S. B. JAIN INSTITUTE OF TECHNOLOGY, MANAGEMENT & RESEARCH, NAGPUR.

Practical No. 10

Aim: Load the mobile mood_data.txt data into a DataFrame, Extract independent variables (Xs) and dependent variables (Ys) into separate data objects, Build a model with Multinomial Naive Bayes, Random Forest, Random Forest (Entr opy), SVM and compare their accuracy for understanding sentiment Analysis.

Name of Student: Shrutika Pradeep Bagdi

Roll No.: CS22130

Semester/Year: IV/VII

Academic Session: 2025-2026

Date of Performance: __________

Date of Submission: __________

AIM: Write a Python program that reads the mood_data.txt (provided on LMS). The following are the given tasks, that has to be taken into consideration while constructing the solution.
Here dataset contains two columns where one is our target ("emotion" has 6 different categories) and another is the independent variable ("Text" contains data in form of sentences).

1. Load the mobile mood_data.txt data into a DataFrame
2. Generate tokens and remove punctuations, stop words and lower all rows
3. Join all the tokens as they were before and store them in a new column named "cleaned_text"
4. Now remove all single characters, extra space, and special characters and store processed data in a new column named "processed_text"
5. Create a final DataFrame containing dependent variable(emotion) and processed text
6. Extract independent variables (Xs) and dependent variables (Ys) into separate data objects
7. Generate tokens and do vectorization
8. Build a model with Multinomial Naive Bayes, Random Forest, Random Forest (Entr opy), SVM and compare their accuracy

OBJECTIVE/EXPECTED LEARNING OUTCOME:

- Understanding Vectorization.
- Understanding sentiment analysis.

HARDWARE AND SOFTWARE REQUIRMENTS:

Hardware Requirement: Computer System with high configurations

Software Requirement: Google Colab

THEORY:

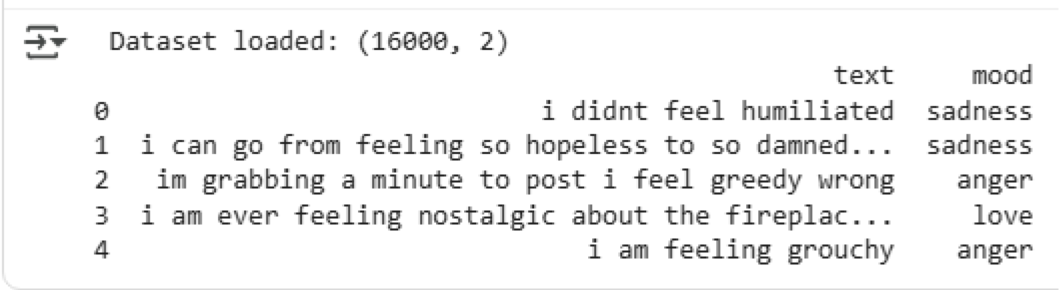
Sentiment analysis is the method of analyzing consumer sentiment using natural language processing, text analysis, and statistics. Many companies are aware of their customers' feelings — what they're doing, how they're saying it, and what they mean. Instead of reading word by word and trying to figure out its sentiment, nowadays with the advance of machine learning, human just let the machine read news or comments for us and it will answer the sentence's sentiment or Sentiment Analysis already has wide applications in our real life, especially in business. One of the most well-known applications of sentiment analysis is to provide a complete 360-degree view of how the name, product, or business is seen by consumers and stakeholders. Product feedback and social networking, for example, are widely accessible media that can reveal key information into whether the company is doing right or wrong. Companies may use sentiment analysis to assess the effectiveness of a new product, ad campaign, or other marketing initiatives meaning in the faster time.

CODE:

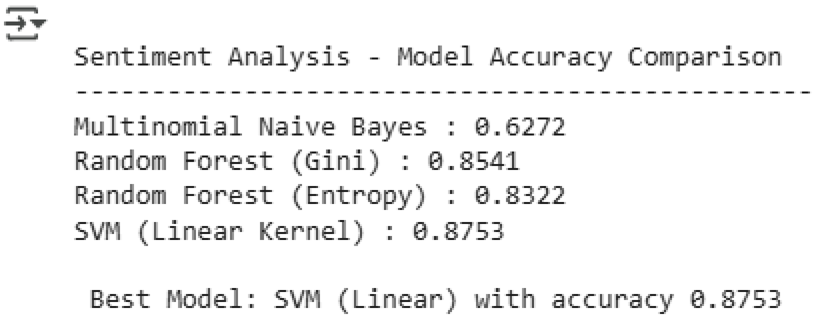
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
df = pd.read_csv("/content/mood_data.txt", header=None)
df = df.rename(columns={0: "text"})
df[["text", "mood"]] = df["text"].str.split(";", expand=True)
df = df.dropna(subset=["text", "mood"])
df = df[df["text"].str.strip() != ""]
print(" Dataset loaded:", df.shape)
print(df.head())
X = df["text"].astype(str)
y = df["mood"].astype(str)
tfidf = TfidfVectorizer()
X_tfidf = tfidf.fit_transform(X)
X_train, X_test, y_train, y_test = train_test_split(
X_tfidf, y, test_size=0.2, random_state=42, stratify=y)
nb_model = MultinomialNB()
nb_model.fit(X_train, y_train)
nb_acc = accuracy_score(y_test, nb_model.predict(X_test))
rf_gini = RandomForestClassifier(criterion="gini", random_state=42)
rf_gini.fit(X_train, y_train)
rf_gini_acc = accuracy_score(y_test, rf_gini.predict(X_test))
rf_entropy = RandomForestClassifier(criterion="entropy", random_state=42)
rf_entropy.fit(X_train, y_train)
```

```
rf_entropy_acc = accuracy_score(y_test, rf_entropy.predict(X_test))
svm_model = SVC(kernel="linear", random_state=42)
svm_model.fit(X_train, y_train)
svm_acc = accuracy_score(y_test, svm_model.predict(X_test))
print("\nSentiment Analysis - Model Accuracy Comparison")
print("------------------------------------------------")
print(f"Multinomial Naive Bayes : {nb_acc:.4f}")
print(f"Random Forest (Gini) : {rf_gini_acc:.4f}")
print(f"Random Forest (Entropy) : {rf_entropy_acc:.4f}")
print(f"SVM (Linear Kernel) : {svm_acc:.4f}")
accuracies = {
"Naive Bayes": nb_acc,
"Random Forest (Gini)": rf_gini_acc,
"Random Forest (Entropy)": rf_entropy_acc,
"SVM (Linear)": svm_acc,
}
best_model = max(accuracies, key=accuracies.get)
print("\n Best Model:", best_model, "with accuracy", f"{accuracies[best_model]:.4f}")
```

OUTPUT (SCREENSHOT):

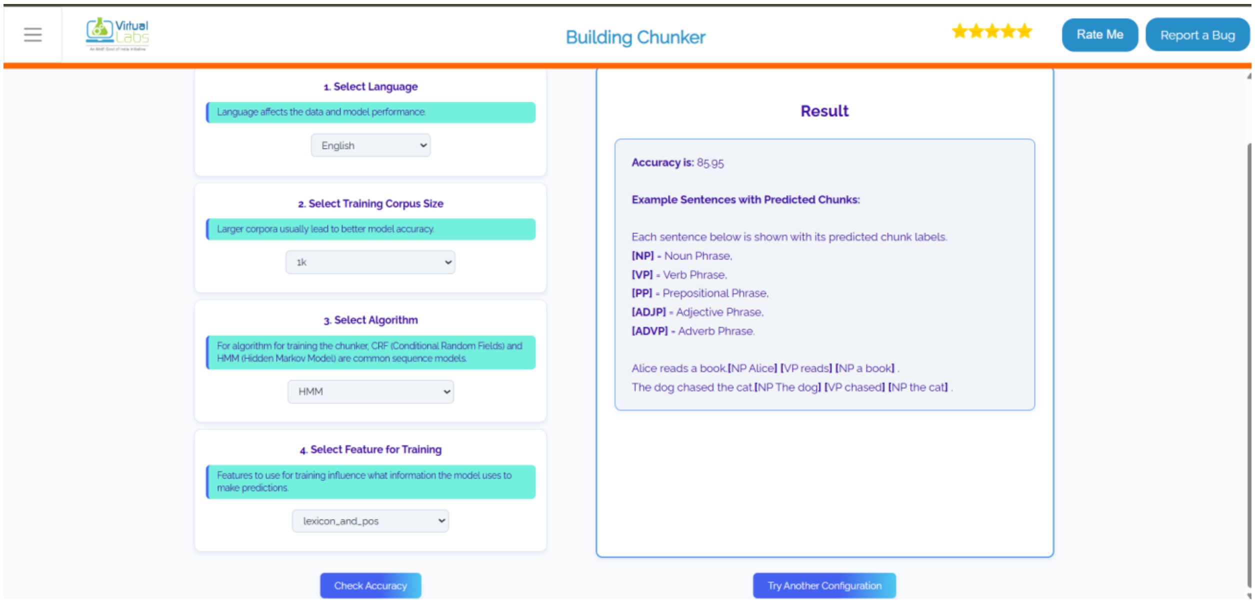


```
Dataset loaded: (16000, 2)
                                                text     mood
0                            i didnt feel humiliated  sadness
1  i can go from feeling so hopeless to so damned...  sadness
2   im grabbing a minute to post i feel greedy wrong    anger
3  i am ever feeling nostalgic about the fireplac...     love
4                               i am feeling grouchy    anger
```



```
Sentiment Analysis - Model Accuracy Comparison
------------------------------------------------
Multinomial Naive Bayes : 0.6272
Random Forest (Gini) : 0.8541
Random Forest (Entropy) : 0.8322
SVM (Linear Kernel) : 0.8753

 Best Model: SVM (Linear) with accuracy 0.8753
```



1. Select Language

Language affects the data and model performance.

English

2. Select Training Corpus Size

Larger corpora usually lead to better model accuracy.

1k

3. Select Algorithm

For algorithm for training the chunker, CRF (Conditional Random Fields) and HMM (Hidden Markov Model) are common sequence models.

HMM

4. Select Feature for Training

Features to use for training influence what information the model uses to make predictions.

lexicon_and_pos

Result

Accuracy is: 85.95

Example Sentences with Predicted Chunks:

Each sentence below is shown with its predicted chunk labels.
[NP] - Noun Phrase,
[VP] - Verb Phrase,
[PP] - Prepositional Phrase,
[ADJP] - Adjective Phrase,
[ADVP] - Adverb Phrase.

Alice reads a book.**[**NP Alice**]** **[**VP reads**]** **[**NP a book**]** .
The dog chased the cat.**[**NP The dog**]** **[**VP chased**]** **[**NP the cat**]** .

CONCLUSION:

I successfully Write a Python program that reads the mood_data.txt (provided on LMS). The following are the given tasks, that has to be taken into consideration while constructing the solution. Here dataset contains two columns where one is our target ("emotion" has 6 different categories) and another is the independent variable ("Text" contains data in form of sentences).

DISCUSSION AND VIVA VOCE:

- What is sentiment analysis?
- What are application of SA?

REFERENCE:

https://www.geeksforgeeks.org/machine-learning/what-is-sentiment-analysis/

https://en.wikipedia.org/wiki/Sentiment_analysis

https://learn.microsoft.com/en-us/azure/ai-services/language-service/sentiment-opinion-mining/overview