



---

# Prediction of Customers' Behaviour Using Machine Learning Algorithms

*Aishwarya.V<sup>a</sup>, Dr. Gomathi alias Rohini.S<sup>b\*</sup>*

<sup>a</sup> B.Sc. AI&ML, Kongunadu Arts and Science College, Coimbatore 641029, India

<sup>b</sup> Associate Professor & Head i/c, Kongunadu Arts and Science College, Coimbatore 641029, India

---

## ABSTRACT :

Understanding customer behavior is paramount for businesses to tailor their products and services effectively. With the proliferation of online platforms, analyzing text reviews has become a crucial method for gaining insights into customer sentiments and preferences. The system was developed using Python 3.12 and Jupyter Notebook. It has pre-processing, model training and model evaluation phases. This study explored the efficacy of three machine-learning models – Decision Tree, Naive Bayes and Ensemble Voting Classifier in predicting customer behavior using text review datasets. They produced an accuracy of 95%, 96% and 100% and provided a detailed summary of the evaluation metrics - precision, recall, F1-score and support for each class respectively.

Keywords: text reviews, evaluation metrics, customer behaviour, model training, visualisation

---

## 1. Introduction :

In today's digital age, understanding customers' behavior is paramount for businesses striving to deliver tailored products and services that meet their preferences. With the proliferation of online platforms, analyzing text reviews has emerged as a powerful tool for gaining insights into customer sentiments and preferences. In this paper, delving into the realm of prediction of customers' behaviour (Serhat Peker, Altan Kocyigit & P. Erhan Eren, 2018) was done using text review datasets, focusing specifically on Amazon electronic review dataset. The efficacy of three machine learning models - Decision Tree, Naive Bayes and Ensemble Voting Classifier was used in predicting customer behavior based on textual inputs.

---

## 2. System Study and Overview:

The available system leverages logistic regression and Naive Bayes algorithms to analyze text reviews and predict customer behavior, enabling businesses to gain insights into customer sentiments and preferences. By identifying patterns and trends in textual data, businesses can make data-driven decisions to improve products, enhance customer satisfaction and optimize marketing strategies (Soumi Ghosh & Chandan Banerjee, 2020). The accuracy of the existing system by logistic regression is 96% and Naive Bayes is 93%. Even though the existing system has higher accuracy, it also has some disadvantages like linear decision boundary, sensitivity to outliers and limited interpretability.

The developed system of customer behavior prediction using text data integrated the two powerful machine-learning algorithms - Decision Tree and Naive Bayes, along with an Ensemble Voting Classifier to enhance predictive performance and robustness (QiongWu, Wen-LingHsu, TanXu, ZhenmingLiu, GeorgeMa, GuyJacobson & Shuai Zhao, 2019). Decision Tree models are well-suited for capturing complex relationships between textual features extracted from customer reviews and predicting customer behavior. Naive Bayes classifiers offer a probabilistic approach to customer behavior prediction, leveraging the assumption of feature independence to efficiently categorize textual data into predefined classes. Ensemble model combines the prediction of both models using voting Classifier technique, which can increase the accuracy.

---

## 3. Software, Libraries and Models Used

- Python 3.12: Python is an interpreted, high-level, general programming language. Its formatting is visually uncluttered, and it often uses English keywords where other languages use punctuation. It provides a vast library for data mining and predictions.
- Jupyter Notebook: It is an open-source cross-platform integrated development environment (IDE) for scientific programming in the Python language.
- NumPy: NumPy was used for building the front-end part of the system.
- Pandas: Pandas were used for data pre-processing and statistical analysis of data.
- Matplotlib: Matplotlib was used for graphical representation and prediction.
- NLTK: NLTK was used for Natural Language Processing.

- Scikit-learn: Scikit-learn was used to provide many unsupervised and supervised learning algorithms.
- TKinder: TKinder was used to create user interface.

#### 4. System Design and Algorithm Used

The system design consists of input dataset, pre-processing the data, training the model using Decision Tree, Naïve Bayes and Ensemble Voting Classifier. See Fig.1 below.

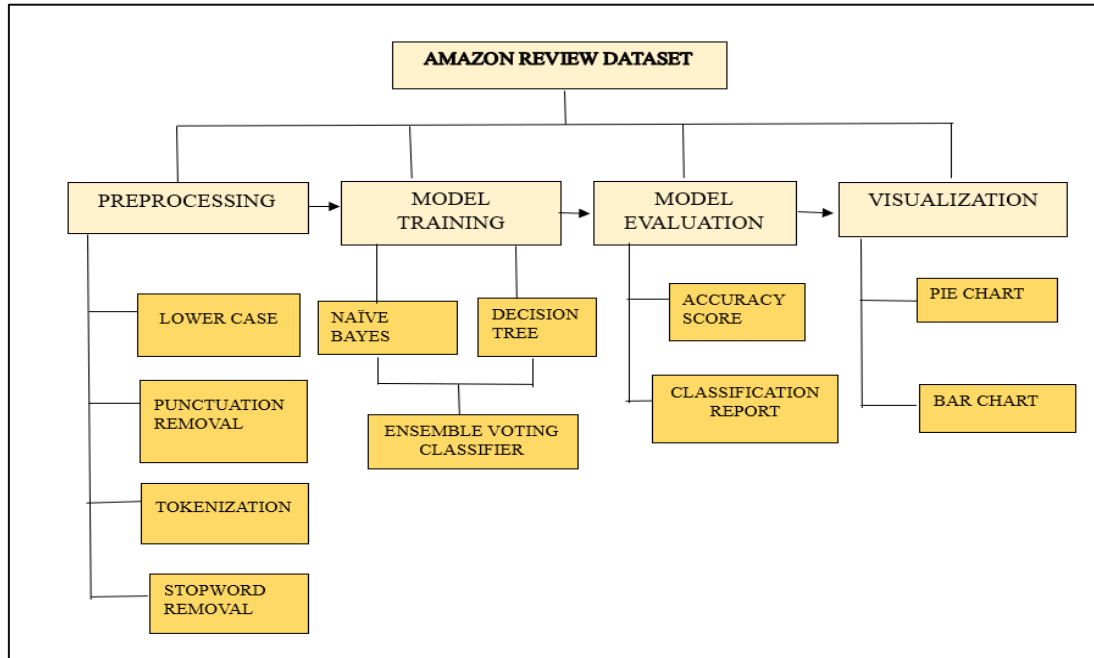


Fig. 1 - System Design

##### 4.1. Model training

- DECISION TREE -The Decision Tree classifier is trained using the Term Frequency -Inverse Document Frequency (TF-IDF) transformed text features to create a tree-like model that predicts the sentiment of the text data.
- NAÏVE BAYES- The Multinomial Naive Bayes classifier is suitable for text classification tasks and is trained using the TF-IDF transformed text features.
- ENSEMBLE VOTING CLASSIFIER-An ensemble voting classifier is constructed using a combination of the trained Naive Bayes and Decision Tree classifiers. The hard voting mechanism is employed, where the predicted class labels from each classifier are combined, and the majority class is selected as the final prediction. See Fig.2 below.

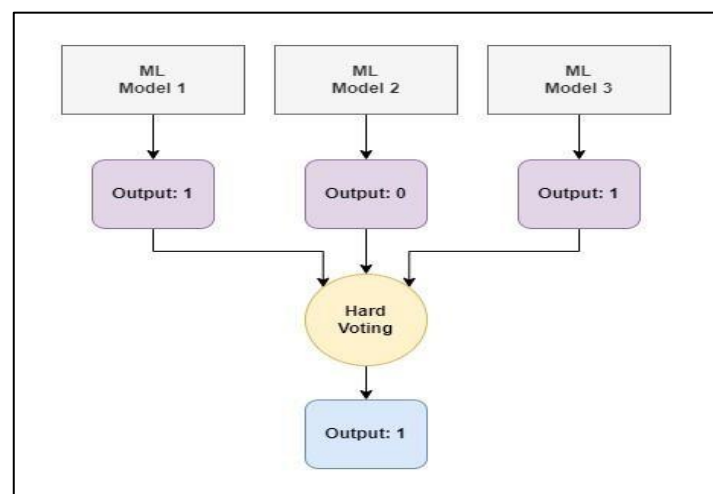


Fig. 2 - Voting Classifier-Hard Voting

## 5. Detailed System Description

### Phase 1: Pre-processing

The Pre-processing involves following steps. See Fig.3.

- LOWERCASE- Changes all the words in the text into lowercase letters.
- PUNCTUATION REMOVAL-removes punctuation marks.
- TOKENIZATION - Splits text sentence into individual words.
- STOPWORD REMOVAL –Removes stop words such as ‘and’, ‘is’, ‘was’ etc.
- JOINING TOKENS- After tokenization and stop word removal the remaining tokens are joined into a single string.



```

•[8]: sid = SentimentIntensityAnalyzer()

•[9]: def preprocess_text(text):
    if isinstance(text, str):
        text = text.lower()
        text = text.translate(str.maketrans('', '', string.punctuation))
        tokens = word_tokenize(text)
        stop_words = set(stopwords.words('english'))
        tokens = [word for word in tokens if word not in stop_words]
        text = ' '.join(tokens)
    else:
        text = str(text)
    return text

•[10]: selected_columns['clean_text'] = selected_columns['reviews.text'].apply(preprocess_text)

•[11]: def get_sentiment_label(sentence):
    score = sid.polarity_scores(sentence)['compound']
    return 'positive' if score > 0 else 'negative' if score < 0 else 'neutral'

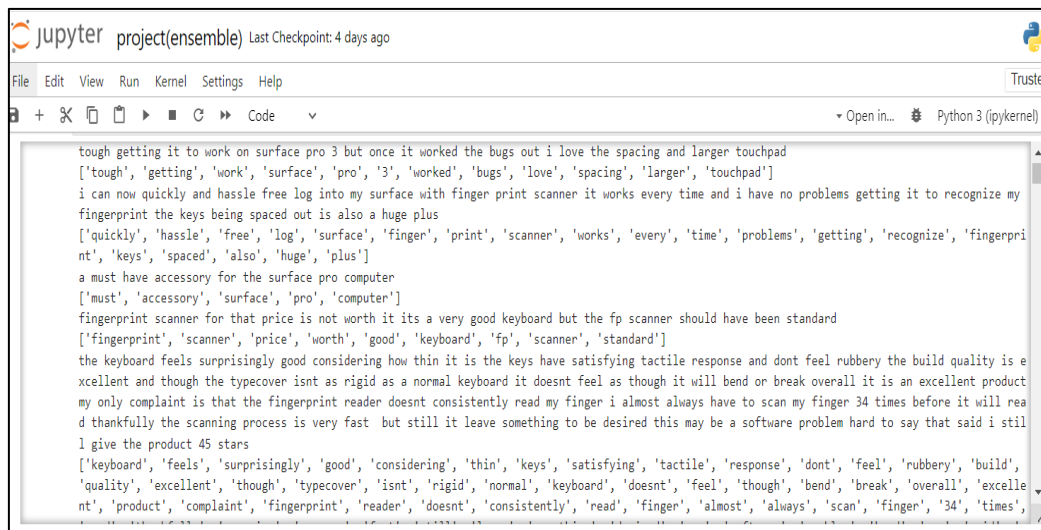
•[12]: selected_columns['sentiment'] = selected_columns['clean_text'].apply(get_sentiment_label)

•[13]: X = selected_columns['clean_text']
    y = selected_columns['sentiment']

[14]: vectorizer = TfidfVectorizer()
    X_tfidf = vectorizer.fit_transform(X)
  
```

Fig. 3 - Script for Pre-processing

Pre-processing produced text data as below. See Fig. 4.



```

tough getting it to work on surface pro 3 but once it worked the bugs out i love the spacing and larger touchpad
['tough', 'getting', 'work', 'surface', 'pro', '3', 'worked', 'bugs', 'love', 'spacing', 'larger', 'touchpad']
i can now quickly and hassle free log into my surface with finger print scanner it works every time and i have no problems getting it to recognize my
fingerprint the keys being spaced out is also a huge plus
['quickly', 'hassle', 'free', 'log', 'surface', 'finger', 'print', 'scanner', 'works', 'every', 'time', 'problems', 'getting', 'recognize', 'fingerprint',
'keys', 'spaced', 'also', 'huge', 'plus']
a must have accessory for the surface pro computer
['must', 'accessory', 'surface', 'pro', 'computer']
fingerprint scanner for that price is not worth it its a very good keyboard but the fp scanner should have been standard
['fingerprint', 'scanner', 'price', 'worth', 'good', 'keyboard', 'fp', 'scanner', 'standard']
the keyboard feels surprisingly good considering how thin it is the keys have satisfying tactile response and dont feel rubbery the build quality is e
xcellent and though the typecover isnt as rigid as a normal keyboard it doesnt feel as though it will bend or break overall it is an excellent product
my only complaint is that the fingerprint reader doesnt consistently read my finger i almost always have to scan my finger 34 times before it will rea
d thankfully the scanning process is very fast but still it leave something to be desired this may be a software problem hard to say that said i stil
l give the product 45 stars
['keyboard', 'feels', 'surprisingly', 'good', 'considering', 'thin', 'keys', 'satisfying', 'tactile', 'response', 'dont', 'feel', 'rubbery', 'build',
'quality', 'excellent', 'though', 'typecover', 'isnt', 'rigid', 'normal', 'keyboard', 'doesnt', 'feel', 'though', 'bend', 'break', 'overall', 'excele
nt', 'product', 'complaint', 'fingerprint', 'reader', 'doesnt', 'consistently', 'read', 'finger', 'almost', 'always', 'scan', 'finger', '34', 'times',
  
```

Fig. 4 - Text Data after pre-processing

### Phase 2: Model training

Model training by Decision Tree, Naïve Bayes and Ensemble Models is done as below. See Fig.5:

#### a. Decision Tree

The Decision Tree model is a popular choice for classification tasks due to its simplicity and interpretability. Decision trees are valuable tool for predicting customers' behavior using review text data. By converting textual reviews into numerical features, such as word frequencies or sentiment scores, decision trees can classify and predict outcomes based on the patterns found in the data. The tree structure allows for the sequential splitting of data into branches based on feature values, making it straight forward to understand which factors most influence customer behavior. Each node in the

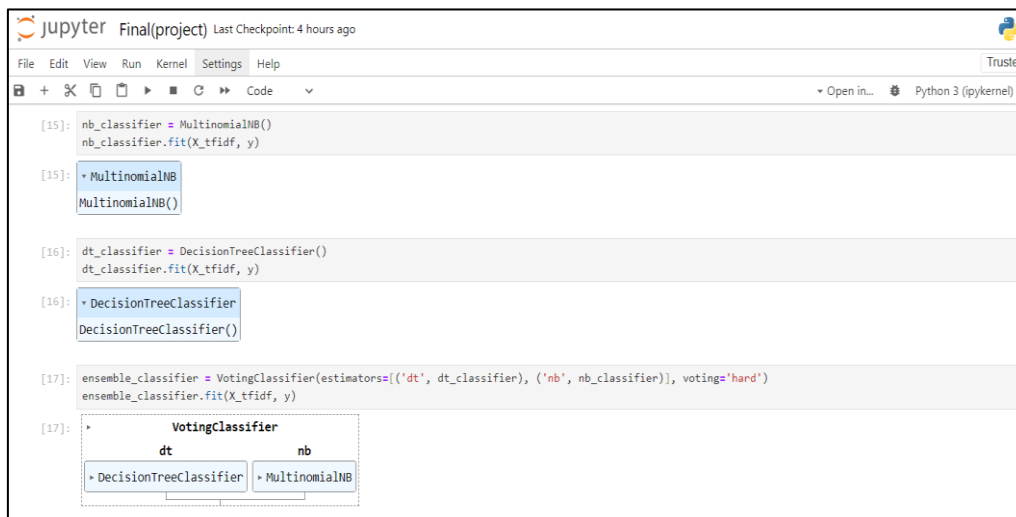
tree represents a decision point that splits the data based on the most significant feature, leading to predictions at the leaves. This method is particularly effective for identifying key words or phrases that signal positive or negative customer sentiments, enabling businesses to better understand and respond to customer needs and preferences. The clear visualization of the decision-making process also aids in explaining the model's predictions, which is essential for gaining trust and actionable insights from stakeholders.

### b. Naïve Bayes

Naive Bayes is based on probabilistic principles and assumes independence between features. Naive Bayes is a powerful algorithm for predicting customer behavior using review text data due to its foundation in probability theory and its efficiency with large datasets. By treating each word in a review as an independent feature, the Naive Bayes classifier calculates the probability that a given review belongs to a specific category (e.g., positive or negative sentiment) based on the frequency of words. Despite the simplifying assumption that all features are independent, which rarely holds true in practice, Naive Bayes often performs surprisingly well. This is because it effectively captures the underlying patterns in the data by combining the individual probabilities of each word to make a final prediction. The algorithm's efficiency and simplicity make it ideal for real-time applications, providing quick insights into customer sentiments and behaviors.

### c. Ensemble Model (Voting Classifier -Hard Voting)

Ensemble methods combine multiple base estimators to improve predictive performance. Using an ensemble model like Voting Classifier with Hard Voting, combination of Decision Trees and Naive Bayes can enhance customers' behavior prediction from review text data by leveraging the strengths of both the algorithms. Hard voting involves each model making a prediction and the final prediction is determined by the majority vote. This approach benefits from the interpretability and hierarchical decision-making of decision trees (Bora Bardük, 2020) along with the probabilistic efficiency and robustness of Naive Bayes. By combining these models, the ensemble can mitigate the individual weaknesses of each method such as the overfitting tendency of decision trees and the independence assumption in Naive Bayes resulting in more accurate and reliable predictions.



```

Jupyter Final(project) Last Checkpoint: 4 hours ago
File Edit View Run Kernel Settings Help Trusted
+ × ↻ ↺ ↻ ⏪ ⏩ Code
Python 3 (ipykernel)

[15]: nb_classifier = MultinomialNB()
nb_classifier.fit(X_tfidf, y)

[15]: ▾ MultinomialNB
MultinomialNB()

[16]: dt_classifier = DecisionTreeClassifier()
dt_classifier.fit(X_tfidf, y)

[16]: ▾ DecisionTreeClassifier
DecisionTreeClassifier()

[17]: ensemble_classifier = VotingClassifier(estimators=[('dt', dt_classifier), ('nb', nb_classifier)], voting='hard')
ensemble_classifier.fit(X_tfidf, y)

[17]: ▾ VotingClassifier
dt nb
▾ DecisionTreeClassifier ▾ MultinomialNB

```

Fig. 5 - Script for Model Training

## 6. System Evaluation and Visualization

### 6.1. System evaluation

The developed system can be evaluated by its accuracy calculation and classification report generation.

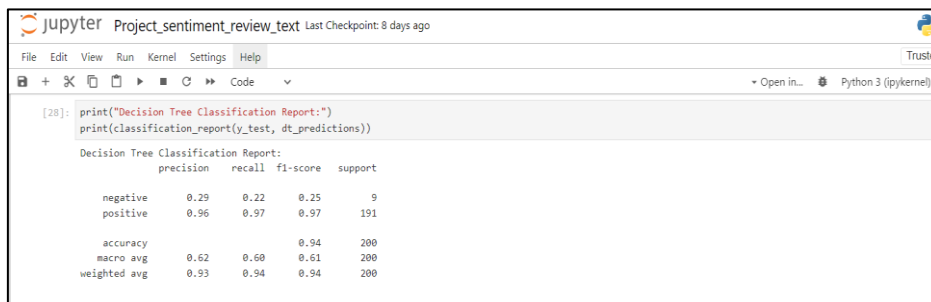
- Accuracy Calculation - Accuracy is a measure of fraction of correctly classified instances out of the total instances. It is calculated for Naive Bayes, Decision Tree and Ensemble Voting Classifier models. They produced an accuracy of 96.5%, 95% and 100% respectively. See Fig.6.
- Classification Report Generation - The classification report provides a detailed summary of various evaluation metrics such as precision, recall, F1-score and support for each class (positive and negative sentiment). It gives insights into the performance of the models across different classes. See Fig.7,8 and 9.

```
[19]: nb_accuracy = accuracy_score(y_test, nb_predictions)
      dt_accuracy = accuracy_score(y_test, dt_predictions)
      ec_accuracy = accuracy_score(y_test, ensemble_predictions)

[20]: print("Naive Bayes Accuracy:", nb_accuracy)
      print("Decision Tree Accuracy:", dt_accuracy)
      print("Ensemble Learning:", ec_accuracy)

Naive Bayes Accuracy: 0.965
Decision Tree Accuracy: 0.95
Ensemble Learning: 1.0
```

**Fig. 6 - Accuracy Calculation**



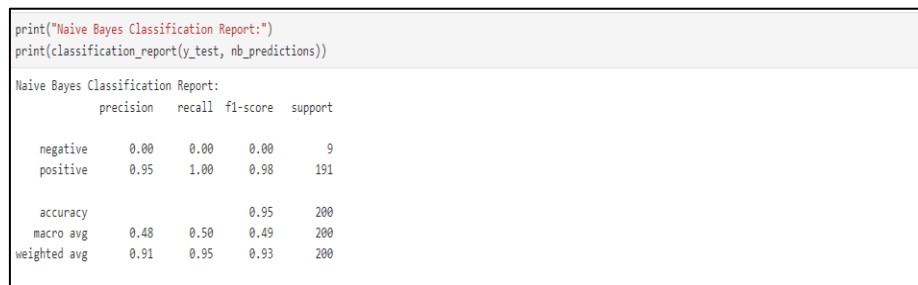
```
[28]: print("Decision Tree Classification Report:")
      print(classification_report(y_test, dt_predictions))

Decision Tree Classification Report:
              precision    recall  f1-score   support

 negative      0.29      0.22      0.25         9
 positive      0.96      0.97      0.97        191

 accuracy            0.94        200
 macro avg           0.62      0.60      0.61        200
 weighted avg        0.93      0.94      0.94        200
```

**Fig. 7 - Classification Report Generation - Decision Tree**



```
print("Naive Bayes Classification Report:")
print(classification_report(y_test, nb_predictions))

Naive Bayes Classification Report:
              precision    recall  f1-score   support

 negative      0.00      0.00      0.00         9
 positive      0.95      1.00      0.98        191

 accuracy            0.95        200
 macro avg           0.48      0.50      0.49        200
 weighted avg        0.91      0.95      0.93        200
```

**Fig. 8 - Classification Report Generation - Naive Bayes**



```
print("Ensemble learning Classification Report:")
print(classification_report(y_test, ensemble_predictions ))

Ensemble learning Classification Report:
              precision    recall  f1-score   support

 negative      1.00      1.00      1.00         7
 positive      1.00      1.00      1.00        193

 accuracy            1.00        200
 macro avg           1.00      1.00      1.00        200
 weighted avg        1.00      1.00      1.00        200
```

**Fig. 9 - Classification Report Generation - Ensemble Model**

**6.2. Visualization**

Visualization is an essential aspect of model evaluation, allowing for a clear understanding of performance differences between models. The following visualizations are generated.

- Pie Chart: A pie chart is plotted to visualize the distribution of accuracy scores for Naive Bayes, Decision Tree and Ensemble Learning models. This visualization provides a contribution of each model for accuracy. See Fig.10
- Bar Chart: A bar chart is plotted to visualise the accuracy scores of each model. This visualization offers a detailed view of the differences in accuracy among the models. See Fig.11.

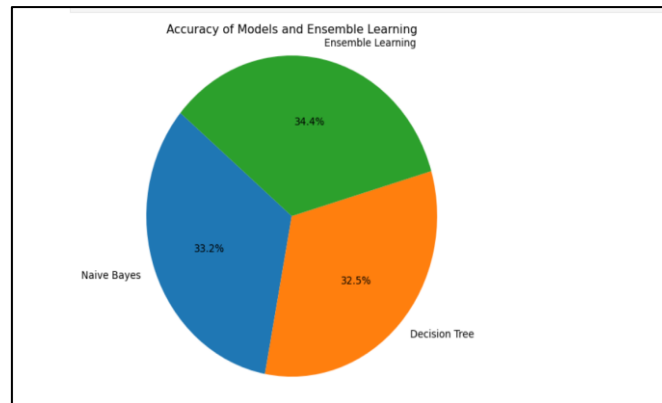


Fig. 10 - Pie Chart

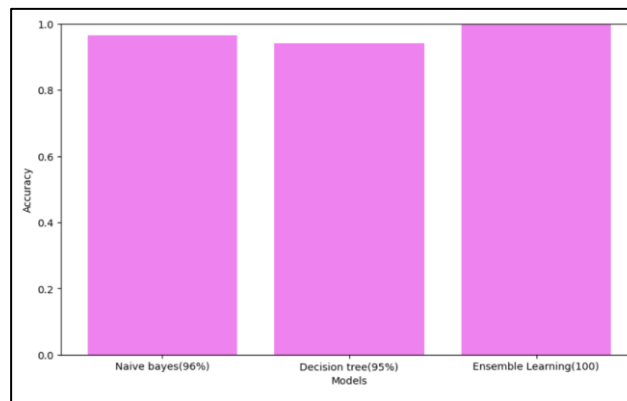


Fig. 11 - Bar Chart

## 7. Conclusion

The objective of this endeavour is to aid organizations in comprehending their clientele and amalgamating targeted marketing strategies to augment their customer base and profits. Behaviour analysis is instrumental in assessing customers' perceptions towards diverse products, facilitating a thorough analysis (Chiaki Doi, Masaji Katagiri, Takashi Araki, Daizo Ikeda & Hiroshi Shigeno, 2018) of product performance in the market. Three classification models for customer analysis on reviews utilizing natural language processing yielded optimal outcomes using decision tree classifier, Naive Bayes classifier and ensemble voting classifier.

## REFERENCES :

1. Serhat Peker, Altan Kocyigit & P. Erhan Eren. (2018). An empirical comparison of customer behavior modeling approaches for shopping list prediction, *IEEE 41<sup>st</sup> International Convention on Information and Communication Technology, Electronics and Microelectronics*.
2. QiongWu, Wen-LingHsu, TanXu, ZhenmingLiu, GeorgeMa, GuyJacobson & Shuai Zhao. (2019). Speaking with actions learning customer journey behavior, *IEEE 13th International Conference on Semantic Computing*.
3. Soumi Ghosh & Chandan Banerjee. (2020). A predictive analysis model of customer purchase behavior using modified random forest algorithm in cloud environment, *IEEE 1st International Conference for Convergence in Engineering*.
4. Bora Bardük. (2020). Modelling time statistics for customer churn prediction, *28th Signal Processing and Communications Applications Conference*.
5. Chiaki Doi, Masaji Katagiri, Takashi Araki, Daizo Ikeda & Hiroshi Shigeno. (2018). Is he becoming an excellent customer for us? a customer level prediction method for a customer relationship management system, *IEEE 32nd International Conference on Advanced Information Networking and Applications*.