Capstone 2 - Milestone Report

28th August 2018

Problem and Client

Santander Group wants to provide their current customers and new customers services not found in other banking providers. The first step in achieving this level of service, Santander Group wants to **determine the amount or value of a customer's transaction.** This will enable them to personalize their services to current and new customers. Customers today are more likely to do business with a company if they provide personalized service; this applies to all sectors of business.

Data Wrangling and Exploratory Data Analysis (EDA)

The data for this project has been provided by the Santander Group in the form a Kaggle competition (https://www.kaggle.com/c/santander-value-prediction-challenge). Santander Group has provided 3 csv files; 1) a sample submission file, 2) a training set file and, 3) a test set file. The training and test files are an anonymized dataset containing numeric feature variables, the numeric target column, and a string ID column.

To understand the size of the train and test data sets the shape of the data frames was explored.

```
print("Train Shape : ", train_df.shape)
print("Test Shape : ", test_df.shape)
```

```
Train Shape :  (4459, 4993)
Test Shape :  (49342, 4992)
```

As shown above the test data set is roughly 10 times larger than the training set. The test set shows a very large number of rows and each set has over 4000 columns as well. Next, to understand the what data looks like we can review the first few rows of the training set.

```
train_df.head()
```

	ID	target	48df886f9	0deb4b6a8	34b15f335	a8cb14b00	2f0771a37	30347e683	d08d1fbe3	6ee66e115
0	000d6aaf2	38000000.0	0.0	0	0.0	0	0	0	0	0
1	000fbd867	600000.0	0.0	0	0.0	0	0	0	0	0
2	0027d6b71	10000000.0	0.0	0	0.0	0	0	0	0	0
3	0028cbf45	2000000.0	0.0	0	0.0	0	0	0	0	0
4	002a68644	14400000.0	0.0	0	0.0	0	0	0	0	0

Looking at the first few rows of the training set shows an ID column and feature columns with anonymized naming conventions. The target column is showing large values which would make sense if we are trying to determine a value for each ID or customer. It's also interesting to see a lot of zero values in the data. It appears that for each ID there could only be a few greater than zero values in the 4000 plus columns.

To get a better idea of the values in the feature columns we can look at some summary statistics.

```
train_df.describe()
```

	target	48df886f9	0deb4b6a8	34b15f335	a8cb14b00	2f0771a37	30347e683	d08d1fbe3
count	4.459000e+03	4.459000e+03	4.459000e+03	4.459000e+03	4.459000e+03	4.459000e+03	4.459000e+03	4.459000e+03
mean	5.944923e+06	1.465493e+04	1.390895e+03	2.672245e+04	4.530164e+03	2.640996e+04	3.070811e+04	1.686522e+04
std	8.234312e+06	3.893298e+05	6.428302e+04	5.699652e+05	2.359124e+05	1.514730e+06	5.770590e+05	7.512756e+05
min	3.000000e+04	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
25%	6.000000e+05	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
50%	2.260000e+06	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
75%	8.000000e+06	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
max	4.000000e+07	2.000000e+07	4.000000e+06	2.000000e+07	1.480000e+07	1.000000e+08	2.070800e+07	4.000000e+07

The summary statistics confirm an earlier assumption, the feature columns are made up of primarily zero values and a few large values. We also need to verify there no feature columns missing data or NA values. The following code checks if there are any missing values in the training set.

```
# missing values check
missing = train_df.isnull().sum(axis=0).reset_index()
missing.columns = ['column_name', 'missing_total']
missing = missing[missing['missing_total']>0]
missing = missing.sort_values(by='missing_total')
missing
```

	column_name	missing_total

The output table shows there are now missing values in the feature columns.

Now we need to determine if there are any feature columns with all zero values. If so, these columns will need to be removed from the training set because they do not provide any value to the model and would only increase the computation time. The code below shows how it determined if any feature columns were all zero values.

```
unique_df = train_df.nunique().reset_index()
unique_df.head()
```

	index	0
0	ID	4459
1	target	1413
2	48df886f9	32
3	0deb4b6a8	5
4	34b15f335	29

```
unique_df.columns = ['col_name', 'unique_count']
constant_col_df = unique_df[unique_df['unique_count']==1]
constant_col_df.shape
```

```
(256, 2)
```

To determine the columns with only zero values the unique numbers in each column were counted. Columns with 1 unique number are zero and those will be removed. 256 columns were identified with constant values. A quick sanity check was run looking at the sum of columns with unique values of 1 and 32.

```
constant_col_df.head()
```

	col_name	unique_count
28	d5308d8bc	1
35	c330f1a67	1
38	eeac16933	1
59	7df8788e8	1
70	5b91580ee	1

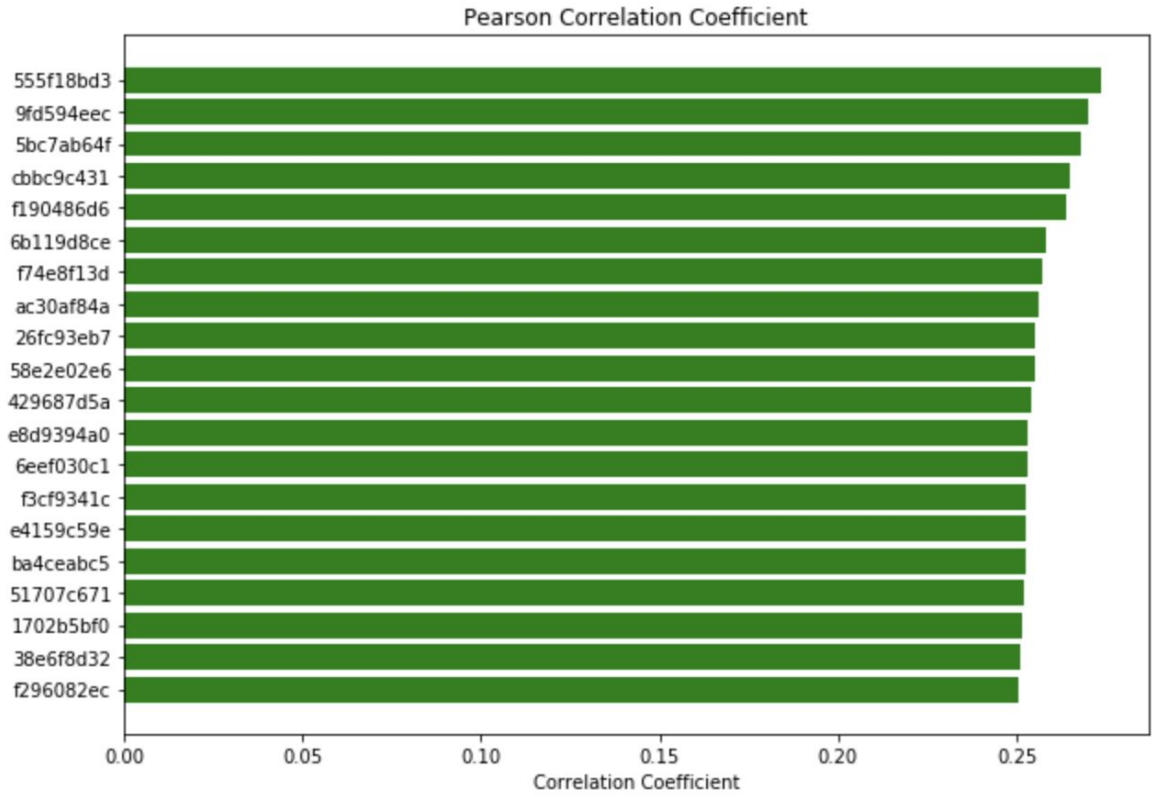
```
print("Column: d5308d8bc, with unique count 1 =", train_df['d5308d8bc'].sum())
print("Column: 5b91580ee, with unique count 1 =", train_df['5b91580ee'].sum())
print("Column: 48df886f9, with unique count 6 =", train_df['48df886f9'].sum())
```

```
Column: d5308d8bc, with unique count 1 = 0
Column: 5b91580ee, with unique count 1 = 0
Column: 48df886f9, with unique count 6 = 65346333.31999999
```

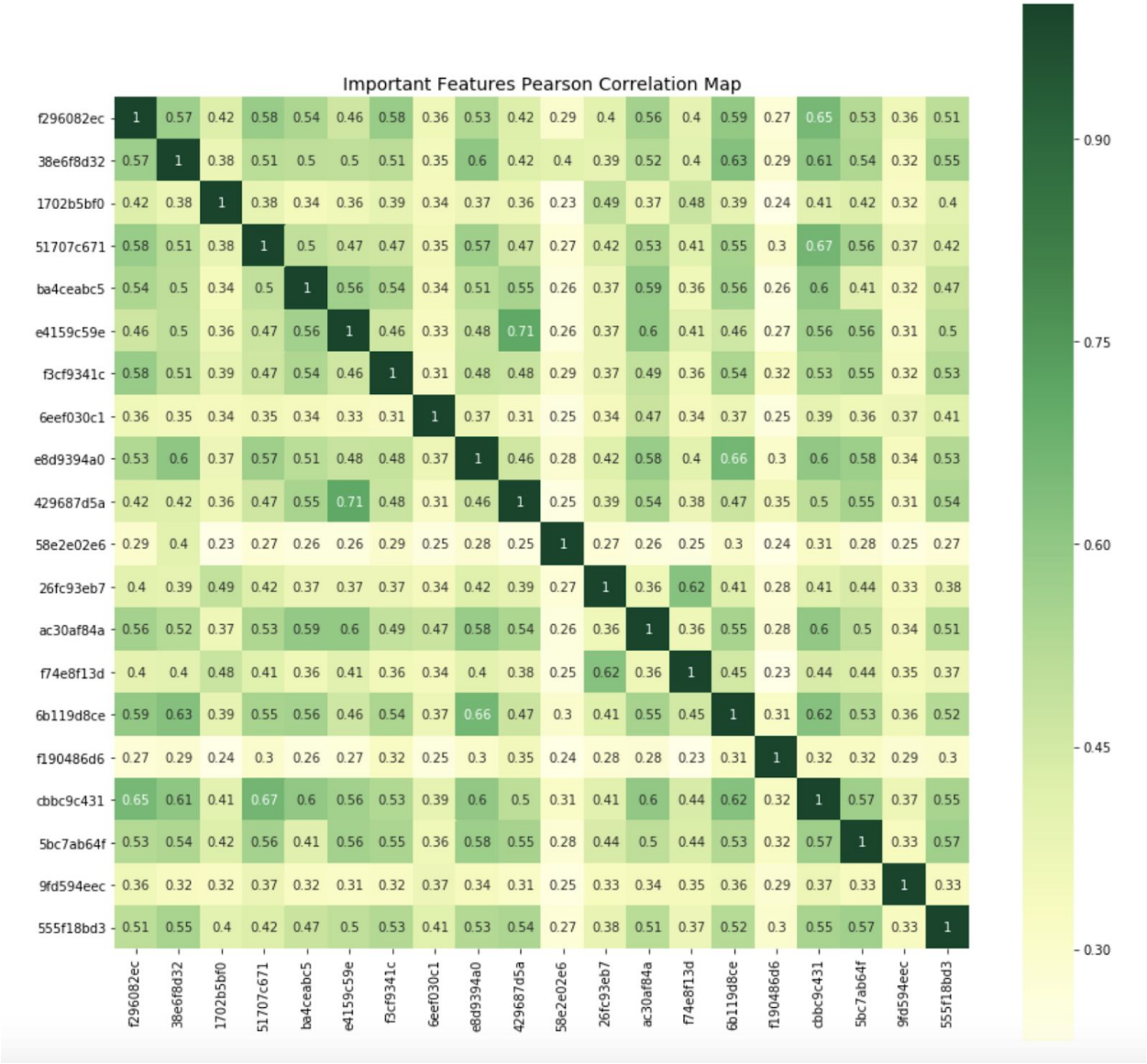
The 256 columns identified as having no values were removed from the training set and test set.

Feature Correlation

The data set has a large number of feature columns > 4000, we can use correlation to understand which features are correlated to the target. This will allow us to identify and use the highly correlated variables in our models. The Pearson correlation method will be used to determine the correlation coefficients. The first chart is a list of the correlation coefficients with an absolute correlation greater than 0.25; 20 feature columns were returned.



Next, we will use the 20 correlation coefficients to create a heat map to find importance.



The correlation map shows 2 features with Pearson correlation higher than 0.7 with each other. Not a lot of highly correlated features are being seen. We can also use feature importance from our machine learning models to verify our correlation is returning similar results.