Assignment

Dependencies

Using Python 3.7

```
pip install ipython
pip install pandas
pip install matplotlib
pip install -U scikit-learn
pip install jupyter notebook
```

Import functions

In [1]:
```
from tools.helpers import read_json, pandas_keep_columns, retrieve_data, cr
from assignment import setup_workspace_retrieve_data, clean_data, select_hi
from assignment import annualized_rate_of_return, logistic_regression
from assignment import plot_hist_for_numeric_col, pie_chart, get_defaults
from assignment import clean_prior_to_regression
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
from sklearn.metrics import roc_auc_score
from sklearn.metrics import roc_curve
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import importlib
plt.rcParams['figure.figsize'] = [15, 5]
%matplotlib inline
```

In [2]:
```
%load_ext autoreload
%autoreload 2
```

In [3]:
```
from matplotlib import pyplot as plt
plt.style.use('ggplot')
```

Read Project Configuration file

This contains download link, file names and columns to use.

In [4]:
```
data = read_json('data/project_config.json')
steps = data['steps']
```

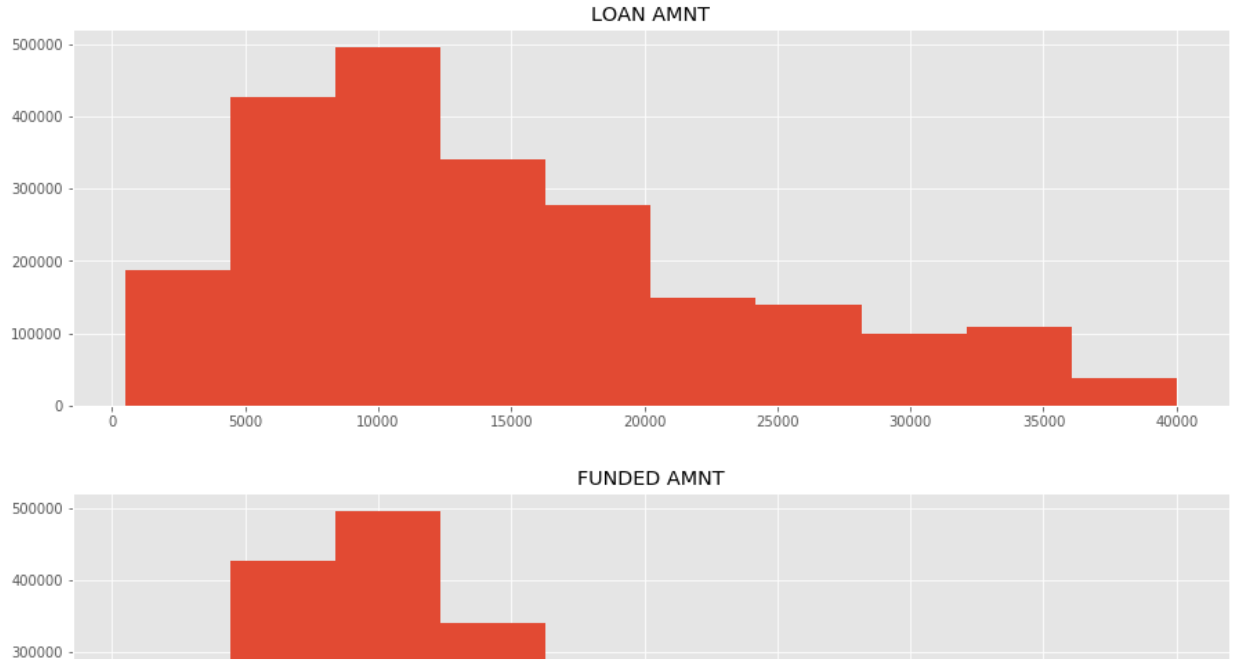
Part 1 Data Exploration and Evaluation

- Clean columns, carry forward `['loan_amnt', 'funded_amnt', 'term', 'int_rate', 'grade', 'annual_inc', 'issue_d', 'dti', 'revol_bal', 'total_pymnt', 'loan_status']`
- Perform any necessary cleaning and aggregations to explore and better understand the dataset.
  - Describe the `data.describe()`
- Describe any assumptions you made to handle null variables and outliers.
  - Remove outliers for Annual Income
  - Remove outliers for DTI
  - Total credit revolving balance (revol_bal)
- Describe the distributions of the features.
  - Include two data visualizations and
  - Two summary statistics to support these findings.

In [5]:
```
df = setup_workspace_retrieve_data(data)
```

```
downloading data.
unzipping data.
unzipping data complete.
```

In [6]:
```
plot_hist_for_numeric_col(df, df.columns, '01_before_filter')
```



In [7]:
```
dfc = clean_data(df)
```

```
creating value-added fields for data.
filtering fields for data.
pre-filter length: 2260668 post-filter length: 2213821 dif. of: 46847
dif. of: 2.07%
```

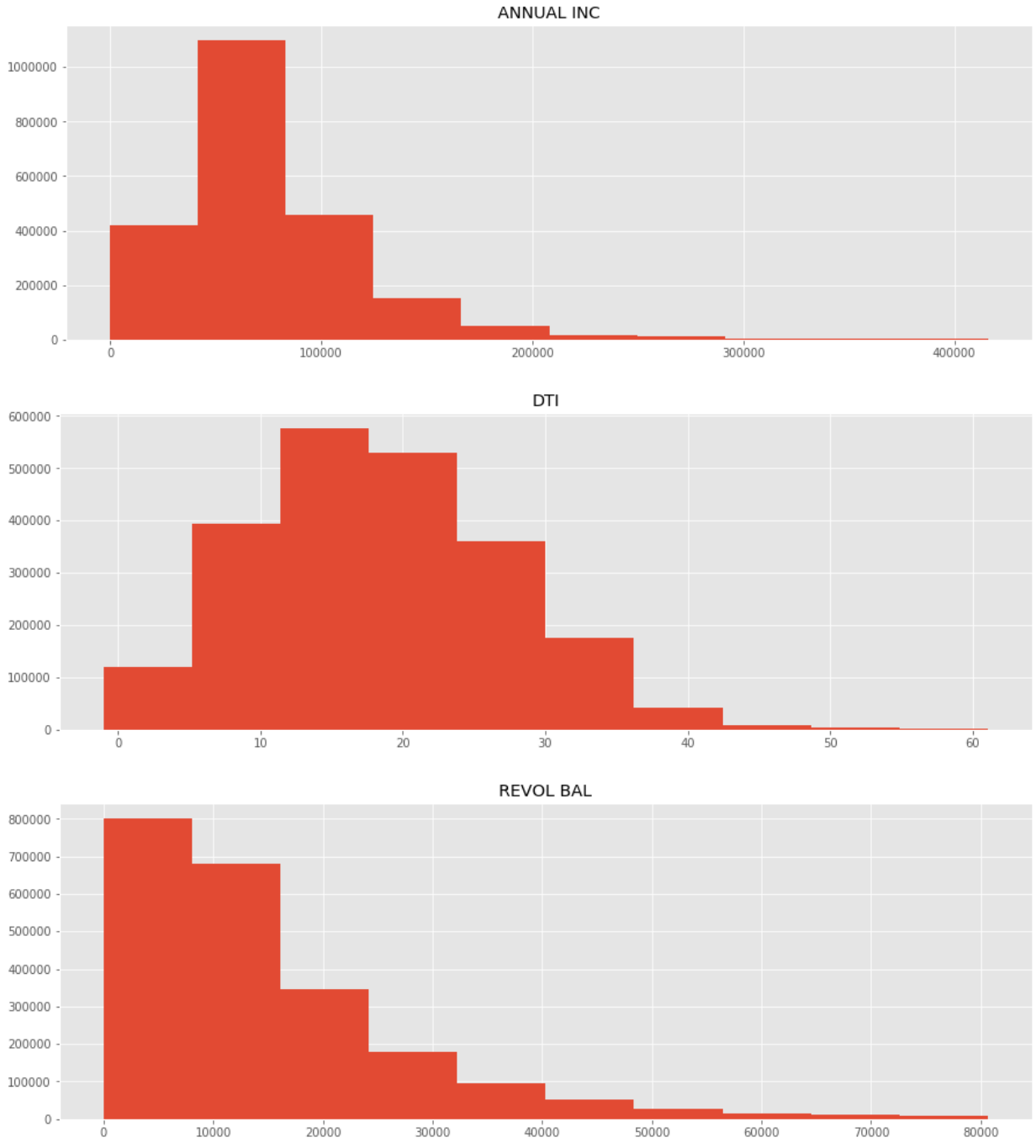
Outliers

Three columns have outliers in the distribution:

- **Annual Income**
- **DTI**
- **Total credit revolving balance**

Rows with >= 3 or <= -3 Z-score were removed. The following are the filtered histograms:

In [8]:
```
plot_hist_for_numeric_col(dfc, ['annual_inc', 'dti', 'revol_bal'], '02_post
```



Data descriptive statistics

In [9]:
```
pd.options.display.float_format = '{:,.2f}'.format

dfc.describe()
```

Out[9]:

	loan_amnt	funded_amnt	int_rate	annual_inc	dti	revol_bal	total_pymnt
count	2,213,821.00	2,213,821.00	2,213,821.00	2,213,821.00	2,213,821.00	2,213,821.00	2,213,821.00
mean	14,872.42	14,867.24	13.11	75,085.37	18.42	14,734.99	11,723.18
std	9,074.88	9,072.98	4.83	43,695.79	8.86	12,727.95	9,790.93
min	500.00	500.00	5.31	0.00	-1.00	0.00	0.00
25%	8,000.00	8,000.00	9.49	46,000.00	11.86	5,874.00	4,252.91
50%	12,500.00	12,500.00	12.62	65,000.00	17.77	11,125.00	9,008.95
75%	20,000.00	20,000.00	15.99	91,000.00	24.37	19,626.00	16,568.51
max	40,000.00	40,000.00	30.99	416,000.00	61.06	80,639.00	63,296.88

Interesting stats

In [24]:
```
mean_loan_amount = dfc.describe().loc['mean']['loan_amnt']
median_annual_income = dfc.describe().loc['50%']['annual_inc']

print(
    'Some interesting data points is the Mean Loan Amount of ${:,.0f}'.form
    'and that the Median Annual Income for loan borrowers is ${:,.0f}, abou
    'than the Census Bureau Median Household Income of $59,039 in 2016.',
)
```

```
Some interesting data points is the Mean Loan Amount of $14,872 and that
the Median Annual Income for loan borrowers is $65,000, about $6,000 more
than the Census Bureau Median Household Income of $59,039 in 2016.
```

Discussion

1. *Describe any assumptions you made to handle null variables and outliers.*

   - For outlier variables, I removed anything with a value of >= 3 or <= -3 Z-score. I didn't drop rows with null values.

2. *Describe the distributions of the features.*

   - Some of the variables have distributions that appear in the family of either Normal, Poisson, Exponential or Binomial. For those with clear outliers, they were filtered as mentioned above. Issue Month - parsed from `issue_d` though for example, appears more Uniform.

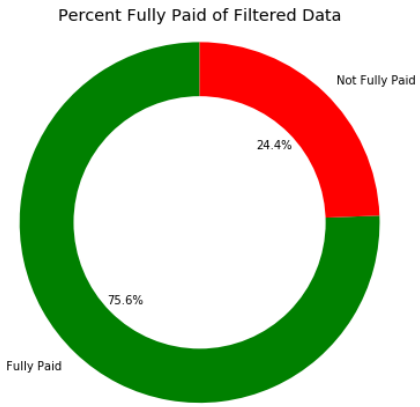
Part 2 Business Analysis

- Assume a 36 month investment period for each loan, and exclude loans with less than 36 months of data available.
- What percentage of loans has been fully paid?

In [10]:
```
dfs, pct_filtered_fully_paid = select_historic_data(dfc)
print('{:.2f}% of loans Fully Paid excluding < 36 months'.format(pct_filter
```

```
75.61% of loans Fully Paid excluding < 36 months
```

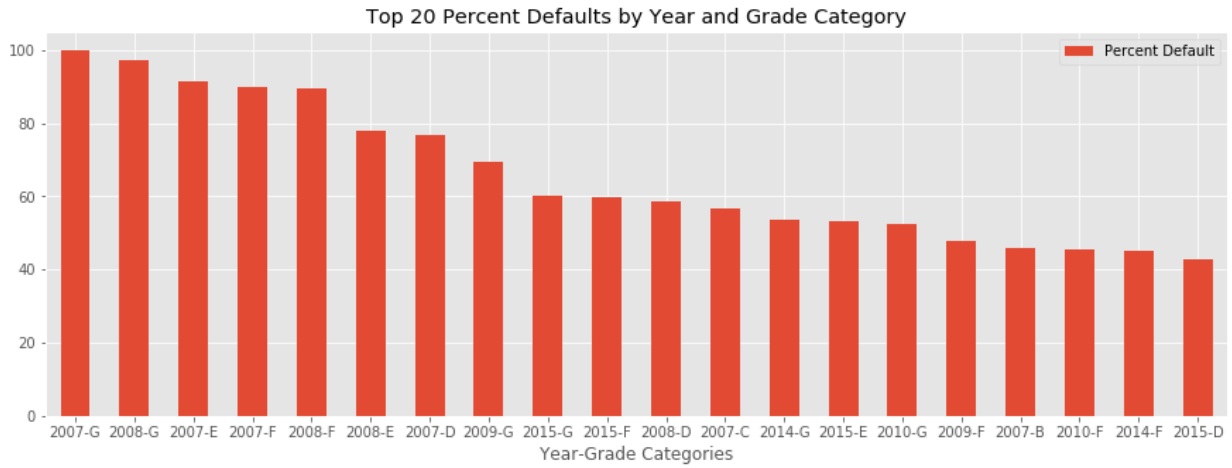
In [11]:
```
pie_chart(
    [pct_filtered_fully_paid * 100, 100 - (pct_filtered_fully_paid * 100)],
    ['Fully Paid', 'Not Fully Paid'],
    ['green', 'red'],
    'Percent Fully Paid of Filtered Data'
)
```



Of the Filtered data, by outliers and excluding records with less than 36 months it appears around 3/4 of the loans have Fully Paid

- ☑ When bucketed by year of origination and grade, which cohort has the highest rate of defaults? Here you may assume that any loan which was not fully paid had “defaulted”.

In [12]:
```
get_defaults(dfs)
```

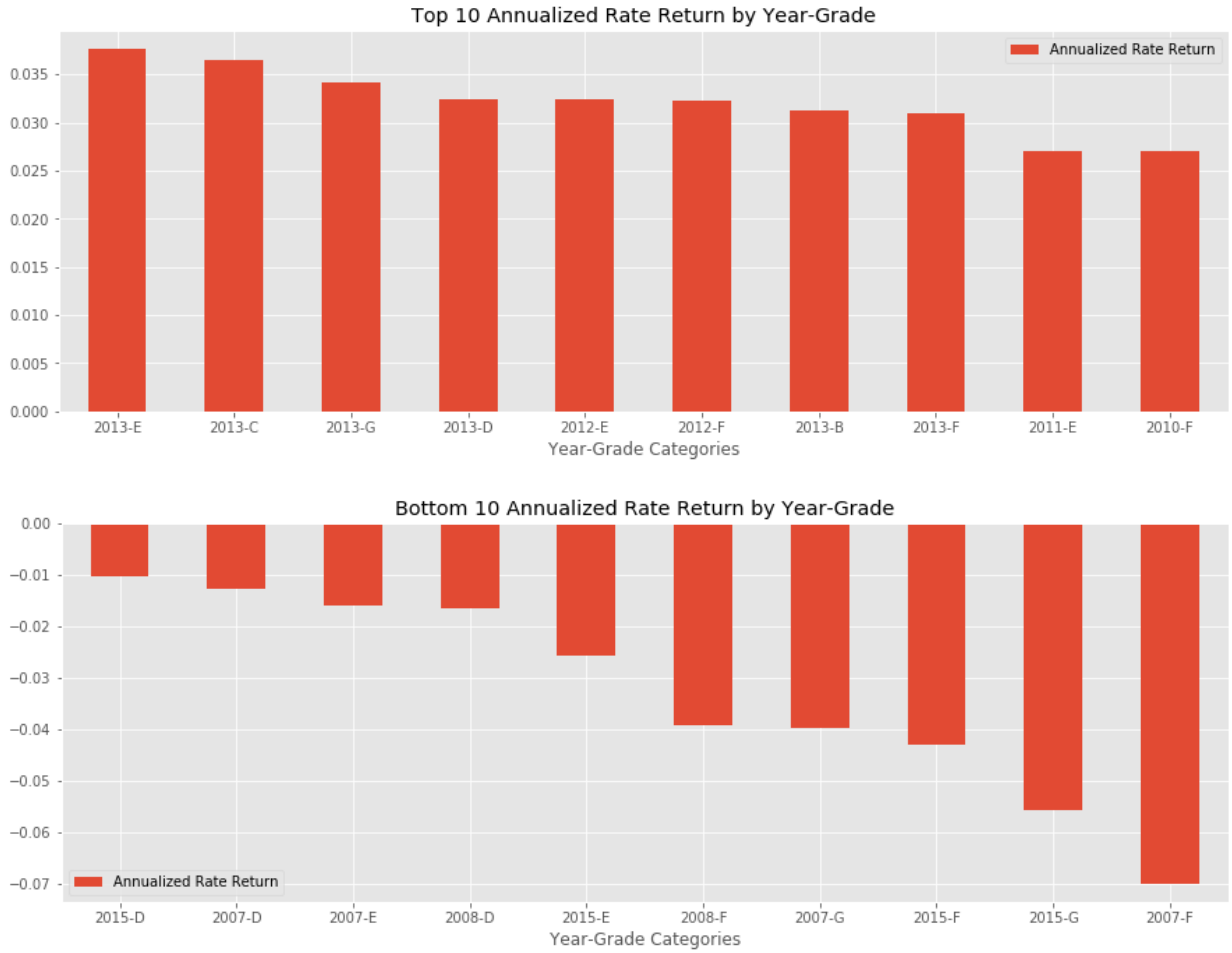


*For the above plot **Top 20 Percent Defaults by Year and Grade Category**, if I had some additional time I would like to check some of the categories close to 100% Defaults, as they seem worthy of further inquiry and sanity-checking.*

- When bucketed by year of origination and grade, what annualized rate of return have these loans generated on average?
  - For simplicity, use the following approximation: `Annualized rate of return = (total_pymnt / funded_amnt) ^ (1/3) - 1`

```
In [13]: rate_return_sort = annualized_rate_of_return(dfs)
```

```
creating annualized rate of return.
```



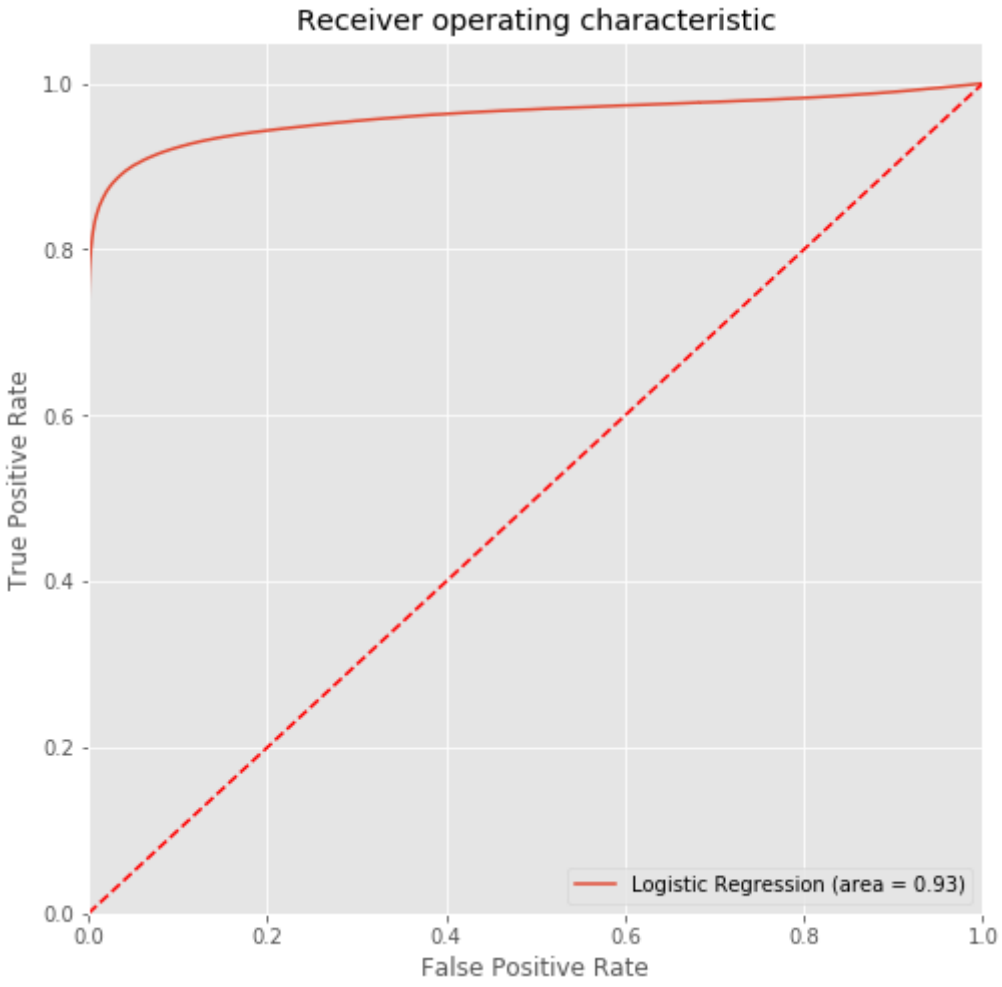
Discussion

Judging from the Defaults and Annualized Rate Return Year-Grade Categories, it appears that the higher (E, G, D, F) Grades and earlier years (2007, 2008) had more Defaults and lower Annualized Rate Returns. This is somewhat expected based on a priori understanding of lending practices around that time.

Part 3 Modeling

- Build a logistic regression model to predict loan defaults
- Assume that
  - (i) You are given the ability to invest in each loan independently
  - (ii) You invest immediately following loan origination and hold to maturity (36 months)
  - (iii) All loan fields that would be known upon origination are made available to you.
- Was the model effective?
  - ☑ Explain how you validated your model and describe how you measure the performance of the model.

In [14]:
```
model_score, confusion_matrix, classification_report = logistic_regression(
    clean_prior_to_regression(dfc)
)
```



Accuracy of logistic regression

In [15]:
```
print('Accuracy of logistic regression on test: {:.2f}%'.format(model_score
```

```
Accuracy of logistic regression on test: 92.36%
```

Confusion Matrix

In [16]:
```
print(confusion_matrix)
```

```
[[297412   9767]
 [ 40968 316000]]
```

Classification Report

In [17]:
```
print(classification_report)
```

```
              precision    recall  f1-score   support

           0       0.88      0.97      0.92    307179
           1       0.97      0.89      0.93    356968

    accuracy                           0.92    664147
   macro avg       0.92      0.93      0.92    664147
weighted avg       0.93      0.92      0.92    664147
```

Discussion

The **accuracy** of the model is **~92%**. The best way to further evaluate that performance would be to attempt to iterate on some feature-engineering, which is adding and removing and selecting different combinations of features to test if accuracy can be improved.

The model is **validated** by the **Confusion Matrix** `confusion_matrix(y_test, y_pred)` . The upper left and lower right values are predictions True Positve and True Negative, respectfully. Upper right and lower left values are False Positive and False Negative respectfully.

The area under the curve is **~0.93**.

> AUC represents degree or measure of separability. It tells how much model is capable of distinguishing between classes. Higher the AUC, better the model is at predicting 0s as 0s and 1s as 1s. https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5 (https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5)

And finally our weighted average for recall, precision and f1-score are **~0.92/~0.93**. Similarly, perhaps some iterating on different combinations of features could yield improvement.

> f1-score gives you the harmonic mean of precision and recall. https://stats.stackexchange.com/questions/117654/what-does-the-numbers-in-the-classification-report-of-sklearn-mean (https://stats.stackexchange.com/questions/117654/what-does-the-numbers-in-the-classification-report-of-sklearn-mean)

Closing points

Ultimately, while ~0.92 is close to 1, it still leaves some uncertainty. So its important when evaluating these types of models, are you trying to predict low-risk things like recommendations to give to shoppers or if it is high-risk or high-value classification, where are there lives at stake or financial security, it is important to be conservative to avoid undue risk in applying a models' predictions. An example where this model could fail if not continuously updated is if the Grade

category assignment changes over time. Categorical variables may change if the application changes or some policy changes in categorization methodology. Machine learning is a helpful tool but must be used with the utmost care.

Code and Information Resource Links

- https://towardsdatascience.com/building-a-logistic-regression-in-python-step-by-step-becd4d56c9c8 (https://towardsdatascience.com/building-a-logistic-regression-in-python-step-by-step-becd4d56c9c8)
- https://medium.com/@kvnamipara/a-better-visualisation-of-pie-charts-by-matplotlib-935b7667d77f (https://medium.com/@kvnamipara/a-better-visualisation-of-pie-charts-by-matplotlib-935b7667d77f)
- https://stackoverflow.com/questions/37487830/how-to-find-probability-distribution-and-parameters-for-real-data-python-3 (https://stackoverflow.com/questions/37487830/how-to-find-probability-distribution-and-parameters-for-real-data-python-3)
- https://stackoverflow.com/questions/30746460/how-to-interpret-scikits-learn-confusion-matrix-and-classification-report (https://stackoverflow.com/questions/30746460/how-to-interpret-scikits-learn-confusion-matrix-and-classification-report)
- https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5 (https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5)
- https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62 (https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62)
- https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html (https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html)
- https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5 (https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5)
- https://stats.stackexchange.com/questions/117654/what-does-the-numbers-in-the-classification-report-of-sklearn-mean (https://stats.stackexchange.com/questions/117654/what-does-the-numbers-in-the-classification-report-of-sklearn-mean)
- https://www.scikit-yb.org/en/latest/api/classifier/classification_report.html (https://www.scikit-yb.org/en/latest/api/classifier/classification_report.html)
- https://stackoverflow.com/questions/30746460/how-to-interpret-scikits-learn-confusion-matrix-and-classification-report (https://stackoverflow.com/questions/30746460/how-to-interpret-scikits-learn-confusion-matrix-and-classification-report)

In [ ]: