Text-Based Prediction of MBTI Personality Types

Project Category: Natural Language Processing
Project Mentor: Michael Xie

Daniel Kim
Stanford University
ddk@stanford.edu

Jenny Mai
Stanford University
mcmai@stanford.edu

Emily Nguyen
Stanford University
eqnguyen@stanford.edu

1 Introduction

Although the concept of personality tests has been around for over 100 years, all of these personality tests follow the same structure: classify the user's personality based on their answers to a fixed set of questions. However, we explore how personality types can be indirectly inferred from text that a user has written. To narrow the scope of our project, we focus on predicting a user's Myers-Briggs Type Indicator (MBTI) from their tweets. Typically, a user's MBTI is determined by taking a introspective questionnaire indicating differing psychological preferences in how they perceive the world and make decisions. Users are assigned four categories: introversion or extraversion, sensing or intuition, thinking or feeling, and judging or perceiving, to ultimately produce a four-letter result (e.g. "ISFJ" or "ENTP") [1].

The input to our algorithm is tweets from a given user. We then apply various machine learning models, particularly multinomial naive Bayes, multiclass logistic regression, and linear support vector classification matrix, to output a predicted MBTI result.

2 Related Work

Many researchers have attempted to predict an author's personality based on their writing. In the realm of social media specifically, researchers in this field have hypothesized that there is a strong correlation between user behavior and personality type.

Sakdipat Ontoum and Jonathan H. Chan's *Personality Type Based on Myers-Briggs Type Indicator with Text Posting Style by using Traditional and Deep Learning* analyzes the efficacy of various machine learning techniques on this problem, including Naive Bayes, SVMs, and Recurrent Neural Networks [2]. Bayu Yudha Pratama and Riyanarto Sarno's *Personality Classification Based on Twitter Text using Naive Bayes, KNN, and SVM* conducts a similar analysis using different techniques for the Big 5 Personality Traits [3]. While Ontoum and Chan find that Recurrent Neural Networks perform best (49% accuracy), Yudha and Sarno find that Naive Bayes performs best (60% accuracy). We classify personality using the Myers-Briggs Type Indicator and use Naive Bayes and SVMs (among other techniques).

Kim Luyckx and Walter Daelemans' *Using syntactic features to predict author personality from text* used a 200,000 word *Personae* Corpus to predict the personality of 145 students [4]. They use Memory-Based Shallow Parsing (MBSP) to extract features from the Corpus. MBSP conducts part-of-speech analysis and look for specific grammatical relations within sentences. They then conduct 4 binary classification tasks, each of which corresponds to the different poles of the Myers-Briggs Type Indicator. We use a similar approach to compartmentalize our personality classification task. However, our decision to use tweets make MBSP infeasible because of the short length and informal language & sentence construction of Tweets.

B. Plank and D. Hovy's *Personality traits on twitter—or—how to get 1,500 personality tests in a week* [5] and B. Verhoeven et. al's *Twisty: a multilingual twitter stylometry corpus for gender and personality profiling* [6] both using n-gram features with logistic regression to predict Myer-Briggs personalities from Tweets . Both studies found Sensing vs. Intuition to be the easiest classification problem and Judging vs. Perceiving to be the most difficult. While we use logistic regression, we do not use n-grams. Our dataset is too small (approx. 8000 Tweets) to train models using n-grams.

3 Dataset and Features

We used the (MBTI) Myers-Briggs Personality Type Dataset from Kaggle [7], which includes over 8600 rows of Twitter users' last 50 Twitter posts and their respective MBTI labels. After exploring the data, we found that our dataset was heavily skewed towards introverted types, and that some of the most popular words for each personality type were the MBTI types themselves. [8]

	type	posts
0	INFJ	'http://www.youtube.com/watch?v=qsXHcwe3krw|||...
1	ENTP	'I'm finding the lack of me in these posts ver...
2	INTP	'Good one _____ https://www.youtube.com/wat...
3	INTJ	'Dear INTP, I enjoyed our conversation the o...
4	ENTJ	'You're fired.|||That's another silly misconce...

Table 1: Raw data

When we cleaned the data, we parsed out each Twitter post from each user and removed any hyperlinks, emojis, special characters, and punctuation marks since we were only interested in the words. In addition, we lemmatized our data using NLTK's WordNetLemmatizer. For example, if we have the word "feet", this would be lemmatized into the word "foot". This process also ensures that words with the same base meanings are classified together to increase the accuracy of the classification. We also removed stop words, or common terms that appear in most or all documents, and made all words lowercase.

After cleaning our data, we randomly divided our data into training and test data with a 80:20 ratio. We decided to numerically represent our data using a measure known as "term frequency-inverse document frequency" (TF-IDF), which measures the relevance of a word is to a document within multiple documents. For a term i within a document j, the TF-IDF weight $w_{i,j}$ is found by

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right), \tag{1}$$

where $tf_{i,j}$ is the frequency of the term i in document j, df_i is the number of documents containing term i, and N being the total number of documents.

We used a TF-IDF vectorizer on the training and test set independently to avoid data leakage. We also ignored words that appeared in less than 1% of all documents, or words that appeared with less than around 69 users, which led our vocabulary to be restricted to around 4781 words.

4 Methods

4.1 Synthetic Minority Oversampling Technique

When exploring our dataset, we found there was a very imbalanced distribution of MBTI types, as shown below in Figure 1. [9]

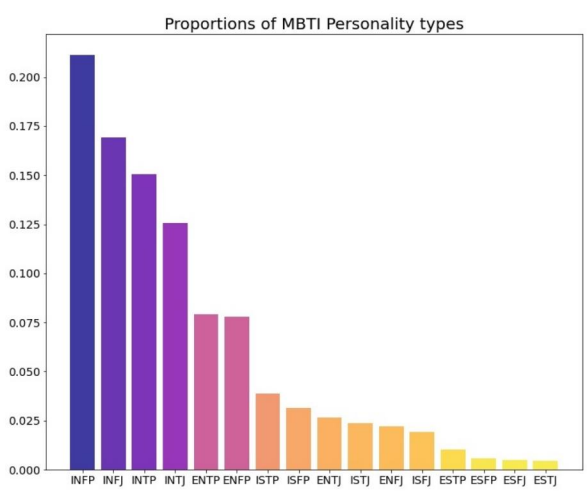

Figure 1: MBTI distribution

To address the class imbalance in our dataset, we used the Synthethic Minority Oversampling Technique (SMOTE) to oversample the minority class. SMOTE works by first randomly selecting a minority class instance, then choosing one if its k nearest neighbors. Then, a synthetic example is created by drawing a line between the examples in the feature space and drawing a new sample at a point along that line. This new synthesized sample is then added to the dataset to increase representation of the minority classes.

4.2 Multinomial Naive Bayes

We used a Multinomial Naive Bayes algorithm as a baseline algorithm for MBTI classification due to its simplicity and ease of implementation. This algorithm is based on Bayes' theorem and calculates the probability of each MBTI type for a given sample and returns the type with the highest probability as output. It assumes that the tokens in a tweet are conditionally independent of each other, given the MBTI type. The optimal parameters are calculated with laplace smoothing as follows:

$$\phi_{j|y=c} = \frac{1 + \sum_{i=1}^{n} 1\{x_j^{(i)} = 1 \wedge y^{(i)} = c\}}{|V| + \sum_{i=1}^{n} 1\{y^{(i)} = c\}} \tag{2}$$

where $c \in \{1, \cdots, 16\}$, referring to the numerical labels for each of the 16 different MBTI personality types, and $|V|$ represents the number of words in the vocabulary.

4.3 Multiclass Logistic Regression

Logistic regression is an algorithm that also calculates the probability of each MBTI type from the Twitter posts, but does so without assuming the words are conditionally independent. Thus, logistic regression makes a prediction based on a direct functional form by calculating weights based on the data given. To classify between multiple classes, we use the one-vs-rest heuristic method which divides a multi-class classification problem into multiple binary classification problems. In our problem, we train sixteen binary classifiers, one for each MBTI personality label. The goal is to find weights θ that maximize the log-likelihood:

$$\underset{\theta}{\operatorname{argmax}} \sum_{i=1}^{n} y^{(i)} \log h(x^{(i)} + (1 - y^{(i)}) \log(1 - h(x^{(i)}) \tag{3}$$

We use the sigmoid function for our hypothesis function.

$$h_\theta(x) = g(\theta^T x) = \frac{1}{1 + \exp -\theta^T x} \tag{4}$$

Stochastic gradient descent is used to find the weights θ that maximize the log-likelihood:

$$\theta_j := \theta_j + \alpha(y^{(i)} - h_\theta(x^{(i)})x_j^{(i)} \tag{5}$$

4.4 Linear Support Vector Machines

Linear Support Vector Machines (SVM) are classification models that linearly separates data into different classes by

optimizing the geometric margin between different data labels. To allow the linear SVM to classfiy between multiple classes, we apply the same measure as in logistic regression and use the one-vs-rest heuristic, separating between one MBTI type versus the other fifteen MBTI types. The goal of the linear SVM classifer is to minimize the functional margins γ of individual examples:

$$\gamma = \min_{i=1,\cdots,n} \gamma^{(i)} \tag{6}$$

The individual geometric margin (with squared-hinge loss and L2 regularization) of (w, b) with respect to a single training example $(x^{(i)}, y((i))$ is given by

$$\gamma^{(i)} = \frac{1}{n}\sum_{i=1}^{n}(\max(0, 1 - y^{(i)}(w^T x^{(i)} + b)) + \lambda||w||_2^2 \tag{7}$$

5 Experiments/Results/Discussion

5.1 Evaluation Metrics

In order to evaluate our models, we used the total accuracy as well as the accuracy of each category. We also used the standard classification metrics for each class: precision, recall, and f1-score metrics. The four metrics are defined as follows:

$$\text{accuracy} = \frac{\text{correct predictions}}{\text{total predictions}}$$

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{f1} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

where TP, TN, FP, FN correspond to the number of true positive, true negative, false positive, and false negative predictions.

5.2 Hyperparameters

For Multinomial Naive Bayes, we used a Laplace smoothing parameter of $\alpha = 0.1$ to account for zero probability of certain tokens. In Multiclass Logistic Regression, we used a one-vs-rest (OvR) to use a binary classification algorithm for our multi-class classification. We optimized with the L-BFGS algorithm and an L2-regularization to prevent overfitting. Finally, in Linear SVM, we again used OvR with L2-regularization and Squared Hinge Loss.

5.3 Experiments and Results

After training all of the models using sklearn's library implementation, we tested each of the models on the testing dataset and calculated the accuracies, precision, recall, and F1-scores with the predicted results. The results are summarized below for each model.

5.3.1 Multinomial Naive Bayes

Multinomial Naive Bayes resulted in an overall accuracy of 50.2%, which is decent, but lower in comparison to the other models and the categorical accuracies. This is likely because Multinomial Naive Bayes relies on assumptions that features are conditionally independent of each other, given the class. However, we see that this assumption is not particularly valid in our case, because the features, in this case tokenized words, are actually dependent.

Multinomial Naive Bayes	
Overall Accuracy	0.502
E/I Accuracy	0.775
N/S Accuracy	0.857
F/T Accuracy	0.783
J/P Accuracy	0.658
Precision	0.52
Recall	0.5

Table 2: Multinomial Naive Bayes Summary

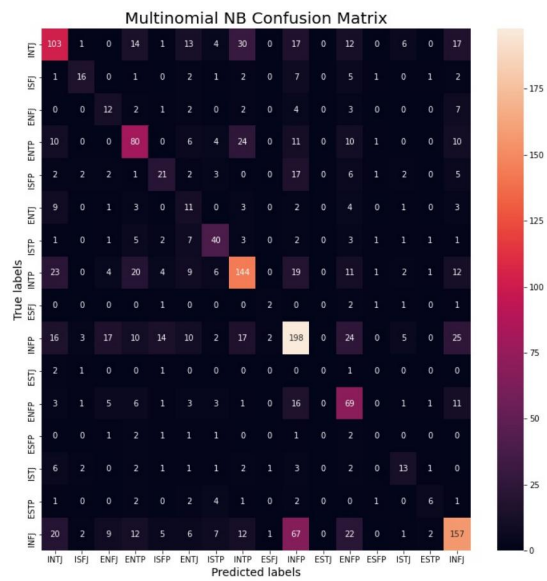

Figure 2: Multinomial Naive Bayes Confusion Matrix

From the confusion matrix, we see that predictions for the 'INFP', 'INTP', and 'INFJ' classes were most accurate, which are the top three most popular MBTI types from our distribution. This makes sense because there is more information on these popular MBTI types, so Multinomial Naive Bayes is able to classify better on these types than on types with less Twitter posts.

5.3.2 Multiclass Logistic Regression

Multiclass Logistic Regression resulted in an overall accuracy of 67.8%, which was the highest among the three models. This makes sense because Multiclass Logistic Regres-

sion is a discriminative model and makes less assumptions than Naive Bayes, and doesn't assume the data are independent. In addition, since we have a sufficiently large dataset, this is in favor of logistic regression as the output is a more generalized model. Overall, Multiclass Logistic Regression has lower bias and higher variance.

Multiclass Logistic Regression	
Overall Accuracy	0.678
E/I Accuracy	0.843
N/S Accuracy	0.883
F/T Accuracy	0.858
J/P Accuracy	0.788
Precision	0.68
Recall	0.68

Table 3: Multiclass Logistic Regression Summary

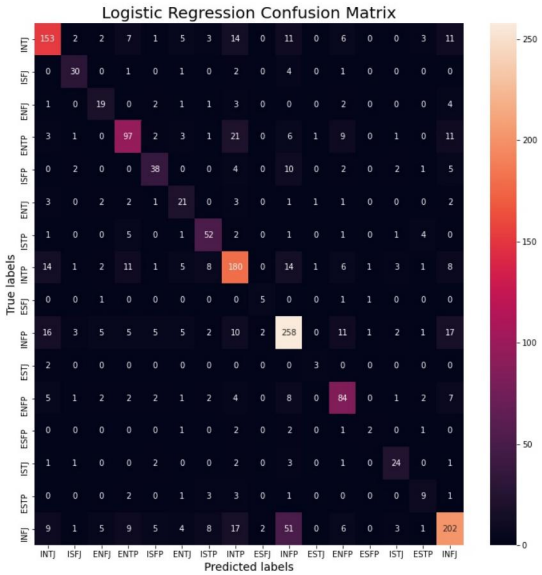

Figure 3: Multiclass Logistic Regression Confusion Matrix

Again, from our confusion matrix, we see that the most precise prediction are from the top three MBTI types: 'INFP', 'INFJ', and 'INTP'. However, unlike Multinomial Naive Bayes, there are less false positives among these classes, i.e. Multiclass Logistic Regression does not make as many predictions in these classes. This is most likely due to the fact that Logistic Regression has less assumptions about the data and thus, with our data, is more accurate as a result.

5.3.3 Linear SVC

Linear SVC resulted in an overall accuracy of 62.1%, which is higher than Multinomial Naive Bayes, but lower than Multiclass Logistic Regression. This is because SVMs also don't make the faulty assumptions that Naive Bayes does, but rather tries to create a hyperplane to geometrically separate the data into classes. However, this is not as effective as logistic regression because the data are not linearly separable because there is correlation among the MBTI personality types.

Linear SVC	
Overall Accuracy	0.621
E/I Accuracy	0.835
N/S Accuracy	0.895
F/T Accuracy	0.831
J/P Accuracy	0.761
Precision	0.62
Recall	0.62

Table 4: Linear SVC Summary

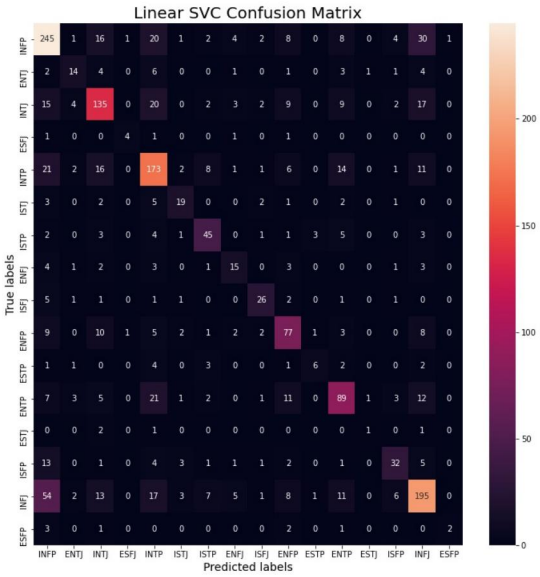

Figure 4: Linear SVC Confusion Matrix

From the confusion matrix, we see the problem in trying to linearly separate the data—there are a lot of false predictions among MBTI personality types that differ by one category. For example, mistaking 'INFP' for 'INFJ' and vice versa, and 'ENTP' for 'INTP' and vice versa. This shows that Linear SVC does not perform as well when the dataset has overlapping classes.

Across all of the models, we see that the categorical accuracies were relatively high (over 80%), but the 'J' (Judging) vs. 'P' (Perceiving) accuracy was significantly lower in comparison. This is interesting because the 'J' and 'P' distribution was relatively balanced in the dataset. Therefore, this suggests the fourth category of the MBTI personality tests may be very related and difficult to distinguish.

6 Conclusion/Future Work

Our results are significant in indirectly inferring personality types from text that a user has written. After preprocessing

multiple Twitter posts from over 8600 users, we applied TF-IDF weighting to each term in all posts and balanced the dataset since it was skewed towards the introverted types. After applying multinomial Naive Bayes, multiclass Logistic Regression, and linear Support Vector Machines, we found that multi-class Logistic Regression does the best with an accuracy of 67%, followed by the linear Support Vector Classifier with 62% accuracy and Multinomial Naive Bayes with 50% accuracy. This makes sense as Naive Bayes has the faulty assumption that the words in each Twitter post are conditionally independent of each other, when in reality they are not independent. In addition, our results indicate that the data itself is not well linearly separable, since the linear Support Vector Machine does worse than logistic regression.

However, there is room for improvement. Given more time, we would experiment more with preprocessing, cleaning the data more with regards to different regular expressions. We would also further explore the lower accuracy in predicting the Judging vs. Perceiving component. In addition, we would expand our dataset to not only include Twitter posts but also other forms of user text, which would make our models more robust to different people's personality types. If we had more time and computational resources, we could also look into using deep learning models, particularly the Bidirectional Encoder Representations from Transformers (BERT) model.

7 Contributions

Jenny worked on data exploration, including looking at the raw data and the distribution of MBTI types among the Twitter posts. She also preprocessed and vectorized the data into TF-IDF weighting to use the models on later. In the report, Jenny worked on the "Dataset and Features" and "Methods" subsections.

Daniel worked on data exploration and the "Related Work" section of the report.

Emily worked on balancing the dataset using SMOTE and training/testing the models. She also did model analysis, calculating accuracy, precision, recall, and F1-scores. In the report, Emily worked on the "Experiments/Results/Discussion" section of the report, as well as designed the poster.

References

[1] The myers and briggs foundation - mbti basics. 2022.

[2] Sakdipat Ontoum and Jonathan H. Chan. Personality type based on myers-briggs type indicator with text posting style by using traditional and deep learning. 2022.

[3] Yudha Bayu Pratama and Riyanarto Sarno. Personality classification based on twitter text using naive bayes, knn and svm. *2015 International Conference on Data and Software Engineering (ICoDSE)*.

[4] Kim Luyckx and Walter Daelemans. Using syntactic features to predict author personality from text. *Proceedings of digital humanities*, pages 146–149, 2008.

[5] Barbara Plank and Dirk Hovy. Personality traits on twitter—or—how to get 1,500 personality tests in a week. *Proceedings of the 6th workshop on computational approaches to subjectivity, sentiment and social media analysis*, 2015.

[6] Ben Verhoeven, Walter Daelemans, and Barbara Plank. Twisty: a multilingual twitter stylometry corpus for gender and personality profiling. In *Proceedings of the 10th Annual Conference on Language Resources and Evaluation (LREC 2016)/Calzolari, Nicoletta [edit.]; et al.*, pages 1–6, 2016.

[7] Michell J. (mbti) myers-briggs personality type dataset. 2017.

[8] The pandas development team. pandas-dev/pandas: Pandas, February 2020.

[9] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020.