**Performance of Multi-Class Classification vs Individual Binary Classification Models**

**Data From:** Lending Club from Kaggle

**Objective:** Exploring into the Lending Club loan data to figure out which algorithm would best suite a multi-class problem to identify defaulters and how much of the principal amount would they repay.

**Methodology:**

* All the variables would be categorized and used based on their importance derived
* Three different classes would be used for the multiclass problem
* Different modelling approach would be looked at to come up with the best fit for a multi-class approach

**Scope:**

* The data used for the project is limited to the lending Club dataset in Kaggle (<https://www.kaggle.com/wendykan/lending-club-loan-data>)

**Data used for the Project**

LendingClub data from kaggle

* Contains complete loan data for all loans issued through the 2007-2018
  + 2.2 million observations and 133 variables
* Active loans removed from data
  + 1.3 million observations and 133 variables
* Variable types:
  + Profiling: Address, Employment Length, Zip Code
  + Bureau: Utilization, Inquiry in past 12 months, Satisfactory Accounts
  + Loan: Interest Rate, Funded Amount

**Running on**: COLAB Notebooks

**Libraries & Frameworks**:

|  |  |  |
| --- | --- | --- |
| * NumPy | * PySpark | * SkLearn |
| * Pandas | * Matplotlib | * Seaborn |

|  |  |  |
| --- | --- | --- |
| **Splits** | Event Rates | |
| Class 1 | Class 2 |
| Out of Time | 12.2% | 14.2% |
| Development | 10.4% | 10.6% |
| Out of Sample | 10.3% | 10.6% |

Cleaning & ETL (Extract, Transform and Load Data): NumPy, Pandas, PySpark.

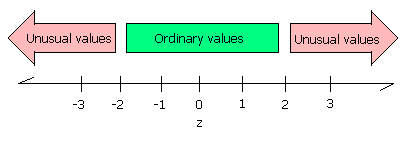
**Methodology Used:**

Data Exploration

* Basic Statistics for All Variables: Min, Max, Percentiles at 25 50 75, stddev
* Plotting variables for Categorical Variables: Analyzing class imbalance
* Analyzing data distribution for Numerical Values: Measure Skewedness of distribution

**Data Pre-Processing:**

* Fill Rate:
* Handling Missing Value:
* OneHotEncoding for Categorical Variables
* Outlier Removal:
  + z-score:



* + DBSCAN (Density Based Spatial Clustering with Application of Noise):

**Variable Creation:** Creating some dependent variables to facilitate class creation.

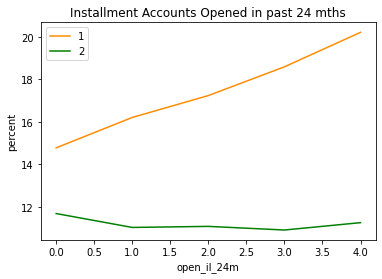
|  |  |  |
| --- | --- | --- |
| **Target** | **Loan Status** | **Amount Repaid** |
| 0 | No Default | 100% |
| 1 | Default | <25% |
| 2 | Default | >25% |

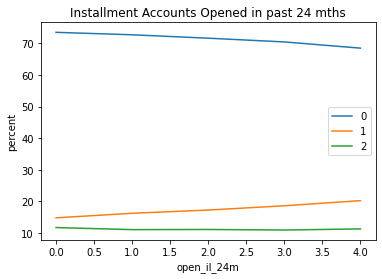
* **Target**:

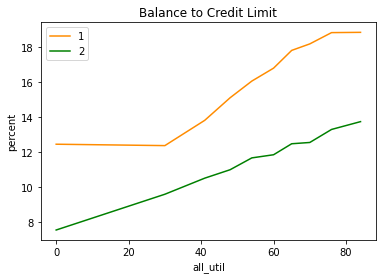
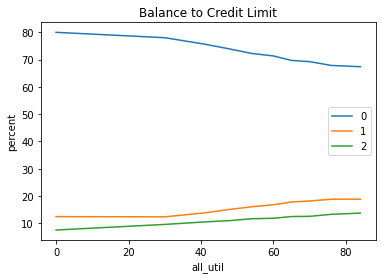
**Variable Selection:**

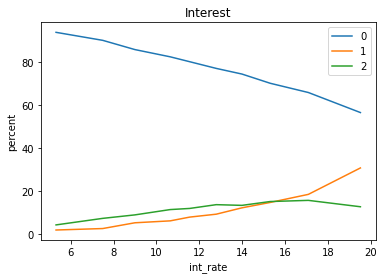
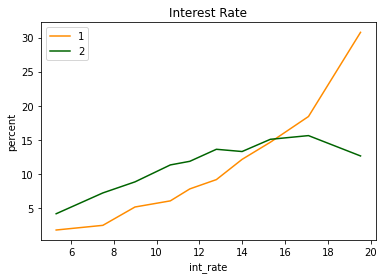
* Gini:
* Information Value (IV):
  + Cut-off of 2%
  + Low IV variables kept seeing business importance
* Pair-wise Correlation: Of highly correlated variables one with higher IV kept

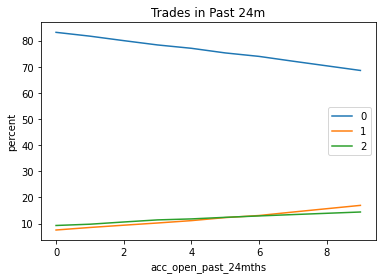
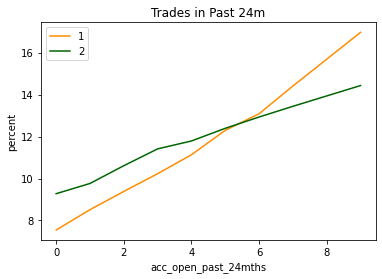
**Data Selection with Event Rates:**

**Ranking Plots:**



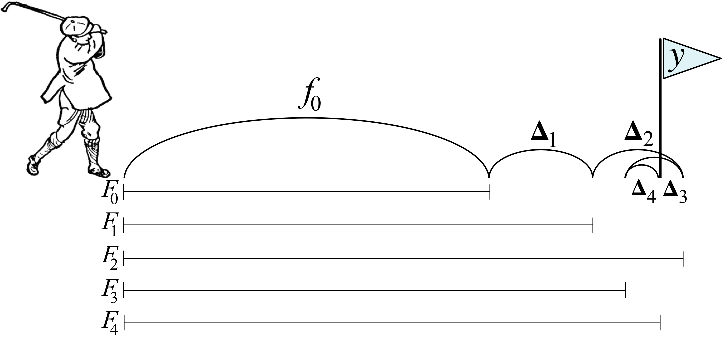






**Algorithms:**

XGBoost

* XGBoost is an implementation of gradient boosted decision trees designed for speed and performance.
  + combines the predictive power of multiple learners.
* Regularization:
* Handling Sparse Data
* Parallel Processing

Variants:

* Simple XGBoost
* RFECV + XGBoost:
  + RFECV

Optimizers:

* GridSearchCV:
* RandomSearchCV:
* Bayesian-Optimization:
* Hyperopt:

Hyperparameters used:

* n\_estimators:
* max\_depth
* alpha
* lambda
* scale\_pos\_weight
* num\_round

Neural Networks

Optimizers:

* Keras:
  + SGD:
  + Adagrad:
  + SGD-Nesterov:
  + Adam:
* Activation Functions: ReLU, Sigmoid, Softmax, Hyperbolic Tangent
* Hyperparameters: Nodes in Layers, Optimizers, Activation functions, Learning Rate
* Bayesian Optimization: Global Optimization

Key Recommendations

* Improvement in performance by applying DBSCAN.
* Bayesian Optimization has better performance than hyperopt, GridSearchCV, RandomSearchCV.
* Bayesian Optimization leads to stable models.
* RFECV leads to simpler models.
* Go with XGBoost with Bayesian Optimization.
* XGBoost because at Amex already familiar with using it.
  + Feature Importance is readily available.
* Neural Nets is black box.
  + Inside working is unknown.

