# SOC MIDTERM REPORT

**Week 1: Python Fundamentals + Introduction to NLP**

Before starting with Natural Language Processing, I dedicated Week 1 to strengthening my **Python programming basics**, ensuring I was comfortable with the language's foundational concepts. This set the stage for implementing NLP algorithms efficiently.

**Topics Covered**

1. **Python Basics Refresher**

* **Data Types & Variables:**
  + Strings, integers, floats, booleans
  + Lists, tuples, dictionaries, and sets
* **Control Flow:**
  + if, elif, else conditions
  + for loops, while loops
* **Functions:**
  + Defined reusable blocks of logic using def
  + Used return statements and parameters effectively
* **String Operations:**
  + .lower(), .upper(), .replace(), .split(), slicing and concatenation
* **List Operations:**
  + Indexing, slicing, appending, iterating
* **Dictionary Operations:**
  + Key-value access, adding/updating values
* **Working with External Libraries:**
  + import nltk, import spacy, import sklearn
  + Used pip install for required packages

1. **Introduction to NLP**

* Understood the definition and **scope of NLP**
* Explored **real-life applications** of NLP:
  + Chatbots, spam filters, sentiment analysis, recommendation systems
* Set up the environment:
  + Practiced using **Google Colab** for running NLP code
  + Installed and imported necessary packages like nltk, spacy, scikit-learn.

**Week 2: Core NLP Concepts (Codebasics Playlist)**

This week focused on **basic NLP preprocessing techniques** using the **End-to-End NLP** playlist by Codebasics. These are essential steps to clean and prepare raw text data before applying any machine learning or deep learning algorithms.

**Topics Covered**

**🔹 Tokenization**

* **Definition:** Splitting a sentence or paragraph into words or sentences.
* **Tools Used:** nltk.tokenize.word\_tokenize, spacy tokenizer
* **Application:** Basic unit for further processing (e.g., stop word removal, vectorization)

**🔹 Stop Word Removal**

* **Definition:** Elimination of frequently used but insignificant words (e.g., "is", "the", "and").
* **Tools Used:** nltk.corpus.stopwords, spacy
* **Purpose:** Focus only on meaningful content

**🔹 Stemming**

* **Definition:** Crude process of chopping off word endings (e.g., “playing” → “play”).
* **Tools Used:** PorterStemmer, SnowballStemmer
* **Limitation:** May reduce words to non-dictionary roots.

**🔹 Lemmatization**

* **Definition:** Converts words to their meaningful dictionary form (e.g., “better” → “good”).
* **Tools Used:** WordNetLemmatizer in nltk, spacy lemmatizer
* **Advantage:** More accurate than stemming

**🔹 Part-of-Speech (POS) Tagging**

* **Definition:** Identifies the grammatical role of words in a sentence (noun, verb, adjective, etc.)
* **Tools Used:** nltk.pos\_tag, spacy
* **Use Case:** Improves lemmatization, parsing, and entity recognition

**🔹 Bag of Words (BoW)**

* **Definition:** Represents a document as a collection of word frequencies (sparse matrix).
* **Tools Used:** CountVectorizer from sklearn
* **Limitation:** Ignores word order and semantics

**🔹 TF-IDF (Term Frequency–Inverse Document Frequency)**

* **Definition:** Weighs words by their importance in a document relative to a corpus
* **Tools Used:** TfidfVectorizer
* **Advantage over BoW:** Reduces the impact of frequent but less important words

**🔹 Word Embeddings (Intro only)**

* **Concept:** Dense vector representation of words capturing semantic meaning
* **Mentioned:** word2vec, GloVe, but implementation postponed to later weeks

**Week 3: NLP with Classification and Vector Spaces (Coursera)**

In this week, I began the **first course of the Coursera NLP Specialization** offered by DeepLearning.AI. The course shifted the focus from text preprocessing to **mathematical representations of text** and **text classification basics**.

**Topics Covered**

**🔹 Vector Space Models**

* **Concept:** Representing text (words or documents) as vectors in a high-dimensional space
* **Why Important:** Enables mathematical operations like similarity, clustering, classification
* **Tools Used:** Custom NumPy implementations and scikit-learn vectorizers

**🔹 Similarity Measures**

* **Cosine Similarity:** Measures angle between vectors (common in NLP)
* **Dot Product and Euclidean Distance**
* **Application:** Document similarity, clustering, search engines

**🔹 Text Classification Intro**

* **What:** Assigning categories to text (e.g., spam vs. ham)
* **How:** Transforming preprocessed text into numeric features → training classifiers

**🔹 BoW and TF-IDF Revisited**

* Applied these techniques for real classification problems
* Understood **feature sparsity** and **dimensionality issues**

**Week 4: Naive Bayes and Logistic Regression**

This week emphasized supervised machine learning algorithms commonly used in NLP classification tasks.

**Topics Covered**

**🔹 Naive Bayes Classifier**

* **Generative model** using Bayes' Theorem and conditional probability
* **Types learned:**
  + Multinomial NB (used for text)
  + Bernoulli NB (used when features are binary)
* **Assumption:** All features are conditionally independent
* **Pros:** Fast, works well for text

**🔹 Logistic Regression**

* **Discriminative model** used for binary and multiclass classification
* **Used sigmoid function** to convert outputs into probability
* **Understood:**
  + Cost/loss functions
  + Gradient descent optimization

**🔹 Evaluation Metrics**

* **Accuracy, Precision, Recall, F1 Score**
* **Confusion Matrix** to analyze classification performance

| **Category** | **Skills Learned** |
| --- | --- |
| Python Programming | Syntax, functions, string ops, data structures |
| Text Preprocessing | Tokenization, stemming, lemmatization, stopwords, POS tagging |
| Feature Engineering | BoW, TF-IDF, vector representations |
| NLP Theory | Vector spaces, document similarity |
| Machine Learning for NLP | Naive Bayes, Logistic Regression |
| Evaluation | Similarity scores, classification metrics |