Outline:

* Title
  + Kicked Car Predictions
* Abstract
* Introduction/Background
  + Hook – Generate Interest
  + What’s our goal and why is it significant
  + Preview
* Data Parsing and Early Interpretations
  + Early Interpretations
    - Possible trends based on plots and histograms
  + Data Parsing
    - Dealing with text heavy data set
    - Dealing with empty values
* Machine Learning Approaches and Results
  + SVM
  + Logistic Regression
  + Generative Models
  + Clustering
* Conclusion
  + Review
  + Significance

Kicked Car Predictions

**Albert Ho Alice Wu Robert Romano**

# 0. Introduction

When you go to an auto dealership with the intent to buy a used car, you want a good selection to choose from and you want to be able to trust the condition of the car that you buy. Auto dealerships purchase many of their used cars through auto auctions with the same goals that you have: they want to buy as many cars as they can in the best condition possible. The problem that these dealerships often face is the risk of buying used cars that have serious issues, preventing them from being sold to customers. These bad purchases are called “kicks”, and they can be hard to spot for a variety of reasons. Many kicked cars are purchased due to tampered odometers or mechanical issues that could not be predicted ahead of time. For these reasons, car dealerships can benefit greatly from the predictive powers of machine learning. If there is a way to determine if a car would be kicked ahead of time, car dealerships can not only save themselves money, but also provide their customers with the best inventory selection possible.

The following paper is split up into 3 main sections describing our approach to solve this problem: Early Interpretation, Data Parsing, and Machine Learning Algorithms. First we identified possible trends by simply graphing data. Next, we came up with strategies for dealing with missing data and data presented in a text format. Finally, we applied various machine learning algorithms to the parsed data set in an attempt to predict which cars would have the highest risk of being kicked.

# 1. Early Interpretation

We obtained our data set from the Kaggle.com challenge “Don’t Get Kicked” hosted by Carvana. Our first step was to graph the data we had to gain some initial insight into which features would be the most important. The features we were provided along with their descriptions are shown in Table 1.1.

<Albert Write more stuff Here>

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **Short Description** | **Feature** | **Short Description** |
| RefID | Unique vehicle identifier | VehOdo | Vehicle Odometer reading |
| IsBadBuy | Was the car kicked? | Nationality | Manufacturer’s country |
| PurchDate | Auction Purchase date | Size | (Compact, SUV, etc.) |
| Auction | Auction provider | PrimeUnit | Car has higher demand? |
| VehYear | Vehicle year | AcquisitionType | (Auction buy, Trade in, etc.) |
| VehicleAge | Years elapsed VehYear | AucGuart | Guarantee provided by Auction |
| Make | Vehicle Manufacturer | ByrNo | Unique identifier for buyer |
| Model | Vehicle Model | VnZip | Zipcode of car purchase |
| Trim | Vehicle Trim Level | VnSt | State of car purchase |
| SubModel | Vehicle Submodel | VehBCost | Acquisition cost to buyer |
| Color | Vehicle Color | IsOnlineSale | Was this an online purchase? |
| Transmission | Vehicle transmission type | WarrantyCost | Warranty price |
| WheelType | Indicates type of wheel |  |  |

|  |  |
| --- | --- |
| **Feature** | **Short Description** |
| TopThreeAmericanName | Is manufacturer one of top three? |
| MMRAcquisitionAuctionAveragePrice\* | Average condition value at auction time |
| MMRAcquisitionAuctionCleanPrice\* | Great condition value at auction time |
| MMRAcquisitionRetailAveragePrice\* | Average condition retail value at auction time |
| MMRAcquisitionRetailCleanPrice\* | Great condition retail value at auction time |
| MMRCurrentAuctionAveragePrice\* | Average condition value currently |
| MMRCurrentAuctionCleanPrice\* | Great condition value currently |
| MMRCurrentRetailAveragePrice\* | Average condition retail value currently |
| MMRCurrentRetailCleanPrice\* | Great condition retail value currently |

Table 1.1

Feature labels and descriptions.

\*MMR refers to Manheim Market Report and is the source of these values

# 2. Data Parsing

Many of the features we were looking at came in a text format, so we had to decide how to properly bin them. Furthermore, some data entries were missing, so we had to make some choices for how to replace them.

***Word Bins***

For data such as the name of the vehicle’s model, manufacturer, and color, we had to assign unique identifiers to specific strings in the feature space. This was straightforward for a feature like transmission since we could assign 0 for Auto and 1 for manual. The process became more involved with features such as the car submodel. We decided that even though there were many different submodels, categorizing them with unique identifiers rather than grouping them was the more conservative option.

***Missing Features***

For many of the samples, particular features were missing. We had the option of throwing out the sample completely, but we believed that it would be a waste. We decided to implement the following rules: if the feature was represented with a continuous value, we would replace the missing value with the average of the feature over the other samples and if the feature was represented with a discrete value, we would create a new value specifically to identify missing data.

# 3. Machine Learning Algorithms

With the Data parsed and some initial insights to guide us, we started to apply some machine learning algorithms that would identify where we needed improvement and what strategy would be most effective

***Support Vector Machine***

First, we tested our data with an SVM. We used the liblinear package v. 1.92. Our initial runs thus far have only been on raw numerical data (not on all post-processed data), so there is lots of room for improvement. We submitted our solution to Kaggle and placed 526 out of 571 teams on our first run.

***Logistic Function***