Capstone Project Report:-

**Sentiment analysis and Text classification**

Table of Contents

1. Problem Statement

2. Project Objective

3. Data Description

4. Data Pre-processing Steps and Inspiration

5. Choosing the Algorithm for the Project

6. Motivation and Reasons For Choosing the Algorithm

7. Assumptions

8. Model Evaluation and Techniques

9. Inferences from the Same

10. Future Possibilities of the Project

11. Conclusion

12. References

Problem Statement:

You are working in an e-commerce company, tasked with analysing customer reviews for various products and creating a report that classifies customer reviews base on the sentiment associated with each review.

Project Objective

The main objective of this project is to analyse customer reviews to:

* Find various trends and patterns in the reviews data.
* Create useful insights that best describe the product quality.
* Classify each review based on the sentiment associated with it.

Data Description

The dataset used in this project is "**Reviews.csv**", which contains the following columns:

|  |  |
| --- | --- |
| **Feature Name** | **Description** |
| Id | Record ID |
| ProductId | Product ID |
| UserId | User ID who posted the review |
| ProfileName | Profile name of the User |
| HelpfulnessNumerator | Numerator of the helpfulness of the review |
| HelpfulnessDenominator | Denominator of the helpfulness of the review |
| Score | Product Rating |
| Time | Review time in timestamp |
| Summary | Summary of the review |
| Text | Actual text of the review |

The dataset comprises 568454 rows and 10 columns.

Data Pre-processing Steps and Inspiration:

In this section, we describe the crucial data exploration and pre-processing steps undertaken in our analysis. These steps are essential for ensuring the reliability and quality of the dataset and for deriving meaningful insights from the data.

**Reading the Dataset and Handling Missing Values:**

* We initiated our analysis by importing the dataset from the "**Reviews.csv**" file using the Pandas library. To ensure that our analysis is based on complete and clean data, we checked for missing values in the dataset.
* To address the issue of missing values, we opted to drop the rows with missing data by using the **dropna()** method. This ensured that our subsequent analysis was based on a complete dataset, free of any missing values.

**Converting the 'Time' Column to Datetime:**

* The '**Time**' column in the dataset initially contained timestamp values. We recognized the importance of converting these timestamp values into a more interpretable format. To achieve this, we applied a conversion using the datetime library in Python, resulting in human-readable date and time information.

**Visualizing the Data with Box Plots and Count Plots:**

* A crucial part of our data exploration process involved visualizing the dataset to identify patterns and outliers. We used Matplotlib and Seaborn libraries to create box plots, which provided insights into the distribution of data, helping us identify potential outliers and variations.
* Additionally, we generated count plots to visualize the distribution of product ratings (Score). These visualizations offered an initial glimpse into the distribution of customer ratings.

**Exploring the Distribution of Product Ratings and Reviews Over Time:**

* We extended our exploration by examining how product ratings and reviews were distributed over time. To achieve this, we plotted a bar graph that displayed the distribution of reviews across different years. This allowed us to understand any temporal trends in customer reviews.

**Investigating the Top-Rated Products and the Most-Reviewed Products:**

* To gain more specific insights, we investigated the top-rated products by calculating the mean product rating for each product and identifying the highest-rated ones. This helped us recognize which products had consistently positive reviews.
* Additionally, we identified the most-reviewed products, shedding light on the products that had garnered the most attention from customers.

Text Pre-processing

In this section, we describe the essential steps taken to pre-process the text data from the customer reviews. Text pre-processing is a crucial aspect of our analysis because it allows us to clean and transform the raw textual information into a format that is more suitable for machine learning and sentiment analysis.

**Why Text Pre-processing Was Required:**

* Raw text data often contains noise, inconsistencies, and elements that are not relevant to our analysis. To make this data useful for sentiment analysis and other natural language processing tasks, it is essential to perform text pre-processing. By doing so, we can:
  + Remove unnecessary elements like HTML tags, which don't contribute to sentiment analysis.
  + Standardize the text by converting it to lowercase, ensuring that the analysis is case-insensitive.
  + Tokenize the text into individual words or tokens for further analysis.
  + Remove common stop words (e.g., "the," "and") that don't carry significant sentiment information.
  + Lemmatize the words to reduce them to their base or dictionary form, allowing for more accurate sentiment analysis.

**Steps in Text Pre-processing:**

* Removing HTML Tags:
  + The initial step involved removing HTML tags from the 'Text' column of the dataset. HTML tags, often found in customer reviews, do not provide meaningful information for sentiment analysis. By using the BeautifulSoup library, we extracted the actual text content from the HTML tags.
* Lowercasing:
  + To ensure case insensitivity in our analysis, we converted all text to lowercase. This transformation ensures that, for example, "Good" and "good" are treated as the same word during analysis.
* Tokenization:
  + Tokenization is the process of breaking the text into individual words or tokens. We tokenized the text to prepare it for further analysis and to create a structured representation of the textual data.
* Stop Word Removal:
  + Common stop words, such as "the," "and," and "is," often appear in text but don't carry sentiment information. Removing these words reduces the dimensionality of the data and helps focus on more meaningful content.
* Lemmatization:
  + Lemmatization is the process of reducing words to their base or dictionary form. This step helps in capturing the core meaning of words, ensuring that variations of the same word (e.g., "running" and "ran") are treated as one word.
* Joining Tokens Back Into Clean Text:
  + After completing these pre-processing steps, we joined the processed tokens back into clean text. This final cleaned text is used for sentiment analysis and machine learning.

Text pre-processing is essential to ensure that our analysis is based on standardized and clean textual data, making it possible to accurately classify customer reviews based on sentiment and extract meaningful insights from the text.

Rating segmentation:

We segment the sentiment as per the score given by the user. It’s a valid approach and we define 3 type of sentiments.

* Positive Sentiment:
  + Reviews with scores of 4 or 5 were categorized as "Positive." The reasoning behind this is that customers who give high scores are likely satisfied with the product, and their reviews generally contain positive sentiments.
* Negative Sentiment:
  + Reviews with scores of 1 or 2 were categorized as "Negative." Customers who rate products poorly often have negative feedback or complaints. Thus, these scores are associated with negative sentiment.
* Neutral Sentiment:
  + Reviews with scores of 3 were categorized as "Neutral." A score of 3 implies that the customer's opinion might be mixed or relatively neutral. This category helps capture reviews that don't express strong positive or negative sentiments.

**Data preparation before modelling**

* First we split our data In testing and training sets
* After splitting, we vectorised the text data using Term Frequency-Inverse Document Frequency (TF-IDF) vectorization. This technique assigned numerical values to each word in the text, reflecting its importance within the dataset.

Choosing the Algorithm for the Project

**Logistic Regression:**

Logistic regression is a statistical method used for classification tasks in machine learning. While the term "regression" is part of its name, logistic regression is primarily employed as a classification algorithm, not for regression tasks. It is particularly useful when the goal is to predict binary outcomes (e.g., yes/no, 1/0) or classify data into multiple classes.

**Binary Classification:**

In binary classification, the goal is to classify data into one of two classes. For instance, determining whether an email is spam or not spam is a binary classification problem. Logistic regression is well-suited for such tasks.

**Multiclass Classification:**

Logistic regression can also be extended for multiclass classification, where the goal is to classify data into more than two classes. This is achieved through techniques like "one-vs-all" or "softmax" regression.

**Probability Estimation:**

At its core, logistic regression models the probability of an input belonging to a particular class. It calculates the probability that a given input belongs to the positive class (usually labelled as 1), and the complementary probability that it belongs to the negative class (usually labelled as 0).

**Sigmoid Function:**

Logistic regression uses the sigmoid (logistic) function to transform a linear combination of input features into a probability score. The sigmoid function "squashes" the linear combination into a value between 0 and 1, making it suitable for probability estimation.

**Decision Boundary:**

Logistic regression then uses a threshold (typically 0.5) to determine the class assignment. If the predicted probability is greater than the threshold, the input is classified as the positive class; otherwise, it's classified as the negative class.

This threshold can be adjusted to control the trade-off between precision and recall, depending on the specific requirements of the classification task.

**Model Training:**

Logistic regression is trained using labelled data, where the input features are associated with known class labels. The model learns the coefficients (weights) that define the linear combination of features and adjusts them to minimize the prediction error, usually using a method called maximum likelihood estimation.

**Interpretability:**

One of the key advantages of logistic regression is its interpretability. The model provides coefficient values for each feature, indicating their influence on the prediction. This allows for a clear understanding of which features are positively or negatively correlated with the outcome.

**Linear Decision Boundary:**

Logistic regression creates a linear decision boundary in the feature space. This means it can separate classes with a straight line (in two dimensions), a hyperplane (in higher dimensions), or a linear combination of features.

**Advantages:**

Logistic regression is simple, computationally efficient, and suitable for both small and large datasets. It is a good choice when the relationship between the features and the class labels is roughly linear.

**Limitations:**

Logistic regression may not perform well when the relationship between features and class labels is highly nonlinear, as it assumes a linear decision boundary.

**Neural Network:**

A neural network is a computer program that's inspired by the way our brains work. It's used to solve problems, make decisions, or recognize patterns in data.

* **Imagine a Network of Neurons:**
  + Think of a neural network as a network of tiny decision-making units, just like the neurons in our brains. These decision-makers are called "nodes" or "neurons."
* **Input and Output:**
  + It has input, like our senses. You can feed it information, like pictures, numbers, or text.
  + It also has output, which are the answers or decisions it gives you.
* **Learning from Examples:**
  + The cool thing about neural networks is that they can learn from examples. You show them lots of examples of something, like pictures of cats and dogs, and they learn to recognize the differences.
* **Hidden Layers:**
  + Inside a neural network, there are layers of neurons, including "hidden" layers. These hidden layers help the network understand more complex things. Think of them as layers of experts, each looking at a different aspect of the problem.
* **Adjusting the Weights:**
  + When the network makes a mistake, it adjusts itself. It changes the "weights" on the connections between neurons to get better at its job. This is like learning from your mistakes and getting better over time.
* **Deep Learning:**
  + If a neural network has many hidden layers, we call it a "deep" neural network. Deep learning is all about using these deep networks to solve really tough problems, like recognizing spoken words or self-driving cars.
* **Use Cases:**
  + Neural networks are used for all sorts of things, from recognizing faces in photos to recommending movies on streaming platforms. They can also beat humans at games like chess and Go.

Model Evaluation and Technique

Techniques Used for Model Evaluation:

**Accuracy:**

How It Works: Accuracy is a straightforward metric that calculates the ratio of correctly classified instances to the total number of instances in the dataset. It provides an overall view of the model's ability to classify data correctly.

Why it’s important: Accuracy is a useful starting point for understanding the model's performance. However, it should be considered alongside other metrics, especially in imbalanced datasets.

**Classification Report:**

How It Works: A classification report provides a detailed breakdown of performance metrics for each class in a multi-class classification problem. It includes precision, recall, F1-score, and support for each class.

Why it’s important: A classification report allows for a more granular assessment of the model's performance, helping to identify areas where it excels and where it may need improvement. It's particularly valuable in multi-class sentiment classification.

**Precision and Recall:**

How They Work: Precision measures the ratio of true positive predictions to the total positive predictions, while recall measures the ratio of true positive predictions to the total actual positives in the dataset.

Why They're Important: Precision and recall provide insights into the model's ability to make accurate positive predictions (precision) and its ability to find all actual positive instances (recall). These metrics are particularly valuable in imbalanced datasets, where one class may dominate the data.

**Confusion Matrix:**

How It Works: A confusion matrix is a table that shows the number of true positives, true negatives, false positives, and false negatives. It helps visualize the model's performance in classification.

Why it’s important: The confusion matrix provides a clear picture of where the model is making correct and incorrect predictions. It helps identify potential issues and areas for improvement.

Inferences from the Project

* We observed that the neural network is able to classify the sentiment easily and more accurately that the logistic regression. NN gives max accuracy of 97% on training data and 88% on testing data.
* It Is due to the fact that NN are able to capture non-linear relationships among the features.
* Our features and rows are very large and complex. Classical ML algorithm like Random Forest and Decision Tree fail to run. So NN becomes our first choice to be used.
* Dataset is imbalanced towards positive reviews as observed in Distribution of product ratings (Score)
* Top rated products have only 5 scores no negative or neutral sentiment

Future Possibilities

While the current project has made significant progress in classifying customer reviews based on sentiment, there are several future possibilities and alternative approaches that could further enhance its scope and capabilities:

**Advanced NLP Models:**

Leveraging state-of-the-art Natural Language Processing (NLP) models, such as BERT, GPT, or RoBERTa, could significantly improve sentiment classification accuracy. These models are pre-trained on massive text corpora and can capture complex language patterns effectively.

**Fine-Tuning Pre-trained Models:**

Instead of building sentiment classification models from scratch, fine-tuning existing pre-trained models for sentiment analysis can save time and potentially yield better results. Fine-tuning can be performed on models like BERT to adapt them to the specific domain and data.

**Aspect-Based Sentiment Analysis:**

Expanding the project to perform aspect-based sentiment analysis would provide more detailed insights. This approach involves classifying not only the overall sentiment of a review but also the sentiment towards specific aspects of a product or service (e.g., price, quality, customer service).

**Sentiment Analysis on User-Generated Content:**

Extending the sentiment analysis to social media data, forum posts, and other user-generated content can provide valuable insights for businesses. Analysing sentiment from diverse sources can enhance the understanding of customer perceptions and feedback.

Conclusion

We were successfully able to analyse and classify the reviews given by the user using a simple Bag of Words approach by using Logistic regression and Deep Neural networks

References

In the course of this project, several resources and platforms have been instrumental in acquiring knowledge and data, as well as in gaining insights into sentiment analysis and machine learning techniques. The following references have significantly contributed to the success of this endeavour:

* Stack Overflow (https://stackoverflow.com/):
  + Stack Overflow has been an invaluable resource for troubleshooting coding issues and seeking solutions to technical challenges encountered during the project. The collective knowledge and expertise of the Stack Overflow community have been a reliable support in problem-solving.
* Kaggle (https://www.kaggle.com/):
  + Kaggle, a well-known platform for data science and machine learning enthusiasts, has provided access to essential datasets, notebooks, and discussions relevant to sentiment analysis. It served as a valuable source of real-world data and insights specific to the sentiment classification project.
* Analytics Vidhya (https://www.analyticsvidhya.com/):
  + Analytics Vidhya, a prominent platform for data science and AI enthusiasts, offered a plethora of articles, tutorials, and datasets related to data pre-processing, machine learning, and sentiment analysis. These resources have been instrumental in understanding and applying key concepts in the field.
* Towards Data Science (https://towardsdatascience.com):
  + "Towards Data Science," a reputable publication on Medium, hosted numerous articles, case studies, and tutorials covering various aspects of data science and machine learning. The publication has enriched the project with valuable insights, particularly in model evaluation and techniques for sentiment analysis.
* GitHub (https://github.com/):
  + GitHub, a platform for version control and collaborative development, has been utilized for storing and sharing code repositories related to the project. It has facilitated collaboration with peers and provided a means to manage project code efficiently.