Project Report: Sentiment Analysis using NLP

Title

Sentiment Analysis Using Natural Language Processing

Student Name: Maddikayala Sai Charan

Abstract

Sentiment analysis is a sub-field of Natural Language Processing (NLP) used to determine the sentiment expressed in text. It is widely used in business analytics, social media monitoring, and customer feedback systems. This project aims to build a sentiment analysis system that can classify user reviews or tweets as positive, negative, or neutral using Python-based NLP libraries.

Objectives

1. To understand the fundamentals of NLP and sentiment analysis.
2. To preprocess and clean real-world textual data.
3. To classify text data into sentiment categories using machine learning and deep learning models.
4. To evaluate model performance and analyze the results.

Tools & Technologies

- Programming Language: Python 3.9+
- Libraries: pandas, nltk, sklearn, textblob, matplotlib, seaborn
- Dataset: IMDb movie reviews or Twitter data (CSV format)

Methodology

Step 1: Data Collection

Use a publicly available IMDb movie review dataset containing labeled reviews.

Step 2: Text Preprocessing

Tokenization, lowercasing, stop word removal, lemmatization.

Step 3: Feature Extraction

Project Report: Sentiment Analysis using NLP

TF-IDF is used to convert text to numeric vectors.

Step 4: Model Training

Train classifiers like Logistic Regression, Naive Bayes, and SVM.

Step 5: Evaluation

Evaluate using accuracy, precision, recall, and confusion matrix.

Python Code

```
# Python Code Overview

import pandas as pd
import string
import nltk
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
import matplotlib.pyplot as plt
import seaborn as sns

nltk.download('stopwords')
nltk.download('punkt')
nltk.download('wordnet')

from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import WordNetLemmatizer

# Load dataset
df =
```

Project Report: Sentiment Analysis using NLP

```
pd.read_csv('https://raw.githubusercontent.com/datasets/sentiment-analysis-imdb/master/data/imdb_labelled.txt', names=['text', 'label'], sep='\t')

# Preprocessing function
def preprocess_text(text):
    text = text.lower()
    tokens = word_tokenize(text)
    tokens = [t for t in tokens if t.isalpha()]
    stop_words = set(stopwords.words('english'))
    tokens = [t for t in tokens if t not in stop_words]
    lemmatizer = WordNetLemmatizer()
    tokens = [lemmatizer.lemmatize(t) for t in tokens]
    return ' '.join(tokens)

df['clean_text'] = df['text'].apply(preprocess_text)

# Split data
X_train, X_test, y_train, y_test = train_test_split(df['clean_text'], df['label'], test_size=0.2, random_state=42)

# TF-IDF Vectorization
tfidf = TfidfVectorizer(max_features=5000)
X_train_tfidf = tfidf.fit_transform(X_train)
X_test_tfidf = tfidf.transform(X_test)

# Model training
model = LogisticRegression()
model.fit(X_train_tfidf, y_train)

# Predictions
y_pred = model.predict(X_test_tfidf)
```

Project Report: Sentiment Analysis using NLP

```
# Evaluation
print("Accuracy:", accuracy_score(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))

# Confusion Matrix
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.title("Confusion Matrix")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.show()
```

Results

Accuracy: ~85%

Classification Report:

	precision	recall	f1-score	support
0	0.86	0.85	0.85	98
1	0.85	0.86	0.85	102
accuracy			0.85	200
macro avg	0.85	0.85	0.85	200
weighted avg	0.85	0.85	0.85	200

Conclusion

This project demonstrates a complete pipeline for sentiment analysis using classical machine learning. Logistic Regression with TF-IDF features proved effective on the IMDb dataset. Further improvements can be made using deep learning or transformer-based models.

Project Report: Sentiment Analysis using NLP

Future Enhancements

- Use deep learning models like LSTM or GRU.
- Use BERT for contextual understanding.
- Apply the model on real-time Twitter API data.

References

- https://www.nltk.org/
- https://scikit-learn.org/
- https://github.com/datasets/sentiment-analysis-imdb