1. **Data exploration:**

There are three rows with missing values. We don’t consider those rows while data visualization as well as model training.

1. **Data Visualization:**

There are think distinct type of features in data set.

1. Date-Time features such as day, year
2. Tweet meta-attributes such as is\_retweet, has\_media, source
3. Tweet-Textual

I did visualization for all three types separately. Observation from the same are as follows:

* 1. **Date time features:**

We see that we don't have tweets for all the days for every author. In fact, we have sampled tweets from every author from a specific time window. As such, only seasonality which we can tell from this data if the authors' tweet pattern follow weekly or hourly trend. Apart from this, other time variables will not tell the actual tweeting pattern of the author. We could see definitely that each author has different twitter activity on different days of week. Donald Trump has maximum tweets on Sunday whereas Obama tweets mostly on week days (Thursdays and Tuesdays). Similarly some authors tweet during morning hours whereas some have more twitter activity at night.

Though, this tells us about the twitter activity of the author, using information about the day and time of tweet we can tell if an author tweets during a particular time in a particular day of the week. However, it does not makes sense to attribute tweet to an author because the tweet appears during a certain period of time in day or particular day of week. Hence, we will not include these variables in building the model.

* 1. **Tweet meta attributes:**

All the tweets in the data are original tweets of the author and none are retweets. Hence, is\_retweet column does not add any value to model training. For all other attributes we see distinctive pattern for each author which is different from others.

We could see most of the tweets are in English language next being followed by und – which presumptively means undecided, and followed by Spanish (es).

* 1. **Tweets:**

Here we first remove punctuation marks and make all the characters lower then we see which are the most common unigram(single word) and bigram (two consecutive words) for each author. We do this in two ways. One after removing stopwords and other with no stopword removal.

It is quite clear that different authors tweet about different topics as indicated by different prominent words used by authors

1. **Model Design:**

As with every machine learning model, our model needs to

1. Approximates the training data as closely as possible.
2. Generalizable for unseen data (Avoids overfitting)
3. Has good runtime performance i.e. it takes less time to run inference and occupies small disk size

Since we are dealing with textual data, feature engineering (pre-processing of text data and feature extraction of data) along with model selection and training is going to be equally important. Our model should be such that it could handle high dimensional sparse data as we are dealing with textual data here.

Considering all this we decided to use tfidf feature transformation along with other features and Support Vector machine as our model.

* 1. **Feature Engineering:**

**Selecting Important features**

As discussed we are not considering date-time related features and is\_retweet feature.

The content of the tweet itself tells about the language of tweet implicitly. So in the final model we also include the language feature. However, language feature will be used in deciding whether to do stemming or not

**Text preprocessing:**

We remove all the punctuation marks from the text and make all the characters lower string. Considering the tweets are of few characters, it is not clear whether to do stemming will improve the model. As such I have written a custom Transformer which can be configured to do stemming by providing a False and True argument. We use cross validation to decide if we should do stemming or not. We do remove stop words from the tweet. We consider all English, Spanish, Portuguese stopwords and in addition the words ‘http’ or ‘https’ because this word indicate the tweet has an url which is already a feature.

**Feature Extraction:**

For tweet column we use tfidf as feature extraction. Tfidf is good representation of textual data for text classification purpose as it gives more weightage to words/terms which are specific to particular class, thereby helping the model classify the text into suitable class. One downside is that such representation results in high dimensional sparse feature space. However, SVM model has good performance for such scenarios.

With respect to tweet meta attributes every feature except for source info is binary. We use one hot encoding for source-info to get numerical features.

It’s seldom that hyperplane in SVM separating the classes would be linear. Mostly, it will be non-linear. However, building SVM with Exact Kernel transformation is computationally intractable for large datasets. As such we use Nystroem Kernel approximation on subset of training data to get high dimensional kernel features and use these approximation as features in linear SVM model to predict the output variable.

* 1. **Model Selection:**

We have selected SVM as our model. The main advantages of using SVM are:

1. In cases, for high dimensional feature space or if the feature space is very sparse, Support Vector Machine (SVM) would be good candidate. In such cases, SVM has much better performance than Random Forest.

2. SVM is also robust to overfitting because it contains the epsilon parameter which acts as regularizer thereby preventing overfitting. Epsilon parameter can be fine-tuned to get optimum model performance on test data.

3. As SVM uses hinge loss to get the optimum hyperplane separating the two classes, it is robust to outlier presence. The soft margin-based hyperplane depends solely on few points known as support vectors which are close to the hyperplane. Other points don't have much of an impact on the optimum hyperplane.

4. We can account for correlation between input features by adding regularization term to the hinge loss. Adding regularization term, the weights of variables which are artificially boosted as they are related to another variable which strongly impacts the outcome variable.

5. Although non-linear decision boundaries are not modelled accurately by SVM, we can use kernel transformation of features to higher dimensional however it is not possible to get feature importance as the features are already transformed due to kernel transformation.

6. SVM is sensitive to variable scale and it gives more weightage variables of higher magnitude. However, we can overcome this by first re-scaling all the variables, so they have the same range

* 1. **Hyperparameter Tuning**

It’s critical to use optimal values of hyper parameters of the model so that model generalizes to unseen data properly. We use randomized grid search cross validation in sklearn to do hyperparameter tuning. The hyper parameters we consider for tuning are:

**SVM model:**

alpha: The value by which the penalty term added to the loss is multiplied. It controls overfitting and feature selection.

l1\_ratio: ratio of l1 penalty to l2 penalty with 0 indicating l2 penalty and 1 indicating l1 penalty.

Epsilon: It controls the hyperplane width and thereby overfitting.

**Feature Extraction:**

do\_stem: whether to do stemming or not values (True, False)

Kernel: (linear, radial-basis function (rbf), polynomial, sigmoid)

Gamma: this defines how far the influence of a single training example reaches, with low values meaning ‘far’ and high values meaning ‘close’

value range (0.0001, 0.001, 0.01, 0.1)

* 1. **Accuracy Metric**

To judge the model performance, we use micro averaged f1 score as metric to evaluate model performance. As there is no class imbalance except for one Author Donald Trump. Using micro averaging for f1-score makes sense. In micro-f1 score we look at the individual predictions and take the average across the overall population. Micro f1-score is useful in scenarios where the datasize varies whereas macro-f1 score is used when there is class imbalance.