David Chang and Jared von Halle

CSC 478

Final Project

Due 6/6/16

# Introduction

We chose to perform data analysis on the World Food Facts data provided by the Open Food Facts Database. The World Food Facts dataset comes in the form of a single table, where each row is a different food. The columns represent facts about each food. Facts include textual information such as what countries the food is produced in, the generic name of the food, the product name, information about packaging, the branding, nutrition labels, and much more. There are also many columns related to the nutriment contents of each food. This data is given in the form of how many grams of the nutriment the food contains per 100 grams of the food.

# Objectives

Our objective, in the most general sense, was to explore the World Food Facts data to determine if we could find patterns between different foods or different qualities of foods. We were particularly interested in exploring what we could predict based on the nutrition of foods.

One angle we took was to see if we could find patterns between the nutriment makeup of foods and the countries that they are produced in. We hypothesized that different countries would have tendencies to produce foods with different nutritional contents. Therefore, we were hopeful that we could predict what country a food was produced in based on the nutrition facts of that food. We used various classification methods to attempt this, which are described further in the sections below.

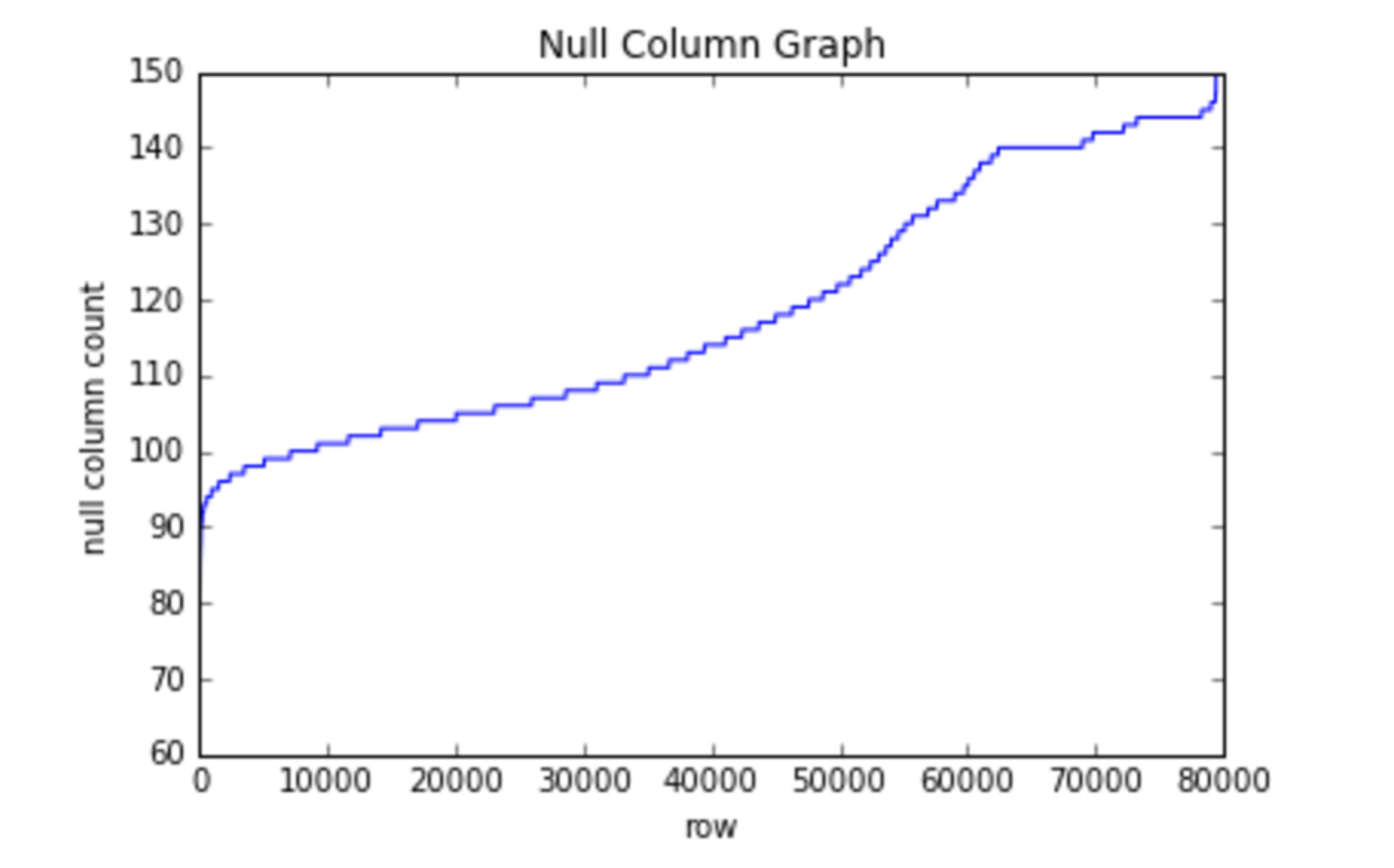
We also sought to predict the nutrition score of a food based on the nutriments that it is made up of. The nutrition score, which was included in the data for each food, is a calculation of the food’s overall health from the UK Food Standards Administration. We used various regression analysis techniques for this prediction, discussed more in the sections below.

Finally, we sought to separate foods into various buckets based on the their nutriment makeup. The data included a “category” field, where each individual food is put into a subjective category of food such as meat, dairy, sugary snacks, etc. We explored techniques for classifying each food into these categories based on their nutriment makeups. We used clustering techniques to accomplish this task.

# Predicting Where a Food is Produced

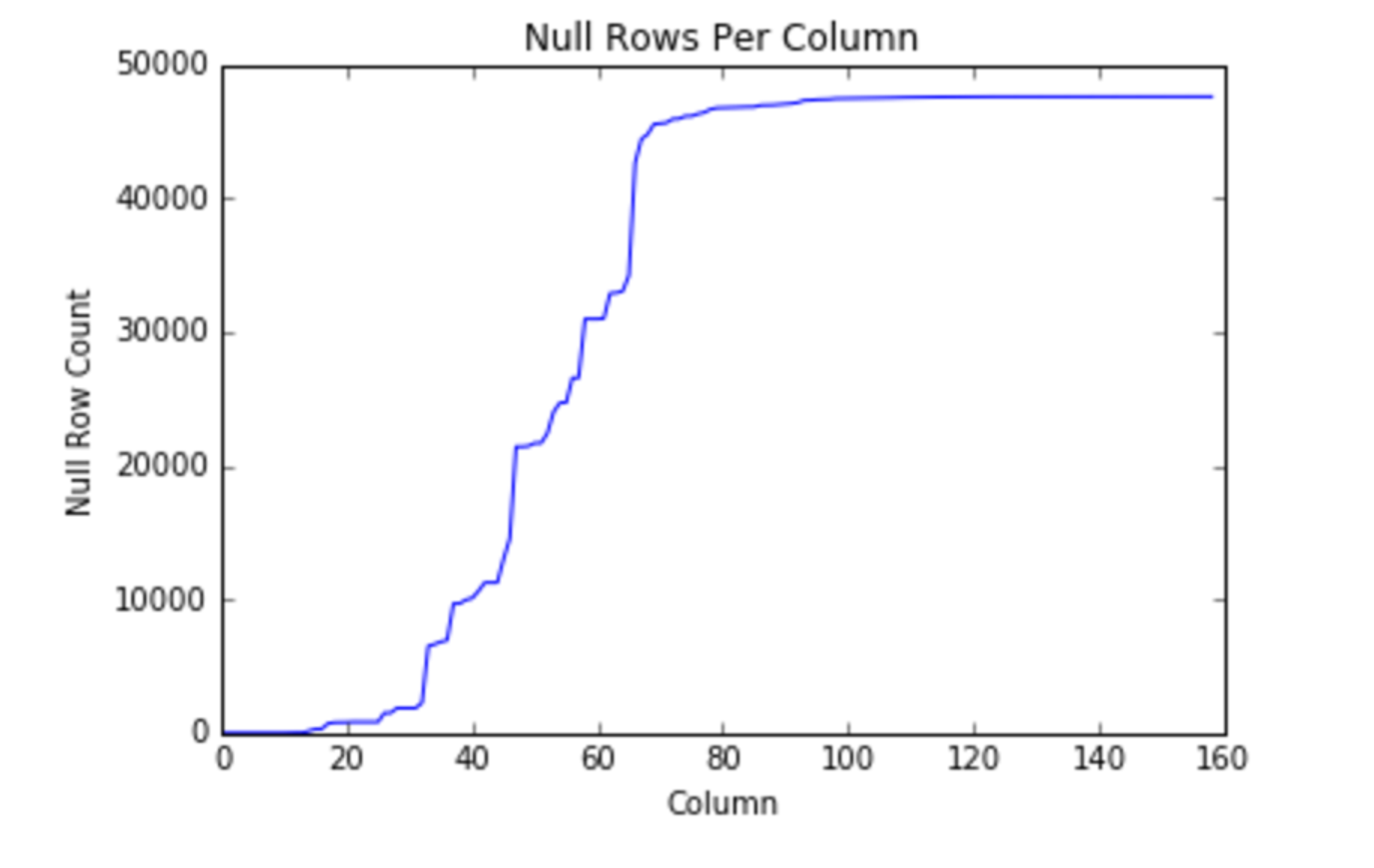
## General Data Preprocessing

The World Food Facts data is very sparse. We did not want (nor could we) completely ignore all sparse fields in the dataset. However, we for this section, we did remove rows and columns that were excessively sparse. We first examined the number of columns that were null for each row. The plot below displays the number of null fields for each row in the data.



There were 159 columns in total in the graph. You can see in the graph above that for some rows, almost all of the fields were null. For this section, we decided to remove any rows that had 120 or more null fields.

We then examined the sparseness of certain fields (columns) in the data. The graph below shows the number of rows that contain a null value for each column.



After removing the sparse rows in the previous step, there were 47,586 rows left in the data. In this case, we decided to remove columns from the dataset where more than 35,000 rows contained a null value for that field.

This trimming left us with a matrix of size 47,586 x 66, which is down from an original dataset of size 79,470 x 159.

For the sake of predicting what country each food was produced in, we did not want to consider foods that were listed as being produced in multiple countries. These foods represented a very small portion of the data (2.8%), so we removed these rows. This left us with a matrix of size 46,210 x 66.

Since our goal was to predict the country a food is produced in based only on its nutriment makeup, we limited the records for this section to include only the columns for nutriment values. This reduced the size of the matrix to 46,210 x 11.

We replaced all null values with the mean from each column and normalized the data. The final pre-processing step was the split the data into training and test sets.

## Nearest Centroid Classifier

The first technique that we attempted for the country classification was Nearest Centroid Classification. This is similar to the Rocchio method that is used for text classification, in that a prototype vector is calculated for each category. In our case, the categories are the countries that Foods are produced in. Our technique differed from Rocchio, however, because we were not dealing with text classification. Therefore, it didn’t make sense for us to use the TFIDF approach. Instead, we calculated the prototype vectors by calculating the average of each vector in the training set pertaining to each category (each country). We accomplished this by adding the vectors in the training set together for each country, and then dividing by the number of vectors that were added together.

Our main motivation for utilizing the Nearest Centroid Classification approach was performance. With this large dataset, we expected a performance gain from this approach because much of the work can be accomplished prior to the actual classification of an individual row. Specifically, we could calculate the prototype vectors using the training set ahead of time. During the actual classification, the row being classified only needed to be compared to the prototype vectors. This is opposed to other approaches where the row being classified must be compared to all other rows in the training set during the actual classification step.

After creating the prototype vectors, we used the test portion of the data to see how well the approach worked. We used cosine similarity as our distance measure, looking for the prototype vector that was closest in distance to each row in the test set.

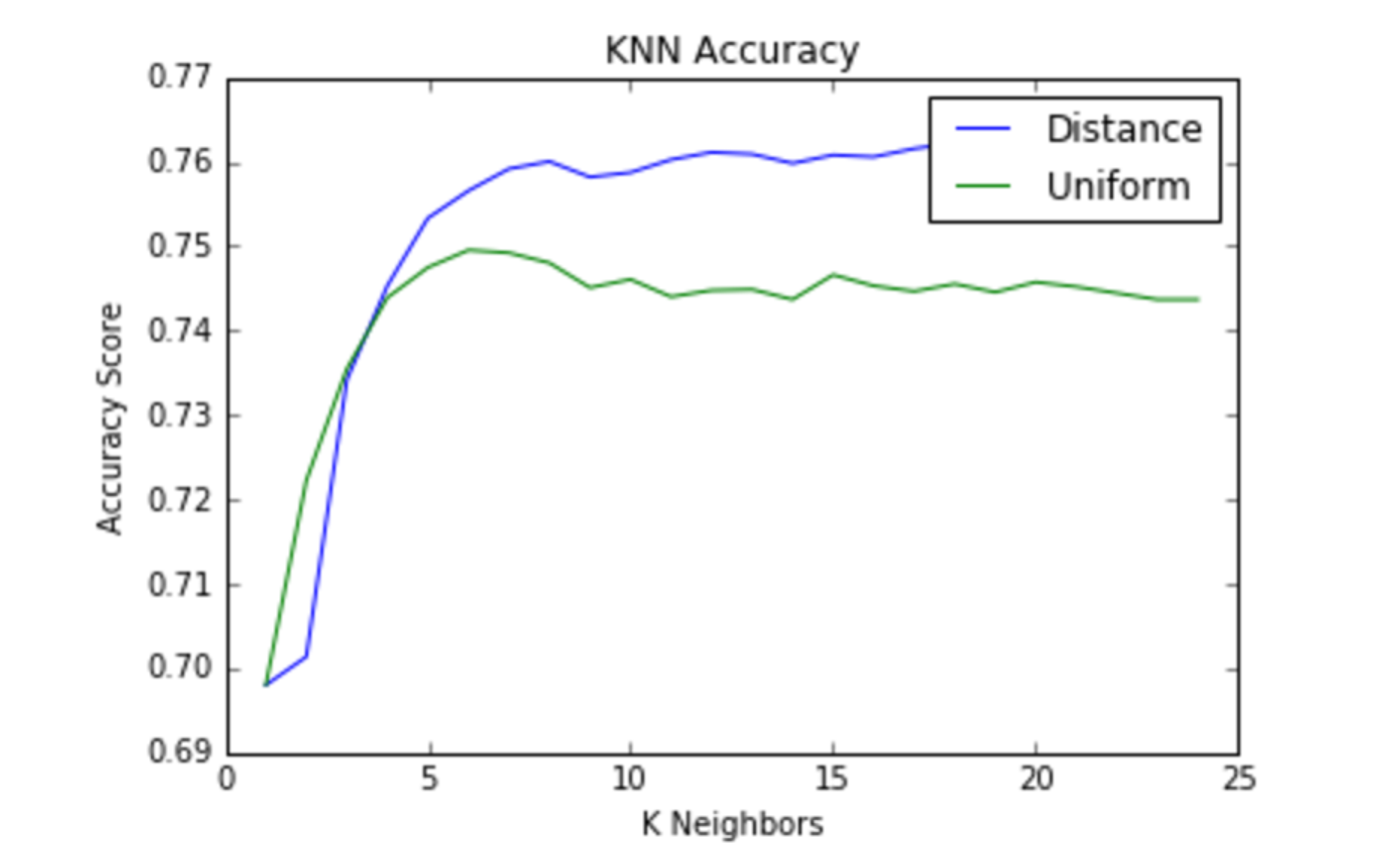
The approach did not work well at all. It got only .065% of the classifications correct. This is far below the number you would expect to get right if you randomly guessed. We believe this failure is due, in part, to the fact that the data is not evenly distributed across the various categories. A huge portion of the foods in the dataset was produced in France (74%), even though there were 59 unique countries in total. This presents an issue with our approach, because the classification predictions are not weighted to favor countries that appear more often. For example, if a test case most closely resembles a one country, but is also very close in distance to France, the most common choice, it might be a better guess to go with France.

To account for some of the skew, we re-tested this approach without France included. In other words, we eliminated all rows where the food was produced in France. This did not result in a dataset with balanced categories, but it was better, at least. For this test, the approach got 8% of the classifications correct. This is certainly not great, but at least it beats the number you’d expect to get right by random chance.

## K Nearest Neighbor Approach

We examined the same question, “can we classify which country a food was produced in based on its nutriment makeup” using the KNN approach. We thought this approach should fare better than the Nearest Centroid Classifier because it naturally has the weighting towards more common categories that was missing in the latter.

We explored which formula (distance or uniform) performed better, as well as the ideal number of neighbors to use. The graph below shows the accuracy results for both formulas across a range of neighbor values.



The best results were obtained using the distance measure and 20 neighbors. When we ran the KNN Classification with these parameters, we got 76% of the classifications correct. This beat the number that you would expect by chance significantly. Another method we used to evaluate how well the approach fared was how it compared to just guessing “France” every time, since France was by far the most common classification answer. Our KNN accuracy of 76% beat the “guess France” technique, but not by much. The “guess France” technique results in an accuracy of 74% for our particular train, test split. Interestingly, the KNN approach resulted in an accuracy of 96% on the training set. This suggests that perhaps the model was over-fit.