**National University of Computer and Emerging Sciences**



**Natural Language Processing**

**Project Report**

Qasim Naveed 21L-5231

Taha Sheikh 21L-5175

Rameez 21L-1775

Afnan Asif 21L-1864

**Introduction**

This project implements an Aspect-Based Sentiment Analysis (ABSA) Recommender System to analyze product reviews and suggest personalized recommendations to users. The system focuses on breaking down reviews into specific aspects (e.g., Fit, Fabric, Price), assessing sentiments for these aspects, and matching them with user preferences to rank products. The approach integrates NLP techniques, machine learning, and recommendation logic.

This report will explain the project in detail, covering each section, its relevance, and additional insights to provide context.

**1. Data Loading and Exploration**

**Purpose**

Data loading is the initial step that provides the foundation for the entire analysis. We import and inspect the dataset to understand its structure and identify potential preprocessing needs.

**What Was Done**

1. **Dataset Description**:
   * The dataset contains columns like Review Text, Rating, Recommended IND, and metadata about clothing items.
   * Each review serves as input to assess user sentiments for various product aspects.
2. **Exploratory Data Analysis (EDA)**:
   * Analyzed the distribution of ratings and reviews to identify trends.
   * Finding out common words present in positive and negative reviews.

**Why It Matters**

EDA helps:

* Understand the characteristics of the data (e.g., review length distribution).
* Spot issues like imbalanced classes or missing values that could affect model performance.
* Provide a baseline for feature engineering.

**2. Data Preprocessing**

**Purpose**

Raw text reviews are unstructured and noisy, requiring cleaning and transformation into a format suitable for analysis. Preprocessing ensures consistency and improves the performance of downstream models.

**What Was Done**

1. **Text Cleaning**:
   * Removed punctuation, special characters, and numbers.
   * Lowercased all text to standardize it.

**Concept**: Text normalization reduces variability caused by casing and irrelevant symbols, simplifying token matching during analysis.

1. **Stopword Removal**:
   * Excluded common words (e.g., "the", "and") that do not convey meaningful information.

**Concept**: Stopwords, while syntactically important, often dilute the semantic relevance of a review.

1. **Sentence Splitting**:
   * Split reviews into sentences using punctuation and conjunctions (e.g., "and", "because").
   * Used SpaCy's sentence segmentation capabilities.

**Why?**: ABSA requires analyzing sentiments for each aspect separately. Splitting reviews enables granular assessment.

1. **Lemmatization**:
   * Reduced words to their base forms (e.g., "running" → "run").

**Concept**: Lemmatization minimizes vocabulary size, improving model efficiency and reducing noise.

**Why It Matters**

Preprocessing transforms chaotic, unstructured text into a clean, structured form. It enables effective tokenization, reduces computational overhead, and improves model interpretability.

**3. Aspect Classification**

**Purpose**

The goal is to classify each sentence into predefined aspects like "Fit", "Fabric", or "Price". This step structures the problem into an Aspect-Based Sentiment Analysis (ABSA) task.

**What Was Done**

1. **Aspect Identification**:
   * Predefined aspects (e.g., Fit, Fabric) are matched with sentences based on keywords and rules.
   * A fallback classification ("None of these") filters irrelevant sentences.
2. **Why AI Based?**
   * Rule-based systems are simpler to implement and effective when aspect categories are well-defined however we do not have such data.
   * Machine learning models (e.g., BERT) can replace this if dataset annotations exist for aspect-specific training which we have created using local LLM.
3. **Challenges and Concept**:
   * Context ambiguity: Sentences like "This is perfect for a formal occasion" could refer to multiple aspects.
   * Manual rules simplify initial implementation but may miss subtle relationships.

**Why It Matters**

Aspect identification structures unstructured reviews into analyzable components. Without this, downstream sentiment analysis cannot pinpoint customer feedback on specific product attributes.

**4. Sentiment Analysis**

**Purpose**

For each identified aspect, the project assesses whether the sentiment expressed in the sentence is positive or negative.

**What Was Done**

1. **Sentiment Scoring**:
   * Positive sentences contribute a score of +1 and negative sentences a score of -1.
   * Irrelevant sentences are excluded from scoring.
2. **Sentiment Aggregation**:
   * Aspect scores are averaged across all sentences for a product.

**Concept**: Aggregating scores provides a normalized measure of customer feedback for each aspect.

1. **Used Approach**:
   * Train a sentiment classifier using Bidrectional LSTM and Glove embeddings.

**Why It Matters**

Sentiment analysis quantifies qualitative feedback, enabling numerical representation of customer opinions. This step is critical for scoring and ranking products effectively.

**5. Recommendation Logic**

**Purpose**

The recommender system matches user preferences with product scores to provide personalized suggestions.

**What Was Done**

1. **User Preferences**:
   * Users rate the importance of aspects like "Fit", "Price", or "Durability".
2. **Product Scoring**:
   * Products are scored based on aspect sentiment analysis.
3. **Similarity Computation**:
   * A weighted dot product calculates the match between user preferences and product scores.
   * Products are ranked in descending order of similarity.
4. **Output**:
   * The system returns the top n recommendations (e.g., top 5).

**Why It Matters**

By aligning user preferences with product features, the system ensures recommendations are relevant and personalized. The dot product ensures efficient computation even for large datasets.

**6. Deployment**

**Purpose**

The project includes a Flask-based web application to expose the recommender system via a user-friendly interface.

**What Was Done**

1. **HTML/CSS/JavaScript**:
   * Designed an interactive dashboard displaying recommended products.
   * Dynamic elements render recommendations in real-time.
2. **Backend Integration**:
   * Flask routes serve APIs to fetch user preferences and product recommendations.
   * SQLite database manages user and product data.
3. **Concept**:
   * Separation of concerns ensures scalability. The backend handles logic, while the frontend enhances user experience.

**Why It Matters**

Deployment bridges the gap between development and real-world application. A user-friendly interface increases accessibility and adoption.

**Conclusion**

This project demonstrates the complete lifecycle of an **Aspect-Based Sentiment Analysis Recommender System**, from data preprocessing to deployment. Throughout our projects we faced various issues as was faced by any one else doing the similar work of sentiment analysis. They ranged from having unbalanced dataset, to using stopwords removal to improve accuracy, to trying out different architectures better suited for your problem. However, we tackled them as much as we can by researching about those problems. In this project we tried a new version of sentiment analysis by going into more granular level of a review and tried to identify and capture the actual emotions and message of the user so that we can have better grasp of user's preferences. Throughout this project we learned new techniques of data preprocessing, develop a better understanding of LSTM, tried new embeddings named as Glove and grasped the concept of transformers and implemented them.