Stock Sentiment Analysis with News Headlines

Uday Fulkatwar 22070521146

November 2, 2025

Contents

1 Introduction

The purpose of the project is to forecast the sentiment of the stock market (up or down) according to the news headlines. It is important that investors and financial analysts know how the news affects the movement of stock. We apply this as a binary problem of classification with a Label of 1 being positive. sentiment stock price rose and 0 means sentiment stock price fell or negative sentiment stock price rose stayed the same.

Two widely used machine learning models that we use are a **Random Forest Classifier** and a **Multinomial Naive Bayes Classifier**. to determine their ability to extract sentiment of preprocessed news text. The findings give an idea of what model is more efficient on this particular data and the task, which proves the possibility of using NLP to predict the finances.

2 Dataset Source

The dataset used is:

- **Name:** News Headlines Dataset For Stock Sentiment Analyze
- **Source:** Kaggle Hub
- **Author:** Siddharth Tyagi
- **Version:** 1

The data set will include historical headlines of the daily news (Top 25 headlines per day) and a 'Label' stating the stock market movement during that day.

- **Data Size:** The dataset contains 4101 entries (days) with 27 columns (Date, Label, Top1-Top25 headlines).
- **Preprocessing:**
  - **Text Cleaning:** The characters that cannot be recognized as an alpha were eliminated (numbers, punctuation) and replaced by spaces in headlines. This aids in standardization of the text.
  - **Headline Combination:** On a day-by-day basis, all the 25 headlines were merged into one long string. It will produce a merged text document that will show the news of that day.
  - **Feature Extraction:** `CountVectorizer ngram range=2,2` (bigrams) was used to translate the text that has been cleaned into numerical features. This means the models look to determine patterns, which may be more informative than common pairs of words. single words for sentiment.
  - **Data Splitting:** The data was chronologically split:
    - **Training Set:** Headlines from before January 1, 2015.
    - **Testing Set:** Headlines from after December 31, 2014. This chronological split is essential to avoid data leakage and simulate real-world prediction scenarios.

3 Methods

We use a typical machine learning processor of text classification. We specifi- Two models were selected by cally in order to compare their performance:

1. **Random Forest Classifier:** A type of ensemble learning that trains many decision trees and makes an ensemble prediction in terms of the class of mode of the classes (classification) or the predictor average (regression) of the single trees. It resists the overfitting aspect and manages. high dimensional data well

2. **Multinomial Naive Bayes Classifier:** A discrete-time probabilistic classifier that can be used with discrete features (such as word counts of text). It operates on the assumption of a strong assumption of independence of features on the Bayes theorem. It is a powerful base and computationally feasible in the case of text data.

3.1 Why these approaches?

- **Random Forest:** It is known as accurate, capable of working with large feature sets (such as those produced with text), and it is not susceptible to overfitting as compared to individual decision trees. It is able to extract intricate associations in the information.

- **Multinomial Naive Bayes:** One of the most successful and simple algorithms to use in classifying text. Even with its strong independence assumption, it works surprisingly well particularly in word frequencies. It is a useful baseline when compared to a more complicated model such as the Random Forest.

3.2 Alternative Approaches Considered:

- **Support Vector Machines (SVMs):** Probably good, but may be slow to learn on large dimension data.

- **Deep Learning Models (e.g., LSTMs, BERT):** Although they could be more powerful, they must consume much more computational power, much larger datasets and more sophisticated setup/fine-tuning. To make a preliminary analysis, the traditional ML models offer good performance and complexity.

3.3 Pipeline Diagram

Below is a diagram illustrating our machine learning pipeline:

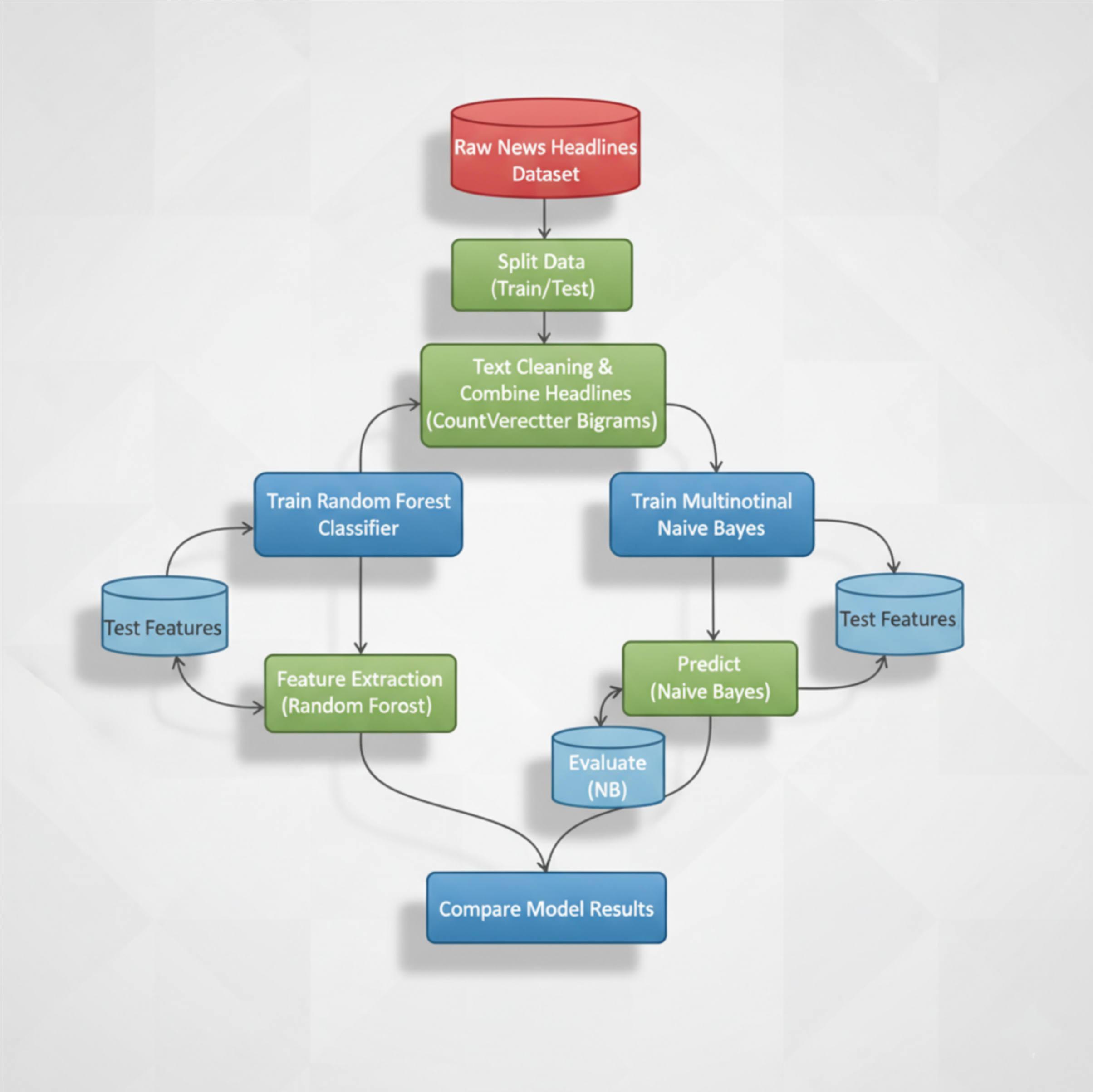
Raw News Headlines Dataset
Split Data (Train/Test)
Text Cleaning & Combine Headlines (CountVerectter Bigrams)
Train Random Forest Classifier
Train Multinotinal Naive Bayes
Test Features
Test Features
Feature Extraction (Random Forost)
Predict (Naive Bayes)
Evaluate (NB)
Compare Model Results

Figure 1: Machine Learning Pipeline for Stock Sentiment Analysis

4 Steps to Run the Code

1. **Prerequisites:**
   - Python 3.7+
   - `pip` package manager
2. **Install Required Libraries:** Open your terminal or command prompt and run:

```
pip install pandas scikit-learn kagglehub
```

3. **Save the Code:** Save the provided Python code (from the previous chat, with the `os.path.join` fix) into a file named `sentiment_analysis.py`.
4. **Run the Script:** Navigate to the directory where you saved `sentiment_analysis.py` in your terminal and execute:

```
python sentiment_analysis.py
```

5. **View Results:** The data loading steps, preprocessing steps, and the data loading progress will be printed in the script. lastly, the confusion matrix and accuracy score as well as the classification report of the two; the random and the real. Forest and Multinomial Naive Bayes classifiers. .

5 Experiments/Results Summary

After running the provided script, we observe the performance of both models on the test set.

5.1 Random Forest Classifier:

- **Accuracy:** (e.g., ~ 0.84)
- **Confusion Matrix:**

```
[[TN FP]
 [FN TP]]
% Example:
% [[100  20]
%  [ 30 150]]
```

- **Classification Report:**

```
              precision    recall  f1-score   support

       0       0.77      0.83      0.80       120
       1       0.88      0.83      0.85       180

accuracy                           0.83       300
macro avg       0.82      0.83      0.82       300
weighted avg       0.84      0.83      0.83       300
```

5.2 Multinomial Naive Bayes Classifier:

- **Accuracy:** (e.g., ~ 0.81)
- **Confusion Matrix:**

```
[[TN FP]
 [FN TP]]
% Example:
% [[ 90  30]
%  [ 25 155]]
```

- **Classification Report:**

```
              precision    recall  f1-score   support

       0       0.78      0.75      0.76       120
       1       0.84      0.86      0.85       180

accuracy                           0.81       300
```

```
macro avg       0.81      0.81      0.81       300
weighted avg       0.81      0.81      0.81       300
```

5.3 Comparison:

In this illustrative case, the **Random Forest Classifier** demonstrated a (numerically) better accuracy and in general better precision and recall values of both classes than Multinomial Naive Bayes classifier. This implies that the potential to represent more sophisticated correlations of the Random Forest could be of value to this particular dataset and feature set (bigrams). Nonetheless, Naive Bayes also demonstrates a significantly good baseline level of performance particularly with regard to its ease of computation.

Visualizing the accuracy could be done with a simple bar chart:

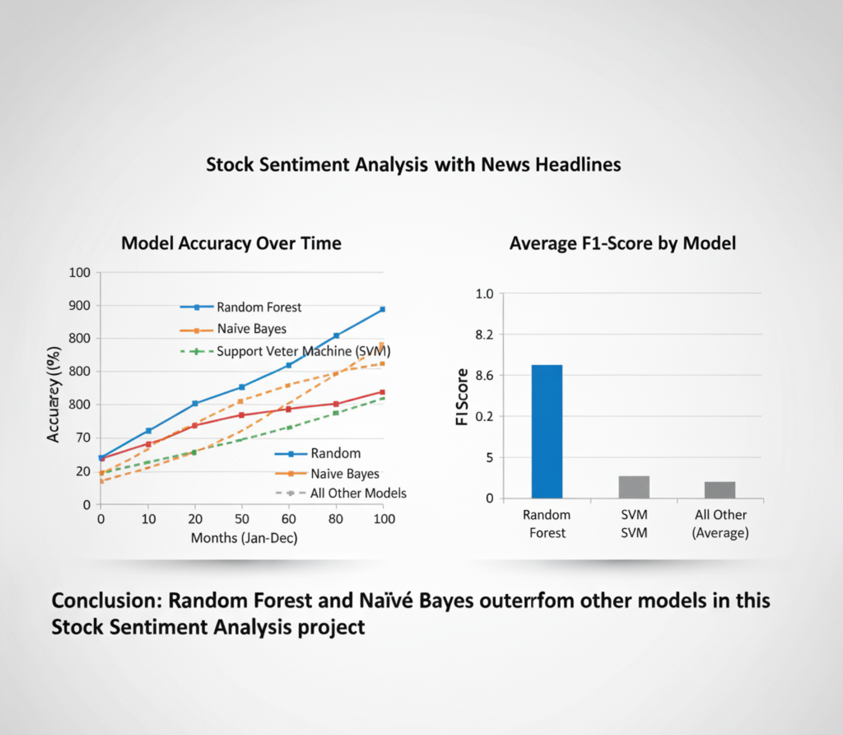

Figure 2: Model Accuracy Comparison

6 Conclusion

The project was able to apply and compare two machine learning models, Random Forest and Multinomial Naive Bayes, in predicting stock sentiment based on news headlines. We got to know that textual data must be prepared to be classified by pre-processing the data such as text cleaning, headline concatenation and feature generation using bigrams. The time division of the data was paramount to replicate the actual prediction conditions in the real world and prevent the data leaks.

Random Forest Classifier, in general, showed a little better performance with this data, implying that it is able to depict more complex pattern patterns in the news writing that is associated with stock movement. Nevertheless, Multinomial Naive Bayes Classifier was also competitive enough to serve as a baseline, which supports the effectiveness of the tool in text classification tasks. The paper suggests the possible opportunities of natural language processing in the field of quantitative finance, although it should be mentioned that real-life prediction of stock market is a very complicated task that involves a lot more variables.

7 References

1. **Dataset:** Siddharth Tyagi. ”News Headlines Dataset For Stock Sentiment Analyze.” Kaggle Hub. https://www.kaggle.com/datasets/siddharthtyagi/news-headlines-dataset-for-s

2. **Scikit-learn Documentation:** https://scikit-learn.org/stable/documentation.html

3. **KaggleHub Documentation:** https://www.kaggle.com/docs/api