Predicting MBTI using Multi-layer Perceptrons

Introduction

The Myers-Briggs Type Indicator is one of a number of pseudoscientific measures in the field of personality theory, the foundation of which is in Carl Jung's work in cognitive psychology and psychological types. The 16 MBTI types are well-defined but unfortunately rely mainly on self-reporting, which is inherently biased.

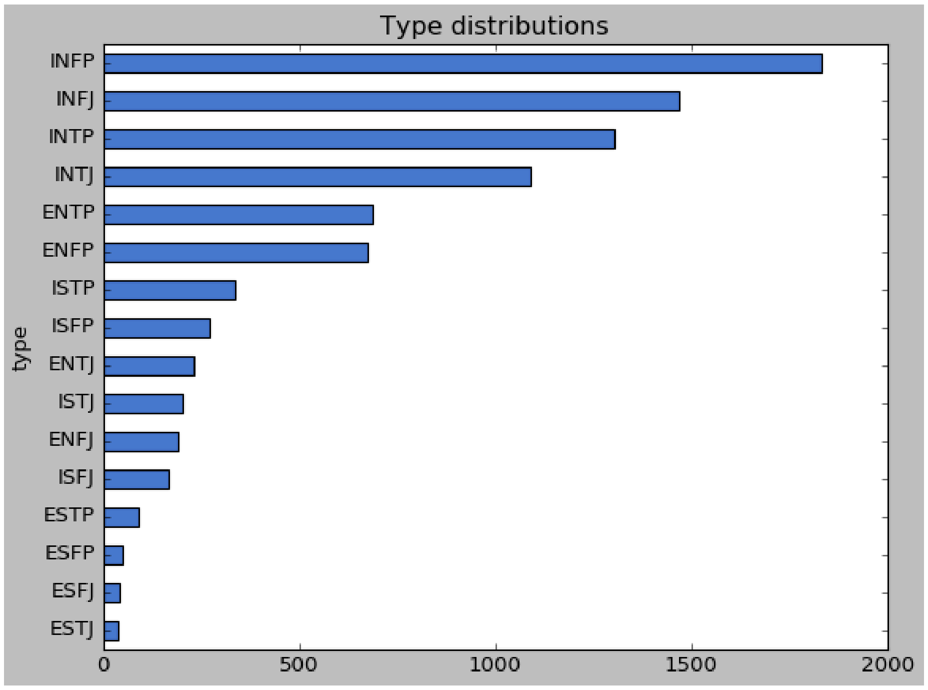
This project seeks to improve type classification using machine learning to build a model that takes in text (in this case, posts from an online forum about personality theory) and returns a prediction of the most likely personality type of the user or writer of that text. The production of an accurate natural language based classifier would be a significant advancement in the field of personality psychology, indicating a demonstrable connection between natural language and personality type.

Dataset

The data for this project were obtained from a Kaggle dataset with nearly 9000 samples from personalitycafe.com, each consisting of a person's MBTI type and a set of text written by that person. Larger datasets which would be usable for this purpose do exist, for example this 600MB dataset containing over one million posts scraped from Reddit, but due to technological limitations I worked with the smaller dataset (26MB) for this project. This project could be done again using the larger dataset or by scraping data from other social media sites, for example, Twitter posts, or even Instagram or Tiktok comments, but this would require the collection of each user's actual MBTI type by other means.

Problems specific to the data used for this project were non-uniform distributions (Fig. 1), which were extremely disproportionate to the generally accepted population distributions, and the presence of non-meaningful text, such as stopwords and hyperlinks.

Fig. 1: Original class distribution

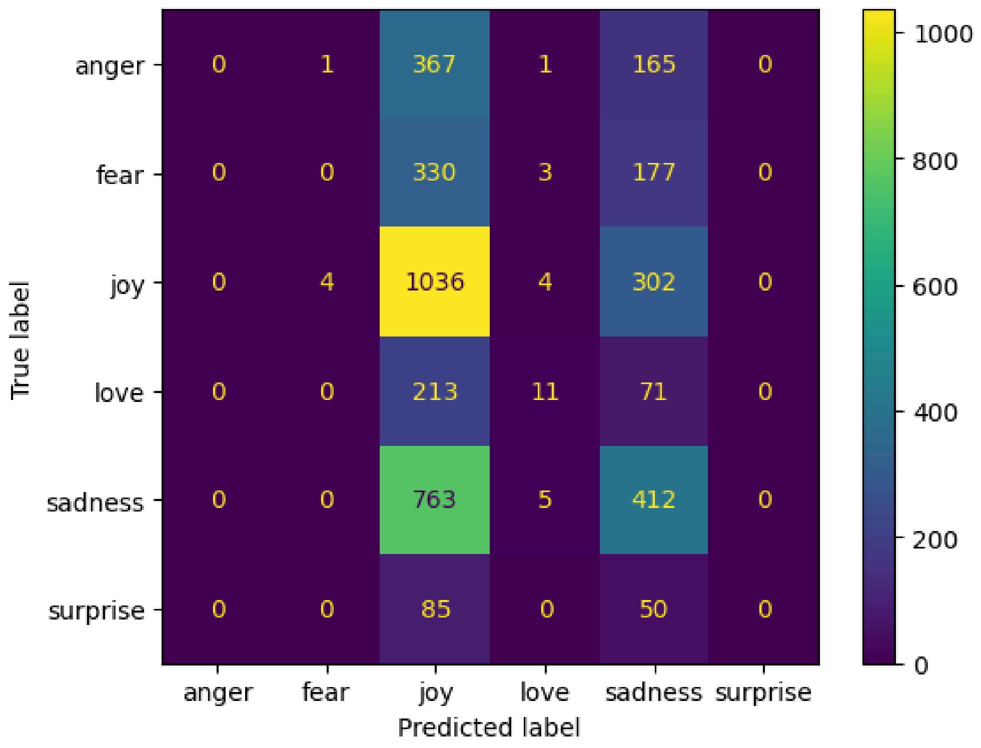


For the first problem, one option would have been to artificially reproportion the dataset to accurately reflect the general population, but I found that this had a huge negative impact on model performance. This technique might be possible given a larger dataset, since with this data it required either dropping a huge number of samples from overrepresented types and/or resampling a huge number of samples from underrepresented types. I chose to use coefficient-based class normalization, which resized classes based on how out of scale they were in relation to other classes in the dataset itself.

For the second problem, I cleaned the data by first dropping non-meaningful text, then lemmatizing the text to correct for inflection when the same word appears in different tenses. I considered also dropping explicit mentions of personality types from the data, in case the model tried to learn to predict type by recognizing direct mentions, but there was no obviously discernible pattern as to whether these mentions were more often someone mentioning their own type or someone mentioning another type, so it seemed that keeping this data could still lend some value to the model.

Other work done on the dataset before modeling also included building in additional variables to the vectorized text data, including variables for emotions (anger, fear, joy, love, sadness, surprise), polarity, and subjectivity. The emotion vectors were created using a sentiment analysis model (Fig. 2, a logistic regression model chosen over a Random Forest classifier for better performance) built on a separate dataset assigning emotion labels to text samples, which was then applied to the type dataset. The polarity and subjectivity vectors were created using TextBlob’s .sentiment( ) method.

Fig. 2: Sentiment analysis model, confusion matrix

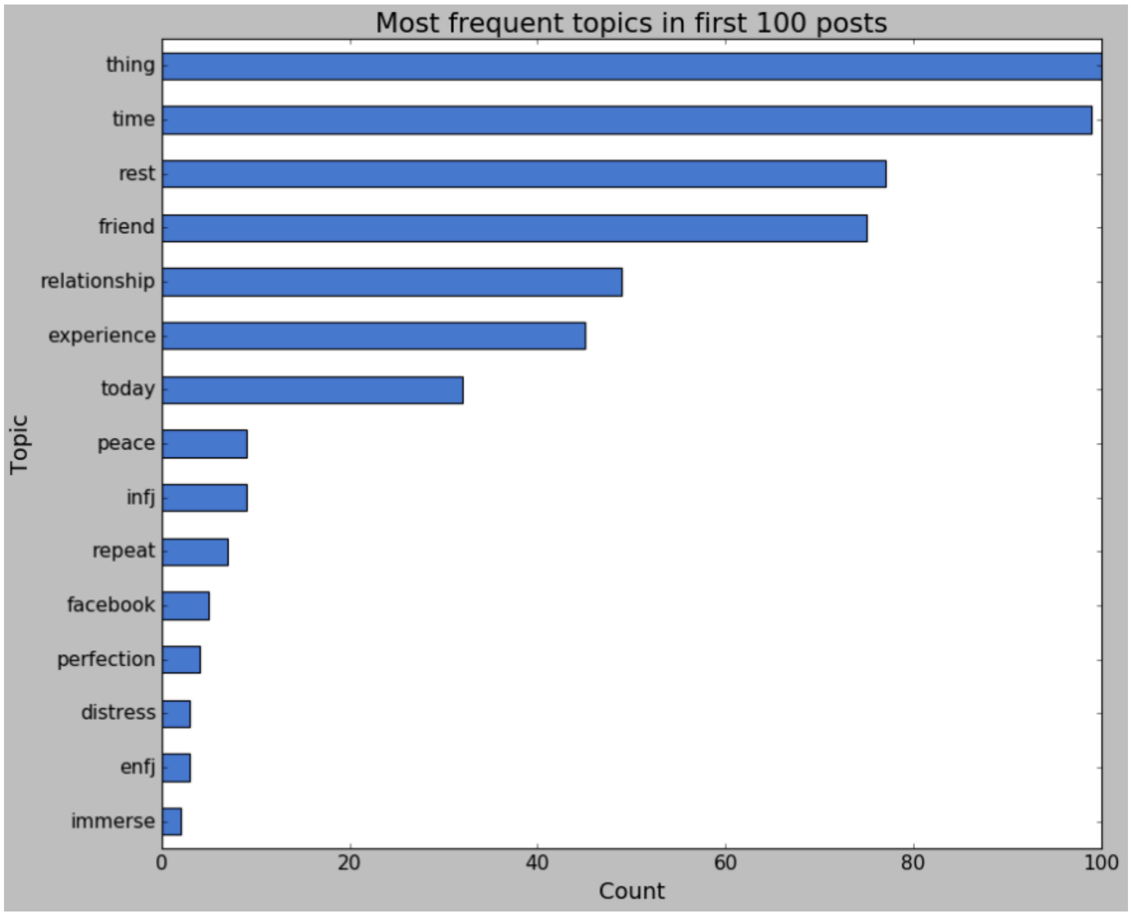


EDA

Topic modeling

I started with topic modeling on the first 100 samples, which gave me a quick look at what topics seemed to be most common in the dataset (Fig. 3). Some were less interesting ("thing" or "today" seemed fairly vague) but the high frequency of topics like "time", "relationship", and "experience" gave an interesting look at what kinds of discussions the forum as a whole tended toward.

Fig. 3: Topic mentions in first 100 samples

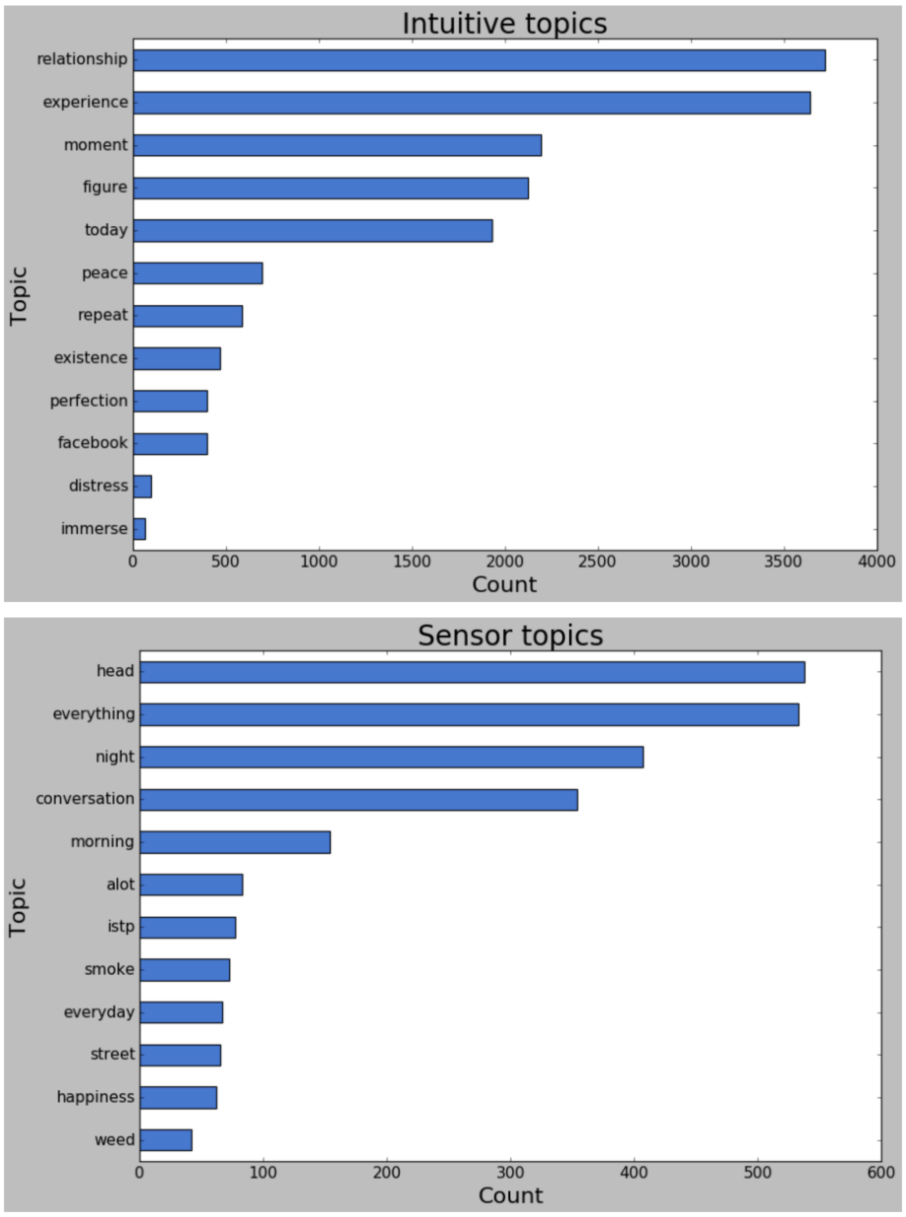


Breaking down the topic modeling by type also led to some interesting discoveries and appeared to give credence to the binary dimensions of the MBTI, especially (or most obviously) the intuition/sensing and feeling/thinking dimensions.

The intuitive types favored the internal/abstract, with words like "theory", "appreciate", "humanity", "relationship", "peace", and "interest" appearing high in their frequency charts, while the sensing types favored the external/concrete, with words like "stuff", "conversation", "power", "order", "food", "money", "drink", "body", "school", and "action" appearing high in their frequency charts.

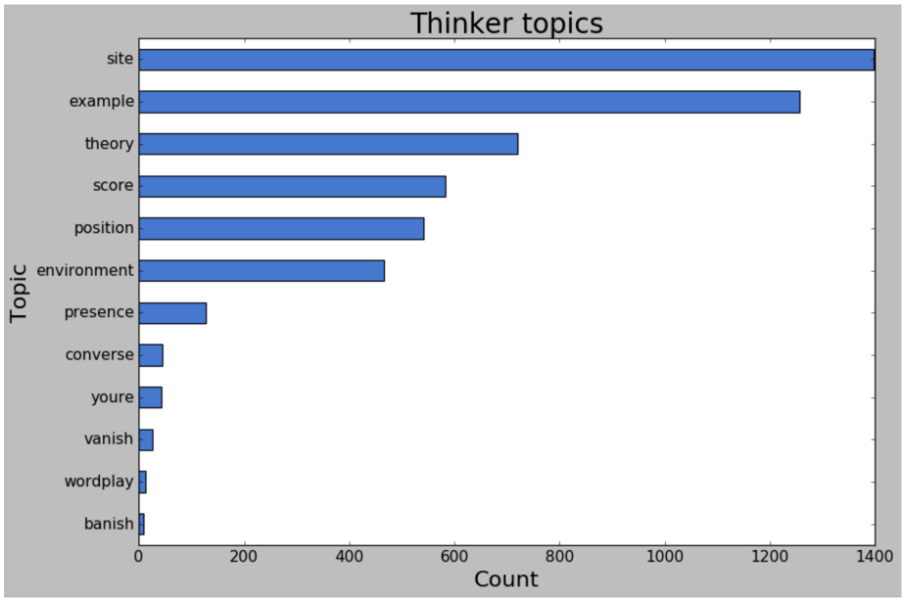
Of course, there was also a great deal of overlap between types, which might be attributed to the overarching bias in the data, which is that it was sourced from a forum dealing specifically in personality theory, likely meaning a majority of the users on the site, intuitive or sensing, lean toward an interest in the abstract to begin with. Directly dividing the data into intuitives and sensors gave a slightly clearer look at the differences between the topics each group favored (Fig. 4). Intuitives frequently mentioned relationships, experiences, peace, existence, and perfection, while sensors frequently discussed conversations, times of day, and physical/sensory topics like smoking and/or smoking weed.

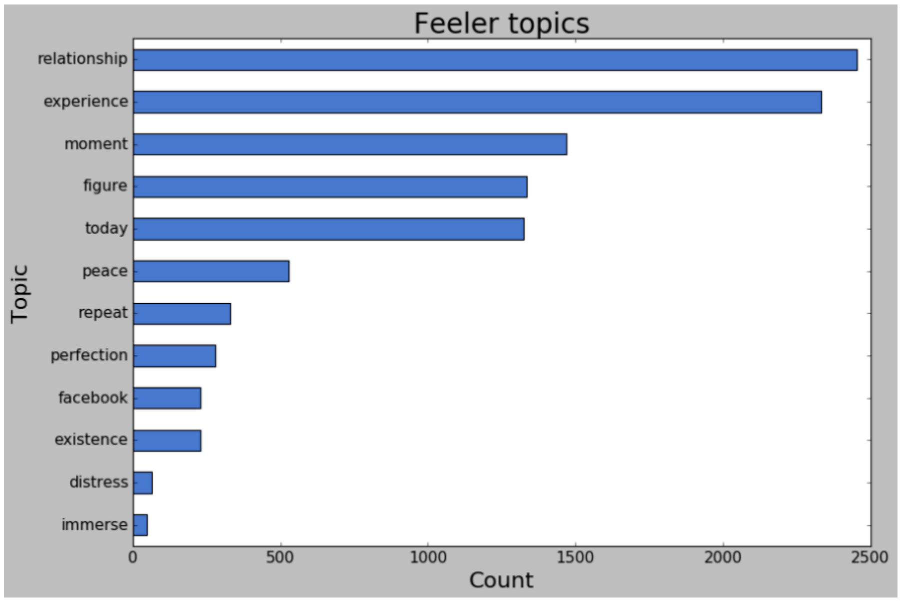
Fig. 4: Topic modeling, intuitives vs. sensors



The divide between thinkers and feelers was also well-defined in the topic modeling. The feeling types had high mentions for words like “care”, “everyone”, “conversation”, “life” “appreciate”, “stress”, “relationship”, “hate”, “feeling”, and “past”, while the thinking types tended to mention things like "course", "question", "function", "example", "theory", "score", "action", "rule", "power", "money", and "intention". Breaking the data into thinkers and feelers made this difference even clearer (Fig. 5).

Fig. 5: Topic modeling, thinkers vs. feelers





The differences between introverts vs. extraverts and judgers vs. perceivers were less defined and seemed to reflect the class distribution more so than individual types or subgroups. For example, introverted intuitives heavily outweighed introverted sensors, so topic modeling on the introverts subgroup heavily overlapped with the results obtained from topic modeling on the intuitives subgroup.

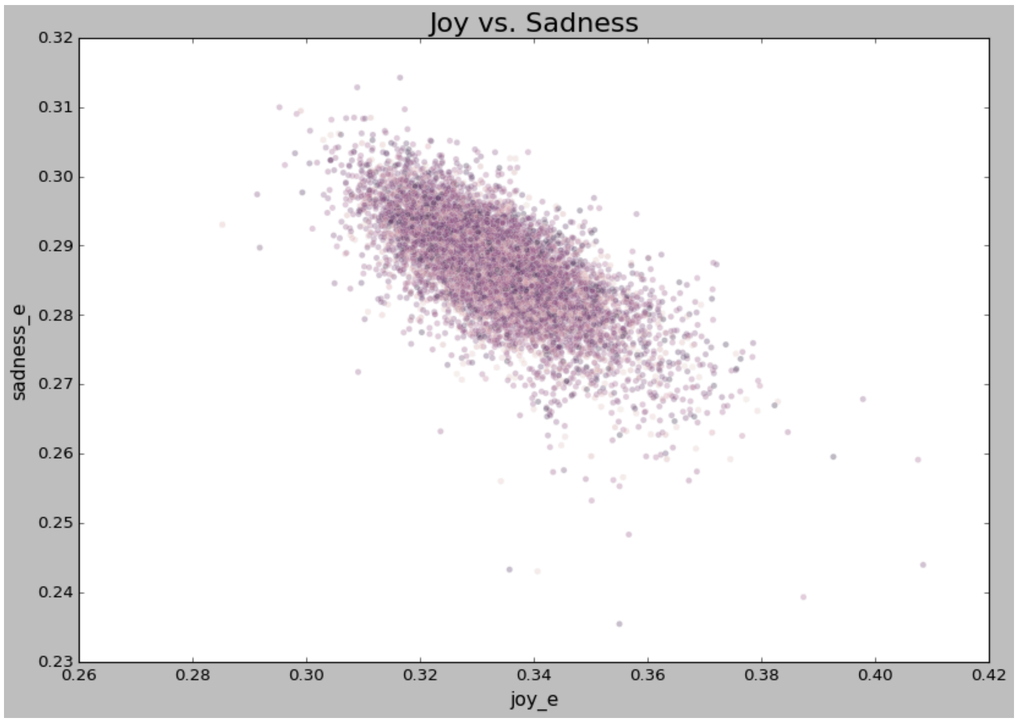
Sentiment analysis

To engineer the emotion features used in modeling, I ran a separate sentiment analysis using a dataset containing text and emotion labels, including anger, fear, joy, love, sadness, and surprise. As the confusion matrix in Fig. 2 shows, the final model used for feature engineering was strongest on predicting joy and sadness, and gave comparatively few predictions for the other emotions, partly due to a class imbalance in the data (joy and sadness represented about 60% of the data that were set aside for testing).

The overall accuracy of the model was 0.362 which, especially considering the fact that the model was only for generating emotion vectors as predictors in the MBTI model, is quite good. In comparison, a random baseline would only give about 16.7% accuracy.

The usefulness of the sentiment analysis (and the original emotions dataset) was also confirmed by the correlation between different emotions. Fig. 6 shows the correlation between probability predictions on text samples labeled "joy" and "sadness" and demonstrates a clear negative correlation. In other words, the model recognized that as the "joy" score increased, the probability of sadness decreased, and vice versa. This supported the ability of the model to predict at least these two emotions.

Fig. 6: Joy and sadness, negatively correlated

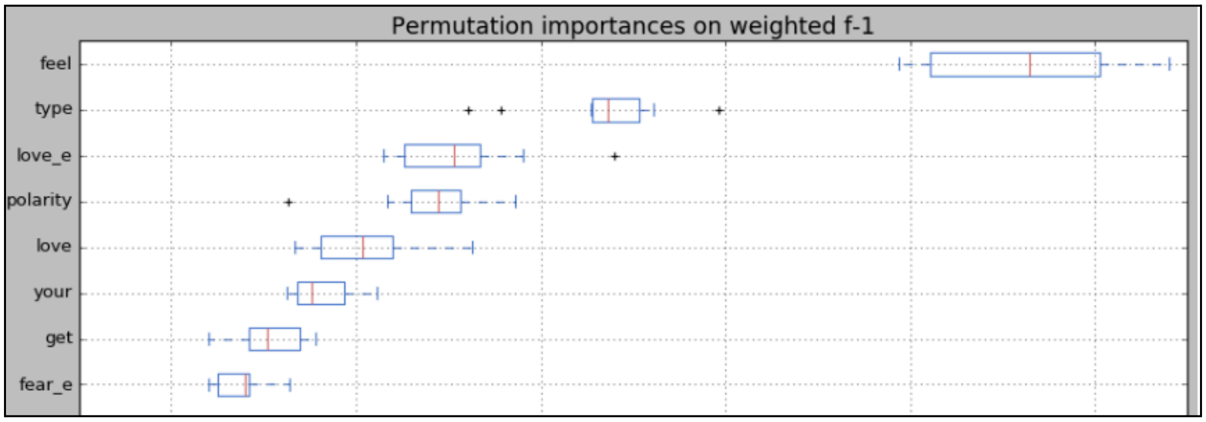


Based on this analysis I had high expectations for the usefulness of the newly created emotion vectors for the MBTI model. To confirm, I next ran a feature importance analysis.

Random Forest feature importance

As my next step in EDA, I ran a permutation importance analysis on a subset of the training data (validating on another subset of the training data) to analyze which features seemed to be most useful in predicting MBTI (Fig. 7). Love (one of the newly created emotion vectors), polarity, and fear (another of the emotion vectors) all scored high in permutation importance, confirming the utility of these features and the sentiment analysis conducted earlier.

Fig. 7: Permutation importance analysis



Preprocessing and TF-IDF

When approaching preprocessing and modeling, I chose to use a TF-IDF (term frequency and inverse document frequency) approach which, in contrast to a bag-of-words approach, takes into account the importance of words as well as information on the count of occurrences in each sample. This preserved the syntactic and semantic aspects of the original text samples, but required a little more processing.

First, the text had to be cleaned, which included removing URLs and non-English words and symbols, including the delimiters that had been added to the original text to separate individual posts within each sample ("|||"). Then the text was lemmatized and converted into a TF-IDF matrix using scikit-learn's TfidfVectorizer. The newly engineered emotion, polarity, and subjectivity features were then added to the matrix, as well as the original MBTI labels after one-hot encoding.

During preprocessing I also corrected for the extreme class imbalance using a coefficient-based class normalization method, downsampling on overrepresented classes and upsampling minority classes to achieve a more balanced dataset. The final dataset that was passed on to model selection and testing consisted of 8664 samples (only about 100 fewer than the original dataset), 108 predictors, and 16 class labels (one for each MBTI type).

Model selection

I experimented with a variety of types of classifiers, including a Random Forest classifier, a K-Nearest Neighbors classifier, a Multinomial Naive Bayes classifier, and a Multi-layer Perceptron network. For assessing model performance I used accuracy as the metric for comparison. As stated above, the baseline for comparison was 16.7% accuracy for a completely random prediction.

The table in Fig. 8 compares the baseline accuracy of each classifier before hyperparameter tuning, and the table in Fig. 9 contains the best cross-validation score for the two classifiers I moved forward with, as well as the hyperparameter settings that produced the best performance for each.

Fig. 8: Model selection, initial performance

Classifier	**Accuracy**
Random Forest	**53.57**
K-Nearest Neighbors	**33.53**
Naive Bayes	**20.62**
MLP (Multi-layer Perceptron)	**52.74**

From initial testing it was evident that the problem had proved too complex for a Naive Bayes algorithm and too high-dimensional and/or imbalanced for KNN. Since I had used TF-IDF rather than bag-of-words it is possible that the predictors are not as independent as a Naive Bayes classifier would be assuming.

Therefore I moved on to hyperparameter tuning with only the Random Forest and MLP classifiers, which both achieved more than 50% accuracy before any adjustments. Using a grid search, I found that the following hyperparameters produced the best performance for each model (Fig. 9).

Fig. 9: Model selection, hyperparameters and scoring

Classifier	**CV accuracy**	**Hyperparameters**

Random Forest	**54.08**	bootstrap = True
		max_depth = None
		max_features = ‘auto’
		min_samples_leaf = 1
		min_samples_split = 2
		n_estimators = 200

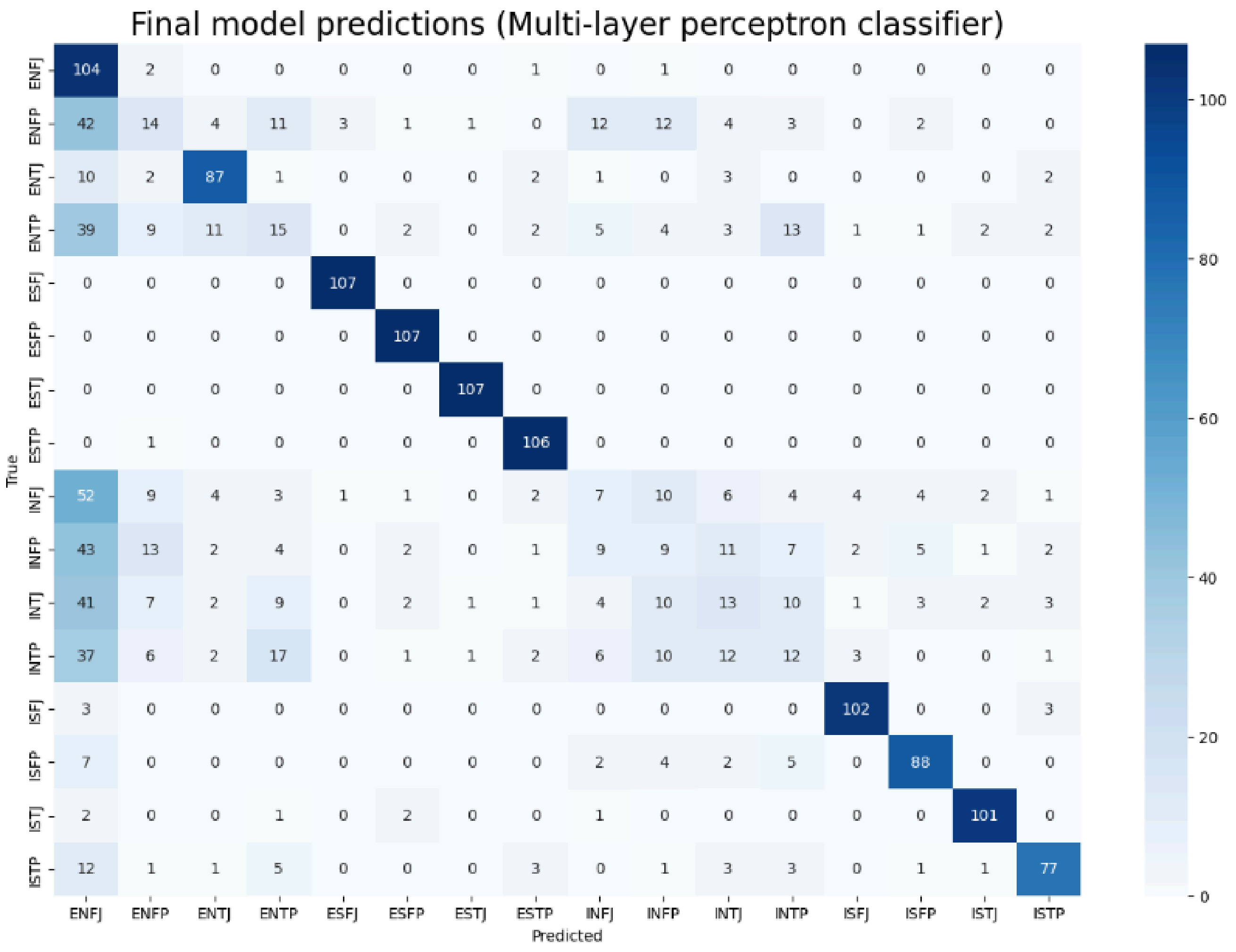
MLP (Multi-layer Perceptron)	**55.26**	activation = ‘tanh’
		alpha = 0.01
		early_stopping = True
		hidden_layer_sizes = (100, 100, 100)
		learning_rate = ‘constant’
		max_iter = 2500
		solver = ‘lbfgs’

Final model

Based on the classification scores above, the final model I moved forward with was the MLP classifier. This model achieved a 0.571 accuracy score on the withheld test data. Tuning hyperparameters for the MLP classifier was a more complex process than with the Random Forest model. I tested various thresholds for maximum iterations, but still had an issue with underfitting (79% accuracy on the training data). Adding another hidden layer of size 50 improved the underfitting problem but hurt performance on the test data. I experimented further with various layer sizes and additional layers, before ending with 3 hidden layers of size 100. There was no improvement in performance when increasing the complexity beyond 3 hidden layers.

The final model performed exceptionally well for minority classes (in stark contrast to some exploratory analysis done before class normalization) and actually gave relatively few predictions for what were the majority classes in the original dataset (Fig. 10).

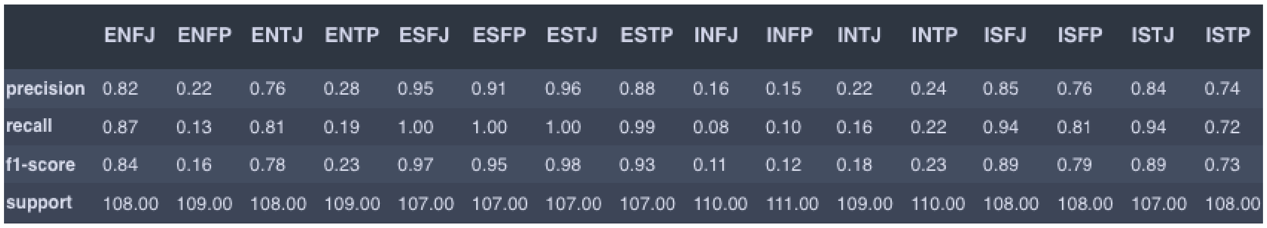
Fig. 10: Confusion matrix, final model



	ENFJ	ENFP	ENTJ	ENTP	ESFJ	ESFP	ESTJ	ESTP	INFJ	INFP	INTJ	INTP	ISFJ	ISFP	ISTJ	ISTP
ENFJ	104	2	0	0	0	0	0	1	0	1	0	0	0	0	0	0
ENFP	42	14	4	11	3	1	1	0	12	12	4	3	0	2	0	0
ENTJ	10	2	87	1	0	0	0	2	1	0	3	0	0	0	0	2
ENTP	39	9	11	15	0	2	0	2	5	4	3	13	1	1	2	2
ESFJ	0	0	0	0	107	0	0	0	0	0	0	0	0	0	0	0
ESFP	0	0	0	0	0	107	0	0	0	0	0	0	0	0	0	0
ESTJ	0	0	0	0	0	0	107	0	0	0	0	0	0	0	0	0
ESTP	0	1	0	0	0	0	0	106	0	0	0	0	0	0	0	0
INFJ	52	9	4	3	1	1	0	2	7	10	6	4	4	4	2	1
INFP	43	13	2	4	0	2	0	1	9	9	11	7	2	5	1	2
INTJ	41	7	2	9	0	2	1	1	4	10	13	10	1	3	2	3
INTP	37	6	2	17	0	1	1	2	6	10	12	12	3	0	0	1
ISFJ	3	0	0	0	0	0	0	0	0	0	0	0	102	0	0	3
ISFP	7	0	0	0	0	0	0	0	2	4	2	5	0	88	0	0
ISTJ	2	0	0	1	0	2	0	0	1	0	0	0	0	0	101	0
ISTP	12	1	1	5	0	0	0	3	0	1	3	3	0	1	1	77

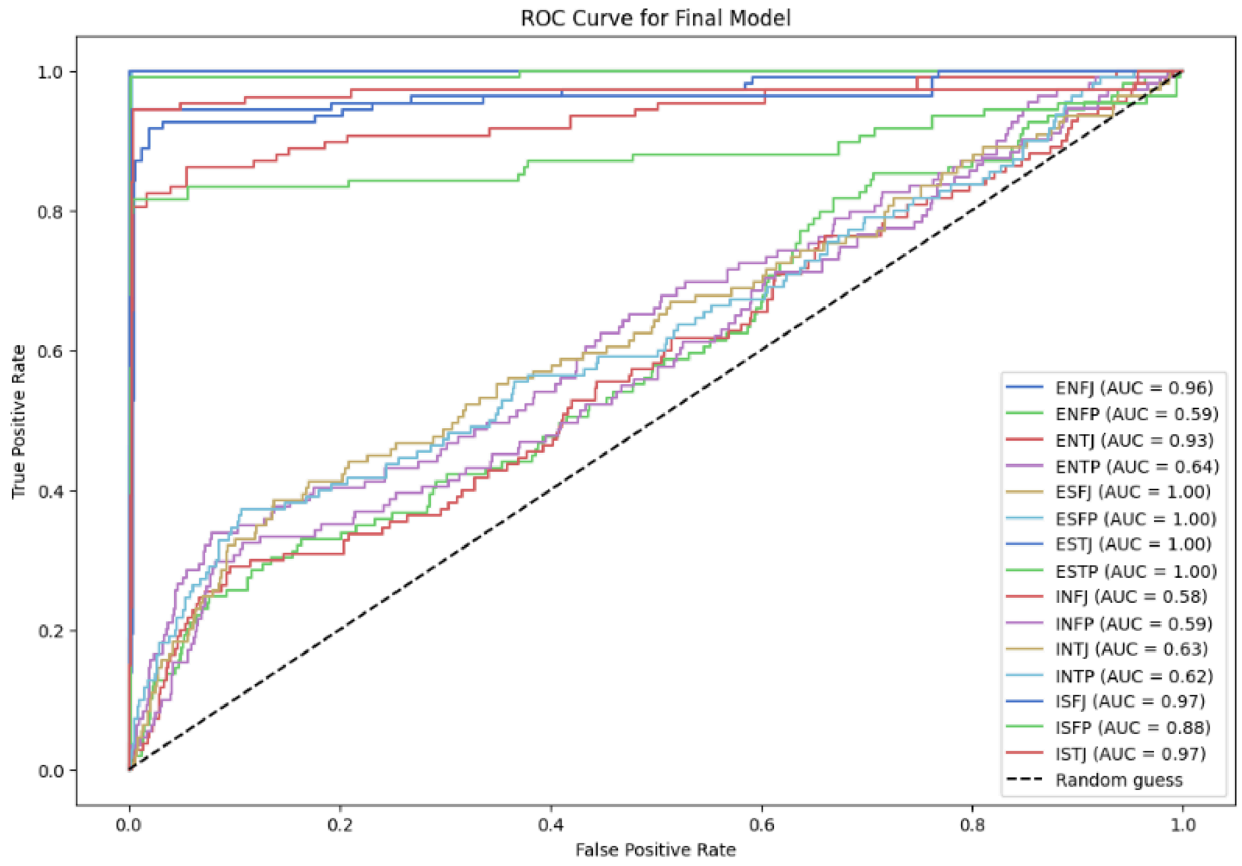
Looking at the classification report on the final model as well as comparing ROC-AUC scores for all classes also showed that there were definitely some classes that the model had learned to predict better than others (Fig. 11a, Fig. 11b). This was about half and half, but on all classes, the model still did better than the baseline.

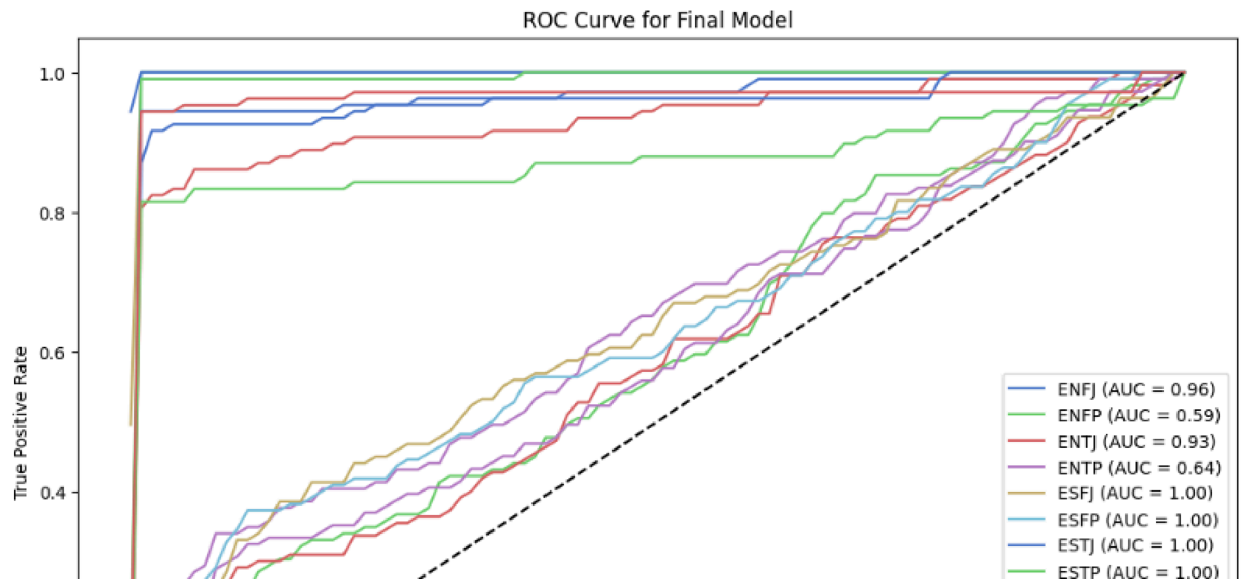
Fig. 11a: Classification report, final model



	ENFJ	ENFP	ENTJ	ENTP	ESFJ	ESFP	ESTJ	ESTP	INFJ	INFP	INTJ	INTP	ISFJ	ISFP	ISTJ	ISTP
precision	0.82	0.22	0.76	0.28	0.95	0.91	0.96	0.88	0.16	0.15	0.22	0.24	0.85	0.76	0.84	0.74
recall	0.87	0.13	0.81	0.19	1.00	1.00	1.00	0.99	0.08	0.10	0.16	0.22	0.94	0.81	0.94	0.72
f1-score	0.84	0.16	0.78	0.23	0.97	0.95	0.98	0.93	0.11	0.12	0.18	0.23	0.89	0.79	0.89	0.73
support	108.00	109.00	108.00	109.00	107.00	107.00	107.00	107.00	110.00	111.00	109.00	110.00	108.00	108.00	107.00	108.00

Fig. 11b: ROC-AUC curve, all classes





Conclusion

For this project, I chose to take a fairly straightforward approach to predicting MBTI type directly from the labeled data. Considering the number of classes and the relative complexity of the problem, the final model performed strongly, an improvement compared to non-ML methods such as online quizzes asking users to score or classify themselves on a large number of unique indicators or questions. Studies show that with these kinds of tests, about 50% of users will receive a different result when taking the test a second time, even in a relatively short test-retest period. So in light of that, the accuracy of this model (57%) represents a great step forward in the field of personality typing.

Alternative approaches to this problem that might result in different performance, even if I were to use the same data, could include building a set of four individual classifiers to predict each binary MBTI dimension (introversion/extraversion, intuition/sensing, feeling/thinking, judging/prospecting) or to predict Jungian cognitive functions (Ni/Ne, Fi/Fe, Ti/Te, Si/Se) which are immediately related to and can be directly derived from the four-letter MBTI codes. It is possible that these other approaches might lead to higher predictive capabilities, but it is also equally likely that these approaches would still fall victim to the inherent problem in personality theory today, which is the self-reporting aspect of any MBTI data that can be collected from the Internet.

If I were to return to this problem, those are two alternatives I would be very interested in testing, although using cognitive functions as classes does present other issues, such as the need to take into account function placement (each MBTI type has a specific 4-function stack) and the existence of only 16 permutations of these functions (in case the model attempts to predict a function stack that does not correlate to any MBTI types). These are problems that I have as yet been unable to solve, which is why I could not use the cognitive functions approach for this project, but I remain confident that there is a solution, even if that might require a more complex model or a data pipeline with more processes.

Creating separate classifiers for each of the four personality dimensions in MBTI would also present an issue with determining model performance, since this would very likely decrease the rate of “perfect” classifications (all four dimensions predicted correctly for a sample) but might increase the overall strength of the model if approximate or close predictions can also be taken into account. The model built in this project can only be correct or incorrect in its predictions, and unfortunately cannot take into account inherent similarities between types, like the relative proximity of INFP to INFJ as opposed to INFP and a more dissimilar type, such as ESTJ. Some applications, including existing MBTI quizzes on the Internet, might prioritize producing an approximate prediction (for example, three out of four dimensions predicted correctly) over producing a perfect prediction.

The draw of social categorization is evolutionary, stemming from the inherent social nature of humanity. Although MBTI and personality theory in general is considered unreliable, there will always be a demand for it, so continuing to refine and improve tools such as this classification model is a valuable endeavor. Given more data, or even data that is less reliant on self-reporting (ie. third-party observational studies assessing behavior to determine type), I think this model and others like it could easily be improved upon, and perhaps even eventually reach a level of accuracy that could elevate personality “theory” into a more recognized “science” of personality, even if in the end, MBTI loses out to other popular theories, such as enneagrams, Jungian cognitive psychology, Keirsey temperaments, and more.