DNNLPersonality
Capstone Project
Danny Clifford
October 2018

Definition

Project Overview

With much better memory than people and the amount of personal information we share with computers, it's amazing they don't appear to understand our personalities better. With the exceptions of saving searches and some companies using AI and machine learning to predict profitability for advertisements, there is little effort to understand personal tendencies to cater content for (and not just to) individuals. Even those multi-billion dollar corporations aim to maximize advertisement profit, not to understand the fundamental features that make each of us unique.

Understanding what connects us all, what makes each of us unique, what are our strengths and how can we harness everyone's strengths to build better lives for individuals and humanity, seems far from the focus of their efforts. For the first time in history, we are able to economically collect and process enough information to understand the patterns of human nature. There have been good attempts in recent history of distinguishing the features of human processing that make certain people different than others, most notably Carl Jung in his book Psychological Types and the subsequent adaptation of his theories to the Myers Briggs Personality Index.

Carl Jung notes that it is difficult for a person who experiences their own bias to accurately judge others, joking that one person creating a system would be like creating a Universal Church with one member. Luckily, since his time wonderful scientists such as Alan Turing, John von Neumann, J.C.R. Licklider, Miller, Moore, Noyce, and countless other have made it incredibly easy to collect, share, and calculate data from around the world almost instantly, not to mention make impressive improvements on models of understanding how agents behave. With the addition of breakthroughs in behavioral psychology by greats like Kahnemann, Tversky, and Thaler, we are quickly building the ability to study the patterns of reason and thought in humans as differentiated by the rational agents which traditional economic theory implies.

Putting these pieces together, as we communicate with computers and people, we are creating valuable information and patterns that, if only captured and studied, would give great insight into how our individual and collective minds work. This project is about helping individuals use computers understand how the patterns in our language reflect our inner personality and in turn how we receive, process, and communicate information; which determines the outcomes in our personal lives and compounded over every individual over time, amounts to the fate of humankind. Ultimately, computers can be our tools to help us learn our

unique patterns and to help us change, supplement, or leverage how we do things to help us solve problems and achieve our goals.

Project Statement

The problem that I am setting out to solve is how to understand someone's personality based on their use of language. If we can accurately predict one of the most fundamental aspects of a person's behavior and uniqueness as measured by the language they use, the ability to communicate information with that person will be drastically improved. The internet is designed based on information that is already programmed into the web page itself; for example, administrators see a website much differently as a new customer or even a logged in user and are determined prior to visiting the webpage. This poses a difficult problem for web designers to incorporate designs that maximize the profit or usefulness to their intended audience rather than communicating information or value to a person on an individual basis. If we can predict learning style or how an individual will react to their environment, then we can better customize the learning experience to their preferences and strengths.
Quantifying personality has been done for us with the Myers Brigg Personality Index 4 letter code. These will be further broken down into their 4 features of a single letter with only 2 options. In addition, it will also allow us to train the weights of determining individual features of personality in a more focused way. NLP allows algorithms to extract meaning from text whether from word count, frequency, parts of speech, and even sentiment in a quantifiable and measurable way. These matrices of language data will be learned by a Deep Neural Network and these patterns recognized during training will be used to predict the personality features of the test group.

1. Download the data from Kaggle
2. Let SpaCy web-medium language model run through the posts to make word vectors
3. Shuffle-Split the data into testing, validation, and training data
4. Run benchmarks training and testing with Logistic Regression, Random Forest, and MLP Classifiers
5. Run training and validation through the DNN
6. Test accuracy of the models on the test set with AUC

Metrics

Training a neural network on language use and their corresponding personality feature labels allows us to measure the AUC. Area under the ROC curve is used to ensure the proper binary classification when it comes to specificity and sensitivity. This will help better quantify individual differences in each of the 4 personality features. Wang uses AUC in order to quantify and measure accuracy of his logistic regression models with 10 fold cross validation. Since the distribution of personalities within the dataset is skewed in both our datasets, this will be a good evaluation metric to use. He also measured accuracy by comparing different models based on the features mentioned above based on AUC, not only breaking them down into dichotomous features (Sensing and Intuitive, Extrovert and Introvert), but also by focusing on features of the language such as POS and word vectors.

On Kaggle, the best performance of a project breaking the problem into 4 binary classification problems was achieved using F1_scoring on 5 fold stratified cross validation.

Therefore I will use 10 fold stratified cross validation and use AUROC (AUC) and F1_score in order to ensure that my models are at least comparable.

Precision answers the question of the people the model labels as extroverted, how many were actually extroverted. Recall answers the question: of the extroverted people, how many did the model label as extroverted. Another way of looking at it is the True Positive Rate (TPR) and False Positive Rate (FPR). A false positive would occur if the model predicted a person was Extroverted person when they actually tested as Introverted; in other words, 1/Precision. A True Positive would obviously be correctly predicting the Extroversion of a person by their Doc Vector and the TPR is the same as recall. These are calculated by predicting the test set and comparing it to the test labels.

Plotting the Receiver Operating Characteristic (ROC) curve is done by calculating the FPR and TPR of the model. By moving threshold for tolerance of all possible TPR, FPR's it creates a curve that demonstrates how robust a particular model is. From this curve, you can calculate the Area Under the ROC Curve (AUC) which gives a score of 0.5 for random guessing and 1.0 for perfect precision and recall regardless of the threshold.

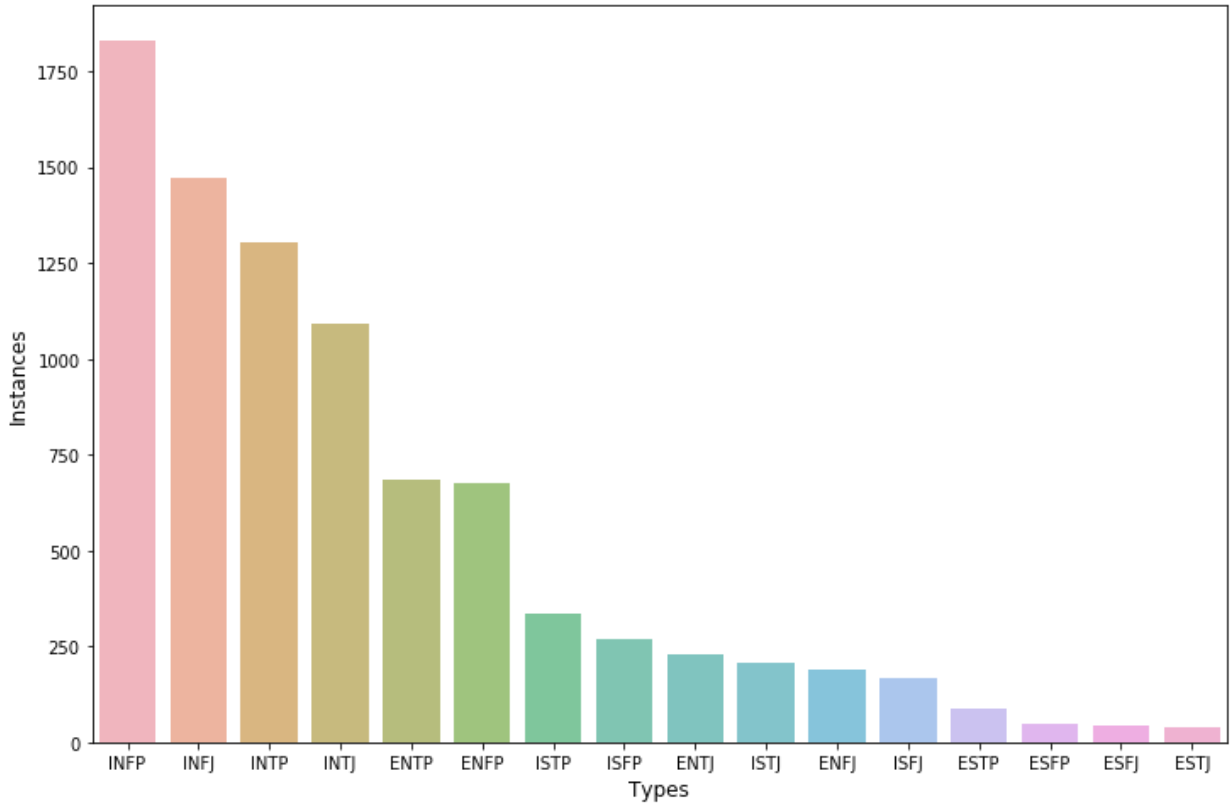
Another way to evaluate the quality of the model is the F1_score. This done by taking the harmonic mean of the precision and recall of the model. Instead of taking the average of precision and recall, which would give models that always predicted Introversion very high scores, F1 gives weight to the lower of the scores. The formula for calculating F1_score is 2* ((precision*recall)/(precision+recall)).

Analysis

Data Exploration

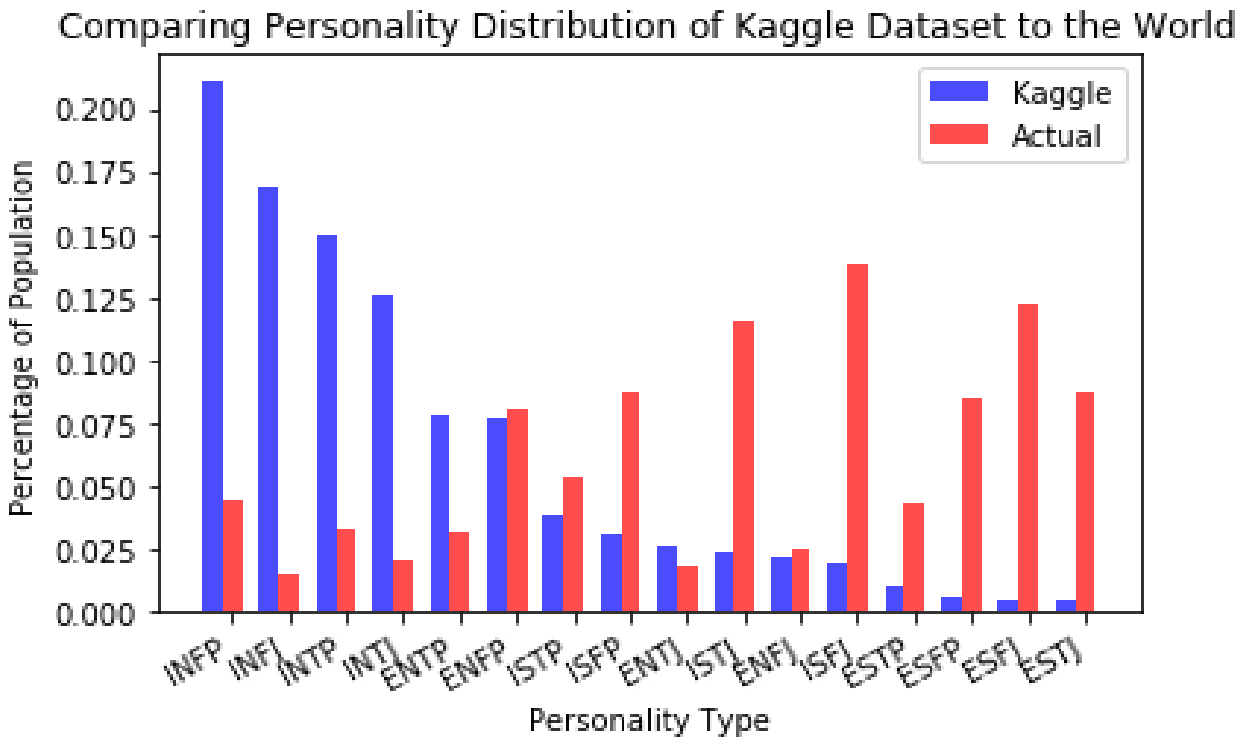
The dataset is taken from Kaggle and contains 8,600 users with 50 recent comments on the Kaggle website each and their corresponding personality type. This was user generated data from the Kaggle website and offers the most labeled personality data connected to their text data (comments) of what I could find online. It is attached and also on the Kaggle website.

Exploratory Visualization



Clearly Introverted and/ or iNtuitive people dominate the Kaggle forums, or at least the ones participating in the creation of the database. This will make it quite difficult to learn about those who are extroverted and sensing types (ESxx). The model would likely minimize error by simply never predicting ESxx labels.

The bar graph below compares the percentage of people who belong to each type on the Kaggle database and Myers-Briggs world estimates. If the data weren't already skewed enough, the people's personalities of this dataset are quite different to the general population. When it comes to creating a system for understanding personality, the data used to train the model would ideally represent the population of personalities it is trying to understand.



Algorithms and Techniques

Preprocessing the data will be the largest part of the technique for predicting the personality of a person based on their text. There are separating characters (|||) that need to be removed, links that will not show up in language and instead will be converted simply into the word 'link'.

As mentioned in the Data Exploration section above, using the bag of words technique to create word vectors of each personality types' language. Using spaCy's medium web language library to create relative frequencies of common language used online, the result will be a model of language use specific to each personality feature. In addition, spaCy can find the similarities of new sentences to each corresponding personality and even between personalities.

The classifier is a Deep Neural Network, that will run over the language and assign the probability for each class based on their word vector representations. Deep Neural Networks take an array of inputs and run them through a series of hidden layers with weights. A very simple form of DNN is what will be used as another comparison for the final model from sklearn called a MLPClassifier. It is a multilayered perceptron with activation functions that send values to new layers. At the end of the DNN there is an output layer that makes the final guess, whether that be a personality type, Introverted / Extroverted, 0 / 1, depending on what problem is being solved.

The other DNNLPersonality models used here have 5-6 layers ranging from the input of 300 neurons to hidden layers of either 128, 64, 32, 16, or 8 neurons with relu activation functions. There are several dropout layers usually of 20% that help reduce overfitting of the

models. Always finishing with the sigmoid activation output layer with 1 neuron. I chose DNN because they are best at seeing patterns within patterns which is what personality ultimately is, especially when trying to understand it through language.

Since the data is quite skewed not only from the population but especially not distributed evenly throughout the personality types, creating a model that will most accurately predict the personality type will likely result in simply guessing the most likely personality features, in this case Introverted and Intuitive (IN). Tuning the model based on the AUC and series of 4 binary classifications (which coincidentally also makes more sense in the study of personality) allows to create the most potentially useful prediction model and avoid overfitting.

Calculating AUC not only tries to get the most predictions right, but tries to ensure that false positives and false negatives don't crop up. In other words, since the accuracy metric would tend to create a prediction where everyone is IN__, the AUC metric will catch that the model systematically falsely assigns Extroverts the Introvert label and will look for patterns to help correct it, even if retroactively. It will be used as an evaluation of the quality of the models and a window into how much information was actually learned.

I also have read that AUC on skewed datasets is overused and doesn't necessarily represent a good way to compare models. To compensate, I added F1_score which was suggested on a few forums.
https://stats.stackexchange.com/questions/210700/how-to-choose-between-roc-auc-and-f1-score

Benchmark

The Benchmark I will be using to predict personality based on the data is a Logistic Regression, Random Forest Classifier, and Support Vector Machine Classification algorithms, as I am trying to make a similar model to Yilun Wang's in his Understanding Personality through Social Media.

The average AUC across the four binary predictions he achieved with his best model was 0.661 using a large repository of Tweets of around 90,000 twitter users. Of course this dataset is much more skewed and smaller. I used another top kernel from kaggle with this same dataset where F1_score was used. His best models created an average F1_score of 0.665. See links for these projects in the conclusion.

Methodology

Data Preprocessing

Implementation

The process for which metrics, algorithms, and techniques were implemented with the given datasets or input data has been thoroughly documented. Complications that occurred during the coding process are discussed.

1. Preprocess the data, splitting the post string into an array of posts.

2. Remove the hyperlinks and replace with the word link
3. Process each post with SpaCy nlp library turning each post into an average word vector over the words of the post.
4. Save a pd.df of all the posts and their respective person's type for simple classifiers
5. Split the posts_df into train and test sets
6. Train on Logistic Regression, Random Forest, and SVC Classifiers
7. Test accuracy, AUC, and F1 Score of different personality types and functions
8. Train on SKLearn MLPClassifier and test Acc, AUC, and F1 of each personality type/ function
9. Split persons_word_vector_df into train, valid, and test datasets
10. Input into keras DNN and train/ validate to create weights for each personality type/ function
11. Test accuracy, AUC, and F1 of keras models
12. Train, (Validate), and Test Multilabel Classification algorithms for each type of algorithm

Refinement

First I started bringing in the simple models of logistic regression, random forest, and svc since it was the model's Simon Wang mentioned in his paper understanding personality through social media and he found they gave similar results.

Then I had a very difficult time trying to get the data to fit into keras into a convolutional neural network. I thought getting a 2D np array of 50 Doc Vectors (one representing each post), which I had to eliminate nearly 1000 people from the dataset who had less and remove any posts over 50 in those with more than 50 posts. I also tried LSTM, but to no avail, as I could not overcome my errors inputting the data properly.

Finally I went back to the basics and implemented an sklearn MLPClassifier since that was at least a Deep Neural Network that I could see if there was any performance. A comparison against the benchmarks are shown in the Results section.

From here I constructed a Deep Neural Network of my own in Keras trying to predict overall personality types (1 of 16; 6.25%). At best my results achieved just over 50% which is fine. However, as I read Carl Jung's Personality Types who's theories Isabel Briggs-Myers based the MBTI on and skeptical papers on the system such as Gregory Boyle's Myers-Briggs Type Indicator (MBTI): Some psychometric limitations, I began to realize this was not even a great goal to strive for. It appears to be far more insightful to understand the individual functions and features of personality than creating an archetypal system that is far more beneficial for creating quick general understanding and a means for everyone to communicate about personality. There have been many questions as to it's statistical foundation and viability in terms of research and nailing down a solution to understanding the nature of humans and their subsets of personalities. Therefore, I concentrated my focus almost exclusively on the identification of certain tendencies of individual functions and features of personality.

Refining the algorithms was a long manual process as I thought I could learn a lot more than a gridsearch by watching the individual changes and in effect training my intuition for

constructing and tuning models. I would change batch_size from 16, 32, 40, 50, 64, 80, 100, and 128. Epochs from 10, 15, 20, 25, 30, 40, 50, 60, 64, 75, 80, 100, 150, 200, 250. Test Size splits .2, .25, .3 between validation and testing. Architectures of dozens of different combinations from 500-256-128-64-32-16, which never worked well when I created more nodes, to as simple as 300-16-1. I finally arrived at 300-128-64-16-8-1 which seemed to give the best results based on how I was doing things.

I did not adjust learning rate or any other variables. This could certainly be fine tuned, but manually doing the hyperparameter tuning beyond this point did not seem as high of a priority as completing the project. I plan to adjust them in the future as I get better datasets and don't have to pay for the privilege to work on this project anymore (haha).

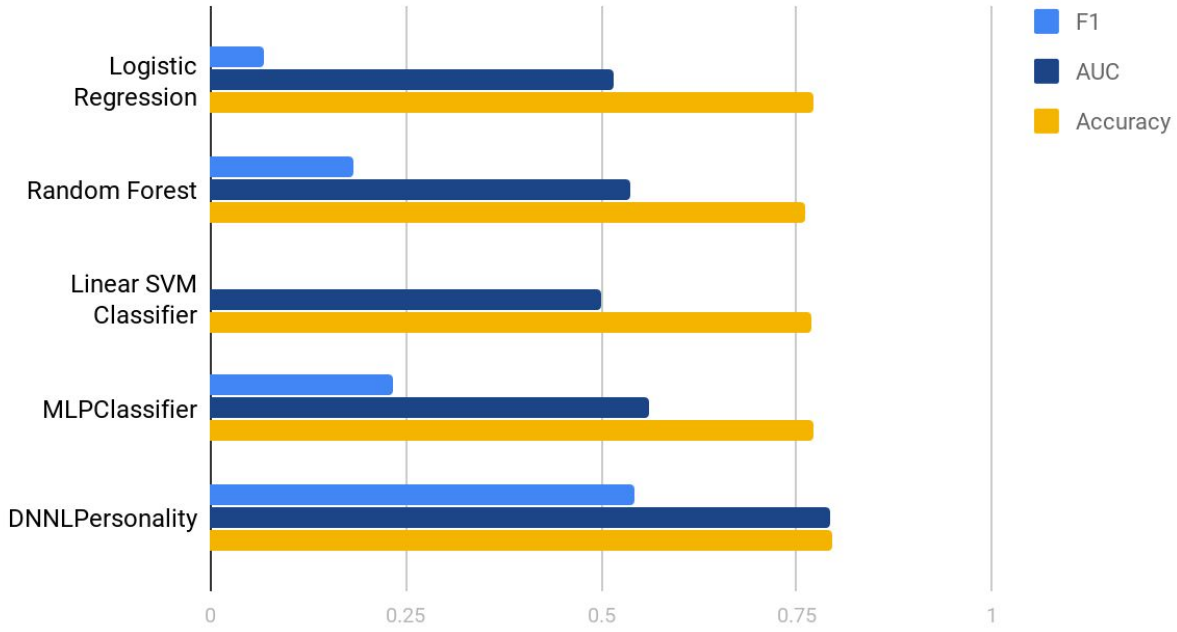
Results

Model Evaluation and Validation

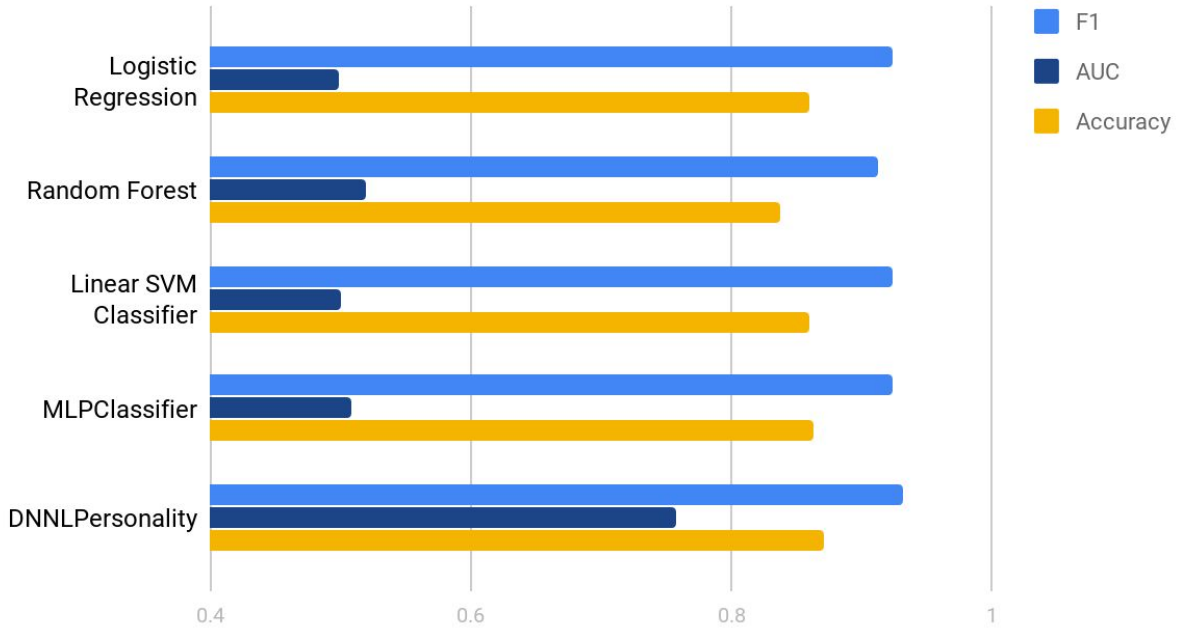
The model evaluation process was done with accuracy, AUC, and F1 Score through KFold Cross Validation. The simple models were run with sklearn cross_val_score() with 5 folds on accuracy. Then once for AUC and F1 score. The keras models run on 10 Fold cross validation measuring Accuracy, AUC, and F1 at each fold to ensure robustness.

Note that the F1 and AUC scores for the models other than DNNLPersonality are likely overstated since no cross validation was done on these values.

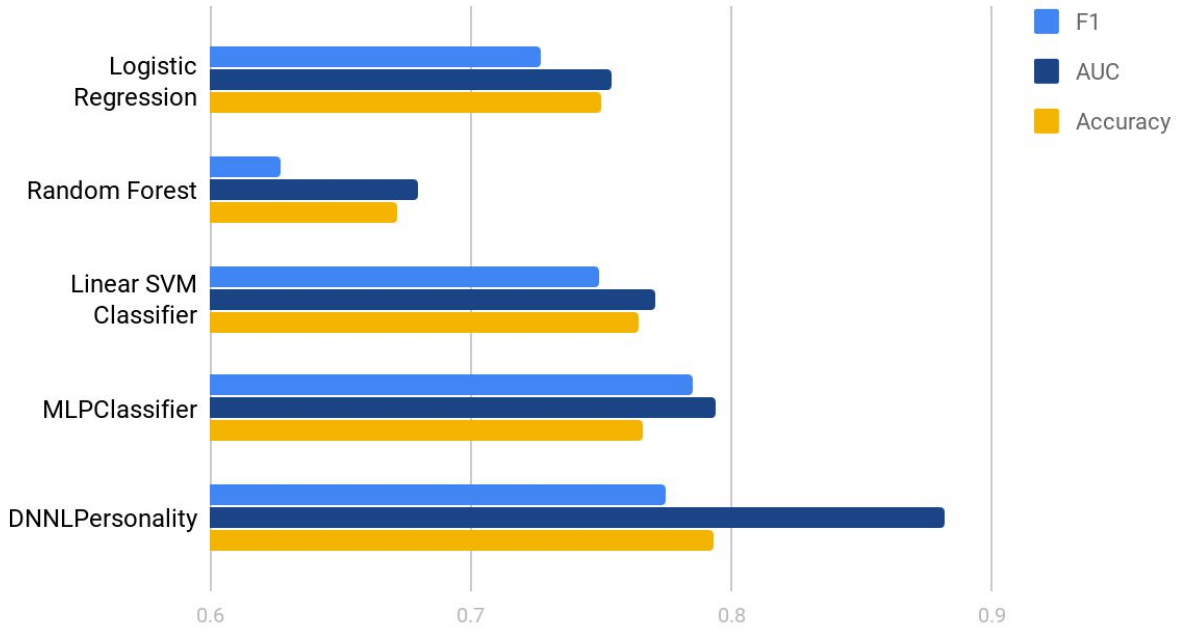
Extroversion vs. Introversion



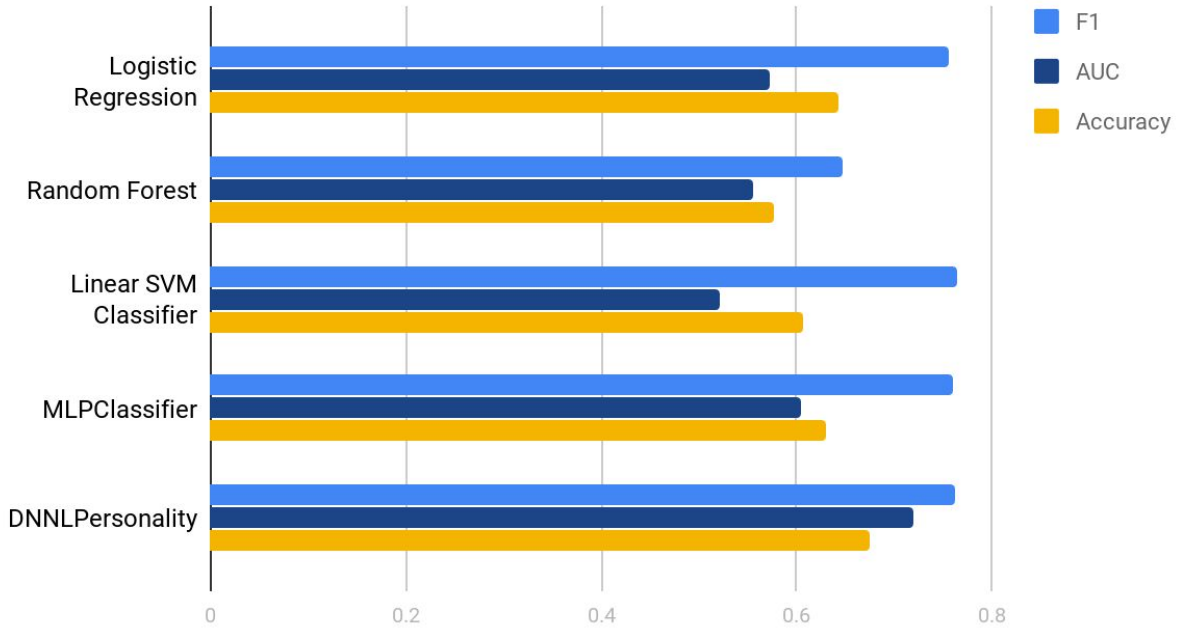
INtuitive vs. Sensing



Thinking vs. Feeling



Perceiving vs. Judging



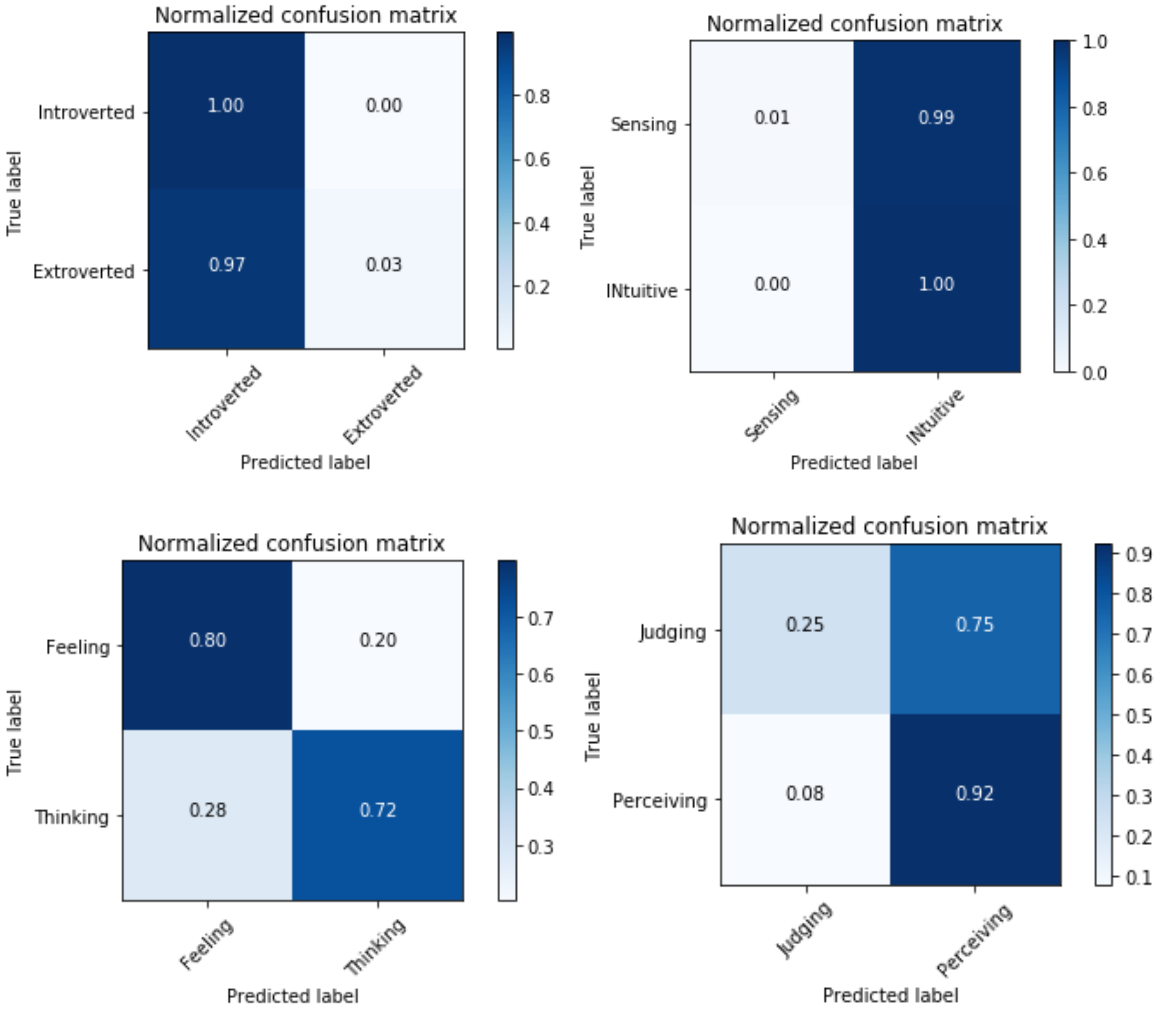
Justification

Although the final model performed much better than the benchmark, I would not say this solution adequately solves the problem of developing a tool to help understand people's personality through their language. Simply getting near 80% accuracy of understanding a generalization of people is not enough. Much work is to be done before we can begin to use our computers as tools for understanding human personality and behavior in a meaningful way.
The final results with the comparison to the benchmarks are shown below.

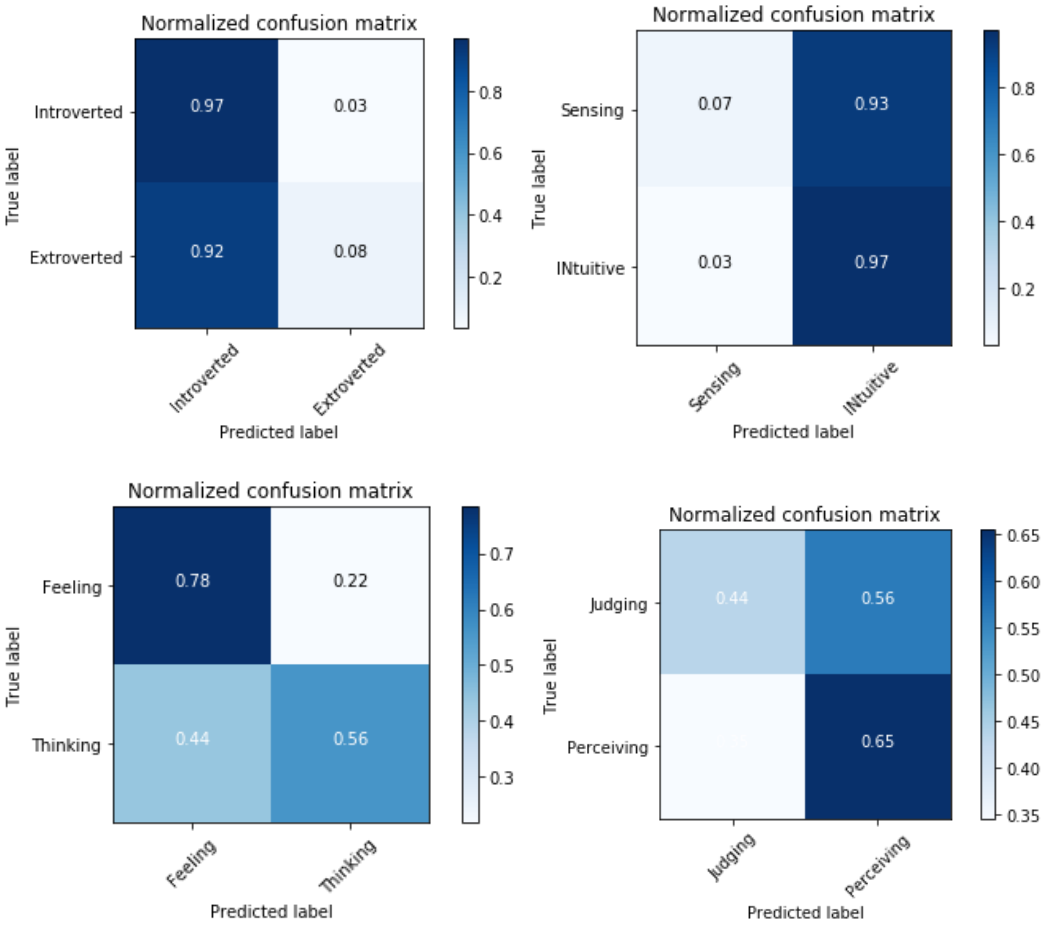
Conclusion

Free Form Visualization

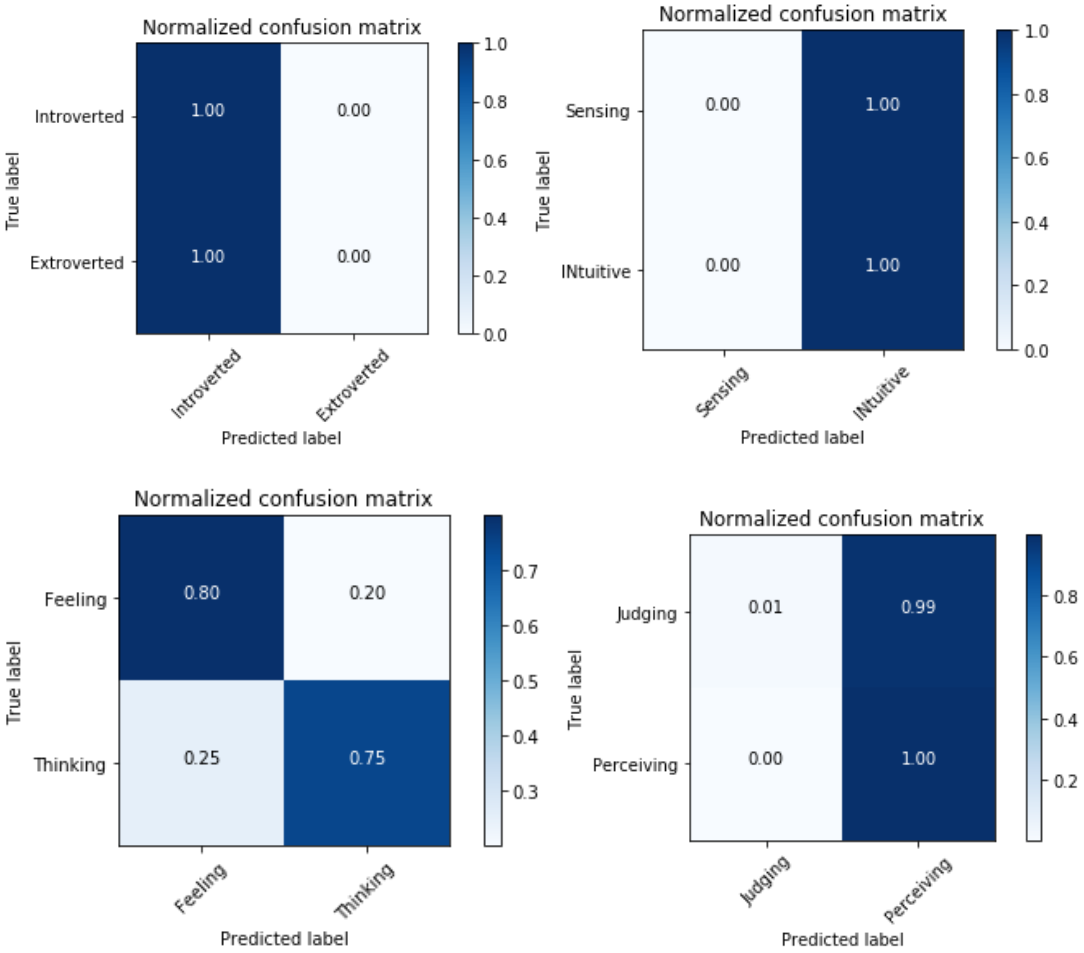
Logistic Regression



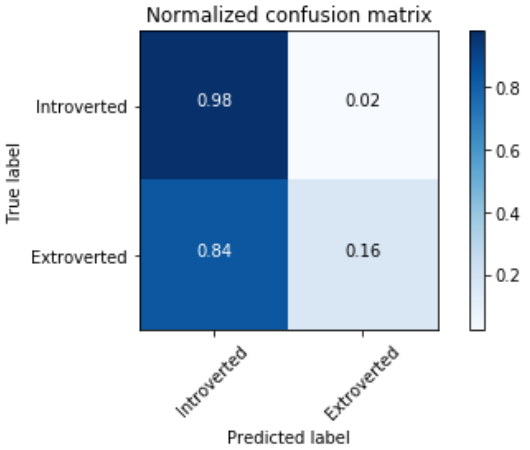
Random Forest

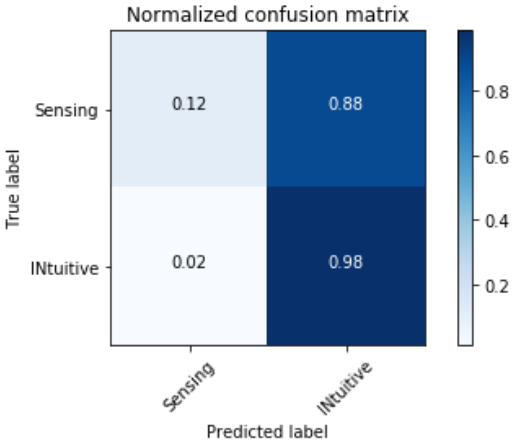


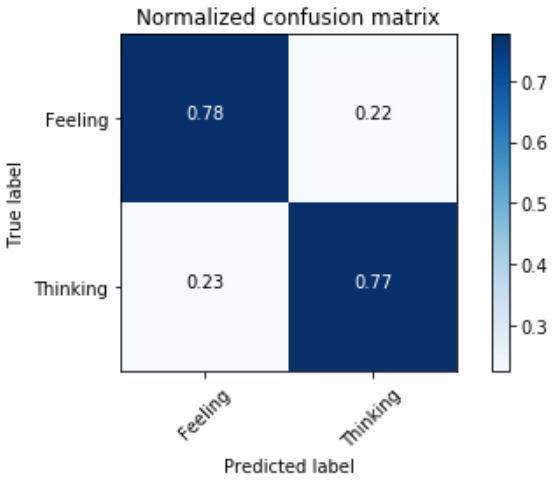
SVC

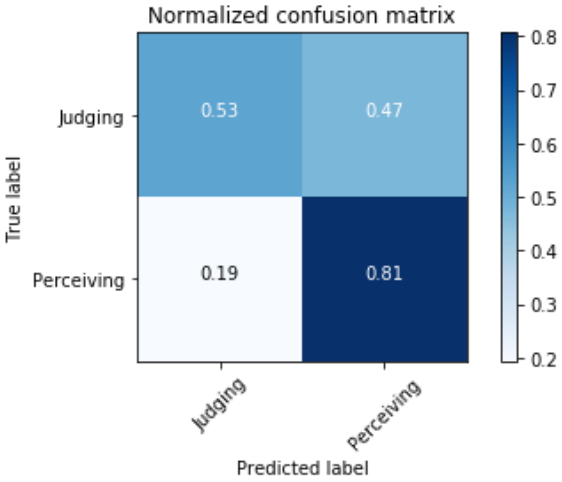

Normalized confusion matrix
Introverted
Extroverted
True label
1.00
0.00
1.00
0.00
Introverted
Extroverted
Predicted label
1.0
0.8
0.6
0.4
0.2
0.0
Normalized confusion matrix
Sensing
INtuitive
True label
0.00
1.00
0.00
1.00
Sensing
INtuitive
Predicted label
1.0
0.8
0.6
0.4
0.2
0.0
Normalized confusion matrix
Feeling
Thinking
True label
0.80
0.20
0.25
0.75
Feeling
Thinking
Predicted label
0.7
0.6
0.5
0.4
0.3
Normalized confusion matrix
Judging
Perceiving
True label
0.01
0.99
0.00
1.00
Judging
Perceiving
Predicted label
0.8
0.6
0.4
0.2

MLPClassifier

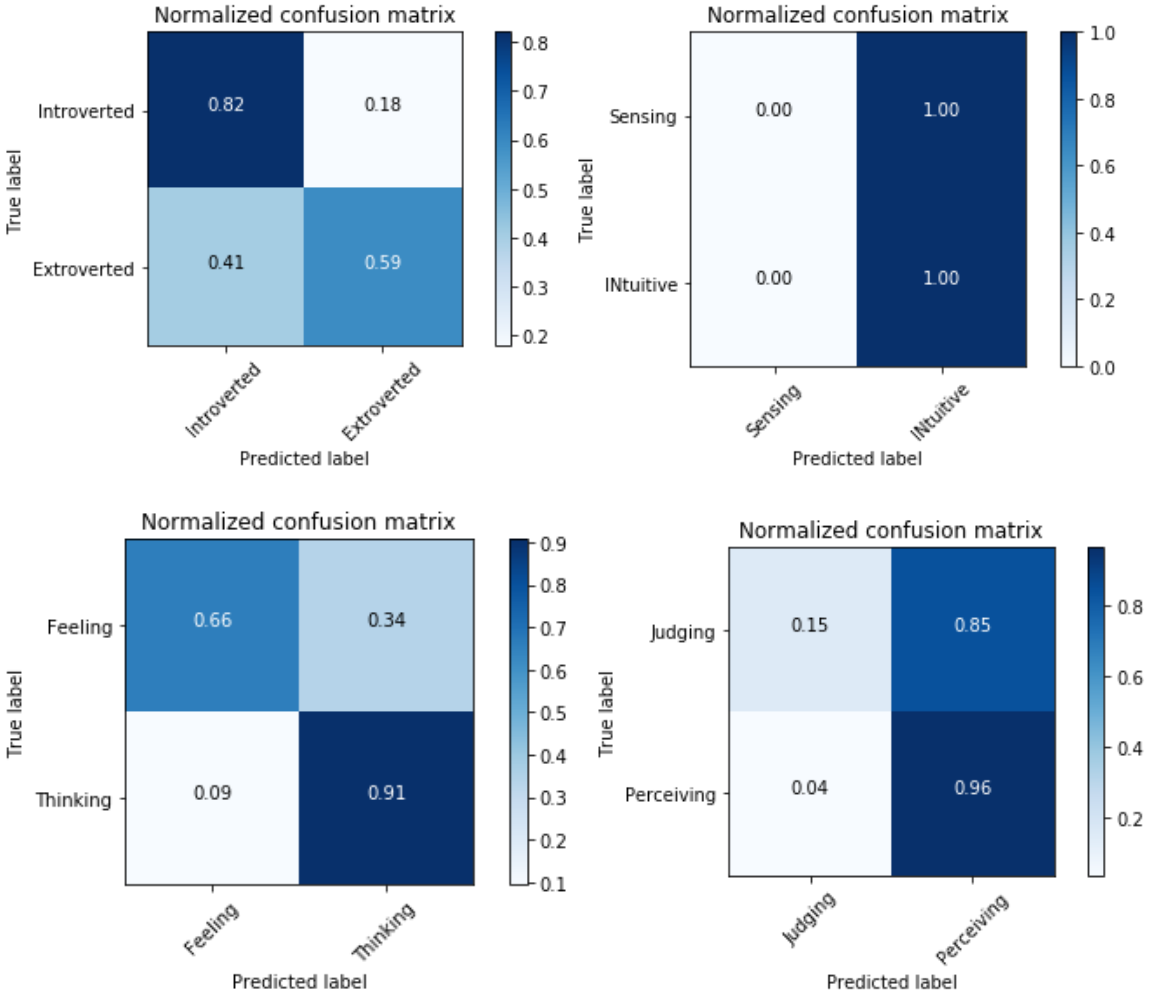


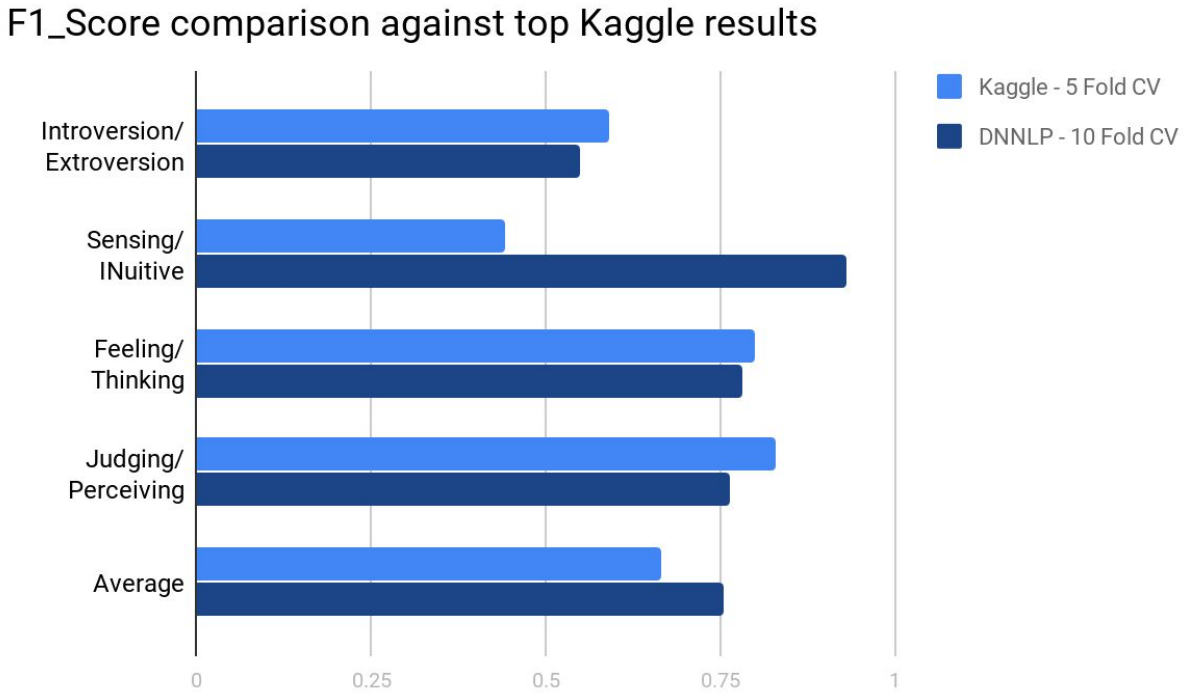




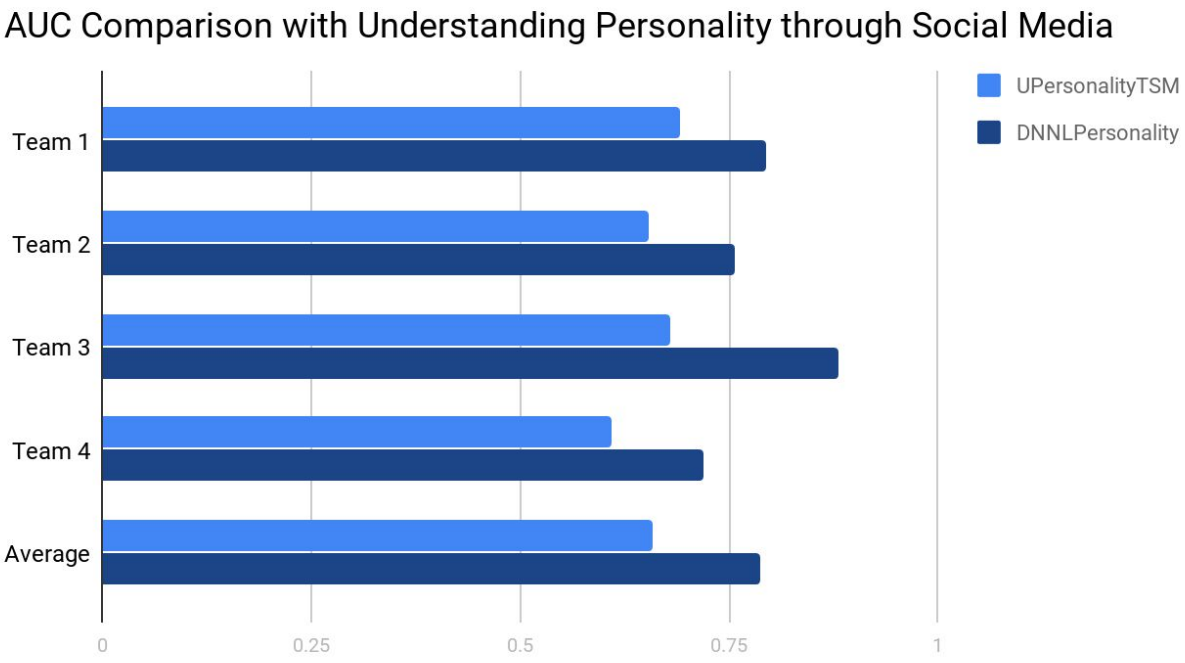


DNNLPersonality





https://www.kaggle.com/depture/multiclass-and-multi-output-classification/notebook



https://nlp.stanford.edu/courses/cs224n/2015/reports/6.pdf

Reflection

Clearly most of the credit for advancements in this field go to SpaCy, since their Doc 2 Vector NLP library is clearly extracting more information from the text than any other implementations that I can find.

To summarize, I clean each person's posts by removing the links and separating characters. Break the 4 letter type into 4 individual letter for binary classification predictions. Run each corpus of posts through SpaCy's Doc2Vec function which outputs a 300 dimension word vector that represents how the language was used against pretrained internet word embeddings.

With these word vectors and labels, I ran logistic regression, random forest, and svc algorithms and tested them based on accuracy, AUC, and F1_score and 5 Fold cross validation before doing the same with a simple DNN MLPClassifier, all of which are from sklearn.

Finally using Keras and Tensorflow I created DNN's to predict the 4 letters of the MBTI. Used 10 Fold Cross Validation and the same metrics as above and achieved excellent results. However there is still plenty of room for improvement on the data usage, cleaning, preprocessing, hyperparameter tuning, and algorithms used to predict and understand people's personalities.

One particular thing I found difficult was preprocessing the data for the Keras/ Tensorflow Neural Network. For the simple classifiers I had to input a 1D stream of data, whereas I needed 3 dimensions for the LSTM NN. At first I was getting very confused, going back and forth between different preprocessing methods getting one to fit and then not the other. I did not realize that I needed to make two separate datasets in order to make this project work. However, what I found interesting was that the SpaCy word vectors take into account dependencies and time series when creating word vectors. Since this is information is encapsulated, then the LSTM can focus on more broad patterns than one understanding and predicting the likelihood of seeing the phrase. Finding the right mix of abstraction of language without losing information in the averaged word vectors will be integral in understanding the patterns of the human psyche through language.

Another interesting thing I had trouble with was one-hot encoding the personality data. Since there are 16 different personality types, one method would be to make each prediction class equivalent to one personality type resulting in an array of length 16. Another way would be to predict whether or not each person was Introverted, Extroverted, iNtuitive, Sensing, Thinking, Feeling, Perceiving, or Judging, resulting in an array of length 8. Or even take it one step further, since one cannot be both Introverted and Extroverted according to the MBTI, we could assign Extroverted a value of 1 and Introverted a value of 0. This would result in an array of length 4.

Improvement

One aspect of the implementation that could have been improved making 2D or 3D tensors and cleaning the data in a way that didn't require losing people with less than 50 posts and any post after the 50th one, losing information from links, or result in a skewed distribution

of classes in the dataset. I ran into trouble just cleaning the data, and never was able to get a 2D or greater tensor to fit into keras. This way, the most information would be preserved by the corpus of the users and the model would result in a more transferrable prediction to the general population, rather than specifically Kaggle members. This would increase the dimensionality of the data further and allow for a convolutional neural network, however, this takes more than 2 days to process on my quad-core processor and requires padding or otherwise losing information in some posts as some people use more words than others.

Other improvements would be to use much better data, this is clearly skewed toward Introverts and Intuitive Functioning personalities. Getting more information from links, removing the types from posts, as well as adding other features like post count, number of words per post, characters per word, weighing certain words more heavily, among many many others. I also tried breaking the problem into smaller sections, learning on introverts to create weights that would better predict the remaining features given the person is introverted. This way, I could create hidden markov models with hidden variables that are extracted from individual filters to create a new model that would predict more accurately.