A Data Analysis to Predict Customer Loyalty

By Ameen Abdelghani

December 2018

Problem Statement

- **Company :**
  - Elo is one of Brazil's top credit and debit card companies
- **Goal :**
  - They're offering a new perk to offer discounts on certain merchants
  - Wish to tailor the discounts to merchants users have actual demand for
- **Obstacle:**
  - How to understand the differences in a customers purchasing pattern
- **Solution:**
  - Elo's desired first step is to accurately predict a customer's loyalty based off their features

Why?

- Will boost customer engagement with Elo
- Give them incentives they care for
- Can create more accurate ad campaigns

" Get discounts to the shops YOU love! " - Elo

Data

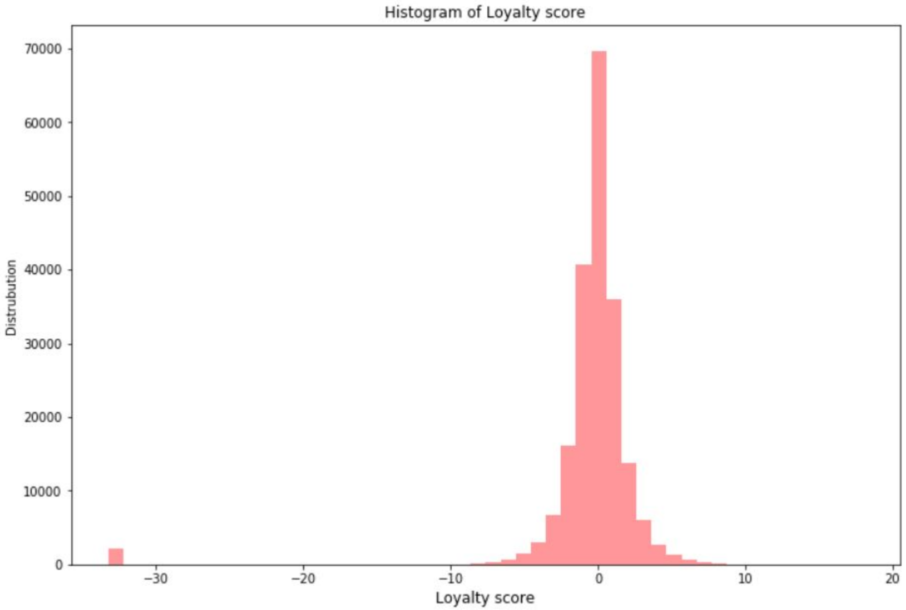
- The data for this analysis is part of a current Kaggle competition
- The data of use is the training, test, historical transaction, and new merchants data
- CSV format
- 200,000 observations approximately
- Majority of features are anonymous

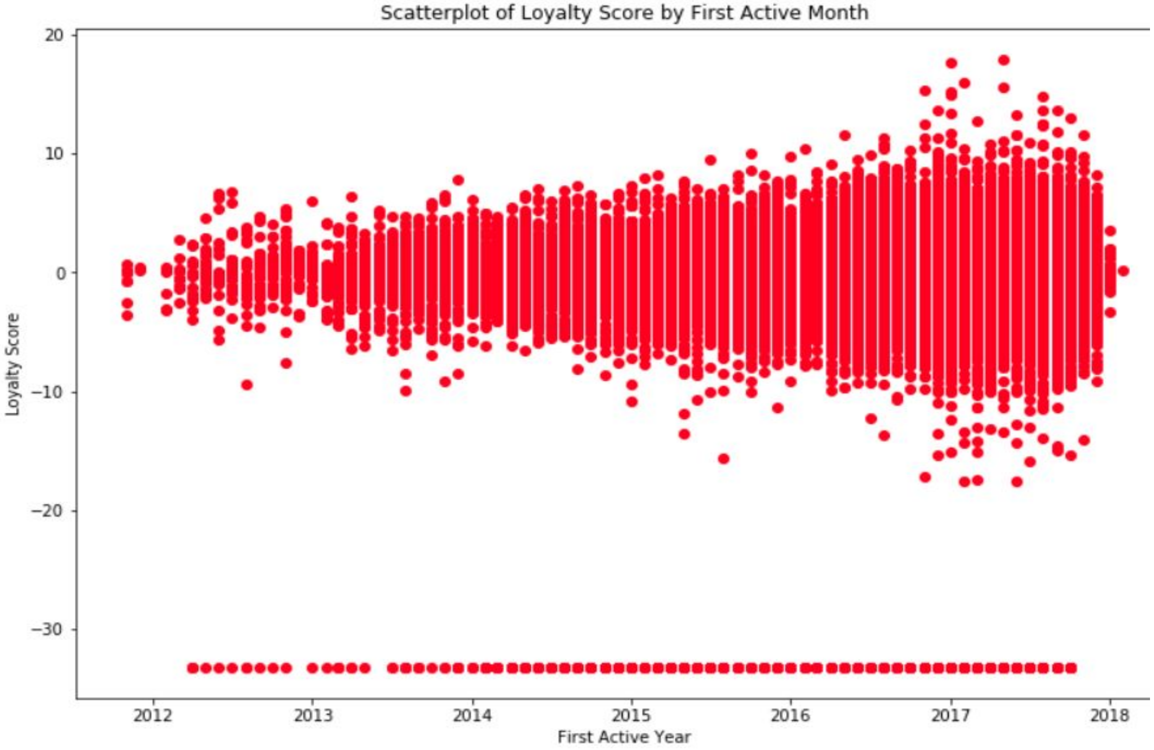
Data

- Columns on training data
  - Feature 1, 2 ,3
  - Card ID
  - First active month
  - Customer Loyalty
- Historical / New Merchants data
  - Authorized Flag (Yes or No)
  - Month Lag (how long before they visited the shop)
  - Purchase Amount
  - City ID / Merchant ID
  - Other anonymous features

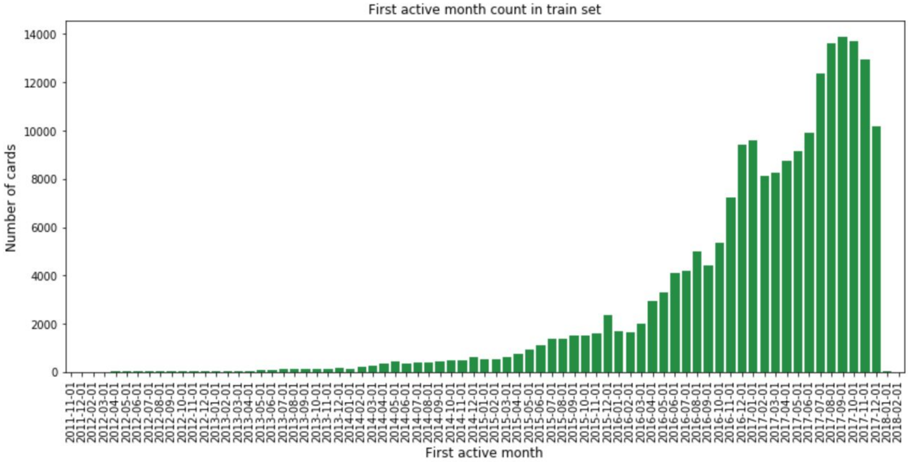
EDA / Data Wrangling

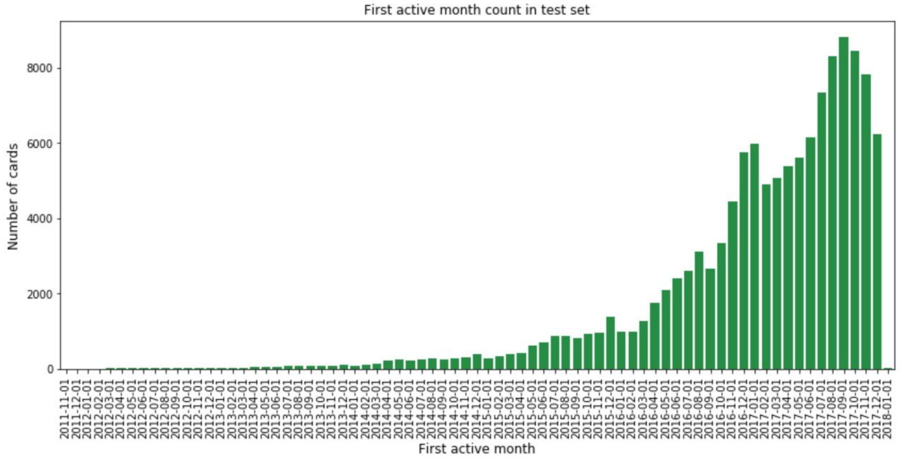
- Converted first active month to datetime
  - Created elapsed feature time from latest date in data
- Observed Loyalty Score range





First Active Year vs Loyalty Score





First active month difference between training and test data

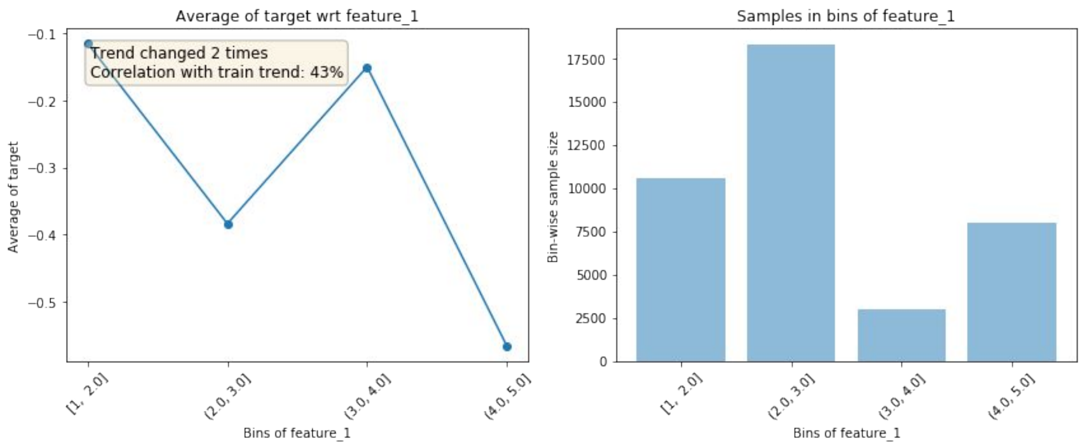
Feature Engineering

- Performed a large groupby on Hist. Transactions to create new features based off each each customer

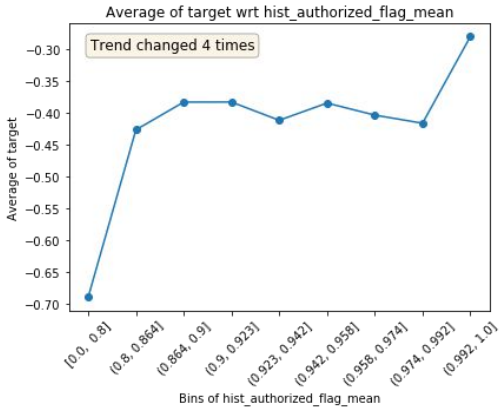
```
def aggregate_historical_transactions(history):

    history.loc[:, 'purchase_date'] = pd.DatetimeIndex(history['purchase_date']).\
                                      astype(np.int64) * 1e-9

    agg_func = {
        'authorized_flag': ['sum', 'mean'],
        'merchant_id': ['nunique'],
        'city_id': ['nunique'],
        'purchase_amount': ['sum', 'median', 'max', 'min', 'std'],
        'installments': ['sum', 'median', 'max', 'min', 'std'],
        'purchase_date': [np.ptp],
        'month_lag': ['min', 'max'],
        'category_1': ['sum', 'mean'],
        'category_2': ['sum', 'mean'],
        'category_3': ['sum', 'mean']

    }

    agg_history = history.groupby(['card_id']).agg(agg_func)
    agg_history.columns = ['hist_' + '_'.join(col).strip()
                           for col in agg_history.columns.values]
    agg_history.reset_index(inplace=True)

    df = (history.groupby('card_id')
          .size()
          .reset_index(name='hist_transactions_count'))

    agg_history = pd.merge(df, agg_history, on='card_id', how='left')

    return agg_history
```

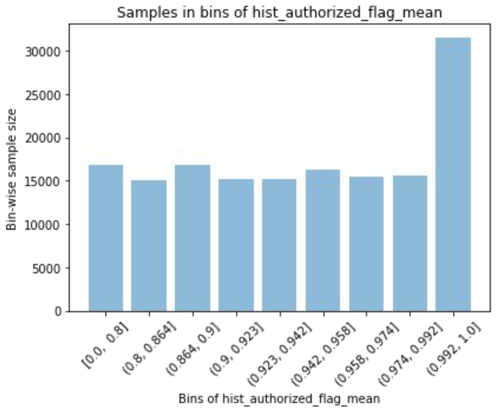
EDA

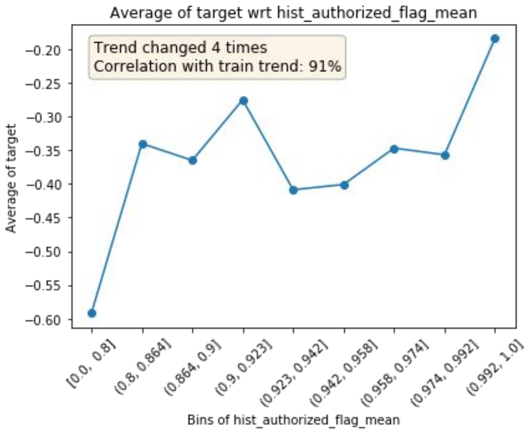
- Utilized library called featexp
- Bins all the feature and shows distribution
- Trend correlation against the target variable
- Compares trend on training vs validation set

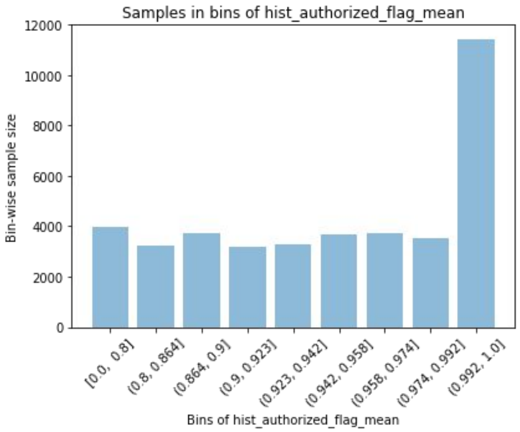


Hist Authorized flag Mean









EDA / Feature Selection

- Featexp module “stats” returns a pandas dataframe summarizing this info

```
stats = stats.sort_values(by='Trend_correlation', ascending=False)
stats
```

	Feature	Trend_changes	Trend_changes_test	Trend_correlation
2	feature_3	0	0	1.000000
3	year	0	0	1.000000
1	feature_2	1	1	0.996557
23	hist_month_lag_max	1	1	0.974305
8	hist_authorized_flag_mean	4	4	0.914882
22	hist_month_lag_min	1	3	0.882759
27	hist_category_2_mean	4	3	0.862556
28	hist_category_3_sum	3	4	0.860611

EDA / Feature Selection

- Variance Inflation Factor
  - Statistical method to identify features causing multicollinearity
- Reduced features with VIF of 5 or below

	VIF Factor	features
0	2.2	feature_3
1	4.3	feature_2
2	2.6	hist_month_lag_max
3	3.1	hist_category_2_mean
4	3.2	hist_merchant_id_nunique
5	2.3	hist_installments_sum
6	2.1	hist_installments_min
7	1.0	hist_purchase_amount_std

Model Prediction

- Extreme Gradient Boosting chosen to predict customer loyalty
- Metric for scoring: Root Mean Squared Error
- Iterated over many varying attempts to achieve best score
  - Created function to easily run model with different training features as input

```
def get_cv_rsme(Features, Target):

    params = {"objective":"reg:linear",'colsample_bytree': 0.3,'learning_rate': 0.1,
              'max_depth': 5, 'alpha': 10}

    data_dmatrix = xgb.DMatrix(data=Features,label=Target) #Improves the performance and efficiency of the model

    cv_results = xgb.cv(dtrain=data_dmatrix, params=params, nfold=3,
                        num_boost_round=70,early_stopping_rounds=20,metrics="rmse", as_pandas=True, seed=123)

    return(cv_results.tail(1))
```

Results

- **Attempt 1**
  - Limit the features to those with the top 15 trend correlation
  - Limit those remaining to those with low VIF scores
  - CV RMSE: 3.81
  - RMSE test: 3.91
- **Attempt 2**
  - Use all applicable features available
  - CV RMSE: 3.75
  - RSME test: N/A (Model overfit due to too many features)
- **Attempt 3**
  - Solely using features with VIF score of around 5 or below
  - CV RMSE : 3.80
  - RMSE Test : 3.91

Model Prediction

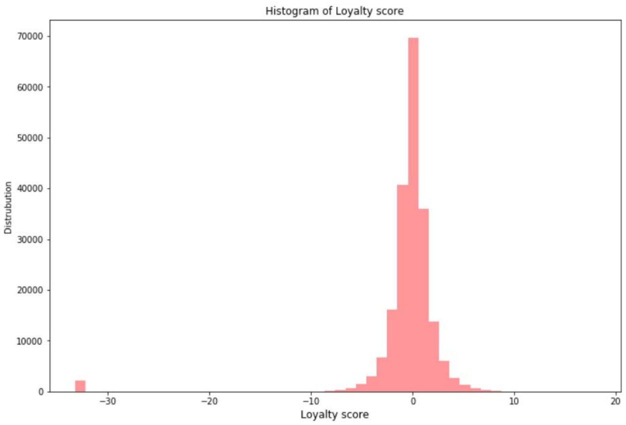
- **Attempt 4**
  - Scale the data
  - Apply PCA
  - Determine Intrinsic Value . Result: 4
  - CV RMSE: 3.81
  - RSME Test: 4.22
- **Attempt 5**
  - Simply tuning the model from attempt 2 by experimenting with the hyperparameters.
  - First attempt was using Gridsearchcv, but took too long.
  - Used trial and error instead
  - CV RMSE : 3.74
  - RMSE Test: N/A (Model overfit)

Conclusion

- RMSE acheived inapplicable to business case
  - Majority of scores between -1 and 1
- With such low accuracy, we cannot make reliable assumptions about customers to make improvements

```
train.target.round().value_counts().sort_values().nlargest(6)
```

```
 0.0    68531
-1.0    38864
 1.0    37614
-2.0    15652
 2.0    14648
-3.0     6685
```



Future Endeavors

- More relevant data needed to predict customer loyalty
    - Customer surveys might give new data that is more telling
- Wrong question / metric being sought by Elo
    - Their customer loyalty scoring currently not useful for customer segmentation
- Goal not achieved, but got to learn a lot :)