Stage 4 Write Up

First, we randomly sampled 400 rows from our candidate set, but our manual labeling returned fewer than 50 matches, showing us that our data faced the problem of skew. We reconsidered our blocker, and found that we were producing far too many negatives by including all combinations that matched on release date. After changing our blocker to not block on this value anymore, we generated a new sample data set, which we manually labeled, finding 96 positives and 304 negatives. With this new blocker, our revised candidate set has 2114 rows. Projecting from the proportion of positives in our sample, we may expect to find more than 500 total positives in our entire candidate set.

Next, we turned our our sample data into feature vectors. The features that we derived are: Jaccard Index on album name, Jaro Distance on album name, Jaccard Index on artist name, Jaro Distance on artist name, and the absolute value of integer difference on release year, for a total of 5 features. It is useful to compare Jaccard measures with Jaro measures, because they are two different methods to approach string similarity, the former being a set similarity, and the latter being an edit distance. Also, we decided to measure the difference in release year, because artists seldomly release multiple albums in the same year, while the specific day of the release can vary depending on region. Also, this allows us to include data that is missing release month and day information. On top of this, it can be an extremely valuable feature for separating original releases with remastered or compiled releases, which are often released at least ten years apart but will have similar album and artist names.

We then split our table of feature vectors into set I of 275 rows, and set J of 125 rows.

Results from cross validation on I

We opted to perform 3-fold cross validation. All models returned very high results (precision, recall, and F-1 scores of 1 in many cases), so we found minimum values to be the most interesting. All 3 results from each cross validation are available for each measure in our Jupyter Notebook. Our cross validation scores for each measure type (F1, precision, recall) are derived from unique trainings.

Minimum values for each cross validation:

Decision Tree:

Precision: 1
Recall: 0.95238095
F-1: 0.97560976

Random Forest:
After our cross validation we have determined that **Decision Tree** is the best learning model to use for our data matching. While all models had exceptional results, the decision tree happened to have the highest minimum F-1 score during our cross validation.

**Debugging and Testing:**

After testing on set J and looking at the data, we noticed two things. First, all five of our classifiers were highly accurate. Second, the only errors our models were producing were false negatives. After going into the data and looking at the false negative examples we determined that these were edge cases that could not be easily accounted for in feature creation. In a business environment, it may be possible and valuable to manually clean such unsystematic abnormalities out of the data, and in this case the models could achieve near-perfect accuracy. We performed no additional rounds of matching after this debugging.

*Here are the results of testing on J after learning on I. Of particular note are the testing results from Decision Tree, which performed the best in our cross validation, and we are aiming to use it as our final matcher.*

**Decision Tree:**

Precision: 31/31 = 1  
Recall: 31/34 = 0.91176470588  
F-1: \( \frac{2 \times (1 \times (31/34))}{1 + 31/34} = 0.95384615384 \)

**Random Forest:**

Precision: 32/32 = 1
Recall: $\frac{32}{34} = 0.94117647058$
F-1: $2 \times \left( \frac{1 \times (32/34)}{1 + 32/34} \right) = 0.96969696969$

SVM:
- Precision: $\frac{33}{33} = 1$
- Recall: $\frac{33}{34} = 0.97058823529$
- F-1: $2 \times \left( \frac{1 \times (33/34)}{1 + 33/34} \right) = 0.98507462686$

Naive Bayes:
- Precision: $\frac{33}{33} = 1$
- Recall: $\frac{33}{34} = 0.97058823529$
- F-1: $2 \times \left( \frac{1 \times (33/34)}{1 + 33/34} \right) = 0.98507462686$

Logistic Regression:
- Precision: $\frac{33}{33} = 1$
- Recall: $\frac{33}{34} = 0.97058823529$
- F-1: $2 \times \left( \frac{1 \times (33/34)}{1 + 33/34} \right) = 0.98507462686$

Our testing on J showed identical results from the SVM, Naive Bayes, and Logistic Regression classifiers, in which each only produced one false negative and no false positives from J. More generally, we are witnessing high results that imply that our data is of high quality and is relatively uniform across both sources. It is also interesting to note that Decision Tree performed worse than the other four methods when tested on J, with 3 total incorrect classifications (specifically, 3 false negatives) from 125 predictions. We believe that this is still highly performant.

Time Estimates:
- a) With the time spent tweaking our blocker to produce a candidate set with a higher proportion of matches, it took us about 5 hours to label our data.
- b) We spent approximately 8 hours converting our sample data into feature vectors and analyzing our 5 learning based matchers.