A heatmap of the confusion matrix was generated to visualize classification accuracy across categories. The matrix shows a low false-negative rate, indicating that few spam emails were misclassified as ham.

Testing Code Report for Email Spam Classifier Using Naive Bayes with Laplace Smoothing

1. Overview

The code for testing is given in NBClassifier_test_code.ipynb

This testing script extends the initial training process by:

1. Evaluating the model's performance on additional email content from external text files.
2. Calculating performance metrics (e.g., accuracy, precision, recall, F1-score) on a balanced test dataset of synthetic spam and ham emails.

2. Function Breakdown

1. `test_concat()`:
2. This function aligns and appends `word_counts` data with the main training dataset to handle cases where test data requires word counts from different email content.
3. `predict_email()`:
4. This function predicts whether a single email is spam or ham. It calculates the log probabilities for both spam and ham classes, taking into account the words found in the email content.

5. It uses the pre-calculated log probabilities for words in each category (`log_p_words_spam`, `log_p_words_ham`, etc.).
6. The function returns a binary classification: `1` for spam and `0` for ham.
7. **`split_words()` and `word_count()`**:
8. `split_words()` cleans the text by converting it to lowercase and removing special characters, preparing it for tokenization.
9. `word_count()` generates a dictionary of word counts from the email text, enabling the calculation of probabilities based on word frequencies.
10. **`process_and_predict()`**:
11. This function processes multiple emails from a specified folder and predicts the class (spam or ham) for each email file.
12. It iterates over `.txt` files in the given directory, reads content, and applies `predict_email()` to classify each email.
13. The function returns a dictionary of predictions and a list of predicted labels for the emails.

3. Test Setup

1. A synthetic test dataset was created:
2. `test_email_spam`: An array of 250 ones (indicating spam).
3. `test_email_ham`: An array of 250 zeros (indicating ham).
4. `test_email_labels`: A concatenated array of spam and ham labels, totaling 500 emails for evaluation.

4. Evaluation Results

1. The model was tested on the synthetic dataset with labels (250 ham and 250 spam emails), achieving the following metrics:
2. **Accuracy**: (98%)
3. **Precision (Ham)**: (100%) (indicating perfect precision for ham emails)
4. **Recall (Ham)**: (96%)
5. **Precision (Spam)**: (96%)
6. **Recall (Spam)**: (100%)
7. **F1-Score (Both Classes)**: (98%)

5. Confusion Matrix Analysis

The confusion matrix highlights that the model is highly accurate:

1. True Negatives (Ham correctly classified): 240
2. False Positives (Ham misclassified as Spam): 10
3. True Positives (Spam correctly classified): 249
4. False Negatives (Spam misclassified as Ham): 1

6. Visualization

A heatmap of the confusion matrix was created to visually confirm classification accuracy:

1. The heatmap shows minimal misclassifications, with almost perfect results for both ham and spam categories.

7. Observations and Recommendations

1. **High Performance**: The model performed exceptionally well, indicating that the Naive Bayes approach with Laplace smoothing was effective on this dataset.
2. **False Positives**: Although minimal, a slight adjustment to reduce false positives could improve overall precision for ham.
3. **Further Testing**: Additional real-world testing on unseen, diverse data would validate the robustness of this classifier.