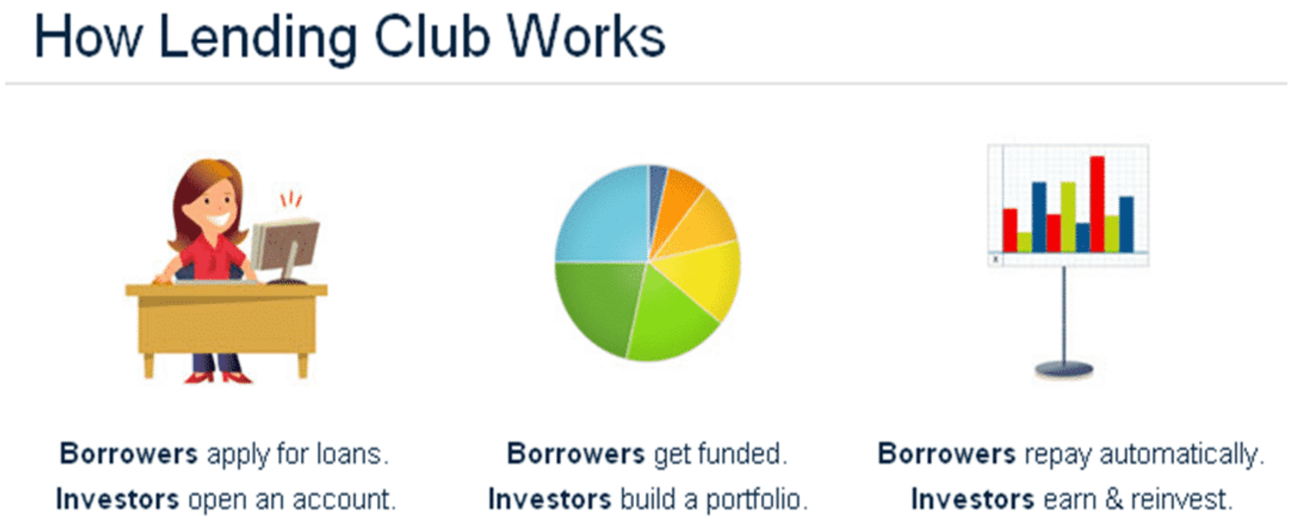
Lending Club Loan Defaulter Prediction

Parag Bhingarkar

Introduction

Lending Club is a peer to peer lending company based in the United States, in which investors provide funds for potential borrowers and investors earn a profit depending on the risk they take. Lending Club provides the "bridge" between investors and borrowers.



How Lending Club Works

Borrowers apply for loans.
Investors open an account.

Borrowers get funded.
Investors build a portfolio.

Borrowers repay automatically.
Investors earn & reinvest.

Objective

The aim of this project is to build a predictive machine learning tool to classify the loan defaulters and non-defaulters based on their profile.

Utilize parallel machine learning concepts to speed up the model development time and optimize the results

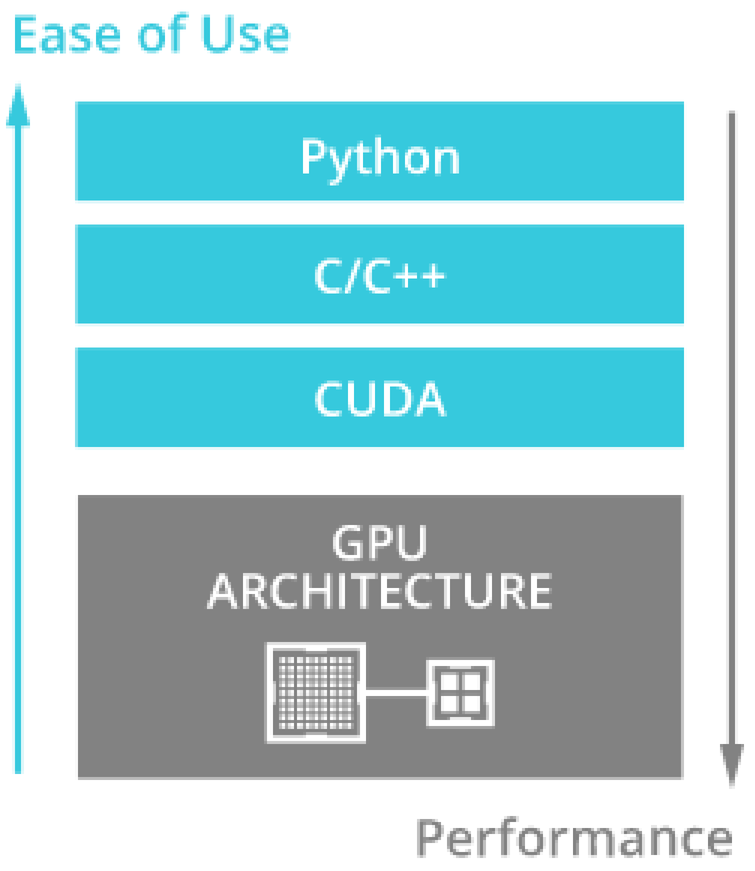
Hardware Specifications

CPU

CPU(s): 4
Thread(s) per core: 2
Core(s) per socket: 2
CPU family: 6
Model name: Intel(R) Xeon(R) CPU @ 2.20GHz

GPU

Tesla P100-PCIE : 1
Memory : 16280MiB
Pascal Architecture



RAPIDS flow

Introduction : Rapids.ai

- RAPIDS utilizes **NVIDIA CUDA®** primitives for low-level compute optimization, and exposes GPU parallelism and high-bandwidth memory speed through user-friendly Python interfaces.
- **Libraries and APIs Overview**
  - **cuDF** — pandas-like dataframe manipulation library
  - **cuML** — collection of ML libraries that will provide GPU versions of algorithms available in scikit-learn
  - **array_interface - Integration with deep learning libraries**

..About Dataset

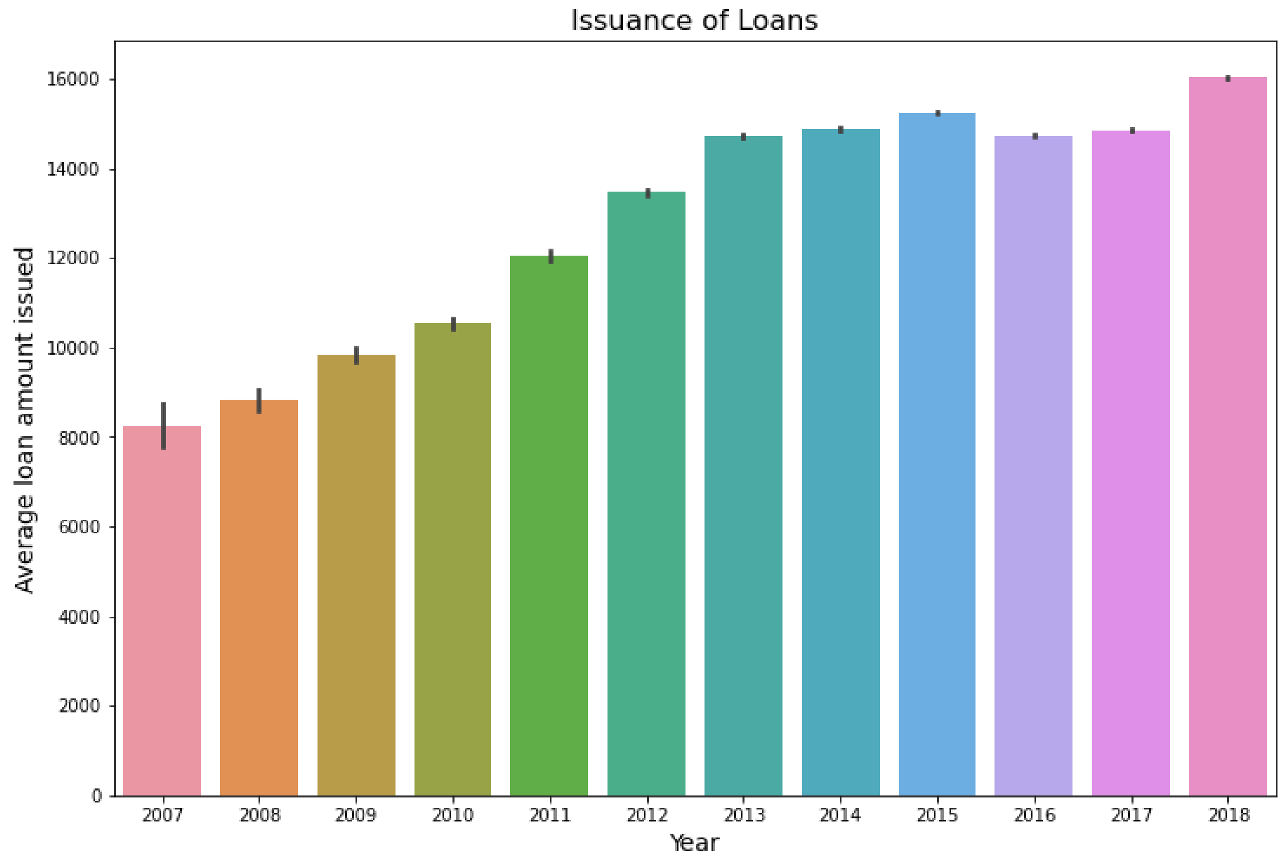
- The dataset contains information about all loans issued through 2007-2018
- Multiple categories to differentiate loan status
- Additional borrower information
  - credit scores
  - Total payment
  - Interest rate
- There are total 152 independent features and 2 million plus entires
- The dataset is imbalanced
  - 80% of total number of loans are paid

Data Preprocessing:

- Nvidia Rapids optimized the data cleaning and preprocessing stage
  - Cuda dataframe loaded data in <5 seconds while pandas took >45 seconds to load
- Multiple features present in the dataset had nulls and missing values
- Data balancing
- Filling up the null values
- Converting dates from improper format to usable features
- Label encoding
- Dummy variables

Exploratory Data Analysis

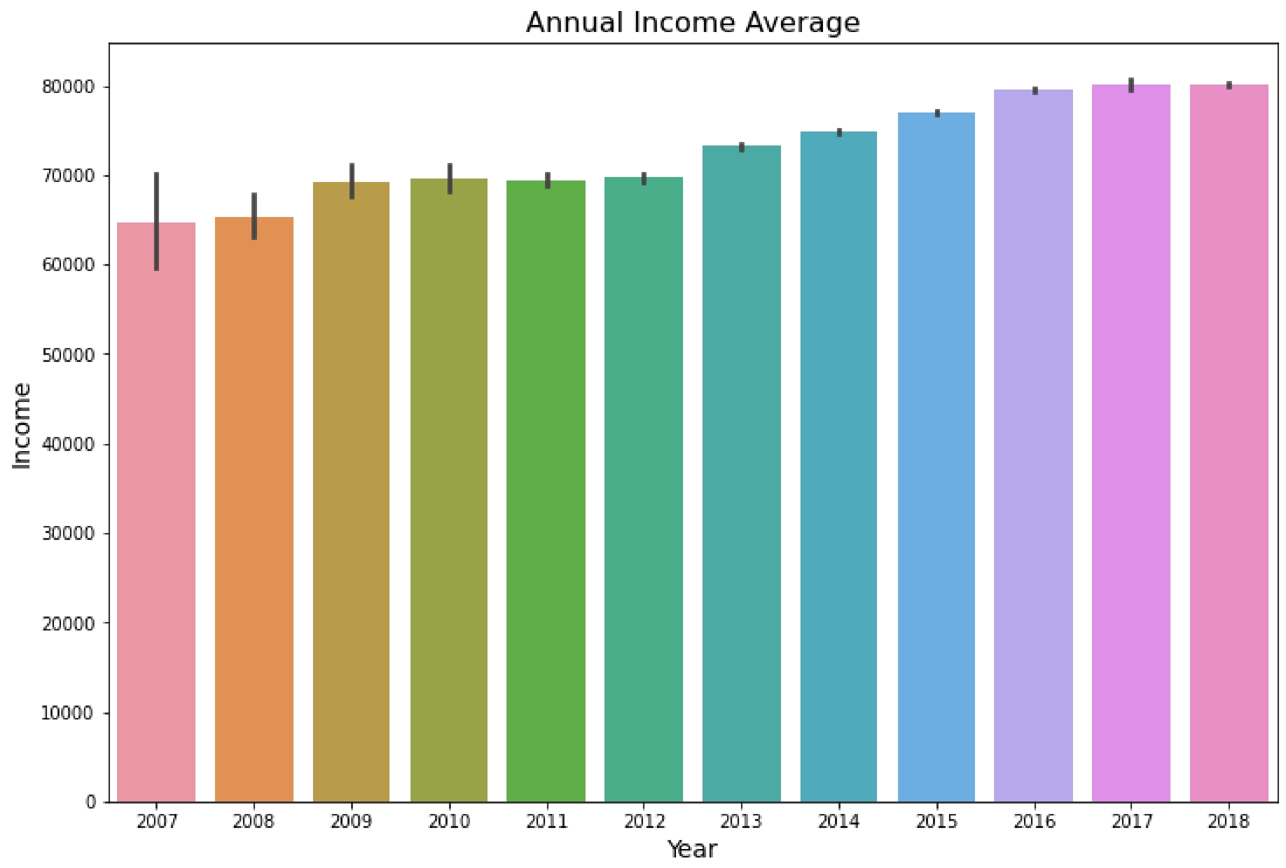
What is the average amount issued by borrowers



Exploratory Data Analysis

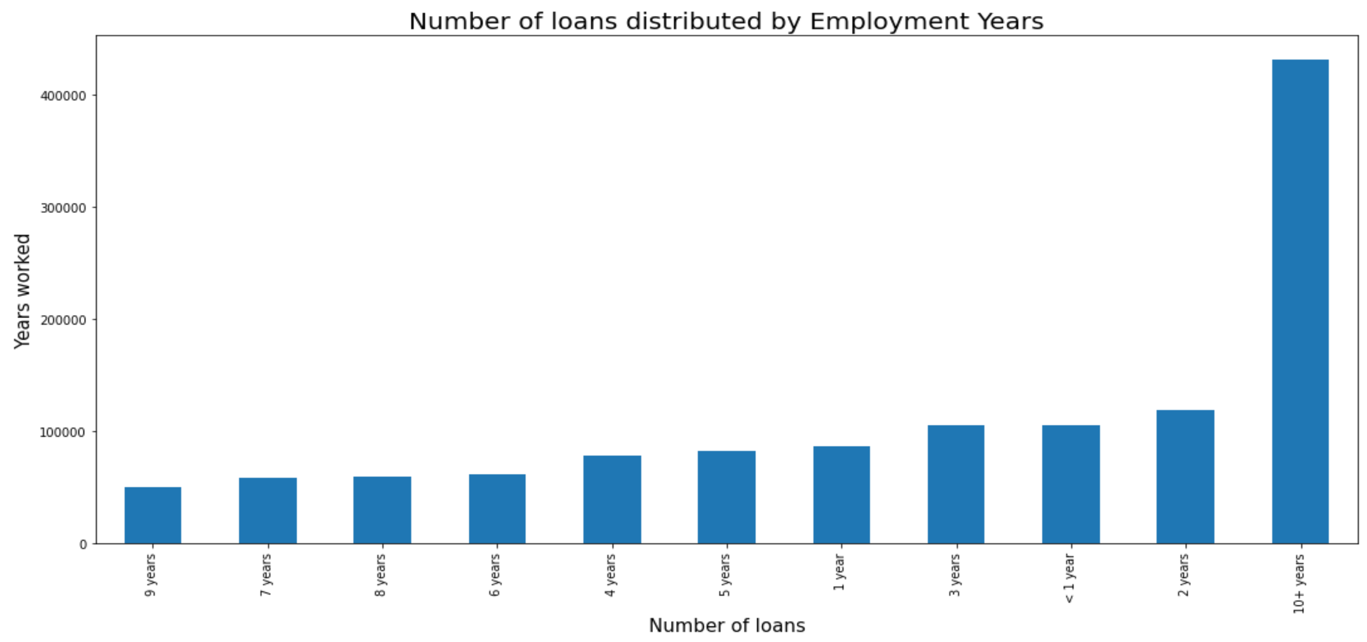
What is the average income of borrowers

So people are borrowing more money but their income has not increased with the same rate



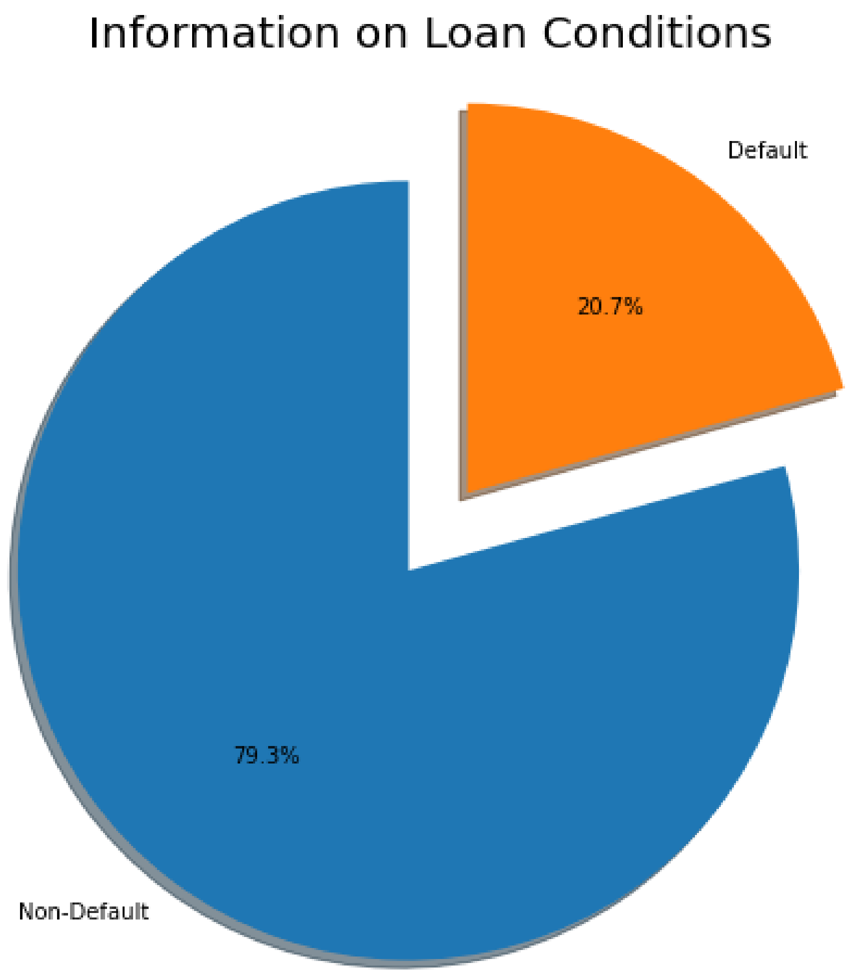
Exploratory Data Analysis

What is the job stability of these people? How many years of experience do they have



Exploratory Data Analysis

How many loans are actually Paid by the borrowers?

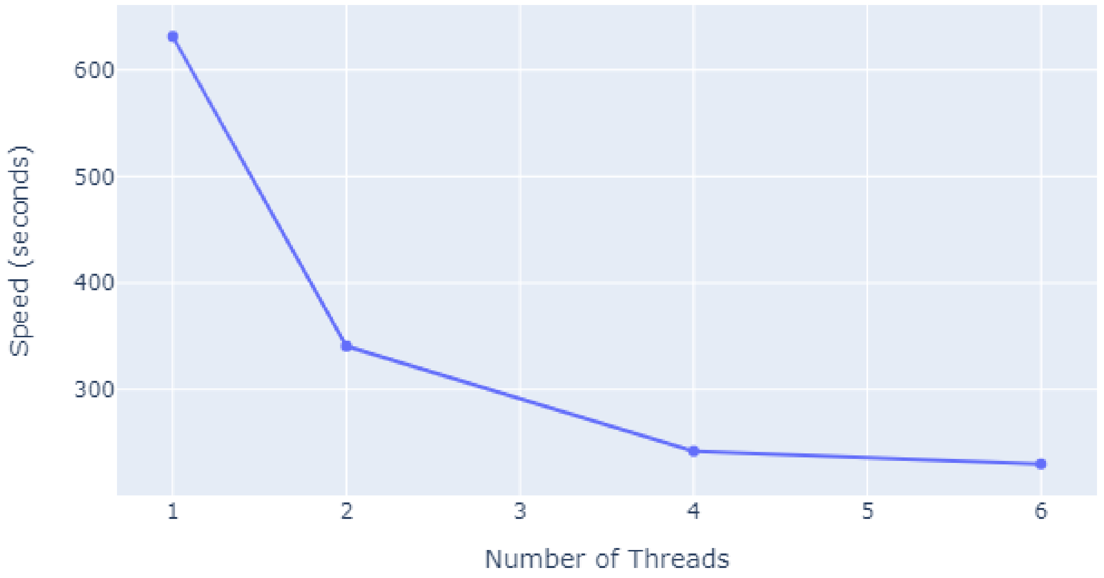


Algorithms and Methodologies

- Since the dataset is huge and the number of features is high as well the model could overfit easily.
- The imbalance in dependent feature could create bias in the model
- To deal with all these I decided to implement ensemble learning (i.e. decision tree-based) algorithms.
  - Random Forest
  - Gradient Boosting
  - Extreme Gradient Boosting

Algorithms and Methodologies

Random Forest



- K-fold cross-Validation validation accuracy 100%
- Model Performance
  - Precision – 1, Recall 0.98, F1-Score 0.99
  - Accuracy – 0.996
- Best time – 130 sec

Algorithms and Methodologies

Gradient Boosting

- Catboost library provides GPU and CPU execution support for gradient boosting
- GPU enables model to train quickly – 3x faster
- No difference in performance

Algorithms and Methodologies

Gradient Boosting

- Gradient boosting with CPU
- K-fold cross-Validation validation accuracy 100%
- Model Performance
  - Precision – 1, Recall 0.99, F1-Score 0.99
  - Accuracy – 0.996
- Best time – 390sec*
- Gradient Boosting reduced the number of misclassified borrowers

*Note: Results are obtained using all cores(6) available on CPU

Algorithms and Methodologies

Gradient Boosting

- Gradient boosting with GPU
- K-fold cross-Validation validation accuracy 100%
- Model Performance
  - Precision – 1, Recall 0.99, F1-Score 0.99
  - Accuracy – 0.996
- Best time – 68sec*
- Gradient Boosting model with GPU reduced the training time significantly
- However there's no change in model performance

*Note: Results are obtained using 1 tesla P-100 (16GB) GPU

Algorithms and Methodologies

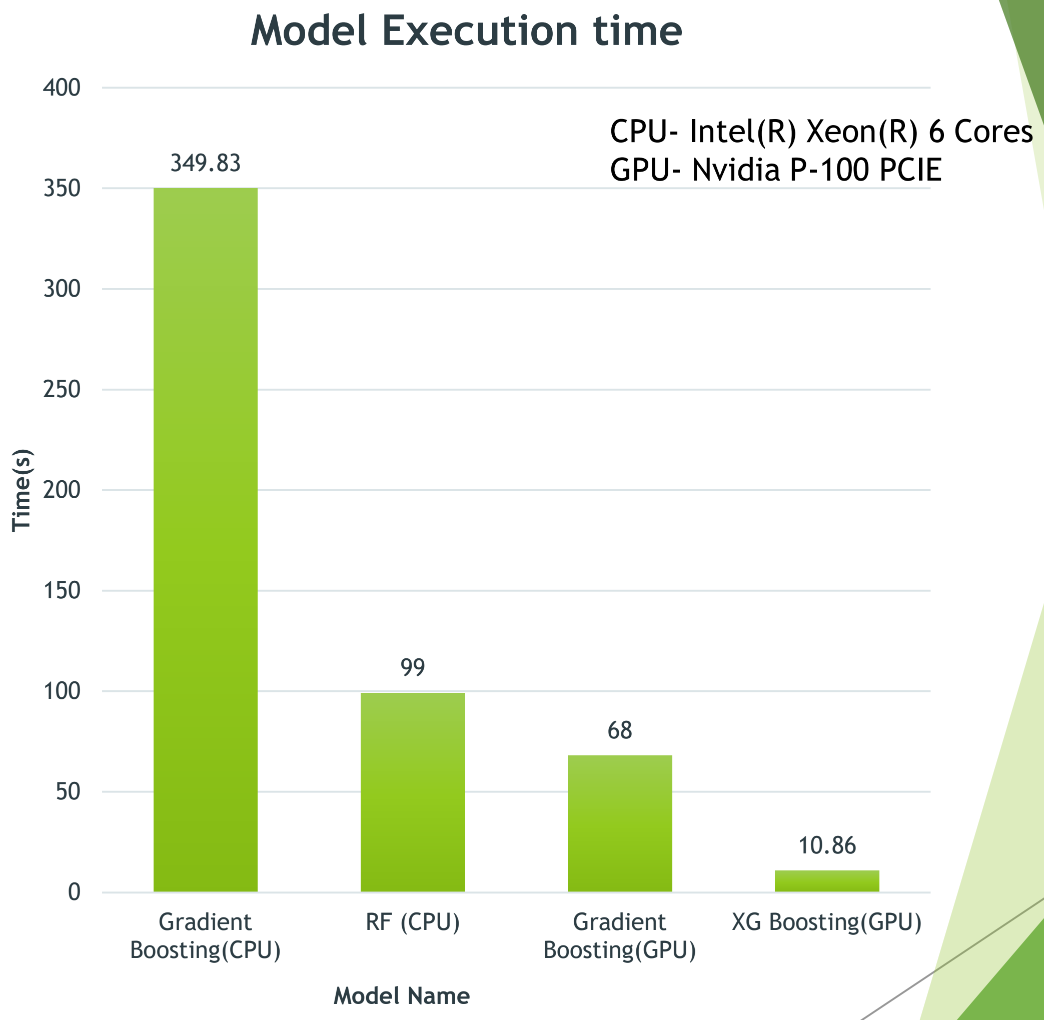
Extreme Gradient Boosting

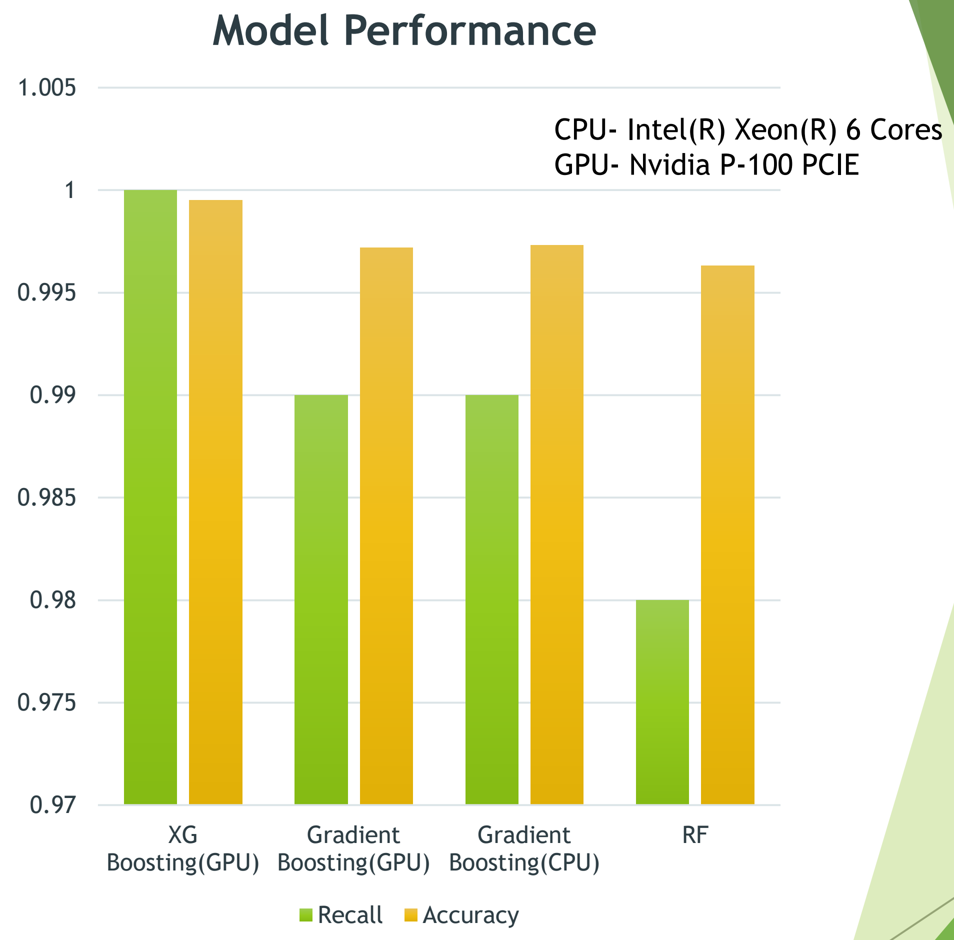
- XG Boosting model improved the model performance and reduced the number of misclassifications.
- Model was trained on GPU with XG Boost D-matrix as input
- Model Performance
  - Precision – 1, Recall 1, F1-Score 1
  - Accuracy – 0.9995
- Best time – 10sec*
- XG boosting reduced the model training time and number of misclassification

*Note: Results are obtained using 1 tesla P-100 (16GB) GPU

Model Performance

- Training time for all the models with maximum number of cores/threads





Model Performance

- XG boost model performed best in every aspect

Conclusion

- XG boost was the best model for predicting the loan defaulters, it would help investors in screening from the list of applicants
- For model development we need to make our priorities clear first
  - CPU will be cost-effective, easily available but at the same time time-consuming
  - GPU will be costly, difficult to maintain but it’ll provide better results
- GPU will always optimize the performance of the model but at a cost
- We need to find the trade-off

Thank You