21MAT301 MATHS PROJECT REPORT

Review Prediction Using Naive Bayes Filter

GROUP MEMBERS

Parvathy G Pillai	**AM.EN.U4AIE21150**
S Anand	**AM.EN.U4AIE21155**

Introduction

The problem we are trying to address is that of review prediction. Our model predicts the nature of the review data of Amazon food reviews using Naive Bayes algorithm which is based on Bayes probability model. The methodology of our project is given below:

To build a generalized prediction model, the first step should be necessary cleaning of data as a part of data preprocessing.

We will perform the following data preprocessing steps.

- Removing Stop-words
- Remove any punctuations or limited set of special characters like , or . or # etc.
- Snowball Stemming the word
- Convert the word to lowercase

Once the data is cleaned to be processed we'll use below Feature generation techniques to convert text to numeric vector.

- Bag Of Words (BoW)
- Term Frequency - inverse document frequency (tf-idf)

Using Naive Bayes algorithm we will build model to predict review polarity for each technique.

Objective: Given a review determine whether a review is positive or negative, by applying Naive Bayes algorithm and deciding the best Feature generation technique with most important features for positive & negative class. We will generate ROC curve for each model to check the sensibility of model.

Mathematical Formulation

Let's define:

- X = vector of features extracted from the text (either BoW or TF-IDF)
- C = set of classes (positive or negative sentiment)
- P(C) = prior probability of class C
- P(X|C) = likelihood of feature vector X given class C
- P(C|X) = posterior probability of class C given features X

Naive Bayes uses Bayes' theorem to calculate the posterior:

- $P(C|X) = P(X|C) * P(C) / P(X)$

Since P(X) is constant for the given X, this simplifies to:

- $P(C|X) \propto P(X|C) * P(C)$

The priors P(C) are estimated by:

- P(C) = Count(documents in class C) / Total documents

The likelihoods P(X|C) are estimated by:

- $P(X|C) = \Pi P(x_i|C)$ (Naive assumption of independence between features)

Where P(xi|C) is calculated by:

- P(xi|C) = (Count(x_i in documents of class C) + α) / (Total term counts in class C + α*|Vocabulary|)

α is the smoothing parameter found by cross-validation.
To make a prediction, we calculate P(C|X) for each class and predict the class with maximum posterior probability.
The accuracy and ROC curve evaluate model performance by comparing the predicted vs true labels.
In summary, Naive Bayes applies Bayes' rule with strong independence assumptions between features to perform probabilistic classification.

Advantages of Naive Bayes model over other models

- Works quickly and saves a lot of time
- Can work on small data and doesn’t need a large dataset to predict accurately
- Most of the time, Naive Bayes finds its uses in-text classification due to its assumption of independence and high performance in solving multi-class problems. It enjoys a high rate of success than other algorithms due to its speed and efficiency.

Pseudocode

Step 1: Data Collection and Preparation

- Assuming we have collected and preprocessed the review data

Step 2: Split Data into Training and Test Sets

- Split the preprocessed data into training and test sets

Step 3: Feature Generation

- Implement text vectorization techniques like Bag of Words (BoW) or tf-idf

Step 4: Naive Bayes Model Training

- Train the Naive Bayes classifier on the training data
  NaiveBayes.train(training_data)

Step 5: Model Evaluation

- Use the trained model to predict sentiment on the test data
  predictions = NaiveBayes.predict(test_data)

Step 6: Evaluation Metrics

- Calculate evaluation metrics (accuracy, precision, recall, F1-score)
  metrics = calculate_metrics(test_labels, predictions)

Step 7: Analyze and Improve

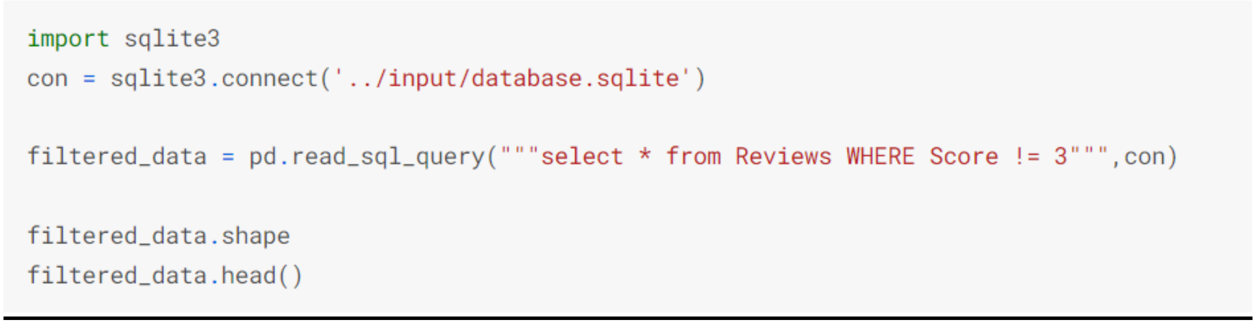
- Analyze misclassified instances and model weaknesses
- Explore enhancements using more advanced techniques like word embeddings or deep learning

Step 8: Final Model Deployment (if satisfactory)

- Deploy the improved sentiment analysis model for customer review classification

Sample Intermediate Results

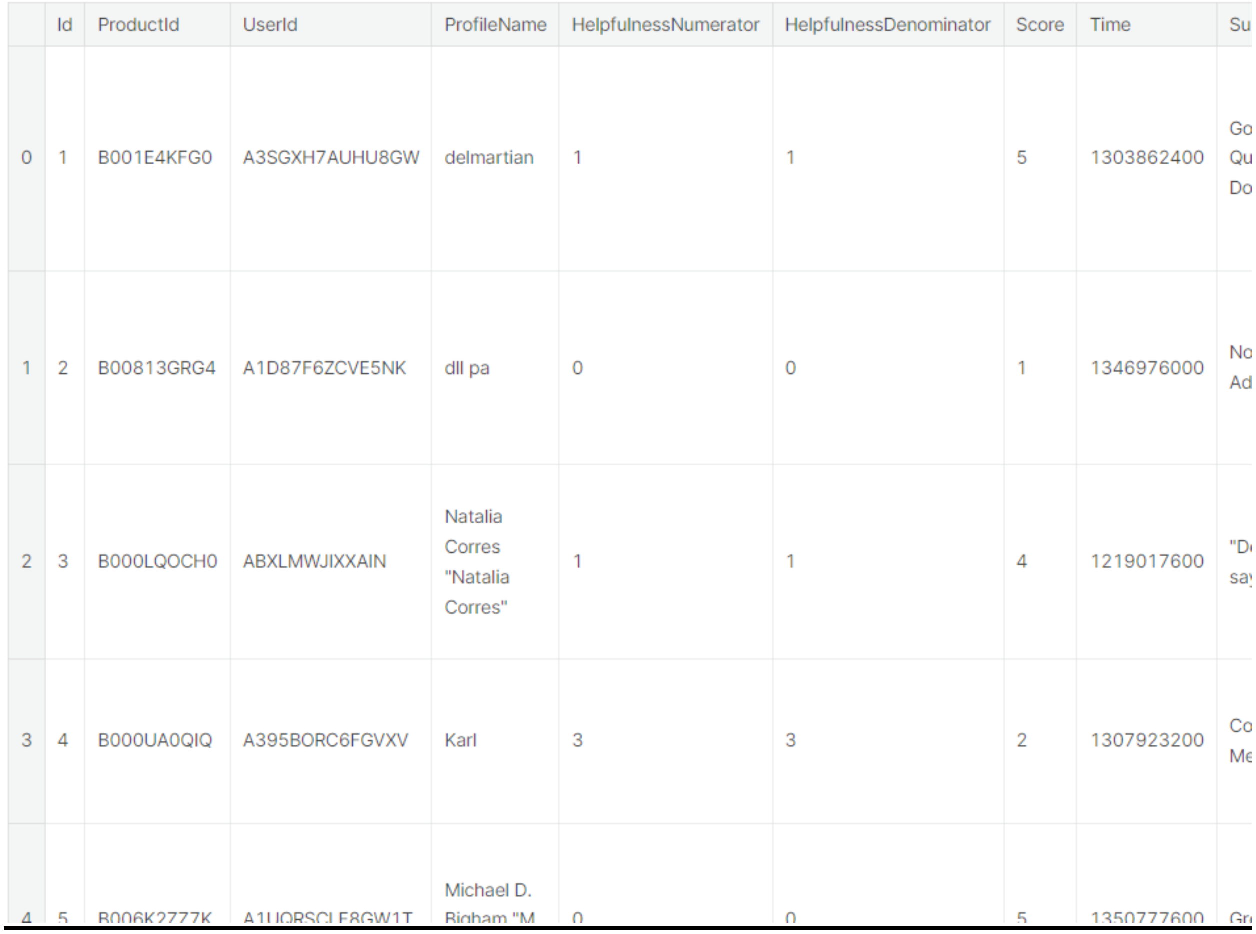
- **Data preprocessing and visualization**



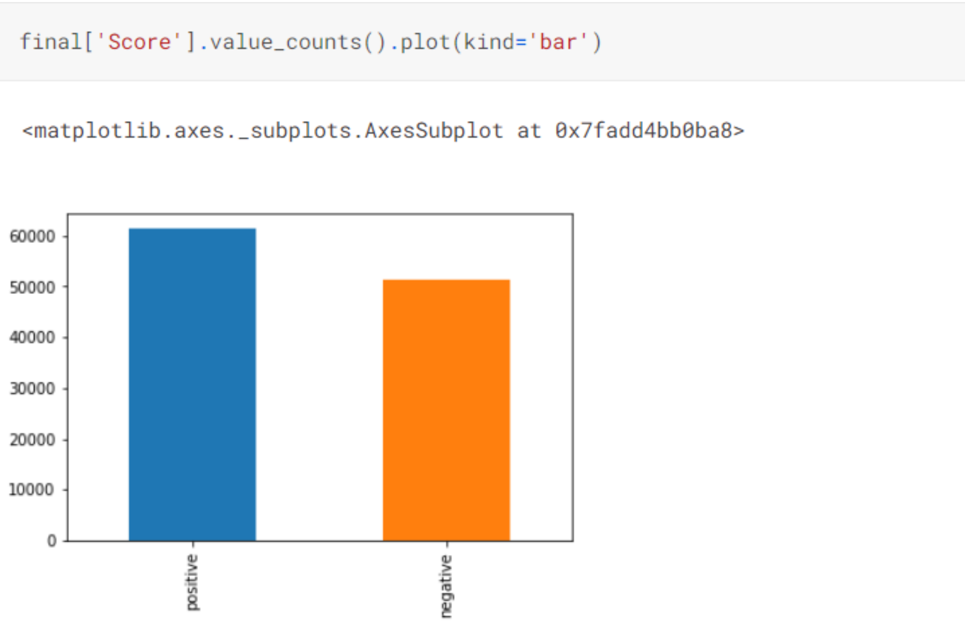
```
import sqlite3
con = sqlite3.connect('../input/database.sqlite')

filtered_data = pd.read_sql_query("""select * from Reviews WHERE Score != 3""",con)

filtered_data.shape
filtered_data.head()
```



	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time	Su
0	1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1	5	1303862400	Go Qu Do
1	2	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0	0	1	1346976000	No Ad
2	3	B000LQOCH0	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	1	1	4	1219017600	"D say
3	4	B000UA0QIQ	A395BORC6FGVXV	Karl	3	3	2	1307923200	Co Me
4	5	B006K2ZZ7K	A1UQRSCLF8GW1T	Michael D. Bigham "M	0	0	5	1350777600	Gr



```
final['Score'].value_counts().plot(kind='bar')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fadd4bb0ba8>
```

****