Intensity Analysis

An intelligent system using NLP to predict the intensity in the text reviews.

Author: Hema Moravapalli

Course: Data Science Boot Camp, UpGrad.

Table of Contents:

1. **Introduction**
2. **Dataset Overview**
3. **Methodology**
   3.1 Exploratory Data Analysis (EDA)
   3.2 Data Augmentation
   3.3 Data Pre-processing
   3.4 Feature Engineering
   3.5 Model Selection
   3.6 Model Training
   3.7 Model Validation
4. **Results and Discussion**
5. **Future Work**
6. **Conclusion**

---

1. Introduction

In an era dominated by textual communication, understanding and analysing text intensity has become increasingly significant. Text intensity reflects the tone or strength of emotional or factual emphasis in communication. Accurately predicting intensity can be crucial for applications in sentiment analysis, customer feedback processing, and behavioural analysis.

This project aims to build a machine learning model to classify text data into varying types of intensity (angriness, happiness, and sadness). By leveraging a labelled dataset and implementing advanced machine learning techniques, the project seeks to provide accurate and actionable predictions for text intensity types.

---

2. Dataset Overview

The dataset used comprises 2039 text samples, each labelled with its corresponding intensity type (e.g., angriness, happiness, and sadness). Given the small dataset size, augmentation techniques were employed to improve model performance.

Dataset Highlights:

- **Total Samples:** Approximately 2,000.
- **Classes:** 3 intensity levels - Angriness, Happiness, and Sadness
- **Key Features:** Text samples, associated metadata like length, punctuation density.
- **Challenges:** High variance in text length.

---

3. Methodology

3.1 Exploratory Data Analysis (EDA)

EDA was performed to uncover patterns and characteristics within the data, including:

- **Descriptive Statistics:** Average text length, word counts, and distribution of intensity levels.
- **Visualizations:** Bar plots of class distributions, word clouds for each intensity level, and histogram plots of text lengths.

3.2 Data Augmentation

Data augmentation involves creating additional training data by modifying or expanding the original data. It helps improve model performance by increasing dataset diversity without requiring manual labelling of more data. This is especially useful when we have a small dataset.

Techniques for Text Data Augmentation:

1. Synonym Replacement: Replace words with their synonyms using a thesaurus or libraries like Word Net.
2. Random Insertion: Insert random words or synonyms at random positions in the text.
3. Random Deletion: Randomly remove words from the text with a certain probability.
4. Back Translation: Translate the text to another language and then back to the original language to generate paraphrased data.
5. Token Shuffling: Randomly shuffle the words in the text while keeping the context intact.
6. Paraphrasing: Use pre-trained models like Pegasus or T5 to generate paraphrased sentences.

3.3 Data Pre-processing

Data pre-processing steps included:

- **Text Cleaning:** Removal of special characters, stop words using nltk library.
- **Tokenization:** Splitting text into individual words for analysis.
- **Stemming and Lemmatization:** Standardizing words to their base forms for better feature extraction.

3.4 Feature Engineering

Feature engineering was critical to enhance model performance:

- **TF-IDF Vectors:** Converting text into numerical form based on term frequency and inverse document frequency.
- **Custom Features:** Incorporating punctuation density and text length as additional predictors.

3.5 Model Selection

Various machine learning and deep learning models were tested:

- **Traditional Models:** Logistic Regression, SVM, Multinomial Naive Bayes, KNN, Decision Trees, Random Forest.
- **Deep Learning Models:** CNN, Bidirectional LSTM.
- **Tuning:** Hyper parameter tuning was conducted using grid search and random search for optimal results.

3.6 Model Training

Data was split into training and testing sets (80/20). Model training involved:

- Applying cross-validation for robustness.
- Fine-tuning model parameters to optimize performance.

3.7 Model Validation

- **Metrics Evaluated**: Accuracy, Precision, Recall, F1 Score, and Confusion Matrices were used for validation.

---

4. Results and Discussion

Key findings from model evaluation include:

- **Best Model:** The Support Vector Classifier achieved the highest accuracy of 96.32%, outperforming other models.
- **Deep Learning Models:** The CNN and LSTM models showed promising results but were slightly outperformed by SVC in this task.
- **Feature Importance:** Sentiment scores and TF-IDF vectors were highly influential in predicting intensity levels.

Model Performance Metrics:

Model Name	Accuracy Score	Precision Score	Recall Score	F1 Score
Best Tuned Support Vector Classifier	0.963220	0.963257	0.963220	0.963224

Best Tuned K-Nearest Neighbors	0.959134	0.959294	0.959134	0.959111
Best Tuned Random Forest Classifier	0.955864	0.955864	0.955864	0.955862
Random Forest Classifier	0.954638	0.954642	0.954638	0.954638
LSTM-Long Short Term Memory	0.950143	0.950377	0.950143	0.950085
LSTM-Long Short Term Memory After Tuning	0.949734	0.950107	0.949734	0.949708
CNN Model After Tuning	0.949734	0.949918	0.949734	0.949759
CNN Model	0.948508	0.948869	0.948508	0.948566
Best Tuned Logistic Regression	0.944013	0.944061	0.944013	0.944023
Support Vector Classifier	0.931753	0.931730	0.931753	0.931720
K-Nearest Neighbors	0.926032	0.927144	0.926032	0.926016
Logistic Regression	0.922763	0.922727	0.922763	0.922638
Decision Tree Classifier	0.922763	0.923346	0.922763	0.922798
Best Tuned Decision Tree Classifier	0.922763	0.923346	0.922763	0.922798
Best Tuned Naive Bayes	0.866776	0.867405	0.866776	0.866983
Naive Bayes	0.852064	0.853558	0.852064	0.852435

Key Takeaways:

1. **Top Performers**:
    - The **Best Tuned SVC** model outperformed all others, demonstrating its robustness and suitability for this classification problem.
    - **KNN (Best Tuned)** and **Random Forest (Best Tuned)** also showed excellent results, making them strong alternatives to SVC.
2. **Deep Learning Models**: While **LSTM** and **CNN** models performed well (above 94% accuracy), they didn't outperform traditional machine learning models like SVC or tuned KNN. This suggests that simpler models are better suited for this specific dataset.
3. **Logistic Regression**: Logistic Regression performed exceptionally well post-tuning, with competitive accuracy and F1 scores, proving its reliability for text classification tasks.
4. **Ensemble Models**: Random Forest demonstrated strong performance, although it was slightly behind the top-performing models like SVC and tuned KNN.
5. **Naive Bayes**: Naive Bayes showed relatively weaker performance, indicating it might not be the best fit for this dataset's complexity

---

5. Future Work

- **Advanced Augmentation**: Explore additional augmentation methods to enhance data diversity further.
- **Real-World Applications**: Investigate the model's performance on real-world datasets.
- **Hybrid Models**: Experiment with ensemble techniques combining traditional and deep learning models.

---

6. Conclusion

This project demonstrated a comprehensive pipeline for text intensity prediction, from data exploration to model evaluation. This project demonstrated that data augmentation could effectively mitigate the challenges of small datasets. The Support Vector Classifier emerged as the top-performing model, showcasing its suitability for text classification. Both traditional and deep learning models performed well, with the Random Forest and tuned CNN models achieving exceptional results.

The insights gained from this analysis pave the way for further enhancements, enabling robust and scalable solutions for text intensity analysis in real-world applications.