Day 18 Training Report

16 July 2025

Introduction to Sentiment Analysis + Simple NLP Model Implementation

On **Day 18**, we applied preprocessing and TF-IDF concepts to **build simple NLP models**, focusing on **sentiment analysis**.

---

1. Objectives

- Understand **sentiment analysis** — identifying **positive, negative, or neutral opinions** in text.
- Implement a **basic NLP classification model** using Python and scikit-learn.

---

2. Steps in Sentiment Analysis

1. **Preprocessing:** Tokenization, stopwords removal, lemmatization.
2. **Feature Extraction:** TF-IDF vectorization of text data.
3. **Model Building:** Train classifiers like Logistic Regression or Naive Bayes.
4. **Evaluation:** Measure accuracy, precision, recall, and F1-score.

---

3. Example Implementation

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

# Sample data
texts = ["I love this product", "This is the worst!", "Amazing experience", "Not good"]
labels = [1, 0, 1, 0]  # 1=Positive, 0=Negative

# Vectorization
vectorizer = TfidfVectorizer()
X = vectorizer.fit_transform(texts)

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, labels, test_size=0.25, random_state=42)

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

# Prediction
```

```
y_pred = model.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
```

Conclusion

By the end of Day 18, students:

- Learned the **workflow for NLP model building**.
- Implemented **simple sentiment classification**.
- Applied preprocessing and TF-IDF for **real text datasets**.