Assessment #1

Overview

This document contains a summary of the machine-learning model deliverable outlined in Assessment #1. It covers initial data exploration, data processing, feature engineering, and model training. This document only provides a high-level overview of the steps taken; for further detail please refer to the Jupyter notebook listed at the beginning of each section.

Data Exploration

notebooks/FindingNulls.ipynb

It is not clear from the assessment description which rows are "corrupted" and which are not. Therefore, the complete file was downloaded from S3 and a preliminary analysis conducted to identify which rows contained corrupted data. The following hypotheses were tested:

1. Corrupted rows had different numbers of fields

If the corrupted rows had the Make field removed, it is possible they would have had different numbers of fields. Therefore, the length of the file header was calculated (19) and each row was inspected to test whether or not it had 19 fields. No rows had less than 19 fields; therefore, this hypothesis was rejected.

2. Corrupted rows contained oddly formatted Make fields

It is possible that corrupted rows had a strangely formatted value in the Make column (e.g. "NULL" or "NA"). A set of unique car makes was compiled and visually inspected to test this hypotheses. No oddly formatted Makes were found; therefore, this hypothesis was rejected.

3. Corrupted records appear in either the first or second half of the file

It is possible that the corrupted rows appear in the first half of the file. A count of rows with a blank string in the Make column was obtained for both the first and the second half of the file. Because there was no great difference in the number of rows matching this criterion in the first half of the file versus the second, this hypothesis was rejected.

Having rejected my other hypotheses, I came to the conclusion that corrupt rows contained a blank string in the Make column. I continue to make this assumption throughout the rest of the analysis.

Model Training

notebooks/SKLearnRandomForest-V2.ipynb

Data loading

Model training first began by loading the uncorrupted segment of the citations dataset into Pandas. This process involved determining the proper data formats in the citations dataset and specifying the data types upon load. All columns requiring additional processing to conform to their expected data type (e.g. Issue Date and Plate Expiry Date) were formatted after the data load.

Data labeling

In order to generate the outcome variable for the machine learning task, the top 25 vehicle makes were identified. All citations involving a vehicle in the top 25 makes were labeled with a 1; otherwise, they were labeled with a 0.

Data preprocessing & feature engineering

First, a count of NULL values was obtained for each column. Columns with a high percentage of nulls were dropped from the dataset under the assumption that they would not contain enough variation to be relevant to the machine learning model.

All columns were scrubbed of NULL values and converted to a numerical format so that they could be used in a machine learning model.

NULL values were dealt with in the following ways:

1. If the column contained numerical data (e.g. integer or float), NULLS were replaced by the column's median value.
2. If the column contained Date data, NULLS were replaced with the number of days since 01/01/2000.
3. If the column contained categorical data, NULLS were replaced with the column's mode value.

After dealing with NULL values, columns were converted to numerical values. The columns were converted in the following ways:

1. Date observations were converted into the number of days since 01/01/2000. This date was prior to the minimum date for all fields and so produces a positive integer for all Date observations.
2. Categorical variables were converted to a numeric value using the Label Encoder provided by Scikit-Learn (e.g. the string category was converted to a number between 1 and the total number of categories).
    1. This approach was chosen in lieu of the one-hot encoding scheme for the following reasons:
        1. Many of the categorical variables had high cardinality and thus would have added too many dimensions.
        2. Converting one of the lower-cardinality variables to a one-hot vector did not lead to a significant increase in accuracy.

Finally, the Latitude and Longitude coordinates were converted into the number of miles from the Los Angeles, CA city center (on the assumption that the data contained LA parking citations). I make the assumption that the Latitude and Longitude coordinates are in the California State Plane format; therefore, this calculation leveraged the Euclidian distance formula. This was used in lieu of the raw Latitude and Longitude measurements largely due to the ease of understanding and explaining one-dimensional features versus two.

Model Training

Due to the high percentage of observations belonging to the top 25 makes, a Random Forest model was chosen to predict the probability that a given citation involved a vehicle in the top 25 makes.

Random Forest was chosen due to its resilience to skewed data compared to linear models as well as its relatively low number of hyperparameters (i.e. ease of use).

Hyperparameters were chosen informally; after increasing their values several times to compare performance, the number of estimators was set at 100 and the maximum tree depth set at 20. Informal methods were used due to the size of the dataset and time required to train the model; given a necessity for fine-tuning and access to additional computing resources a full grid search would have been implemented.

Model Evaluation

The Random Forest model was evaluated on the 25% of the uncorrupted citations dataset held out for testing.

Given the highly skewed nature of the data (91.4% of observations belonged to the top 25 makes), the Random Forest prediction model was compared to a baseline of simply guessing a "1" for each observation.

The Random Forest model implemented for this assessment exhibited a slight improvement over the baseline guess at about 92.4% accuracy. While this is only a 1% improvement over the baseline guess, the classifier was successful at identifying about 15% of citations that did not involve a top-25 Make vehicle.

It is important to take into account the importance of each of the features in this model. The most important feature is the vehicle's Body Type, which would indicate that atypical vehicle makes are associated with a particular physical shape.

However, the next most important features are the Route, distance from the LA city center, Issue Time, and Location. There is no *a priori* reason to expect that these variables would be associated with a vehicle's make. Most likely one of two factors is at play:

1. Atypical cars tend to belong to specific owners, who park their vehicles in similar locations throughout the city.
2. Police officers monitoring a certain route at a certain time of day are likely to misspell vehicle makes on the citations they issue (e.g. ACRU versus ACRA for Acura), leading to atypical makes being erroneously reported by specific officers.

For these reasons, this model is unlikely to generalize beyond the geographic region where the data was collected. Furthermore, the model would have to be retrained over time as owners of atypical cars either sold their vehicles or moved to different locations.

Finally, given the highly skewed nature of the data, this model is not particularly effective in identifying atypical vehicles (i.e. those outside of the top 25 makes). If there is a high cost associated with failing to identify an atypical make, a different algorithm (e.g. anomaly detection) should be considered.

Future Development

The XGBoost algorithm was not compared to the Random Forest algorithm for this assessment, largely due to the size of the data, the time required to train the model, and the introduction of additional hyperparameters in XGBoost. Future iterations would compare the performance of this model to XGBoost.

Additional feature engineering could also be done in this case. For example, the Label Encoder could be tested against one-hot encoding the categorical variables. The Latitude and Longitude coordinates could be also grouped into discrete clusters rather than mapped to a continuous variable representing their distance from the LA city center.

Finally, additional effort could be spent identifying which car make codes are actually different and which are simply misspelled by the issuing police officer.