Basics of NLP, Text Processing, and Sentiment Analysis

Natural Language Processing (NLP) allows computers to **understand, interpret, and generate human language**. It is the backbone of applications like chatbots, machine translation, and sentiment analysis.

Basics of NLP

- NLP combines linguistics, computer science, and machine learning to process text and speech data.
- The primary goal is to convert human language into a structured format that machines can analyze.
- Key tasks include: tokenization, part-of-speech tagging, named entity recognition, and text classification.

Text Processing

Text data is often unstructured and requires preprocessing for analysis or modeling. Common text processing steps include:

1. **Tokenization**
   - Splitting text into individual units like words or sentences.
   - Example: “I love AI” → [“I”, “love”, “AI”]
2. **Lowercasing**
   - Converts all text to lowercase to maintain uniformity.
3. **Stop Words Removal**
   - Eliminates common words that do not carry significant meaning (e.g., “the”, “is”, “and”).
4. **Stemming and Lemmatization**
   - **Stemming:** Reduces words to their root form (e.g., “running” → “run”).
   - **Lemmatization:** Converts words to their dictionary form considering context.
5. **Vectorization**
   - Converts text into numerical representations for machine learning models.

    - Techniques: Bag-of-Words, TF-IDF, Word Embeddings (Word2Vec, GloVe).

Sentiment Analysis

Sentiment analysis is a **NLP task that identifies the emotional tone of text**—positive, negative, or neutral. It is widely used in social media monitoring, product reviews, and customer feedback analysis.

Steps for Sentiment Analysis:

1. Preprocess text (tokenization, stop word removal, stemming/lemmatization).
2. Convert text into numerical form (vectorization).
3. Train a classifier (e.g., Logistic Regression, Naive Bayes, or deep learning models).
4. Predict sentiment on new data.

Python Example (Basic Sentiment Analysis using TextBlob):

```
from textblob import TextBlob


text = "I love this product! It works perfectly."

blob = TextBlob(text)


# Sentiment polarity (-1 to 1)

print("Sentiment Polarity:", blob.sentiment.polarity)


# Sentiment classification

if blob.sentiment.polarity > 0:

    print("Positive sentiment")

elif blob.sentiment.polarity < 0:

    print("Negative sentiment")

else:

    print("Neutral sentiment")
```

Applications of Text Processing and Sentiment Analysis

- Social media monitoring to gauge public opinion.
- Customer feedback and product review analysis.
- Market research and brand reputation management.
- Chatbots and virtual assistants for personalized responses.

By mastering **text processing and sentiment analysis**, you can extract valuable insights from unstructured text data and build AI systems that understand human emotions and opinions.