### Data Cleaning

[Lending Club Predicting Loan Result (Data Cleaning).ipynb](https://colab.research.google.com/drive/1WeP_TIA7EDejAOsO2Dx9u3A-i62z0DF9#scrollTo=KrH3s9OOmPEf) - my colab book of a DQ project

## Goals

\* become familiar with the columns in the dataset

\* select the target column

\* decide on the type of model

\* remove many columns that aren't useful for modeling

#**SIDE NOTE**: *feature selection process notes* [here](https://docs.google.com/document/d/1Mz-MnySe9beaAkryPsiWb5TuRidj1mQczc6ri8IC4AM/edit#heading=h.hz9ndh1doipb)

look for any features that:

\* disclose information from the future (after the loan has already been funded)

don't affect a borrower's ability to pay back a loan (e.g. a randomly generated ID value by Lending Club)

\* need to be cleaned up and are formatted poorly

\* require more data or a lot of processing to turn into a useful feature

\* contain redundant information

# remove useless columns

## Feature Categories for Target Column

Since we're trying to predict whether a loan is either Fully Paid or Not, it should be a binary classification model. Thus all other 'loan\_status' categories (other than 'Fully Paid' & 'Charged Off') are irrelevant to the model. Those other categories pertain to the ongoing nature of the loan.

# remove useless categories out of target col

# this distils model into a binary classifier

# convert the loan categorical classifications to numerical categories

# Removing Single Value Columns

# columns with 0 variance convey no information the model

### feature prep

Lending Club Predicting Loan Result (feature prep).ipynb

src: <https://colab.research.google.com/drive/1_Rw_UMNfwbwumRcZv3BfmYWrV3NNcMPl#scrollTo=MewrrZ5sW5_->

## Goal: Prepare our feature columns for the model

\* handle missing values

\* convert categorical columns to numerical values

\* drop any extraneous columns throughout

# drop columns with high null values

# drop rows with other null values

# drop columns with low variance (columns almost all 1 value)

# resolve columns with overlapping values

# clean 'object'/string columns & convert them to numeric values (for model)

# convert categorical columns into numeric values (for model)

### model training

Lending Club Predicting Loan Result (model training).ipynb

src: <https://colab.research.google.com/drive/18nVyAEbvw4SyKgHmxYoJ8irsuJAonE_B#scrollTo=bIzkq0PeWNRm>

## Goal

Train a model that can predict whether or not a loan will be paid off on time.

## Selecting an Error Metric

Since false-positives (loans given out that are never paid back) are so costly (100% of the loan value) compared to the return on paid back loans(~+10% loan value), we should select a metric that minimizes false-positives.

\*\*Optimize\*\* for:

\* low [fall-out](<https://en.wikipedia.org/wiki/Information_retrieval#Fall-out>) (false positive rate)

\* high [recall](<https://en.wikipedia.org/wiki/Precision_and_recall#Recall>) (true positive rate)

\*\*false\_positive\_rate\*\* = false\_positive\_cnt / total\_true\_negative\_cnt

\*\*true\_positive\_rate\*\* = true\_positive\_cnt / total\_true\_positive\_cnt

\*NOTE:\*

\* we want to minimize the false\_positive\_rate (fpr)

\* we want to maximize the true\_positive\_rate (tpr)

## Training a model on skewed data

With our target column being so heavily biased toward positive predictions (loans payed back), a model will likely overfit to predicting all 1's

# separate target\_col from feature dataframe

# testing a simple Logistic Regression model without a train-test split

# metrics true\_positive\_rate & false\_positive\_rate

# testing a simple Logistic Regression model with cross\_validation

# metrics true\_positive\_rate & false\_positive\_rate

## Correcting for a skewed dataset

### Use oversampling and undersampling

\* ensure that the classifier gets input that has a balanced number of each class

\* Throw out many rows of data to match the lower skewed amount

\* copy the minority classification to match the majority

\*NOTE:\* this effectively weights the minority more

\* generate fake data. generate variations of minority set

### penalize misclassifications of the less prevalent class more

\* weigh the lower skewed category,

\* aka set model parameter class\_weight='balanced'

\* [more info on sklearn's class\_weight='balanced'](<http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html#sklearn-linear-model-logisticregression>)

\* manually set 'class\_weight' to problem specific penalties (penalty\_dict)

# model class\_weight parameter to balanced

## adjusting classification penalties to match their monetary reward/loss

The disparity in profit for a models loan prediction is due to the large(100%) monetary loss of false positives vs the gain of successful loans(10%).

This should drive the model's weights more than the skewed category of defaulted loans.

# try training a model on a problem specific penalty dict

## Ideas for improvement

\* tweak the penalties further.

\* try models other than a random forest and logistic regression.

\* use some of the columns we discarded to generate better features.

\* ensemble multiple models to get more accurate predictions.

\* tune the parameters of the algorithm to achieve higher performance.