AMAZON PRODUCT REVIEWS SENTIMENT ANALYSIS USING MACHINE LEARNING

A Project Report
In the partial fulfilment of the award of the degree of

B.Tech

Under

Academy of Skill Development

ASD
Skills For Employment

Submitted by:

RICHIK DEY

KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY
BHUBANESWAR, ODISHA

ASD

Skills For Employment

CERTIFICATE FROM THE MENTOR

This is to certify that **RICHIK DEY** has completed the project titled AMAZON PRODUCT REVIEWS SENTIMENT ANALYSIS USING MACHINE LEARNING under my supervision during the period from May to June which is in partial fulfilment of requirements for the award of the B.Tech and submitted to School of Computer Engineering of KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY.

DATE: **Signature of the Mentor**

ACKNOWLEDGMENT

I take this opportunity to express my deep gratitude and sincerest thanks to my project mentor, **Mr. Joyjit Guha Biswas** for giving the most valuable suggestions, helpful guidance, and encouragement in the execution of this project work.

I would like to give a special mention to my colleagues. Last but not least I am grateful to all the faculty members of the **Academy of Skill Development** for their support.

1.	Name of the Student:	RICHIK DEY
2.	Title of the Project:	**AMAZON PRODUCT REVIEWS SENTIMENT ANALYSIS USING MACHINE LEARNING**
3.	Name and Address of the Guide:	**MR. JOYJIT GUHA BISWAS** MCA, Sr. Subject Matter Expert & Technical Head (Python) - Academy of Skill Development (An ISO 9001:2015 Certified) Module-132, SDF Building Salt Lake Sector-V, Kolkata, West Bengal - 700 091
4.	Educational Qualification of the Guide:	Ph.d* ☐ M.tech* ☐ B.E*/B.Tech* ☐ MCA* ☐ M.Sc* ☐
5.	Working and Teaching Experience of the Guide:	**……. Years**
6.	Software used in the Project:	**a. Google Colab** **b. Python** **c. Jupyter Notebook** **d. Scikit-learn**

Signature of the Guide

Date:

Name: Mr. Joyjit Guha Biswas

MCA, Sr. Subject Matter Expert &

Technical Head (Python)

Academy of Skill Development

Signature, Designation, Stamp of the

Project Proposal Evaluator

APPROVED ☐ NOT APPROVED ☐

SELF- CERTIFICATE

This is to certify that the dissertation/project proposal entitled "Amazon Product Reviews Sentiment Analysis Using Machine Learning" is done by me, is an Information Technology project under the guidance of Mr. Joyjit Guha Biswas. The matter embodied in this project work has not been submitted earlier for award of any certificate to the best of our knowledge and belief.

Name of the Student:

Richik Dey

Signature of the Student:

Richik Dey

<u>CERTIFICATE BY GUIDE</u>

This is to certify that this project entitled "<u>Amazon Product Reviews Sentiment Analysis Using Machine Learning</u>" submitted in partial fulfillment of the certificate of Bachelor of Technology through Academy of Skill Development, done by Richik Dey, is an authentic work carried out under my guidance & best of our knowledge and belief.

Richik Dey

Signature of the Student

Date:

Signature of the Guide

Date:

CERTIFICATE OF APPROVAL

This is to certify that this proposal of Minor project, entitled "**Amazon Product Reviews Sentiment Analysis Using Machine Learning**" is a record of bona-fide work, carried out by Richik Dey, under my supervision and guidance through the Academy of Skill Development. In my opinion, the report in its present form is in partial fulfillment of all the requirements, as specified by the Kalinga Institute of Industrial Technology of Information Technology as per regulations of the Academy of Skill Development. In fact, it has attained the standard, necessary for submission. To the best of my knowledge, the results embodied in this report, are original in nature and worthy of incorporation in the present version of the report for Bachelor of Technology.

Guide/Supervisor

Mr. Joyjit Guha Biswas

MCA, Sr. Subject Matter Expert & Technical Head (Python)
Academy of Skill Development (An ISO 9001:2015 Certified) Module-132,
SDF Building
Salt Lake Sector-V, Kolkata, West Bengal - 700 091

External Examiner(s)

Head of the Department

School of Computer Engineering
(Kalinga Institute of Industrial Technology)

ABSTRACT

This project presents a detailed study on sentiment analysis of Amazon product reviews using traditional machine learning algorithms. The main goal is to classify customer reviews as either positive or negative. This helps businesses understand customer satisfaction and how their products are perceived. The analysis includes preprocessing a balanced dataset of reviews, converting text to numerical features using TF-IDF vectorization, and applying six classification algorithms: Logistic Regression, Naive Bayes, Linear SVM, Random Forest, Decision Tree, and K-Nearest Neighbors. The models were evaluated based on accuracy, confusion matrices, ROC curves, and classification metrics. Among them, Random Forest achieved the highest accuracy at 90.37%. It was closely followed by Logistic Regression and SVM. Additionally, visualization tools like word clouds and ROC curves were used to improve clarity. This study shows how effective classical machine learning techniques can be for binary sentiment classification tasks and provides insights into how well the algorithms perform on real-world e-commerce review data.

Keywords: Sentiment Analysis, Amazon Reviews, Machine Learning, Text Classification, TF-IDF, Logistic Regression, Random Forest, Natural Language Processing, Binary Classification, ROC-AUC.

AMAZON PRODUCT REVIEWS SENTIMENT ANALYSIS USING MACHINE LEARNING

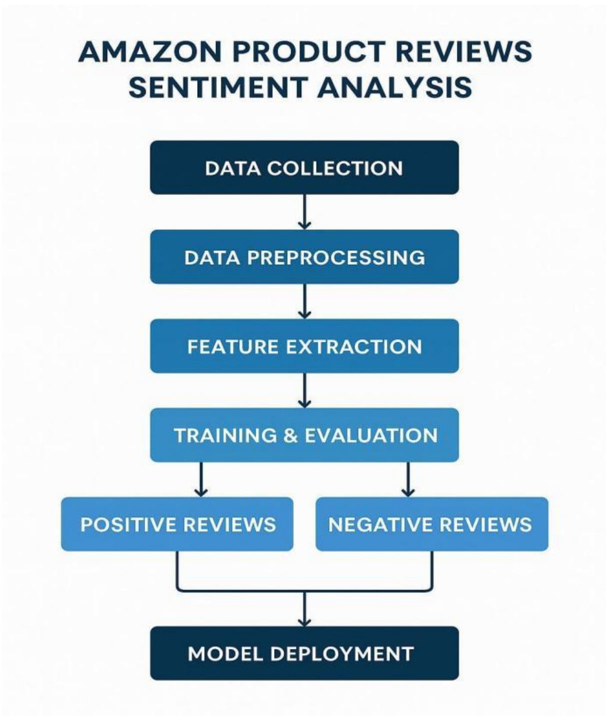
TABLE OF CONTENT

1. INTRODUCTION

In the digital age, large amounts of customer feedback are generated on e-commerce platforms through product reviews. Analyzing this data can provide businesses with important insights into customer satisfaction, product issues, and market needs. However, analyzing this data manually is slow and prone to mistakes.

Sentiment analysis, also called opinion mining, automates the classification of text into sentiment categories like positive, negative, or neutral. This project focuses on binary sentiment classification, comparing positive and negative sentiments using machine learning (ML) algorithms. We examine how different ML classifiers perform using TF-IDF vectorization on a labeled review dataset. Our goal is to compare their effectiveness and find the best model for predicting sentiment.

Sentiment analysis is a part of Natural Language Processing (NLP) that identifies and extracts feelings from text data. With the growing number of customer reviews on platforms like Amazon, automated sentiment classification has become essential for businesses. This project looks into using traditional machine learning algorithms to classify Amazon product reviews.



2. LITERATURE REVIEW

Literature in sentiment analysis has progressed from the early use of bag-of-words and Naive Bayes classifiers to newer embedding-based and transformer-based models. Traditional classifiers like Logistic Regression, SVM, and Naive Bayes have delivered strong results on binary sentiment tasks, particularly with feature engineering using TF-IDF or n-grams (Jagdale et al., 2019; Arkesha Shah, 2021).

The literature indicates that:

- Text preprocessing greatly improves model performance.
- TF-IDF is better than bag-of-words for sparse text data.
- Ensemble methods such as Random Forest usually perform better than simpler models.
- Proper evaluation metrics, including ROC-AUC, precision, recall, and F1-score, provide a clear view of model effectiveness.

Recent studies focus on hybrid approaches that combine neural networks with pre-trained embeddings. CNNs trained on FastText and Word2Vec have shown higher accuracy because they can capture context. Recently, transformer-based models like GPT-4 have been compared to classical ML models (Godia & Tiwari, 2024; Ghatora et al., 2024), showing that traditional methods still work well for short, well-labeled datasets.

This study fits into that framework by using classical algorithms for efficient, interpretable sentiment classification while setting the stage for future experiments with deep learning.

3. RELATED WORKS

Several studies have examined sentiment analysis of product reviews using machine learning and deep learning techniques. Jagdale et al. (2019) used Naive Bayes and SVM on Amazon product categories and achieved an accuracy of 98.2% with Naive Bayes in specific domains. Arkesha Shah (2021) compared Naive Bayes, SVM, and CNN with FastText embeddings. She concluded that CNN performed better than traditional models, achieving 91.2% accuracy.

Adarsh Godia and L.K. Tiwari (2024) conducted a detailed evaluation of Flipkart reviews. They applied SMOTE to address imbalance and tested traditional models alongside GPT-4. They found that SVM was reliable for short reviews, while GPT-4 performed better on longer texts. Ghatora et al. (2024) compared ML classifiers with zero-shot GPT-4 models. Their findings also showed that ML models remain competitive for simple, short-form sentiment classification.

These studies together show that while LLMs are powerful, traditional ML methods are still effective for structured, domain-specific sentiment analysis, which is the focus of our current project.

4. PROBLEM STATEMENT

With the growing number of customer reviews on e-commerce sites, businesses need an effective way to analyze and sort feedback. This is important for improving customer satisfaction and product quality. Manual analysis does not work well because of the large volume and personal bias. Thus, there is a need for an automated system that can classify product reviews as either positive or negative.

This project aims to develop and test machine learning models that can carry out binary sentiment analysis on Amazon product reviews. It will use structured preprocessing, TF-IDF feature extraction, and classification techniques. The objective is to compare different machine learning classifiers and identify the best model for predicting sentiment.

5. OBJECTIVES

The main objectives of this project are::

- To collect and preprocess real-world Amazon review data labeled with sentiment, either positive or negative.
- To clean and normalize the text using NLP techniques.
- To convert text into numerical features using TF-IDF with n-grams.
- To build and evaluate six machine learning models for sentiment classification.
- To compare the models based on accuracy, confusion matrix, and ROC-AUC.
- To visualize results with ROC curves and word clouds.
- To analyze insights and find the best-performing model.
- To interpret the results in a business context for future deployment.

6. METHODOLOGY

6.1 DATA COLLECTION

I. Resource:

Kaggle

https://www.kaggle.com/datasets/sashmindairanga/amazon-reviews-balanced-dataset

II. Dataset Description:

- Two columns: Review (text), Sentiment (Positive/Negative).
- Clean, balanced dataset with equal distribution across classes.
- Used for supervised binary classification.

```
# Display Data
df.head()
```

	Review	Sentiment	label
0	best candy corn on the planet ill keep this sh...	Positive	1
1	cat food my cats eat it that is all i can say ...	Positive	1
2	onions overwhelm otherwise lowkey flavor the o...	Negative	0
3	yummy tasted good spicy those that dont like s...	Positive	1
4	good flavor the product is the same as what we...	Positive	1

```
# Sentiment Distribution Count
df['Sentiment'].value_counts()
```

	count
Sentiment	
Positive	82037
Negative	82037

dtype: int64

6.2 DATA PREPROCESSING

I. Text Cleaning:

- Text Cleaning: Removal of HTML, URLs, punctuation, numbers, and stopwords; conversion to lowercase.
- Vectorization: TF-IDF with n-grams (1, 2) and max features = 7000

```
# Clean text for ML models
def clean_text(text):
    text = str(text).lower()
    text = re.sub(r"<.*?>", "", text)
    text = re.sub(r"http\S+", "", text)
    text = re.sub(f"[{re.escape(string.punctuation)}]", "", text)
    text = re.sub(r"\d+", "", text)
    text = re.sub(r"\s+", " ", text).strip()
    return text

df['clean_text'] = df['Review'].apply(clean_text)
```

6.3 FEATURE EXTRACTION

I. TF-IDF Vectorization:

The TF-IDF score for a term *t* in document *d* is calculated as:

$$\text{TF-IDF}(t,d) = \text{TF}(t,d) \times \log\left(\frac{N}{\text{DF}(t)}\right)$$

Where:

- TF(*t*, *d*): Term frequency of term *t* in document *d*
- DF(*t*): Number of documents containing term *t*
- *N*: Total number of documents

```
# TF-IDF Vectorization
tfidf = TfidfVectorizer(stop_words='english', max_features=7000, ngram_range=(1, 2))
X = tfidf.fit_transform(df['clean_text'])
y = df['label']
```

6.4 EXPLORATORY DATA ANALYSIS

I. Data Visualization for Statistical Analysis:

```
# Generate word clouds for Positive reviews
positive_text = " ".join(df[df['label'] == 1]['clean_text'].values)

plt.figure(figsize=(8, 3))
wordcloud_pos = WordCloud(width=800, height=400, background_color='white', colormap='Greens').generate(positive_text)
plt.imshow(wordcloud_pos, interpolation='bilinear')
plt.axis('off')
plt.title('Word Cloud - Positive Reviews')
plt.show()
```

```
# Generate word clouds for Negative reviews
negative_text = " ".join(df[df['label'] == 0]['clean_text'].values)

plt.figure(figsize=(8, 3))
wordcloud_neg = WordCloud(width=800, height=400, background_color='white', colormap='Reds').generate(negative_text)
plt.imshow(wordcloud_neg, interpolation='bilinear')
plt.axis('off')
plt.title('Word Cloud - Negative Reviews')
plt.show()
```

II. Business Insights:

- Most reviews are either positive or negative; they are rarely neutral.
  ➢ Helps businesses understand customer polarity and adjust support/product strategies accordingly.

- Words like 'great', 'excellent', 'perfect' strongly indicate a positive sentiment.
  ➢ Indicates key drivers of customer satisfaction; useful for marketing and product highlights.

- Words like 'waste', 'poor', 'bad' show strong negative sentiment.
  ➢ Identifies recurring dissatisfaction themes, guiding improvements or recalls.

- Random Forest and similar models perform well across various metrics.
  ➢ Informs decision-makers which ML model is most reliable for deployment in business applications.

- Classical ML models are fast and easy to understand, making them ideal for real-time sentiment applications.
  ➢ Suggests the system's suitability for live business environments like dashboards, CRM tools, etc.

6.5 MACHINE LEARNING MODEL CREATION

I. Data Splitting:

```
# Split data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

II. Model Initialization:

```
# Model Initialization
models = {
    "Logistic Regression": LogisticRegression(max_iter=1000),
    "Naive Bayes": MultinomialNB(),
    "Support Vector Machine": LinearSVC(),
    "Random Forest": RandomForestClassifier(),
    "Decision Tree": DecisionTreeClassifier(),
    "K-Nearest Neighbor": KNeighborsClassifier()
}
```

III. Model Training:

```
# Train and Evaluate Models
for name, model in models.items():
    print(f"\n=== {name} ===")
    model.fit(X_train, y_train)
    preds = model.predict(X_test)
    acc = accuracy_score(y_test, preds)
    print(f"Accuracy: {acc:.4f}")
    print("Classification Report:\n", classification_report(y_test, preds))
    cm = confusion_matrix(y_test, preds)
    sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
    plt.title(f'{name} - Confusion Matrix')
    plt.xlabel('Predicted')
    plt.ylabel('Actual')
    plt.show()
```

6.6 ALGORITHMS USED:

I. Logistic Regression:

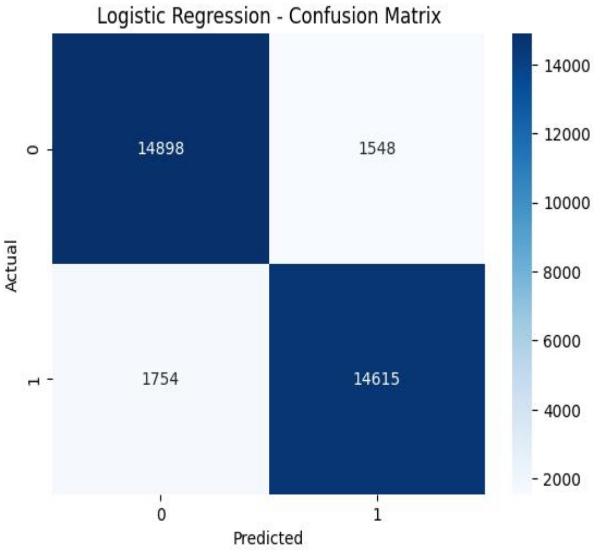
$$P(y=1|x) = \frac{1}{1+e^{-(\beta_0+\beta_1 x_1+\cdots+\beta_n x_n)}}$$

- Outputs probability of class 1 (positive).
- Decision boundary at 0.5 threshold.

```
=== Logistic Regression ===
Accuracy: 0.8994
Classification Report:
              precision    recall  f1-score   support

           0       0.89      0.91      0.90     16446
           1       0.90      0.89      0.90     16369

    accuracy                           0.90     32815
   macro avg       0.90      0.90      0.90     32815
weighted avg       0.90      0.90      0.90     32815
```



II. Naive Bayes:

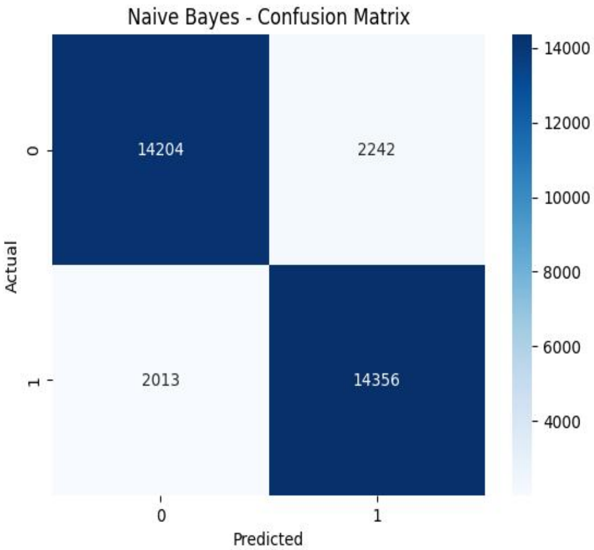
$$P(c|d) \propto P(c) \prod_{i=1}^{n} P(w_i|c)$$

- Assumes independence between features.
- Good baseline for text classification.

```
=== Naive Bayes ===
Accuracy: 0.8703
Classification Report:
              precision    recall  f1-score   support

           0       0.88      0.86      0.87     16446
           1       0.86      0.88      0.87     16369

    accuracy                           0.87     32815
   macro avg       0.87      0.87      0.87     32815
weighted avg       0.87      0.87      0.87     32815
```



IV. Support Vector Machine:

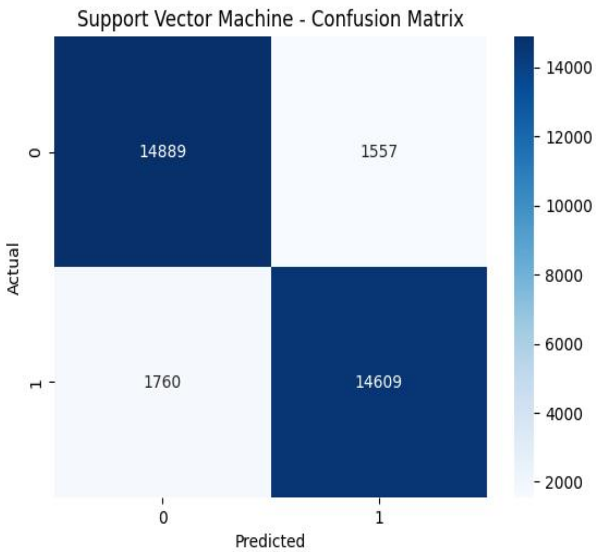
$$f(x) = w^T x + b$$

- Maximizes margin between positive and negative classes.
- Fast and effective for high-dimensional data like text.

```
=== Support Vector Machine ===
Accuracy: 0.8989
Classification Report:
              precision    recall  f1-score   support

           0       0.89      0.91      0.90     16446
           1       0.90      0.89      0.90     16369

    accuracy                           0.90     32815
   macro avg       0.90      0.90      0.90     32815
weighted avg       0.90      0.90      0.90     32815
```



V. Random Forest:

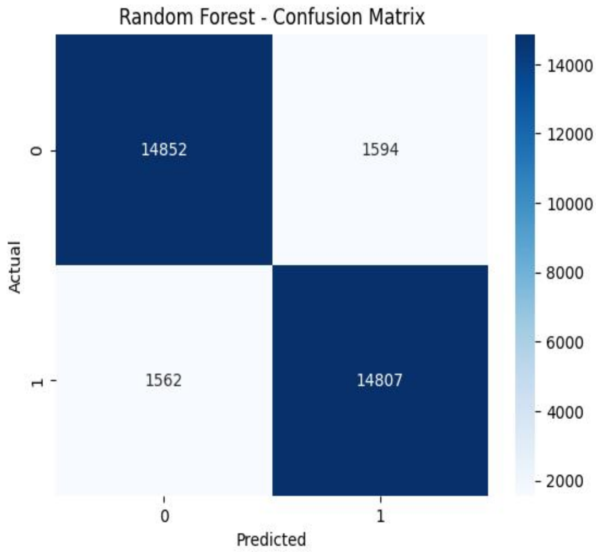
$$\hat{y} = \mathrm{MajorityVote}(T_1(x), T_2(x), \ldots, T_K(x))$$

- Ensemble of decision trees.
- Reduces overfitting and variance.

```
=== Random Forest ===
Accuracy: 0.9038
Classification Report:
              precision    recall  f1-score   support

           0       0.90      0.90      0.90     16446
           1       0.90      0.90      0.90     16369

    accuracy                           0.90     32815
   macro avg       0.90      0.90      0.90     32815
weighted avg       0.90      0.90      0.90     32815
```



VI. Decision Tree:

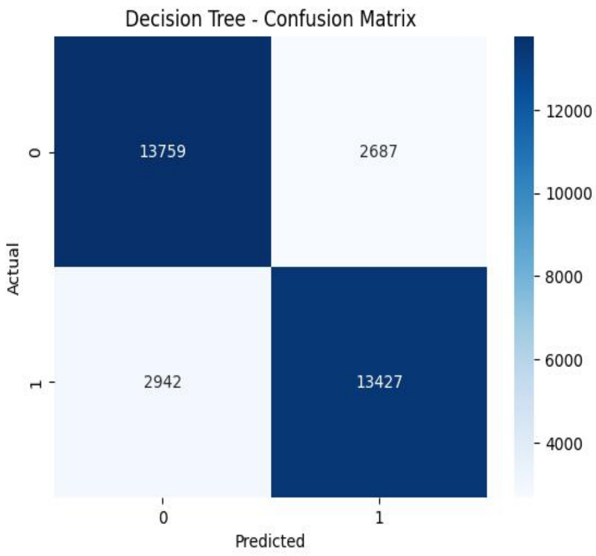
$$Gini(D) = 1 - \sum_{i=1}^{C} (p_i)^2$$

- Recursive binary split based on information gain or Gini impurity.
- Simple, interpretable, but prone to overfitting.

```
=== Decision Tree ===
Accuracy: 0.8285
Classification Report:
              precision    recall  f1-score   support

           0       0.82      0.84      0.83     16446
           1       0.83      0.82      0.83     16369

    accuracy                           0.83     32815
   macro avg       0.83      0.83      0.83     32815
weighted avg       0.83      0.83      0.83     32815
```



VII. K-Nearest Neighbour:

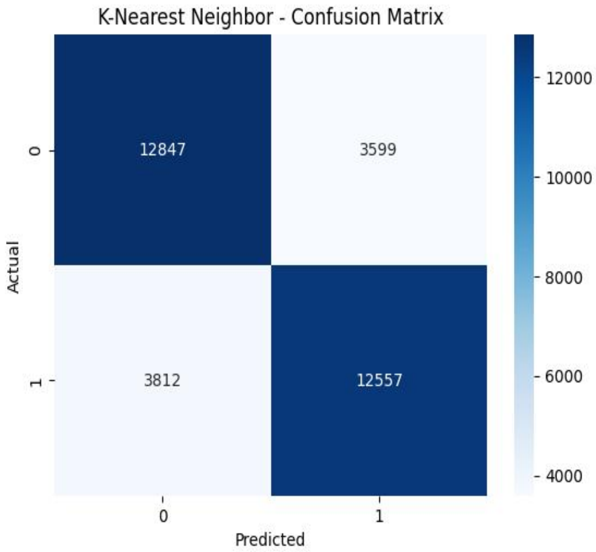
$$d(x, x_i) = \sqrt{\sum_{j=1}^{n}(x_j - x_{ij})^2}$$

- Classification based on majority vote from k closest neighbors.
- No training; lazy learner.

```
=== K-Nearest Neighbor ===
Accuracy: 0.7742
Classification Report:
              precision    recall  f1-score   support

           0       0.77      0.78      0.78     16446
           1       0.78      0.77      0.77     16369

    accuracy                           0.77     32815
   macro avg       0.77      0.77      0.77     32815
weighted avg       0.77      0.77      0.77     32815
```



6.7 RESULTS AND EVALUATION:

I. Accuracy: The proportion of correct predictions.

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$$

II. Precision: The proportion of true positive results among all the positive predictions.

$$\text{Precision} = \frac{TP}{TP+FP}$$

III. Recall: The proportion of true positive results among all the actual positives.

$$\text{Recall} = \frac{TP}{TP+FN}$$

IV. F1-Score: The harmonic mean of precision and recall.

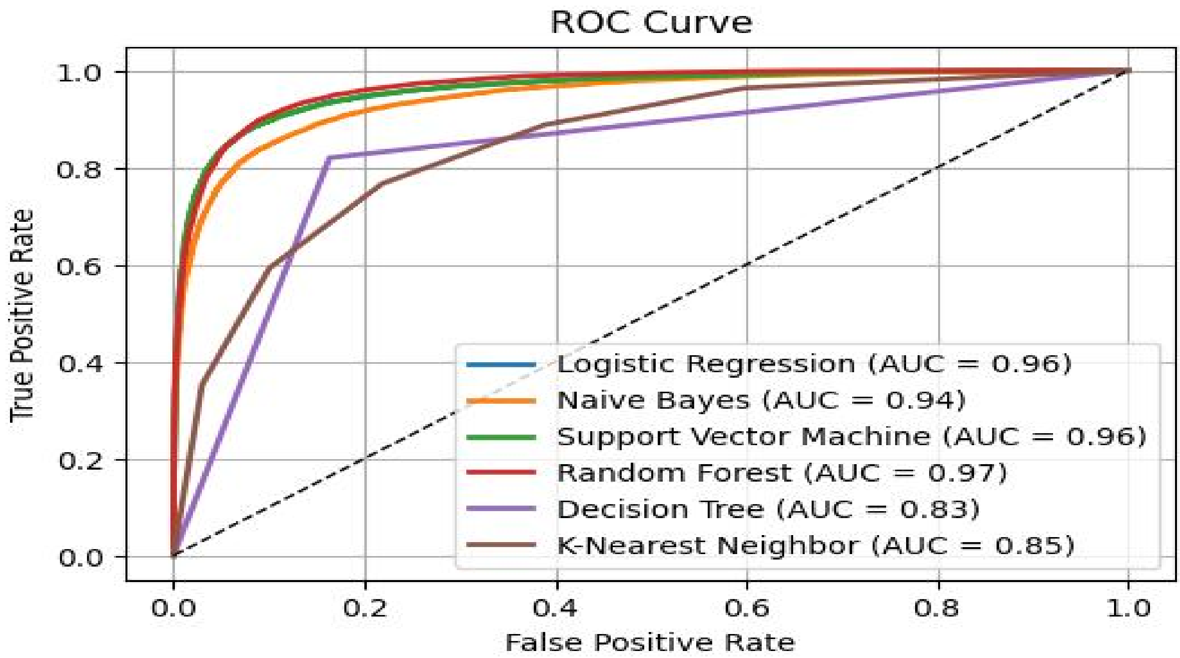
$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

V. ROC-AUC: The area under the receiver operating characteristic curve.

$$\textbf{AUC} = \frac{1}{2}\left(\frac{TP}{TP+FN} + \frac{TN}{TN+FP}\right)$$

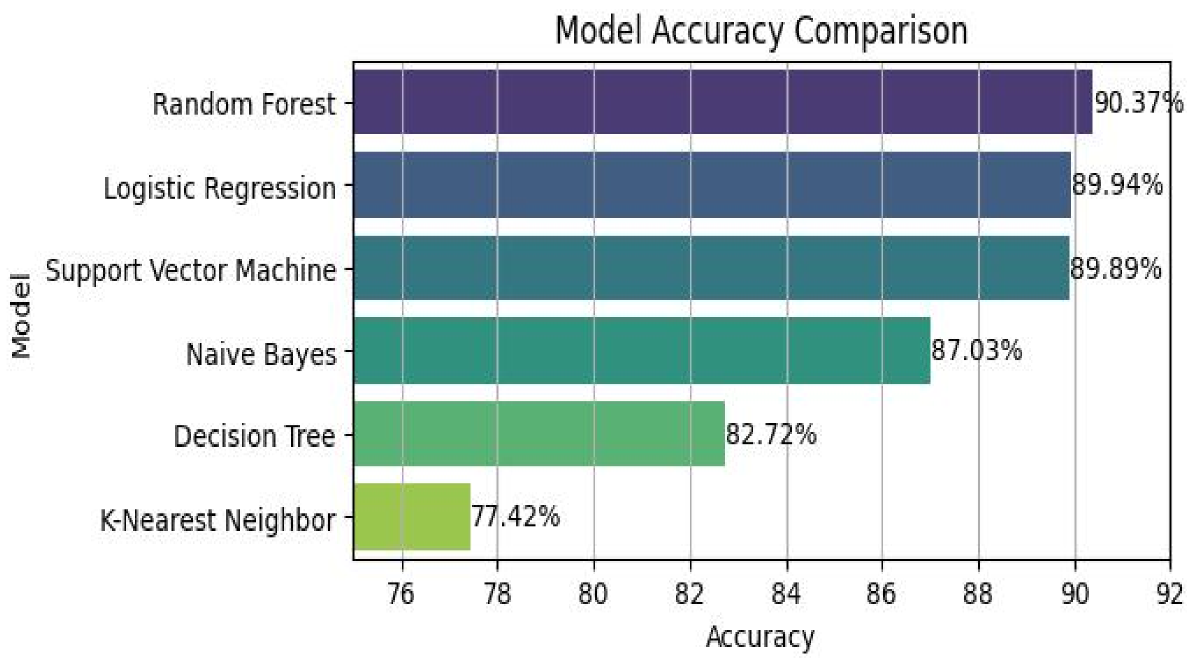
```python
# ROC Curve Comparison for All Models
lb = LabelBinarizer()
y_test_bin = lb.fit_transform(y_test).ravel()
plt.figure(figsize=(6, 4))
for name, model in models.items():
    if hasattr(model, "predict_proba"):
        probs = model.predict_proba(X_test)[:, 1]
    elif hasattr(model, "decision_function"):
        probs = model.decision_function(X_test)
        probs = (probs - probs.min()) / (probs.max() - probs.min())
    else:
        continue
    fpr, tpr, _ = roc_curve(y_test_bin, probs)
    roc_auc = auc(fpr, tpr)
    plt.plot(fpr, tpr, lw=2, label=f'{name} (AUC = {roc_auc:.2f})')

plt.plot([0, 1], [0, 1], 'k--', lw=1)
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve')
plt.legend(loc='lower right')
plt.grid()
plt.show()
```




```python
# Accuracy Bar Chart by Model
model_names = ['Random Forest', 'Logistic Regression', 'Support Vector Machine', 'Naive Bayes', 'Decision Tree', 'K-Nearest Neighbor']
accuracies = [0.9037, 0.8994, 0.8989, 0.8703, 0.8272, 0.7742]
accuracy_df = pd.DataFrame({'Model': model_names, 'Accuracy': [round(acc * 100, 2) for acc in accuracies]})

plt.figure(figsize=(6, 3))
barplot = sns.barplot(data=accuracy_df, x='Accuracy', y='Model', hue='Model', palette='viridis')
for container in barplot.containers:
    barplot.bar_label(container, labels=[f'{v.get_width():.2f}%' for v in container], label_type='edge')
plt.xlabel('Accuracy')
plt.title('Model Accuracy Comparison')
plt.xlim(75, 92)
plt.grid(axis='x')
plt.show()
```



7. CONCLUSION

This project shows that traditional machine learning models can classify sentiment in Amazon product reviews. By using the structured preprocessing, TF-IDF feature extraction, and testing six ML algorithms, we found that Random Forest had the highest accuracy at 90.37%. Logistic Regression and SVM followed closely behind. These results demonstrate the efficiency, clarity, and strong performance of classical ML models for binary sentiment classification. The findings confirm that these models are useful for real-time analysis in business settings.

8. FUTURE SCOPE

- Incorporating deep learning models like LSTM or BERT improves contextual understanding.
- Expanding to multi-class sentiment classification: positive, neutral, negative.
- Deploying the model as a web application or API allows real-time sentiment analysis.
- Including explainable AI (XAI) tools helps interpret model predictions.
- Integrating a feedback loop allows continuous learning from new incoming reviews.

9. LIMITATIONS

- Incorporating deep learning models like LSTM or BERT improves contextual understanding.
- The current model only manages binary classification: positive or negative. It does not address neutral or mixed sentiments.
- Contextual nuances, like sarcasm or idioms, are not captured well by the TF-IDF.
- The dataset is balanced, but it is limited to text-based product reviews from Amazon and may not work well for other platforms or domains.
- The approach assumes that reviews are in English, which limits its use in multiple languages.

10. BIBLIOGRAPHY

- Shah, A. (2021). *Sentiment Analysis of Product Reviews Using Machine Learning and Deep Learning*. International Journal of Engineering Research & Technology.
- Godia, A., & Tiwari, L.K. (2024). *Sentiment Analysis and Classification of Product Reviews using ML and GPT-4*. Proceedings of the AI Research Conference.
- Ghatora, M., Prakash, A., & Rani, S. (2024). *Comparative Study of Sentiment Analysis using Machine Learning and Large Language Models*. Journal of Intelligent Systems.
- Jagdale, R.S., Shirsat, V.S., & Deshmukh, S.N. (2019). *Sentiment Analysis on Product Reviews using Machine Learning Techniques*. International Journal of Computer Applications, 180(20).
- "Sentiment Analysis using Machine Learning Techniques" – IEEE Conference
- "Comparative Study of Text Classification Algorithms" – Springer Journal
- Kaggle Dataset – Amazon Reviews – *https://www.kaggle.com/datasets*
- Scikit-learn Documentation – *https://scikit-learn.org/stable/contents.html*
- WordCloud Documentation – *https://amueller.github.io/word_cloud/*
- Matplotlib Documentation – *https://matplotlib.org/stable/contents.html*
- Google Colab Notebook – *https://colab.research.google.com/drive/1yc7jMnlf-J6tPbu1l0sVSe_fzPKgNQ72?usp=drive_link*