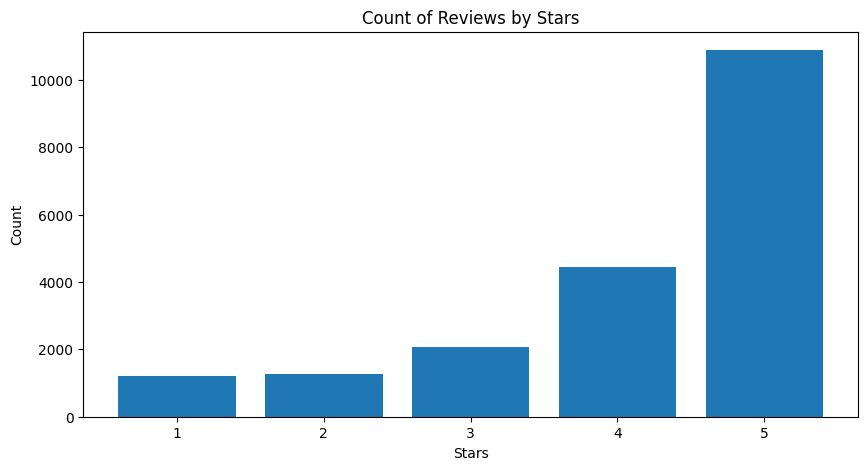
DATA MINING CA2 REPORT

1. **Introduction**

This report showcases a critical analysis of sentiment analysis conducted on Yelp restaurant reviews sourced from the dataset "Yelp Restaurant Reviews.csv". The dataset comprises customer reviews, alongside corresponding ratings on a scale of 1 to 5. A customer rating of 4 or 5 implies that he/she would recommend the restaurant, whereas ratings below 4 suggest that the customer may not recommend it. Leveraging machine learning techniques, namely the Support Vector Classifier (SVC) and Naive Bayes Classifier (NBC), this analysis aims to determine sentiment polarity in the reviews—whether they convey positive or negative sentiments—thereby facilitating informed decision-making for the dessert parlour.



1. **Text Cleaning**

The text cleaning process is essential in preparing the data for analysis. In this context, the Python code provided implements a series of steps to cleanse the reviews before converting it into a structured numerical format.

* HTML Tags Removal: The code utilizes the BeautifulSoup library to remove HTML tags from the reviews. HTML tags are often present in web-based text data and can introduce noise into the analysis. By eliminating these tags, the code ensures that only the textual content of the reviews is considered for further processing.
* URL and Special Symbols Removal: Following HTML tag removal, the code employs regular expressions to replace URLs and special characters with whitespace. This step eliminates extra information that is not relevant for the model. Special characters can twist the representation of words and should be removed to maintain the integrity of the text data.
* Tokenization and Lowercasing: The text is tokenized using the nltk.word\_tokenize() function, which splits the reviews into individual words or tokens. Additionally, the code converts all tokens to lowercase. Lowercasing ensures consistency in word representations, as it treats uppercase and lowercase versions of the same word as identical.
* Stop Word Removal: Commonly occurring words such as "the," "is," and "and" are removed from the text using the nltk.corpus.stopwords module. These words are considered non-informative for sentiment analysis and are thus excluded from the analysis to reduce noise and load.
* Lemmatization: The code employs lemmatization to reduce words to their base or root form. This process involves identifying the lemma or base form of each word based on its context. For example, it changes "running" to "run" and "better" to "good." This helps group together different forms of the same word, so the analysis isn't thrown off by variations in word endings.

It is important to note that even though the code implements an extensive text cleaning routine, it has some limitations. For example, removing special characters could cause important information to be lost, particularly when punctuation conveys emotion. Furthermore, lemmatization might not always yield accurate results, especially for irregular word forms, even though it is often effective in doing so.

Therefore, during the text preprocessing stage, it is crucial to strike a balance between thorough cleansing and the protection of meaningful language indications.

1. **Creation of Structured Data**

Once the text has been cleaned, the next step involves turning it into a format that a machine can understand and analyse. This is crucial for applying machine learning techniques effectively. The method used for this conversion in the code is called TF-IDF (Term Frequency-Inverse Document Frequency).

TF-IDF is a technique commonly used in natural language processing. It essentially allocates a numerical value to each word in a document, based on how important that word is in the document and how unique it is across all the documents in the dataset.

* Term Frequency (TF): This measures how often a word appears in a document. If a word appears many times, it's likely to be important in that document.
* Inverse Document Frequency (IDF): This measures how unique a word is across all documents. If a word appears in many instances, it's less likely to be unique and therefore less important in distinguishing one from another.

By combining these two, TF-IDF assigns a score to each word-document pair, indicating its importance. Words that appear frequently in a document but rarely across all documents will have high TF-IDF scores, indicating their significance in that document.

In the code, the TfidfVectorizer class from the sklearn.feature\_extraction.text is used to perform this transformation. Parameters such as ‘min\_df’ and ‘ngram\_range’ help control which words are considered and how they are grouped together.

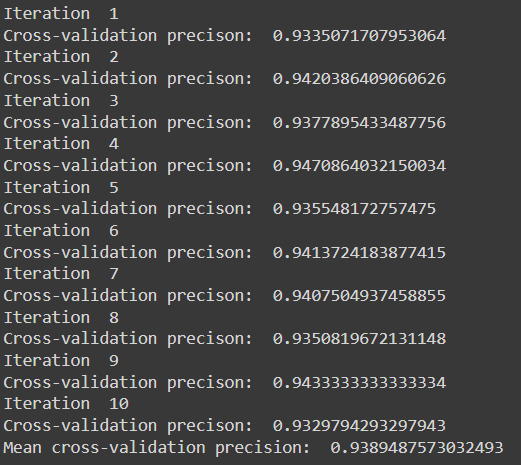
However, it's important to note that TF-IDF has its limitations. It doesn't fully capture the meaning of words and ignores the order in which they appear. Despite these limitations, TF-IDF is widely used because it provides a structured numerical representation of text data, which is essential for machine learning algorithms to work effectively.

1. **Model Performance Evaluation**

When evaluating the effectiveness of our models, we focused our attention on precision, a crucial measure indicating how accurately we can identify positive reviews. We opted to focus on precision as our main evaluation measure because it directly reflects the owner's goal of accurately identifying positive reviews. Precision tells us how many of the reviews predicted as positive by the model are actually positive. Since the owner wants to find positive feedback to improve their business decisions and see who recommends their restaurant further, it's crucial to minimize mistakes like falsely identifying negative reviews as positive. By emphasizing precision, we ensure that the model's predictions are trustworthy, allowing the owner to rely on them confidently when making decisions. Therefore, precision is the most relevant metric for this task as it directly addresses the owner's objectives and requirements.

To thoroughly assess the models' performance, both the Support Vector Classifier (SVC) and the Naive Bayes Classifier (NBC) underwent a rigorous evaluation process called 10-fold cross-validation. This methodology involves dividing the dataset into ten equal parts and testing the models on different parts each time. By doing so, we gain a comprehensive understanding of how well the models perform across diverse subsets of the data.

Examining the precision scores obtained, we observed that the SVC model achieved scores ranging from approximately 0.933 to 0.947, with an average score of around 0.939.



In comparison, the NBC model exhibited precision scores spanning from about 0.943 to 0.958, with a slightly higher average of approximately 0.950.

A screenshot of a computer program

Description automatically generated

These precision scores provide valuable insights into the models' ability to accurately identify positive reviews. While both models demonstrated commendable performance, the NBC model showcased a slightly superior precision. This implies that the NBC model has a marginally better capability in distinguishing positive sentiments from the reviews.

1. **Recommendation Conclusion**

For our task, we aim to accurately identify positive reviews to help the dessert parlour make informed decisions. We focus on precision, which means minimizing the chances of falsely identifying negative reviews as positive. The NBC performs slightly well in this aspect while also being easier to understand and use compared to the SVC.

While the Support Vector Classifier (SVC) is indeed known for its effectiveness in handling high-dimensional data, our decision to choose the Naive Bayes Classifier (NBC) over the SVC was driven by several factors specific to our task requirements and the nature of Yelp reviews.

The NBC is known for its simplicity and ease of interpretation. This means that it's straightforward to implement and understand how it makes predictions. This transparency is crucial for the dessert parlour owner to trust the model's recommendations and make effective decisions. Additionally, the NBC is computationally efficient, meaning it doesn't require as much processing power as the SVC. This is important for handling large amounts of data or making predictions quickly.

In tasks involving text data, like ours with Yelp reviews, the NBC's ability to handle words or tokens effectively is advantageous. Its method of considering each feature (word) independently works well for analysing text data, making it suitable for sentiment analysis.

Taking all these factors into account, the Naive Bayes Classifier (NBC) is the best choice for our task. Its simplicity, interpretability, computational efficiency, and effectiveness in handling text data make it well-suited to helping the dessert parlour understand customer sentiment.