Group 5: Yifan Zhang, Jackson Zhao, Yueming Xu, Zi Ting Ina Leung, Prisha Samdarshi

Social Media Post Classification by MBTI Personality Type

1. Problem Statement

The growing use of social media has generated a large amount of data that can reveal important insights into human behavior and personality. By analyzing the relationship between MBTI[1] personality types and social media behavior using NLP and ML techniques, our group hopes to identify distinct linguistic patterns for each personality type. The project then aims to develop models to predict MBTI classifications based on social media text data.

The project's findings can be applied to various areas, such as enhanced personalization, improved communication strategies, human resource management, and mental health support.

2. Data Description

The dataset used for this project contains more than 8,600 rows of data in Kaggle[2], with each row representing an individual's MBTI type and their last 50 social media posts. This rich dataset is ideal for NLP analysis, allowing us to study the relationship between personality type and language usage.

3. Exploratory Data Analysis

In order to explore the data further, we removed the 20 most common words as the top 20 words were very similar among all 16 MBTI types.

As shown in ***Figure 1***, most of the language used across the posts uses neutral language, which suggests that the content is more descriptive and informational than emotional. Higher positive than negative sentiment suggests that people express themselves more when experiencing positive emotions. This may be due to the nature of social media use, where people may only want to show the best moments of their lives. Since there aren't any obvious differences between each MBTI type, it suggests that personality type may not significantly impact the emotional tone of people's posts.

We further analyzed the sentiment ratio (positive vs. negative) by MBTI type ***(Figure 2)***. This chart shows a clearer difference where Feeling types have a higher positive-to-negative ratio whilst Thinking types have lower ones. This suggests that Feeling types which are more associated with expressing their feelings use more positive language compared to Thinking types which are more associated with logic and analysis language.

In ***Figure 3***, we analysed the length of each social media post to see if there are differences between the MBTI types. Most MBTI types wrote posts with around 600 words on average. Some MBTI types wrote longer posts whilst others wrote shorter ones. However, if we compare based on dichotomy, there were no obvious differences in the number of words.

[1] The Myers-Briggs Type Indicator (MBTI), a well-known personality model, categorizes individuals into 16 distinct types based on four categories: extraversion/introversion, sensing/intuition, thinking/feeling, and judging/perceiving.

[2] https://www.kaggle.com/datasets/datasnaek/mbti-type/code

To further dissect the word choice by each MBTI type, we created unigram and bigram word clouds. In ***Figure 4***, the most common unigrams among all MBTI types are "friend", "want", and "way". There were slight differences in word choice between each dichotomy pair. For example, extroverts used more action-oriented words, indicating their willingness to engage in social interaction, whilst introverts used more reflective words, suggesting a focus on internal thoughts and feelings. In ***Figure 5***, bigrams associated with memories were most common, highlighting that people were reflecting on past and present experiences, and relationships that may be important in forming their personalities.

4. Preprocessing

Cleaning and Sampling

To prepare social media posts for analysis, we cleaned the data to ensure consistency and suitability for NLP tasks. This involved removing special characters, URLs, and stopwords, converting text to lowercase, tokenizing into individual words, and lemmatizing to group similar terms. These steps were crucial for identifying linguistic patterns and improving model performance in predicting MBTI personality types. The 16 MBTI types were label-encoded and the data training, validation, and test sets, bringing the overall split to 60-20-20. Including a validations set allowed for hyperparameter tuning limiting the risk of contamination on the test set.

Feature Engineering and Class Balancing

Our first step in feature engineering was using TF-IDF to capture the importance of each word across posts relative to the 16 MBTI personality types. In order to choose the most important words, we limited TF-IDF to the top 5,000 features. Next, we employed elastic-net regularization with cross validation to further select important features and reduce noise. A variance threshold of 0.0001 was set to filter out features with low variance that will not be useful predictors in the final model.

Our data cleaning process ended with handling data imbalance. We used SMOTE to generate synthetic samples so models were not biased towards the majority classes (ESFJ, ENFJ, ENFP, etc,). The number of samples for each class were balanced at 1,099.

5. Classification Models

Model selection

For the model selection process, we used the F1-score as the metrics. Due to the imbalanced MBTI dataset, the F1-score was prioritized to account for the balance between precision and recall. This ensures the chosen models perform well in identifying both classes effectively. Then we performed the K-Nearest Neighbors (KNN), Decision Tree, Kernel SVM, Logistic Regression, AdaBoost, XGBoost, Random Forest, and Multi-Layer Perceptron (MLP) models for evaluation.

The kernel SVM, XGBoost, logistic regression, and Random Forest models demonstrated the highest F1-scores, with 61.44%, 61.28%, 62.19%, and 61.17%, respectively. These scores indicate their superior ability to handle the imbalanced data compared to other models. Other models, such as Decision Tree, and Multi-Layer Perceptron, underperformed F1-score. Models like AdaBoost and KNN showed notably poor results, making them unsuitable for further consideration.

We focused on tuning the hyperparameters of Logistic Regression, Random Forest, Kernel SVM, and XGBoost to optimize their performance and tried to find the best model to predict people's MBTI.

Tunning Hyperparamter

Our group used the Random Search Cross-Validation to identify promising hyperparameter configurations for multiple models. Random Search was selected for its efficiency in exploring a wide range of hyperparameter values within a limited time. This method helps identify high-performing regions in the hyperparameter space without exhaustive grid searches.

The results showed that the Random Forest model outperformed other models, achieving the highest F1-score of 92.10%. The Kernel SVM followed closely with an F1-score of 90.79%, while XGBoost achieved 89.85% with a tree depth of 6 and a learning rate of 0.24. Logistic Regression demonstrated performance with an F1-score of 87.89%, leveraging L1 regularization and a regularization strength of C=8.07. These results provided a solid foundation for further tuning, shown in ***Figure 6***.

After building on the results of Random Search, we utilized Bayesian Optimization to refine the hyperparameters of the Random Forest model. Bayesian optimization was chosen because it systematically explores the hyperparameter space, balancing exploration and exploitation. This approach is particularly effective for fine-tuning models with complex parameter interactions, like Random Forest. As a result, the optimization process improved the model's performance, increasing its F1-score from 92.10% to 92.57%. The optimal hyperparameters included a maximum tree depth of 31, minimum samples per leaf of 1, minimum samples for a split of 2, and 300 estimators. These values were then used to train the Random Forest model on the entire dataset for deployment and testing.

Test Set Evaluation

The final evaluation on the test dataset showed that random forest had an F1-score of 62.97% . While the tuned model demonstrated strong performance during cross-validation, the metrics on the test set highlighted the potential for further improvements.

6. Conclusions and Future Work

Overall, the random forest had the best F1-score on test data among all the models, and our group concluded that random forest can best identify distinct linguistic patterns for each personality MBTI type. The results underscore the importance of combining multiple tuning methods to achieve optimal performance. Despite the improvements from tuning, the gap between validation and test set performance suggested that we may have overfit the validation set and would need further enhancements to reduce overfitting. Future steps may include addressing overfitting and improving feature engineering. These refinements could further improve the model's reliability and application to real-world data.

Appendix

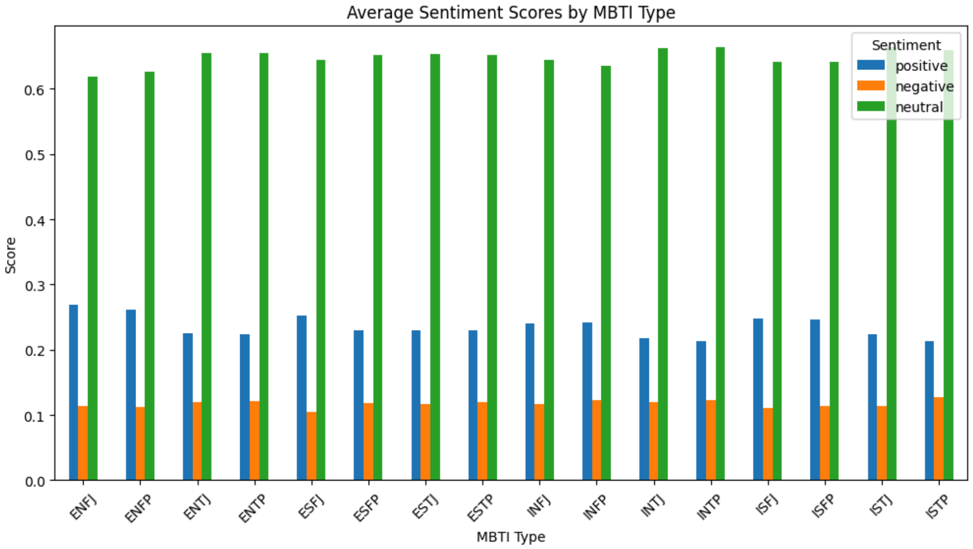


Figure 1: Average Sentiment Scores by MBTI Type

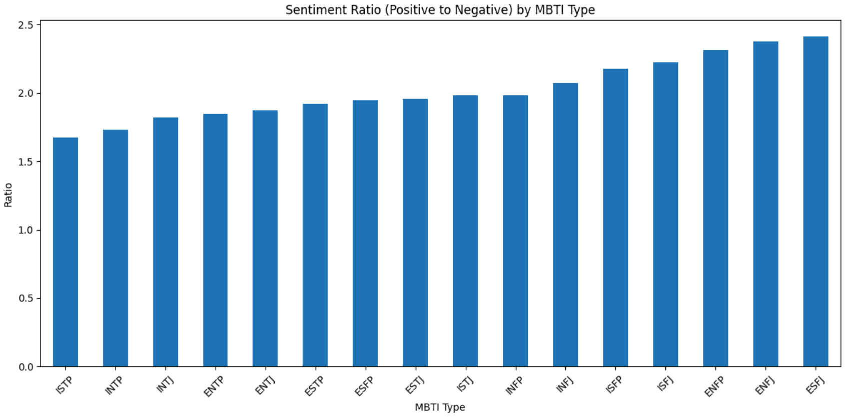


Figure 2: Sentiment Ratio by MBTI Type

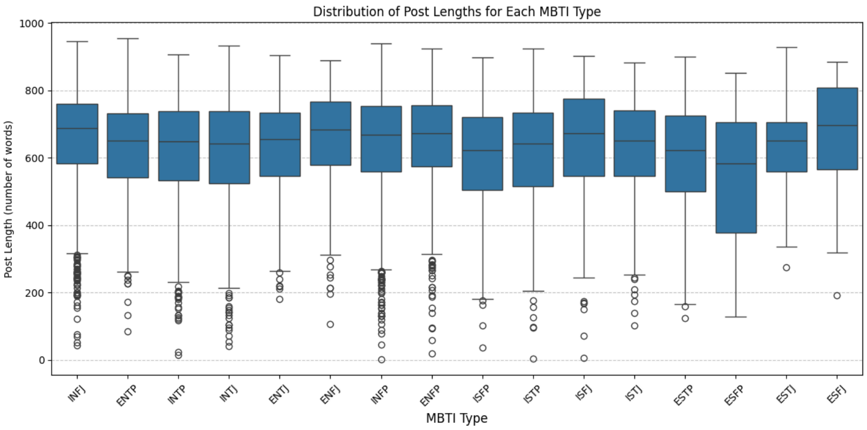
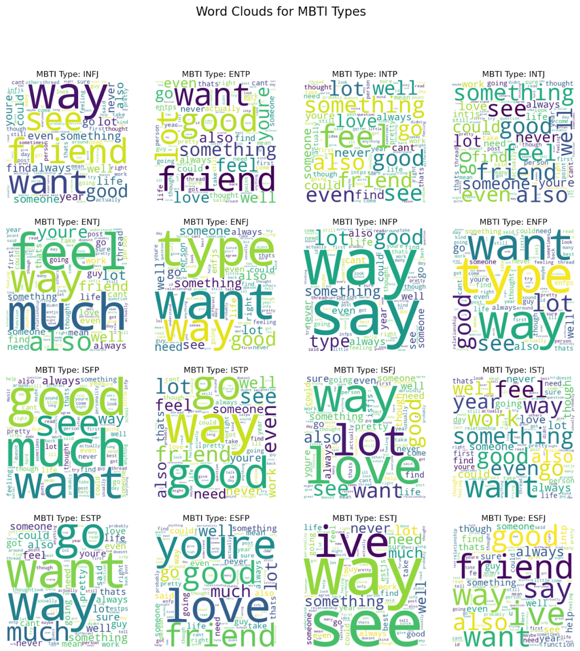
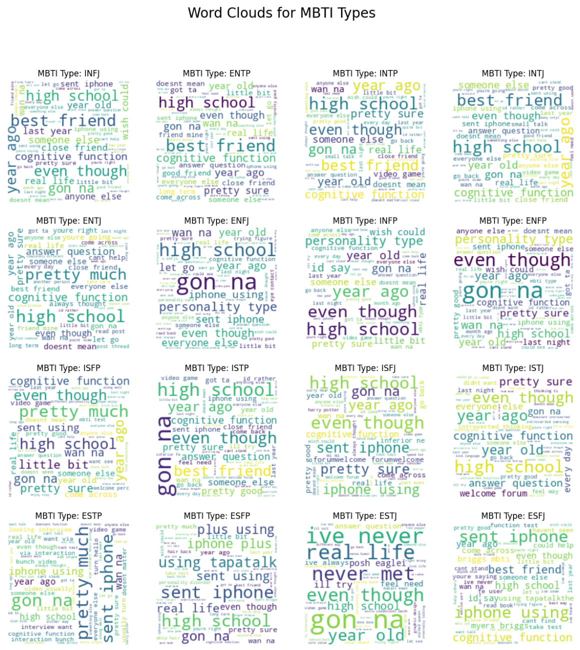


Figure 3: Distribution of Post Lengths for Each MBTI Type


Word Clouds for MBTI Types

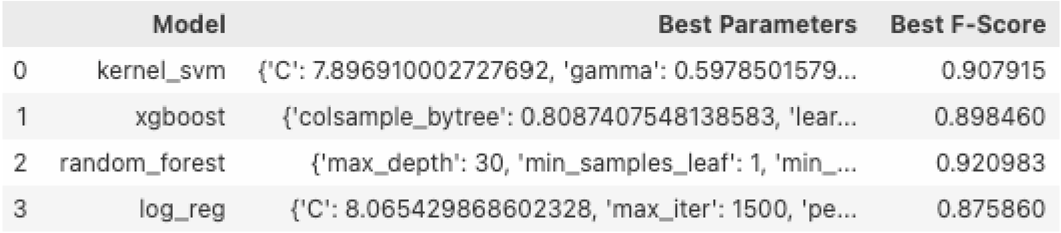
MBTI Type: INFJ · MBTI Type: ENTP · MBTI Type: INTP · MBTI Type: INTJ · MBTI Type: ENTJ · MBTI Type: ENFJ · MBTI Type: INFP · MBTI Type: ENFP · MBTI Type: ISFP · MBTI Type: ISTP · MBTI Type: ISFJ · MBTI Type: ISTJ · MBTI Type: ESTP · MBTI Type: ESFP · MBTI Type: ESTJ · MBTI Type: ESFJ

Figure 4: Word Cloud (Unigram)


Word Clouds for MBTI Types

MBTI Type: INFJ · MBTI Type: ENTP · MBTI Type: INTP · MBTI Type: INTJ · MBTI Type: ENTJ · MBTI Type: ENFJ · MBTI Type: INFP · MBTI Type: ENFP · MBTI Type: ISFP · MBTI Type: ISTP · MBTI Type: ISFJ · MBTI Type: ISTJ · MBTI Type: ESTP · MBTI Type: ESFP · MBTI Type: ESTJ · MBTI Type: ESFJ

Figure 5: Word Cloud (Bigram)



	Model	Best Parameters	Best F-Score
0	kernel_svm	{'C': 7.896910002727692, 'gamma': 0.5978501579...	0.907915
1	xgboost	{'colsample_bytree': 0.8087407548138583, 'lear...	0.898460
2	random_forest	{'max_depth': 30, 'min_samples_leaf': 1, 'min_...	0.920983
3	log_reg	{'C': 8.065429868602328, 'max_iter': 1500, 'pe...	0.875860

Figure 6: Parameter tuning by RandomSearchCV