**Text Mining Project Report: Movie Review Sentiment Analysis**

**1. Introduction**

This project implements a complete text mining pipeline for sentiment analysis of movie reviews. The system collects reviews from Letterboxd, processes them through various stages of text normalization and cleaning, and uses machine learning to classify sentiments into positive, negative, and neutral categories.

**2. Dataset Description**

**Data Collection Process**

1. **Web Scraping Implementation**

* Used LetterboxdScraper.py to collect reviews from Letterboxd
* Targeted 25 different films including:
* Blue Ruin, The Rover, Cold in July
* A Hijacking, Loveless, Victoria
* Timecrimes, The Guilty, Calibre
* And more...
* Collected both highest and lowest rated reviews for each film
* Implemented rate limiting (0.5-1.5s delay between requests)
* Used BeautifulSoup for HTML parsing

1. **Language Filtering**

* Implemented FilterEnglishComments.py using langdetect
* Ensured all reviews are in English
* Handled encoding issues with UTF-8
* Removed non-English content to maintain data quality

**Dataset Characteristics**

* **Size**: Over 300 reviews (meeting course requirements)
* **Balance**: Equal distribution between positive and negative reviews
* **Format**: CSV with columns:
* film: Movie title
* sort\_type: Review rating category
* review: Original review text
* clean\_text: Processed review text
* sentiment: Label (positive/negative/neutral)

**3. Methods**

**Text Preprocessing Pipeline**

1. **Normalization (NormalizationCode.py)**

* Custom stopwords removal (extensive list of 150+ words)
* Slang dictionary replacement (100+ entries)
* Character repetition reduction
* Special character removal
* Number removal
* Whitespace normalization

1. **Feature Extraction**

* Word count
* Average word length
* Punctuation analysis
* Sentence count
* Emphasis ratio
* Capitalization analysis
* Unique word ratio
* Long/short word ratios

**Vectorization**

* **TF-IDF Implementation**
* max\_features: 100,000
* ngram\_range: (1, 6)
* min\_df: 1
* max\_df: 0.99
* sublinear\_tf: True
* strip\_accents: 'unicode'
* analyzer: 'word'
* token\_pattern: r'(?u)\b\w+\b'

**Model Implementation**

1. **Naive Bayes Classifier**

* MultinomialNB implementation
* Grid Search optimization:
* alpha: [0.00001, 0.0001, 0.001, 0.01, 0.1]
* max\_features: [80000, 100000, 120000]
* ngram\_range: [(1, 5), (1, 6), (2, 6)]
* min\_df: [1, 2]
* max\_df: [0.95, 0.99]

1. **Cross-validation**

* 5-fold cross-validation
* F1-weighted scoring

**4. Experimental Results**

**Model Performance Metrics**

* Accuracy Score
* F1 Score (Weighted)
* Precision Score (Weighted)
* Recall Score (Weighted)

**Visualizations Generated**

1. **Sentiment Distribution**

* Bar plot showing class distribution
* Saved as 'sentiment\_distribution.png'

1. **Confusion Matrix**

* Heatmap visualization
* Class-wise performance analysis
* Saved as 'confusion\_matrix.png'

1. **Learning Curves**

* Training vs validation performance
* Model convergence analysis
* Saved as 'learning\_curves.png'

1. **Feature Importance**

* Top features for each class
* Heatmap visualization
* Saved as 'feature\_importance\_heatmap.png'

1. **Cross-validation Scores**

* Distribution plot
* Performance stability analysis
* Saved as 'cv\_scores\_distribution.png'

**Feature Analysis**

* Top 100 most important features for each class
* N-gram analysis (1-6 grams)
* Feature importance visualization

**5. Conclusions**

**Key Achievements**

1. **Data Quality**

* Successfully collected and processed movie reviews
* Implemented robust text normalization
* Maintained balanced dataset

1. **Model Performance**

* Effective sentiment classification
* Good balance between precision and recall
* Stable cross-validation results

**Technical Implementation**

* **Code Organization**
* Modular design with separate scripts for each stage
* Comprehensive error handling
* Detailed logging and output generation
* **Output Management**
* Automated file naming with timestamps
* Organized output directory structure
* Comprehensive logging of metrics

**Future Improvements**

1. **Model Enhancement**

* Implement deep learning models (LSTM, Transformer)
* Experiment with word embeddings
* Try ensemble methods

1. **Feature Engineering**

* Add more sophisticated text features
* Implement sentiment-specific features
* Explore domain-specific preprocessing

1. **Data Expansion**

* Increase dataset size
* Add more movie genres
* Include more diverse review sources

**Technical Requirements**

* Python 3.x
* Key Libraries:
* scikit-learn
* pandas
* NLTK
* BeautifulSoup
* langdetect
* matplotlib
* seaborn