Maddie Clubb, Danish Bokhari, Will Xue, Mohamed Elhussiny, Armando Mendez

Predicting the Timeliness of Imperial Marble and Granite Customer Payments

Section 1: Problem Statement and How the 5 Elements of Machine Learning Apply

The dataset we are going to use is from Imperial Marble and Granite, a countertop-installation business. The features we have available to us include: 'CONTRACTOR', 'CUSTOMER NAME', 'PO#', 'STONE COLOR', 'DATE INSTALLED', 'PLACE INSTALLED', 'SQFT', 'PROJECT COST', 'DEPOSIT', 'CHECK #', 'DATE CASH PAYMENT', 'FINAL PAYMENT CHECK#', 'DATE PENDING', 'PAYMENT DAYS OUTSTANDING', 'TOTAL PAID', and 'AFTER EXPENSES'. For this final project, we decided to build a model that predicts whether or not payment for a given job will be late – made more than 12 days after installation. In terms of the models we explored, we tested several classification algorithms and compared results to determine the best model for this problem. The algorithms we experimented with included SVM, Decision Tree, Random Forest, and Logistic Regression. The source of this data is the business itself since one of our groupmates is a relative of the business owner. The problem we are trying to solve is significant because family businesses need to be conscious of how much money they have available at a given time to put back into the business – thus, knowing when payments are likely to be late or on time is important in terms of the business' financial decisions. There has been no previous attempt to solve this problem.

The 5 elements of learning fit into this problem: beginning with the input, X, comprising of all of the scaled data for the features we selected ('CONTRACTOR', 'STONE COLOR', 'DATE INSTALLED', 'PLACE INSTALLED', 'SQFT', 'PROJECT COST', 'DEPOSIT'). The output, y, is the class label we are trying to predict – 0: if a customer will make a payment on time by paying before or within 12 days of installation, or 1 if after. The target function, F: X -> y, also known as the function that the model aims to find, is the unknown function that perfectly maps the feature data inputs to the actual label outputs; the data is all of the raw business data from 'Production2019-2021.csv'; the hypothesis, G: X -> y, is the function that our model selects as the best function from the hypothesis set, a set determined by our feature choices and depending on each of our various models, including SVM, Decision Tree, Random Forest, and Logistic Regression.

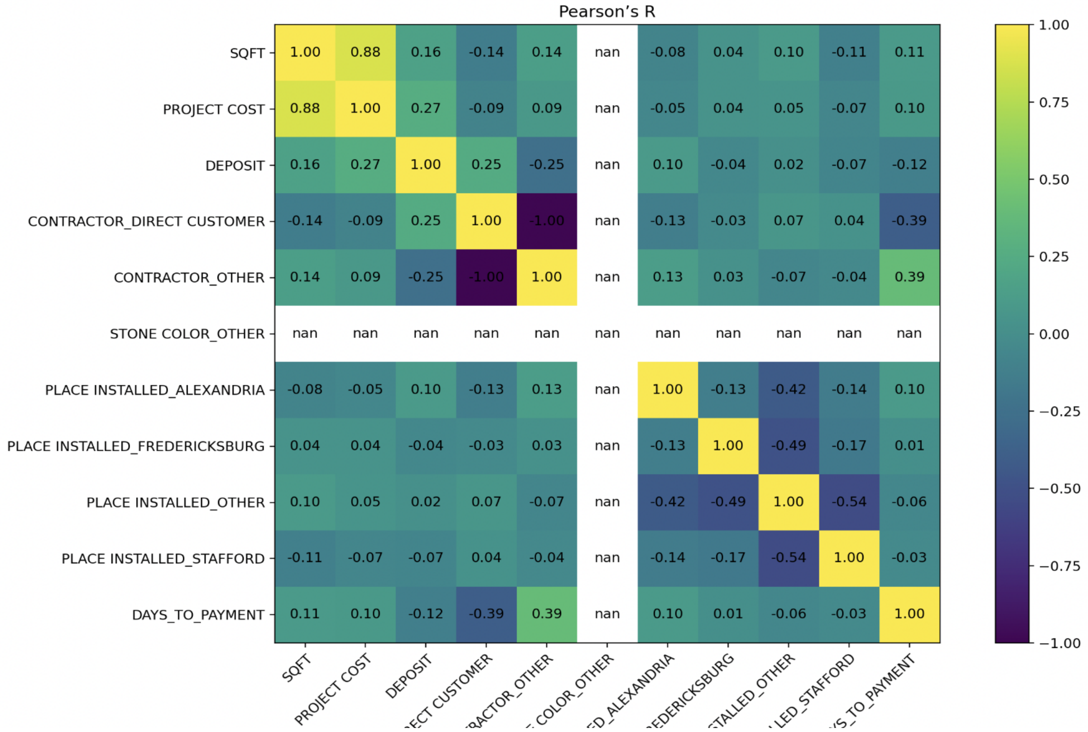
Section 2: Exploratory Data Analysis

Removed features with too many missing values

‘MATERIAL COST’, ‘SINK AND FAUCET COST’, and ‘CREDIT CARD PAYMENTS’ are just a few of the features that are missing the majority of their values.

Included features that logically made sense and/or had a relatively high absolute value according to our Pearson’s R Correlation Heatmap

The features we included were: ‘CONTRACTOR', 'STONE COLOR', 'DATE INSTALLED', 'PLACE INSTALLED', 'SQFT', 'PROJECT COST', 'DEPOSIT', 'PAYMENT DATE'.



Handling ‘DEPOSIT’ values

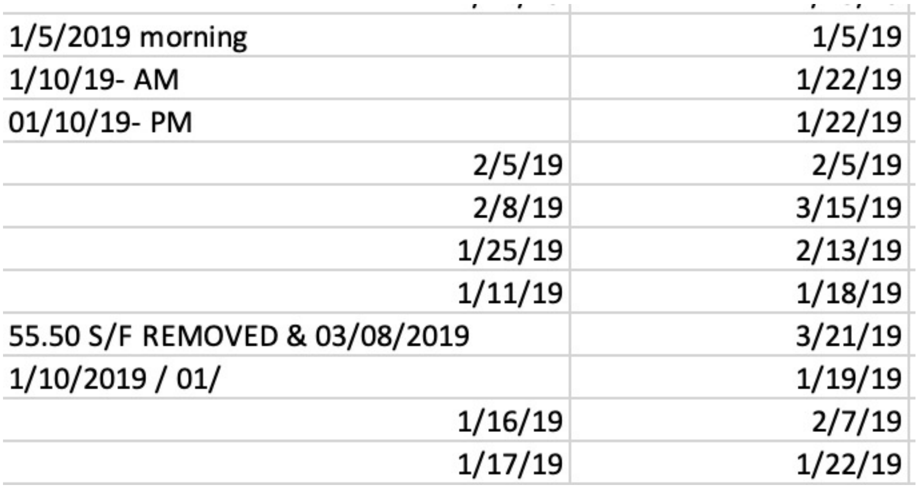
We imputed missing values in ‘DEPOSIT’ to be 0, and changed all other values to be 1, which makes ‘DEPOSIT’ a binary categorical variable instead of a continuous variable. We did this because we figured the actual value of the deposit to be correlated to ‘PROJECT COST’, while the act of putting down a deposit or not might be a better indicator of how long it would take to make the payment.

Dropped rows with null values

Only ~50 rows had null values. We decided to drop these null values from our dataset since they would have a negligible effect on predicting the target variable.

Fixed dates not in correct format

Some of the dates were not in standard format as shown below so we manually fixed them.

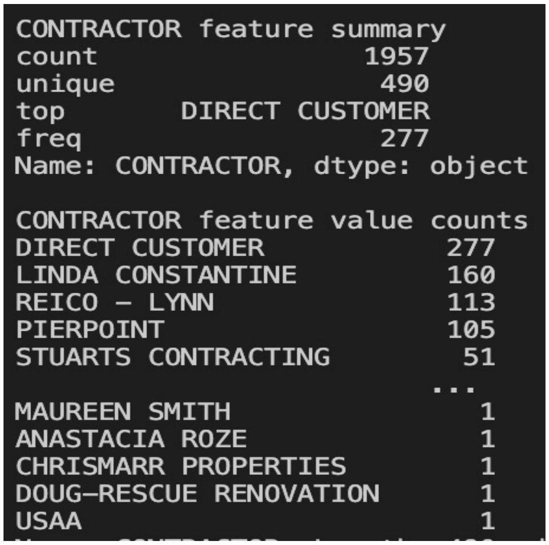


1/5/2019 morning	1/5/19
1/10/19- AM	1/22/19
01/10/19- PM	1/22/19
2/5/19	2/5/19
2/8/19	3/15/19
1/25/19	2/13/19
1/11/19	1/18/19
55.50 S/F REMOVED & 03/08/2019	3/21/19
1/10/2019 / 01/	1/19/19
1/16/19	2/7/19
1/17/19	1/22/19

Binned CONTRACTOR, STONE COLOR, and PLACE INSTALLED features

These categorical features had too many unique values, so we decided to bin the lowest occurrences of values into an 'OTHER' category.

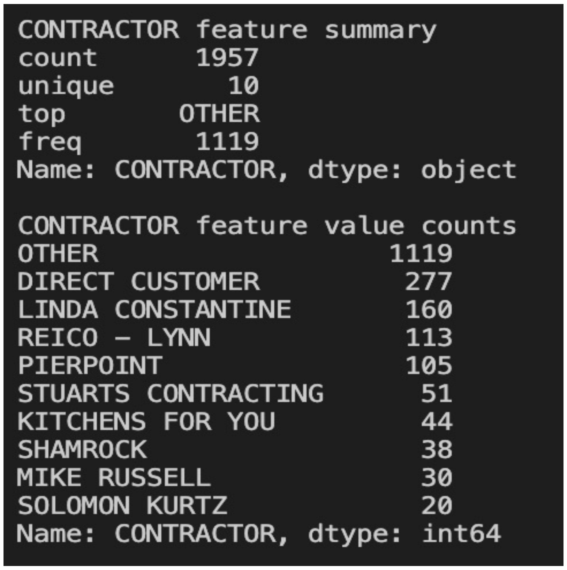
Pre-bin:



```
CONTRACTOR feature summary
count                  1957
unique                  490
top        DIRECT CUSTOMER
freq                    277
Name: CONTRACTOR, dtype: object

CONTRACTOR feature value counts
DIRECT CUSTOMER            277
LINDA CONSTANTINE          160
REICO - LYNN               113
PIERPOINT                  105
STUARTS CONTRACTING         51
                          ...
MAUREEN SMITH                1
ANASTACIA ROZE               1
CHRISMARR PROPERTIES         1
DOUG-RESCUE RENOVATION       1
USAA                         1
```

Post-bin:



```
CONTRACTOR feature summary
count       1957
unique        10
top        OTHER
freq        1119
Name: CONTRACTOR, dtype: object

CONTRACTOR feature value counts
OTHER                       1119
DIRECT CUSTOMER              277
LINDA CONSTANTINE            160
REICO - LYNN                 113
PIERPOINT                    105
STUARTS CONTRACTING           51
KITCHENS FOR YOU              44
SHAMROCK                      38
MIKE RUSSELL                  30
SOLOMON KURTZ                 20
Name: CONTRACTOR, dtype: int64
```

For pre-bin, there were 490 unique values in the CONTRACTOR feature. By changing the values with less than 50 count to ‘OTHER’, we reduced the number of unique values to 10, as can be seen in the post bin image above.

Engineering and Binning labels

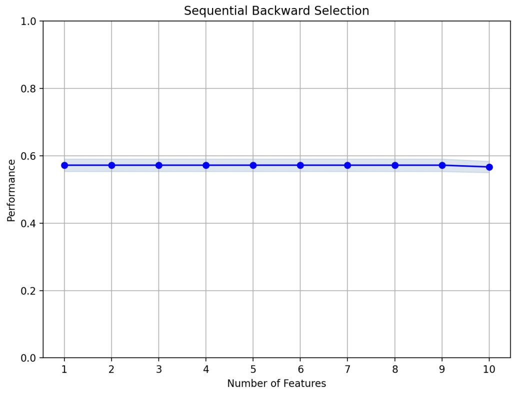
We feature engineered ‘DAYS_TO_PAYMENT’ by subtracting ‘DATE INSTALLED’ from ‘PAYMENT DATE’. We then binned ‘DAYS_TO_PAYMENT’ into two bins: payments before 12 days, and payments after 12 days. This is also our target variable, as mentioned in section 1.

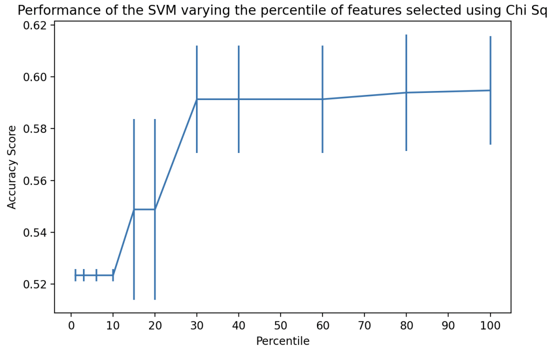
Outlier removal with Elliptic Envelope

We used Sklearn Elliptic Envelope to remove outliers. We experimented with the contamination value and found 0.1 to yield the highest accuracy.

Feature Selection: Chi Squared v. Sequential Backwards Selection

For feature selection, we utilized both sequential backwards feature selection and chi-squared feature selection. In both cases, accuracy did not pass 60%, however, the metrics differed at lower numbers of features used. This indicated the existence of features will less correlation.





We used a 70% train 30% test split on our data.

Scaling and Normalization

We used Sklearn’s MaxAbsScaler because it does not destroy sparsity between data.It does suffer from outliers, which is why we removed them with Elliptic Envelope.

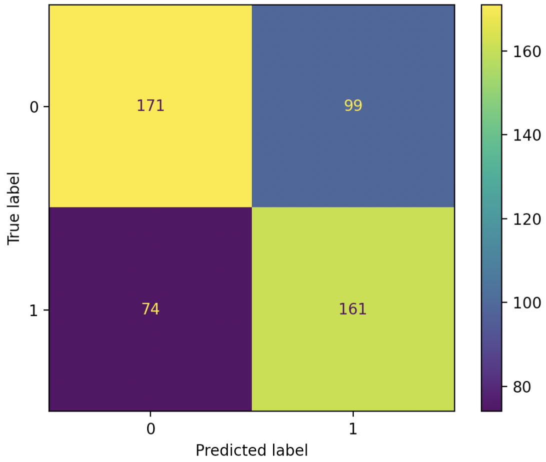
Section 3: Modeling and Results

Comparing different models:

We then created and tested out different models for classifying the payments. We used logistic regression, SVM, a decision tree, and a random forest. Here are the metrics of the different models and their confusion matrices:

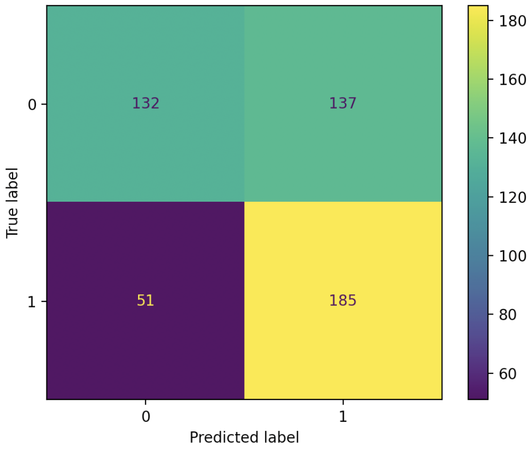
Logistic regression:

Accuracy: 0.65
Precision (weighted): 0.66
F1 (weighted): 0.65
Recall (weighted): 0.65



Decision Tree:

Accuracy: 0.62
Precision (weighted): 0.65
F1 (weighted): 0.62
Recall (weighted): 0.62



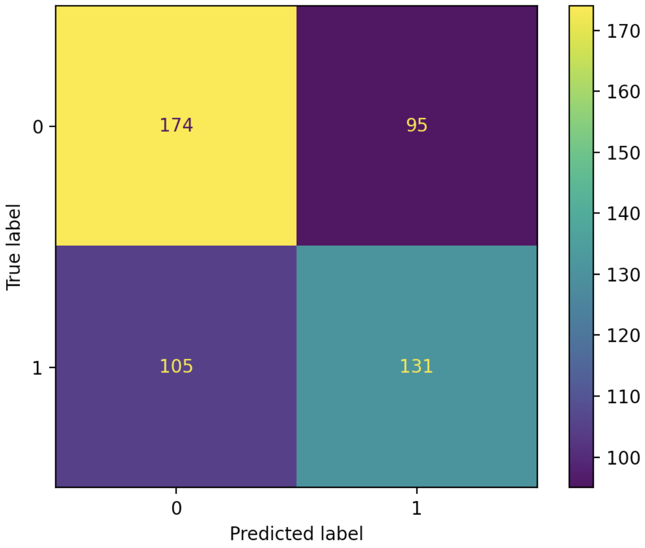
Random Forest:

Accuracy: 0.60

Precision (weighted): 0.60

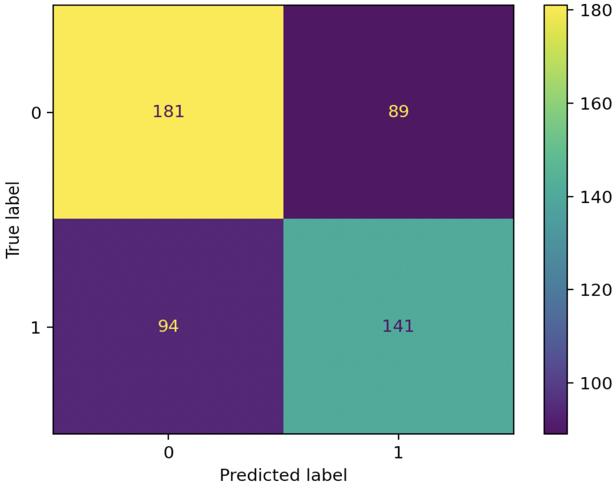
Recall (weighted): 0.60

F1 (weighted): 0.60



SVM:

Accuracy: 0.63
Precision (weighted): 0.63
F1 (weighted): 0.63
Recall (weighted): 0.63



Results:

Logistic regression performed the best across all metrics, with an accuracy, precision, recall, and F1 scores of 0.65. This shows that our model isn't a good predictor of timeliness of payment for Imperial Marble and Granite. We think the low metrics are likely due to the lack of data. We only had around 1800 data points, while we had 10 features, likely causing our model to suffer from the curse of dimensionality. However, using less features also reduced our metrics, which suggests that in addition to not having sufficient data, there might not be any relationship between our features to begin with. Our feature selection analysis supports this, as accuracy did not increase or decrease significantly from removing features.

Future steps:

As can be seen from our results above, the dataset that we had was not sufficient to classify payments as being late or not with more than about 65% accuracy. Although this is certainly better than random guessing, it is not exactly as high as we had initially hoped. Thus, if we were to revisit this problem in the future, we would hopefully have access to more data. Having additional features such as customer credit scores would have also been helpful as they are better predictors of timeliness of payments.