



# Credit Card Approval Prediction

DS105 – Final Project Presentation

# Table of Content

1. Problem Statement & Goal
2. Dataset Review
3. Machine Learning Approach & Challenges Anticipated
4. Sub-Goals
5. Machine Learning Process



# Problem Statement & Goal

## Problem Statement

Banks heavily rely on credit score to assess applicant creditworthiness that may **lose it predictive power** due to large economic fluctuation

Credit score's creditworthiness **do not paint a complete picture** of the applicant such as their personal information. Only rely on historical data such as payment history and credit utilization

## Goal

To **build a Machine Learning Model** to predict “good” or “bad” credit card applicant **based on the collected personal information's** from the applicant with will not lose it predictive power.



## Dataset Review

Column, bar, and pie charts compare values in a single category, such as the number of products sold by each salesperson. Pie charts show each category's value as a percentage of the whole.

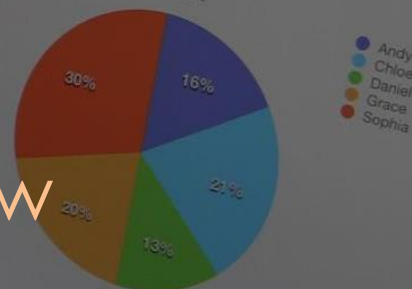
Fundraiser Results by Salesperson

PARTICIPANT	UNITS SOLD
Andy	11
Chloe	15
Daniel	9
Grace	14
Sophia	21

Column Chart



Pie Chart





# Dataset Review

## Datasets

1. Application\_record.csv
2. Credit\_record.csv

<https://www.kaggle.com/rikdifos/credit-card-approval-prediction>

## Description

- Datasets are **connected by customer IDs**.
- **Application\_record.csv** contains applicant personal information, can be use as **features**. *(Total 18 columns and 439k rows)*
- **Credit\_record.csv** records the applicant behaviours of credit card, can be use as **label**. *(Total 3 columns and 1.05m rows)*

# Dataset Review

## Variable Types - Application\_record.csv

No.	Feature name	Description	Variable Type	Data Type	Variable Category
1	ID	Client number	-	Numeric	Continuous
2	CODE_GENDER	Gender	Predictor	Numeric	Categorical
3	FLAG_OWN_CAR	Is there a car	Predictor	Character	Categorical
4	FLAG_OWN_REALTY	Is there a property	Predictor	Character	Categorical
5	CNT_CHILDREN	Number of children	Predictor	Numeric	Continuous
6	AMT_INCOME_TOTAL	Annual income	Predictor	Numeric	Continuous
7	NAME_INCOME_TYPE	Income category	Predictor	Character	Categorical
8	NAME_EDUCATION_TYPE	Education level	Predictor	Character	Categorical
9	NAME_FAMILY_STATUS	Marital status	Predictor	Character	Categorical
10	NAME_HOUSING_TYPE	Way of living	Predictor	Character	Categorical
11	DAYS_BIRTH	Birthday count backwards from current day (0), -1 means yesterday	Predictor	Numeric	Continuous
12	DAYS_EMPLOYED	Start date of employment count backwards from current day(0). If positive, it means the person currently unemployed.	Predictor	Numeric	Continuous
13	FLAG_MOBIL	Is there a mobile phone	Predictor	Numeric	Categorical
14	FLAG_WORK_PHONE	Is there a work phone	Predictor	Numeric	Categorical
15	FLAG_PHONE	Is there a phone	Predictor	Numeric	Categorical
16	FLAG_EMAIL	Is there an email	Predictor	Numeric	Categorical
17	OCCUPATION_TYPE	Occupation	Predictor	Character	Categorical
18	CNT_FAM_MEMBERS	Family size	Predictor	Numeric	Continuous

Both datasets will be connected with Client Number

# Dataset Review

## Variable Types - Credit\_record.csv

No.	Feature name	Description	Variable Type	Data Type	Variable Category
1	ID	Client number	-	Numeric	Continuous
2	MONTHS_BALANCE	The month of the extracted data is the starting point, backwards, 0 is the current month, -1 is the previous month, and so on	Predictor	Numeric	Categorical
3	STATUS	Payment Status 0 : 1-29 days past due 1 : 30-59 days past due 2 : 60-89 days overdue 3 : 90-119 days overdue 4 : 120-149 days overdue 5 : Overdue or bad debts, write-offs for more than 150 days C : paid off that month X : No loan for the month	Target	Character	Categorical

Both datasets will be connected with Client Number

Label for credit card approval based on the acceptable past due range



# Dataset Review

Snapshots - Appication\_record.csv

	ID	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	NAME_INCOME_TYPE	NAME_EDUCATION_TYPE
0	5008804	M	Y	Y	0	427500.0	Working	Higher education
1	5008805	M	Y	Y	0	427500.0	Working	Higher education
2	5008806	M	Y	Y	0	112500.0	Working	Secondary / secondary special
3	5008808	F	N	Y	0	270000.0	Commercial associate	Secondary / secondary special
4	5008809	F	N	Y	0	270000.0	Commercial associate	Secondary / secondary special

NAME_FAMILY_STATUS	NAME_HOUSING_TYPE	DAYS_BIRTH	DAYS_EMPLOYED	FLAG_MOBIL	FLAG_WORK_PHONE	FLAG_PHONE	FLAG_EMAIL
Civil marriage	Rented apartment	-12005	-4542	1	1	0	0
Civil marriage	Rented apartment	-12005	-4542	1	1	0	0
Married	House / apartment	-21474	-1134	1	0	0	0
Single / not married	House / apartment	-19110	-3051	1	0	1	1
Single / not married	House / apartment	-19110	-3051	1	0	1	1



# Dataset Review

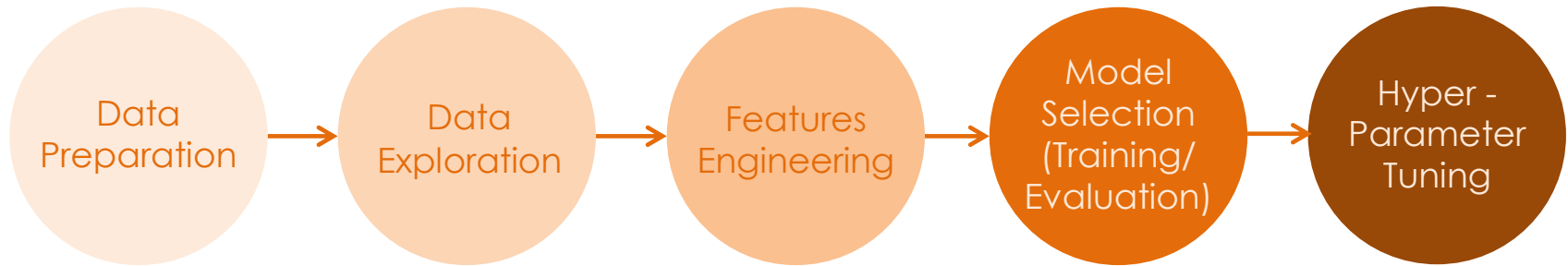
Snapshots - Credit\_record.csv

	ID	MONTHS_BALANCE	STATUS
0	5001711	0	X
1	5001711	-1	0
2	5001711	-2	0
3	5001711	-3	0
4	5001712	0	C



# Machine Learning Approach & Challenges Anticipated

# Machine Learning Approach & Challenges Anticipated

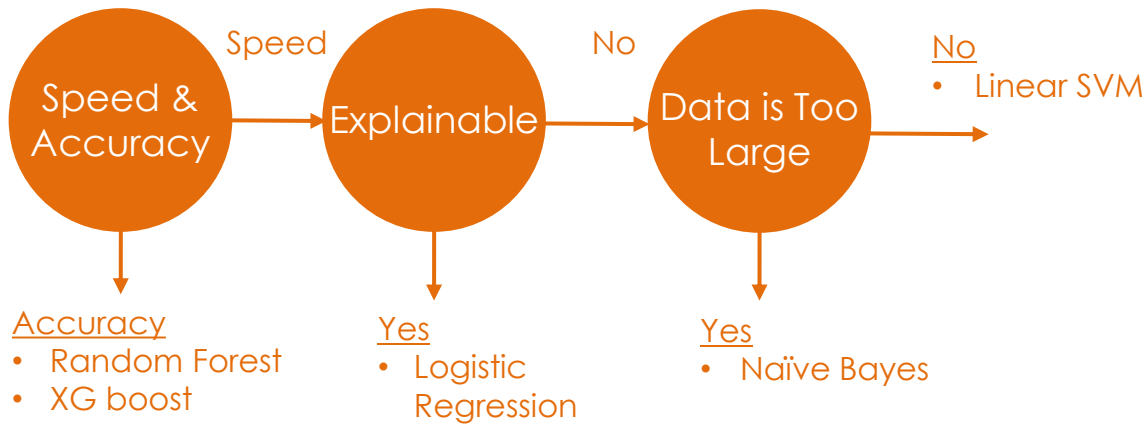


## Challenges

- Null values treatment
- Duplicate records
- Joining of datasets
- Relationship between variables
- Create Label
- Create new meaningful features
- Reduce unused features
- Data unbalance.
- Encode categorical features
- Scale overall dataset
- Choose the best Classification ML model based on different evaluation method
- Choose the best hyper parameter

# Machine Learning Approach & Challenges Anticipated

## Classification Models Selection



## Evaluation Methods

- Confusion Matrix
  - Accuracy
  - Recall
  - Precision
  - F1-Score
  - ROC / AUC



■ Sub-Goals



# Sub-Goals

1. Method of creating label from credit\_record.csv?
2. Encoder used for all categorical features?
3. Clear segregation between the “good” and “bad” credit card applicants?
4. The best classification machine learning model for prediction?
5. The best hyper-parameters to give the best prediction result?
6. The feature that give the major contribution to the prediction?





# Machine Learning Process

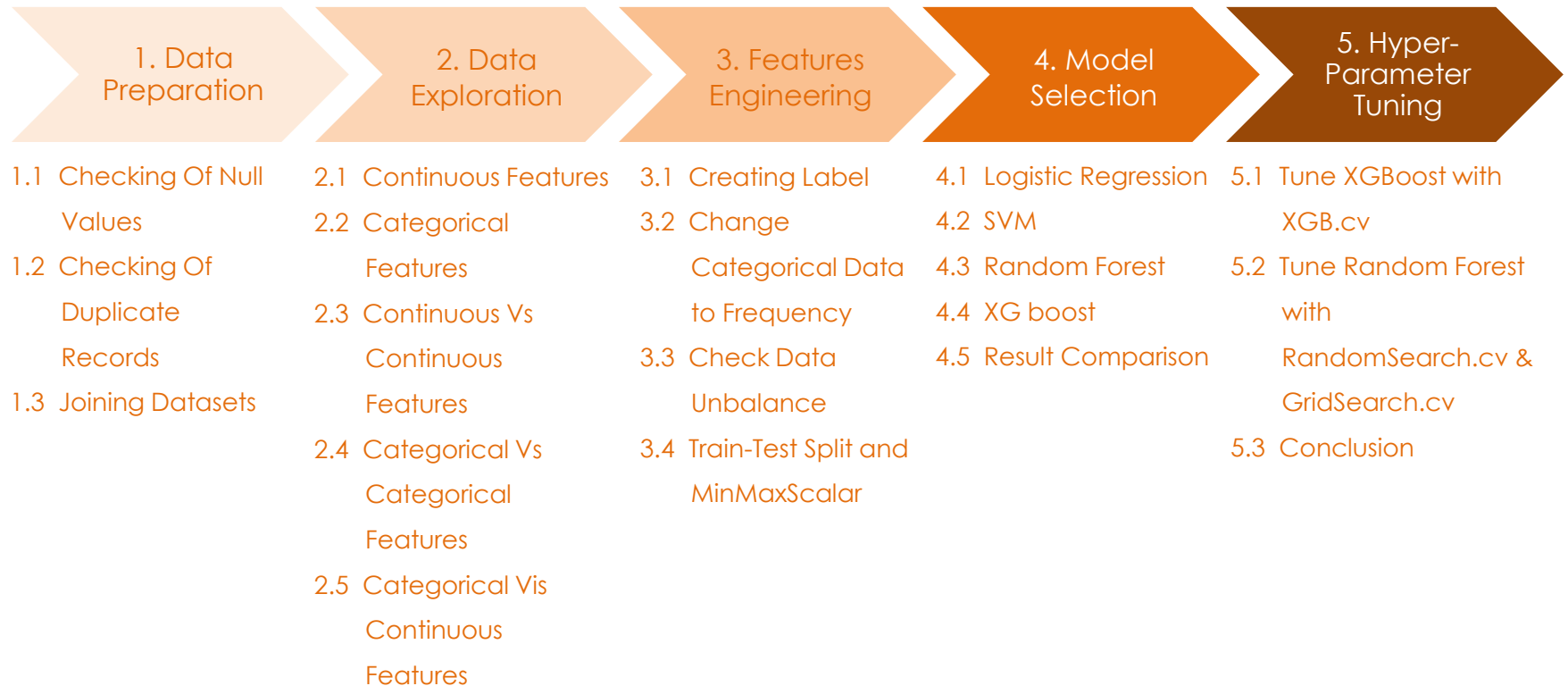
```
3 require File.expand_path("../..", __FILE__)
4 # Prevent database truncation if the environment is production
5 abort("The Rails environment is running in production mode!")
6 require 'spec_helper'
7 require 'rspec/rails'
```

```
8
9 require 'capybara/rspec'
10 require 'capybara/rails'
```

```
11
12 Capybara.javascript_driver = :webkit
13 Category.delete_all; Category.create
14 Shoulda::Matchers.configure do |config|
15   config.integrate do |with|
16     with.test_framework :rspec
17     with.library :rails
18   end
19 end
```

```
20 # Add additional requires below this line. Please do not
21 # Requires supporting ruby files with static require data.
22 # spec/support/ and its subdirectories. Files matching
23 # run as spec files by default. You can use the
24 # in _spec.rb will both be required when running
25 # run twice. It is recommended that you use
26 # end with _spec.rb. You can use the
27 # option on the command line to run
28 # No results found for 'mongoid'
```

# Machine Learning Process



# Machine Learning Process

1. Data Preparation

2. Data Exploration

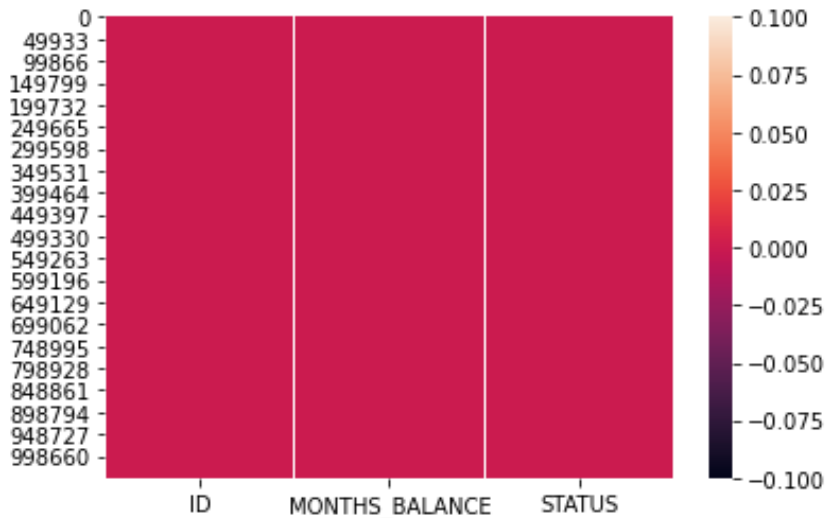
3. Features Engineering

4. Model Selection

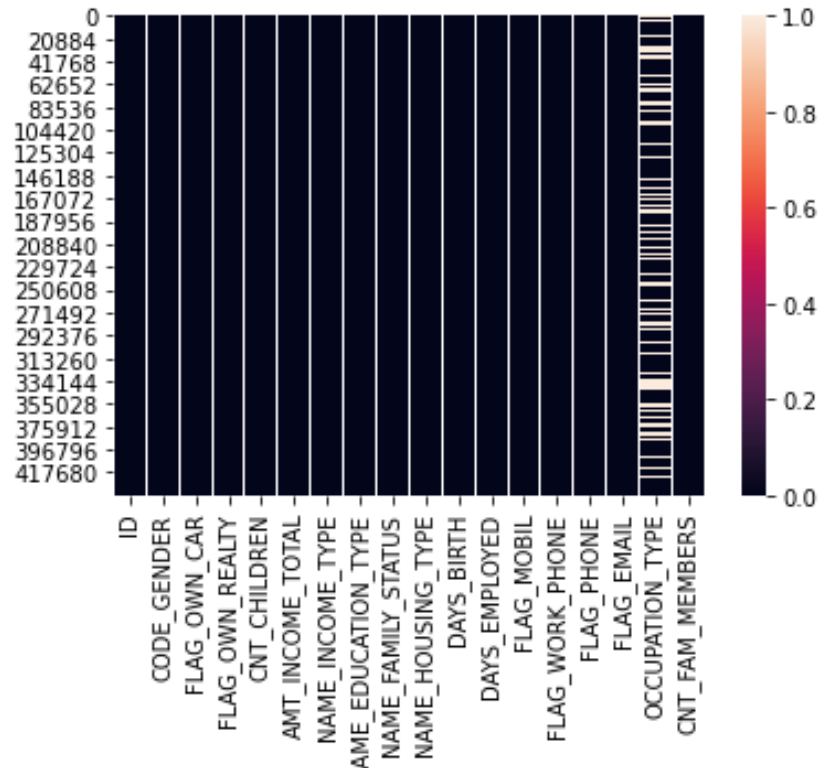
5. Hyper-Parameter Tuning

## 1.1 Checking Of Null Values

app\_record = Application\_record.csv



credit\_record = Credit\_record.csv



- 30.6% null value in "OCCUPATION\_TYPE" column
- <50%, hence, **fillna** with cat "OTHER"

# Machine Learning Process

1. Data Preparation

2. Data Exploration

3. Features Engineering

4. Model Selection

5. Hyper-Parameter Tuning

## 1.2 Checking Of Duplicate Records

app\_record

Columns:438557 - UniqueID:438510 = 47

	ID	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	NAME_INCOME_TYPE	NAME_EDUCATION_TYPE
426818	7022197	M	Y	Y	3	135000.0	Working	Secondary / secondary special
425023	7022197	F	N	Y	0	450000.0	Commercial associate	Higher education
431545	7022327	F	N	Y	0	135000.0	Commercial associate	Secondary / secondary special
431911	7022327	M	Y	Y	0	256500.0	Commercial associate	Higher education

94 rows x 18 columns

- **app\_record['ID'].duplicated** found duplicate records (47 x 2 = 94 rows)
- Different data in each features, hence, might be issue when assigning IDs to new applicant
- Decided to drop every 2nd record by **app\_record.drop\_duplicates()**

credit\_record

Columns:1048575 - UniqueID:45985 = 1002590

	ID	MONTHS_BALANCE	STATUS
0	5001711	0	X
1	5001711	-1	0
2	5001711	-2	0
3	5001711	-3	0
4	5001712	0	C

- Remove multiple monthly entries and only keep the latest record.

**credit\_uniq = credit\_record.groupby('ID').max().reset\_index()**

# Machine Learning Process



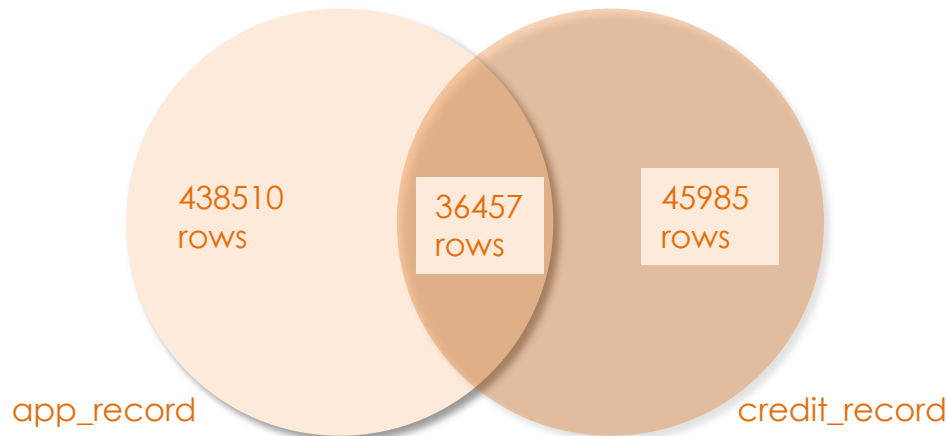
## 1.3 Joining Datasets

- Checking how many ID do two datasets share?  
36457 shared the same ID.

```
len(set(app_record['ID']).intersection(set(credit_uniq['ID'])))
```

- Inner join both datasets together by ID.

```
df = app_record.join(credit_uniq.set_index('ID'), on='ID', how='inner')
```



# Machine Learning Process

1. Data Preparation

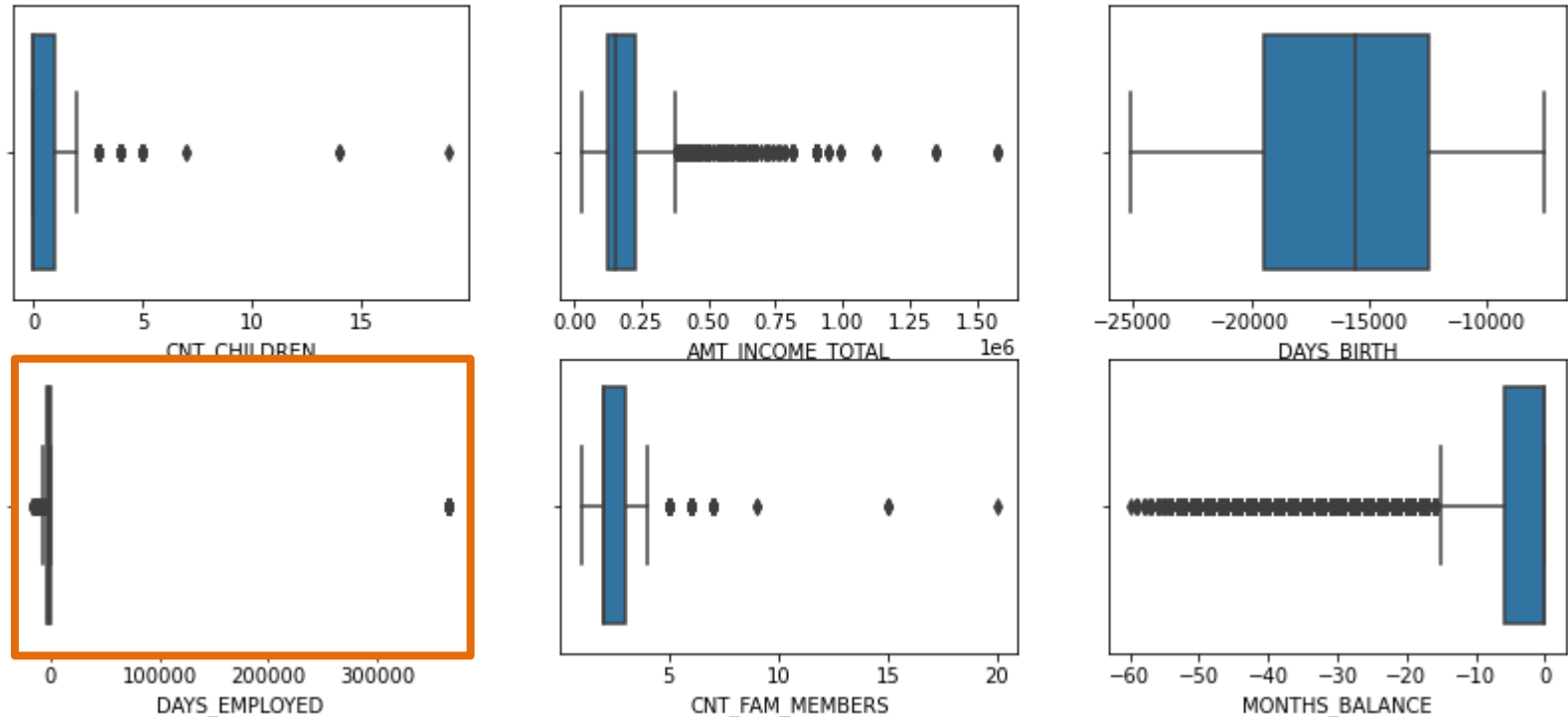
2. Data Exploration

3. Features Engineering

4. Model Selection

5. Hyper-Parameter Tuning

## 2.1 Continuous Features (6 columns)



- “DAYS\_EMPLOYED” have outliers of >30,000days (82years) which is impossible.
- Other features have outliers that more then 1.5x IQ however, the values are still reasonable
- Drop all data in "DAYS\_EMPLOYED" that > 30,000days. `df2 = df[df['DAYS_EMPLOYED'] < 300000]`



# Machine Learning Process

1. Data Preparation

2. Data Exploration

3. Features Engineering

4. Model Selection

5. Hyper-Parameter Tuning

## 2.2 Categorical Features (13 columns)

We can use frequency table to understand distribution of each category

===== CODE\_GENDER =====

	count	percent%
F	19195	63.3
M	11127	36.7

===== FLAG\_OWN\_CAR =====

	count	percent%
N	17766	58.59
Y	12556	41.41

===== FLAG\_OWN\_REALTY =====

	count	percent%
Y	19786	65.25
N	10536	34.75

===== NAME\_INCOME\_TYPE =====

	count	percent%
Working	18819	62.06
Commercial associate	8490	28.00
State servant	2985	9.84
Pensioner	17	0.06
Student	11	0.04

===== NAME\_EDUCATION\_TYPE =====

	count	percent%
Secondary / secondary special	19867	65.52
Higher education	8858	29.21
Incomplete higher	1352	4.46
Lower secondary	214	0.71
Academic degree	31	0.10

===== NAME\_FAMILY\_STATUS =====

	count	percent%
Married	21137	69.71
Single / not married	4148	13.68
Civil marriage	2575	8.49
Separated	1758	5.80
Widow	704	2.32

===== NAME\_FAMILY\_STATUS =====

	count	percent%
Married	21137	69.71
Single / not married	4148	13.68
Civil marriage	2575	8.49
Separated	1758	5.80
Widow	704	2.32

===== FLAG\_MOBIL =====

	count	percent%
1	30322	100.0

===== FLAG\_WORK\_PHONE =====

	count	percent%
0	22100	72.88
1	8222	27.12

===== FLAG\_PHONE =====

	count	percent%
0	21342	70.38
1	8980	29.62

===== FLAG\_EMAIL =====

	count	percent%
0	27265	89.92
1	3057	10.08

===== STATUS =====

	count	percent%
X	16281	53.69
C	9840	32.45
0	3538	11.67
1	574	1.89
5	42	0.14
2	38	0.13
3	7	0.02
4	2	0.01

===== OCCUPATION\_TYPE =====

	count	percent%
Laborers	6211	20.48
Others	5188	17.11
Core staff	3591	11.84
Sales staff	3485	11.49
Managers	3012	9.93
Drivers	2138	7.05
High skill tech staff	1383	4.56
Accountants	1241	4.09
Medicine staff	1207	3.98
Cooking staff	655	2.16
Security staff	592	1.95
Cleaning staff	551	1.82
Private service staff	344	1.13
Low-skill Laborers	175	0.58
Waiters/barmen staff	174	0.57
Secretaries	151	0.50
HR staff	85	0.28
Realty agents	79	0.26
IT staff	60	0.20

- All of features are important since there is very fine classification in each column.
- Their effectiveness cannot be judged at this moment.
- Will drop "FLAG\_MOBIL" as it 100% count 1 which no point to include into the model later on.  
**df3 = df2.drop(["FLAG\_MOBIL"],axis='columns')**

# Machine Learning Process

1. Data Preparation

2. Data Exploration

3. Features Engineering

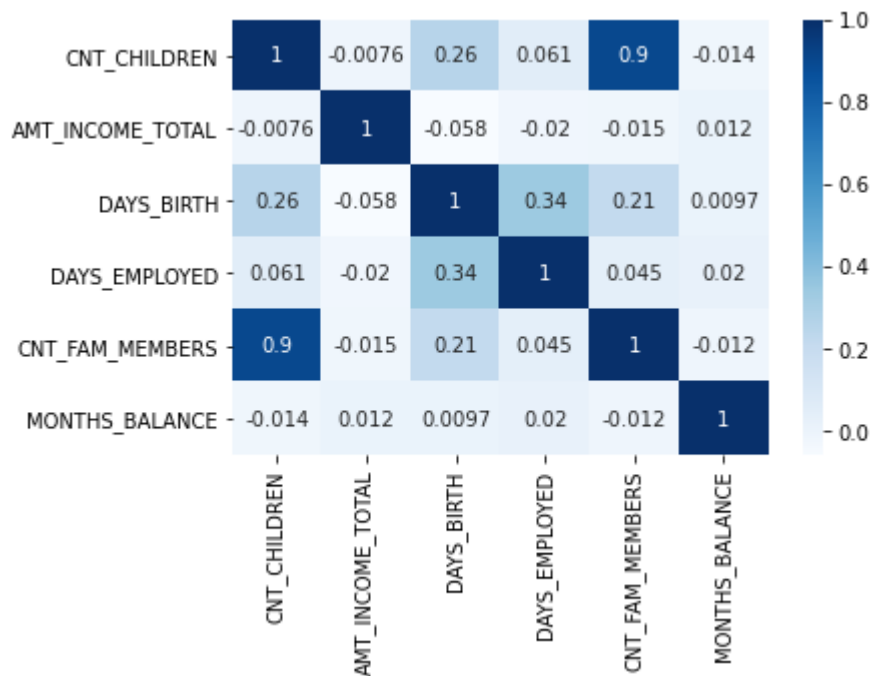
4. Model Selection

5. Hyper-Parameter Tuning

## 2.3 Continuous Vs Continuous Features

We can use standard Pearson coefficient (**df.corr**) to understand correlation between each continuous variables

- -1: perfect negative linear correlation
- +1: perfect positive linear correlation and
- 0: No correlation



- “CNT\_CHILDREN” & “CNT\_FAM\_MEMBERS” are more correlated to each other.
- The no. of children is within the count of family members.
- Others are close to no correlation

# Machine Learning Process

1. Data Preparation

2. Data Exploration

3. Features Engineering

4. Model Selection

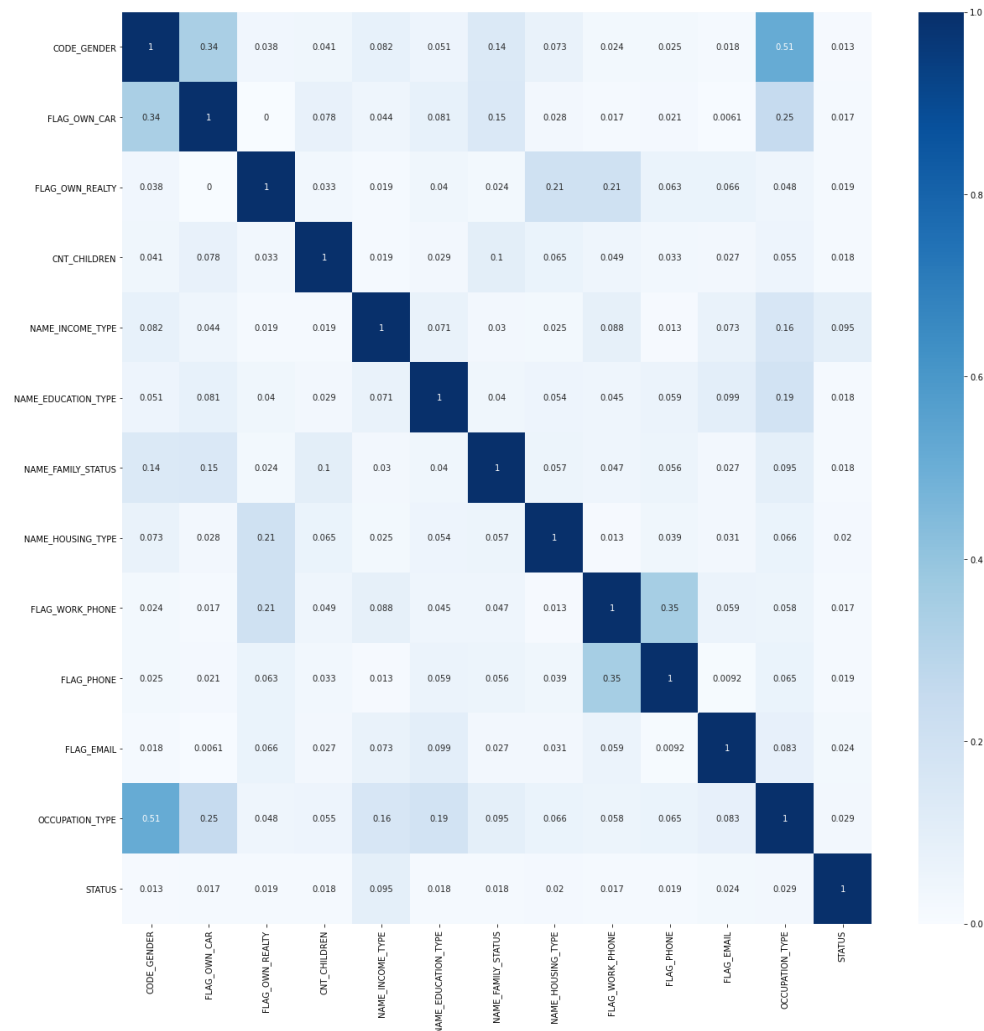
5. Hyper-Parameter Tuning

## 2.4 Categorical Vs Categorical Features

We will use “Cramer's V” to find degree of association between categorical variables

- 0: The variables are not associated
- 1: The variables are perfectly associated
- 0.25: The variables are weakly associated
- 0.75: The variables are moderately associated

- Most are weakly associated (<0.25) between columns.
- The highest (0.51) is between Gender & Occupation type
- The next highest are between Gender & Own a car
- Both association lower than moderate (<0.75)



# Machine Learning Process

1. Data Preparation

2. Data Exploration

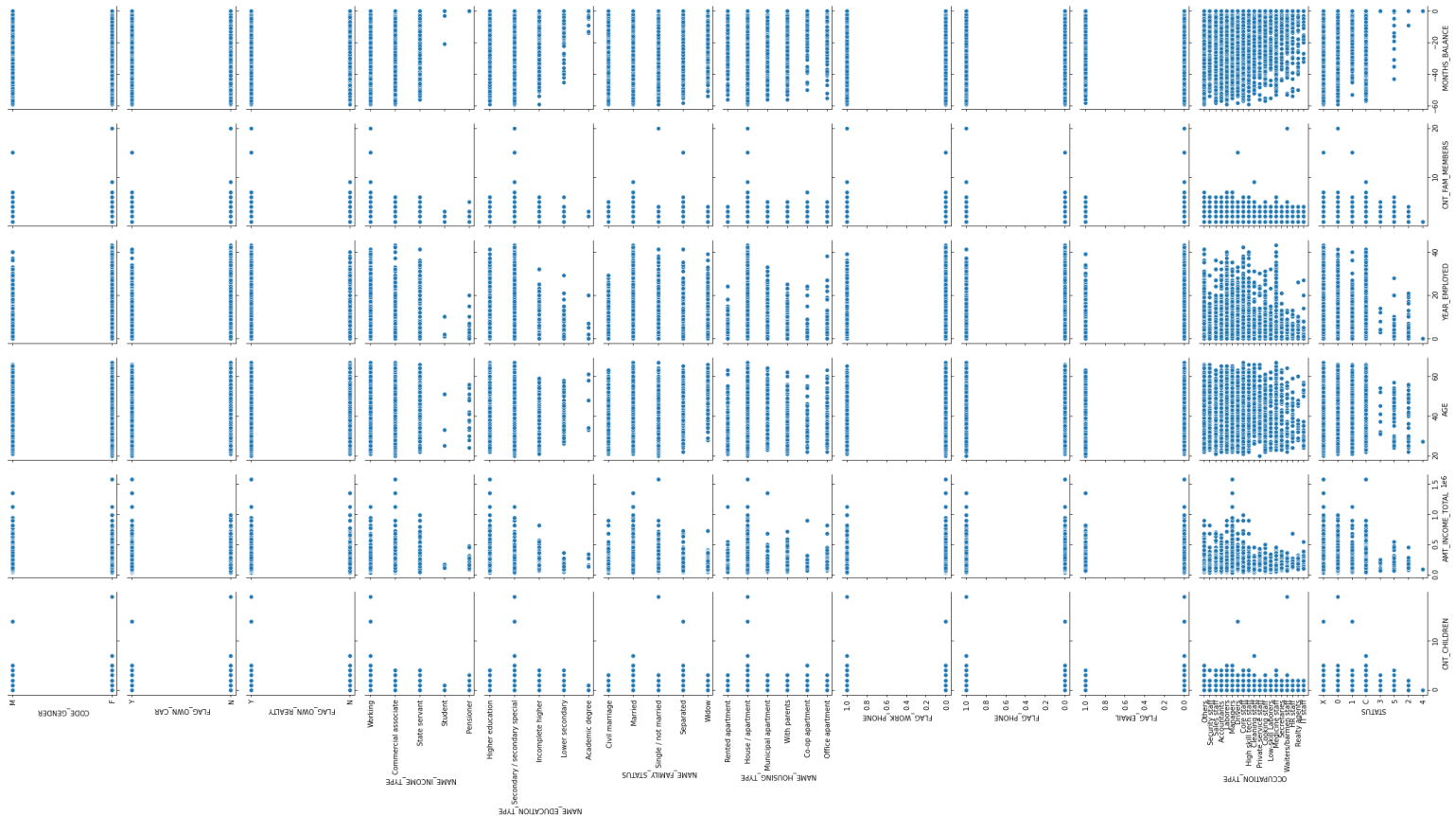
3. Features Engineering

4. Model Selection

5. Hyper-Parameter Tuning

## 2.5 Categorical Vis Continuous Features

We will use strip plot to understand the distribution of each continuous to categorical variables.



Categorical Features (13 columns)

Continuous Features (6 columns)

# Machine Learning Process

1. Data Preparation

2. Data Exploration

3. Features Engineering

4. Model Selection

5. Hyper-Parameter Tuning

## 3.1 Creating Label

- We will be creating binary label. (1 : Bad Customer, 0 : Good Customer)
- Refer to distribution table of 'STATUS'. Almost 98% users have not more than 29 days overdue (only 2% for >29 days overdue), which is too common, thus, it's inappropriate to be our standard.
- Whereas if we use >89 days overdue (in most bank standard), its only 0.17%. If we use that, we will left out many bad customers from our analysis.
- Hence, we will define that overdue >1 day will be the bad customer

### Payment Status :

X : No loan for the month

C : paid off that month

0 : 1-29 days past due

1 : 30-59 days past due

2 : 60-89 days overdue

3 : 90-119 days overdue

4 : 120-149 days overdue

5. >150 days

### ===== STATUS =====

	count	percent%	
X	16281	53.69	
C	9840	32.45	98%
0	3538	11.67	
1	574	1.89	
2	38	0.13	
3	7	0.02	
4	2	0.01	0.2%
5	42	0.14	

# Machine Learning Process

1. Data  
Preparation

2. Data  
Exploration

3. Features  
Engineering

4. Model  
Selection

5. Hyper-  
Parameter Tuning

## 3.2 Change Categorical Data to Frequency

We will use frequency encoder as the categorical data are all nominal type.

	ID	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	NAME_INCOME_TYPE	NAME_EDUCATION_TYPE
0	5008804	0.366961	0.414089	0.65253	0	427500.0	0.620638	0.292131
1	5008805	0.366961	0.414089	0.65253	0	427500.0	0.620638	0.292131
2	5008806	0.366961	0.414089	0.65253	0	112500.0	0.620638	0.655201
3	5008808	0.633039	0.585911	0.65253	0	270000.0	0.279995	0.655201
4	5008809	0.633039	0.585911	0.65253	0	270000.0	0.279995	0.655201
...	...	...	...	...	...	...	...	...
434808	5149828	0.366961	0.414089	0.65253	0	315000.0	0.620638	0.655201
434809	5149834	0.633039	0.585911	0.65253	0	157500.0	0.279995	0.292131
434810	5149838	0.633039	0.585911	0.65253	0	157500.0	0.000561	0.292131
434811	5150049	0.633039	0.585911	0.65253	0	283500.0	0.620638	0.655201
434812	5150337	0.366961	0.585911	0.65253	0	112500.0	0.620638	0.655201



# Machine Learning Process

1. Data Preparation

2. Data Exploration

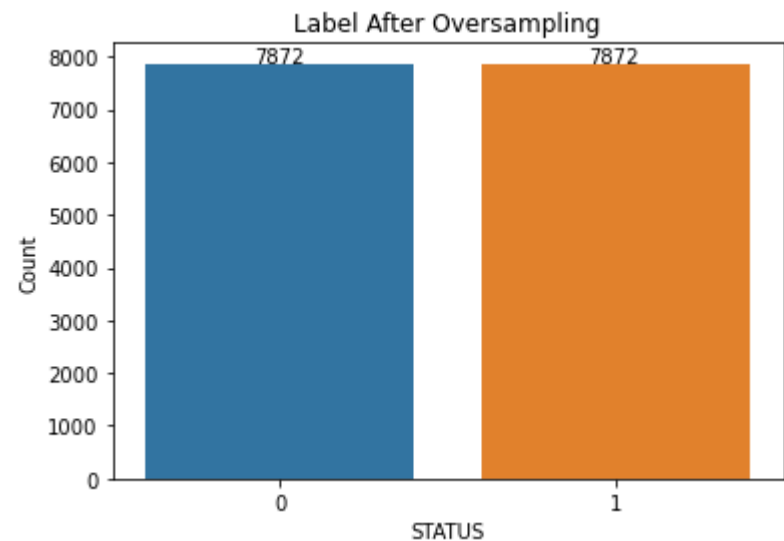
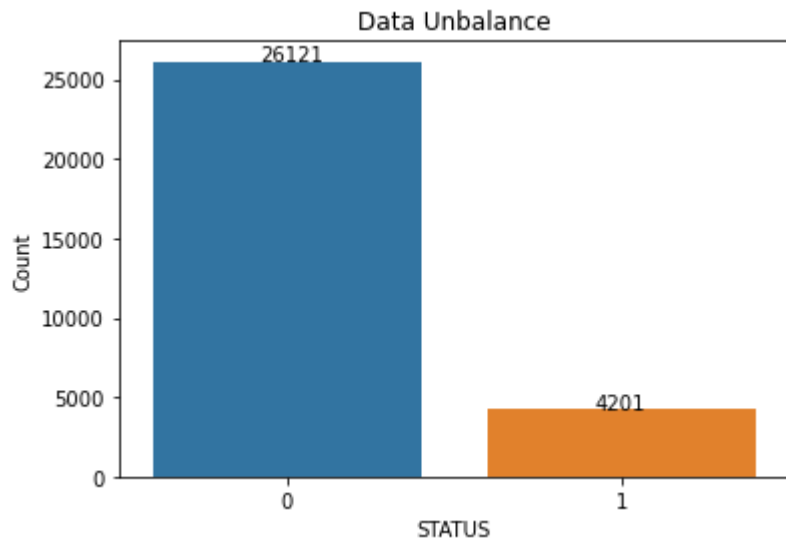
3. Features Engineering

4. Model Selection

5. Hyper-Parameter Tuning

## 3.3 Check Data Unbalance

- The dataset is imbalanced.
- Classification model will give a false accuracy rate as the prediction uses the most common class without performing any analysis of the features.
- We will use Synthetic Minority Oversampling TEchnique (SMOTE) to make the dataset balance



```
oversample = SMOTE()  
X_balanced, y_balanced = oversample.fit_resample(X_scaled, y_train)  
X_test_balanced, y_test_balanced = oversample.fit_resample(X_test_scaled, y_test)
```

# Machine Learning Process

1. Data  
Preparation

2. Data  
Exploration

3. Features  
Engineering

4. Model  
Selection

5. Hyper-  
Parameter Tuning

## 3.4 Train-Test Split and MinMaxScaler

- Split the data for 70% training and 30% testing.
- Fit and transform the data into a scaler for accurate reading and results.

```
1 X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.3)
2
3 mms = MinMaxScaler()
4 X_scaled = pd.DataFrame(mms.fit_transform(X_train), columns=X_train.columns)
5 X_test_scaled = pd.DataFrame(mms.transform(X_test), columns=X_test.columns)
```

# Machine Learning Process

1. Data Preparation

2. Data Exploration

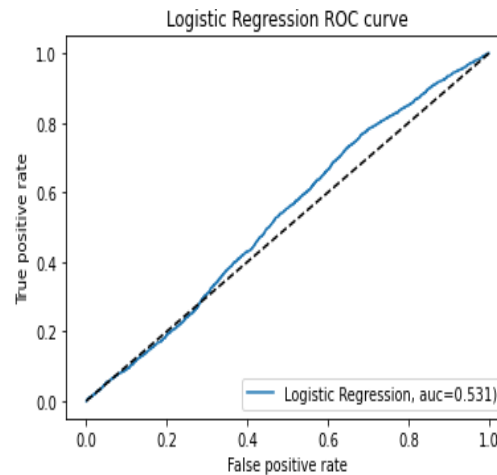
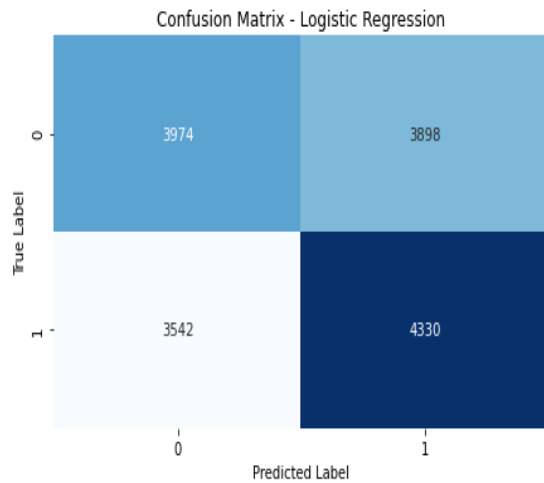
3. Features Engineering

4. Model Selection

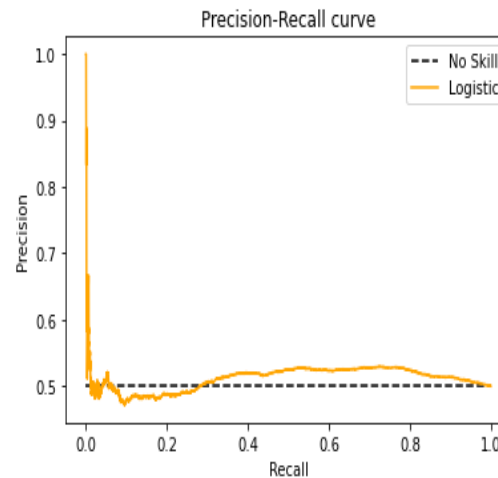
5. Hyper-Parameter Tuning

## 4.1 Logistic Regression

Metrics	Result
Accuracy	0.53
Precision	0.53
Recall	0.55
F1-Score	0.54
AUC	0.53



Classifier is not able to distinguish between Positive and Negative



Both 0 & 1 close to overlapping each other.

# Machine Learning Process

1. Data Preparation

2. Data Exploration

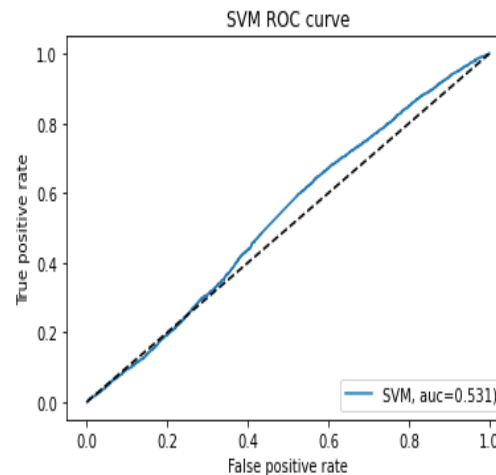
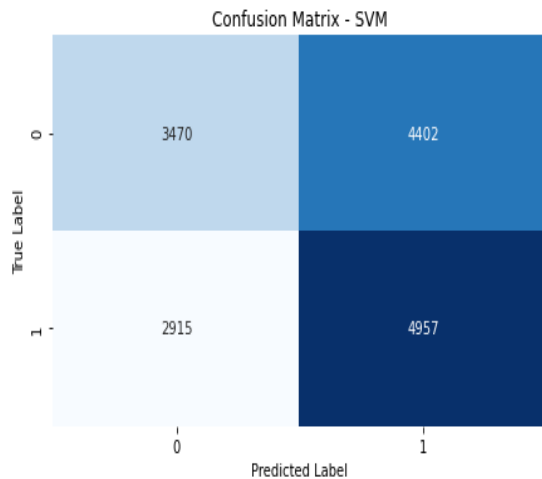
3. Features Engineering

4. Model Selection

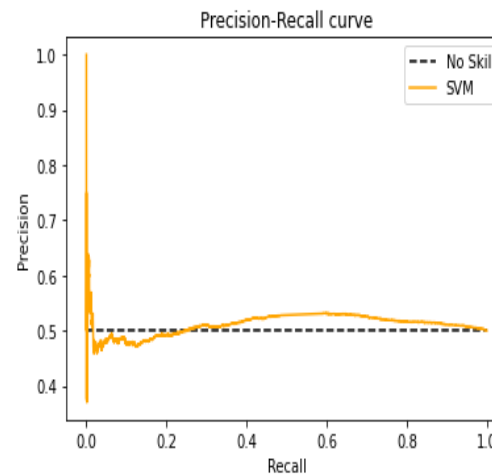
5. Hyper-Parameter Tuning

## 4.2 Support Vector Machine

Metrics	Result
Accuracy	0.54
Precision	0.53
Recall	0.63
F1-Score	0.58
AUC	0.53



Classifier is not able to distinguish between Positive and Negative



Both 0 & 1 close to overlapping each other.

# Machine Learning Process

1. Data Preparation

2. Data Exploration

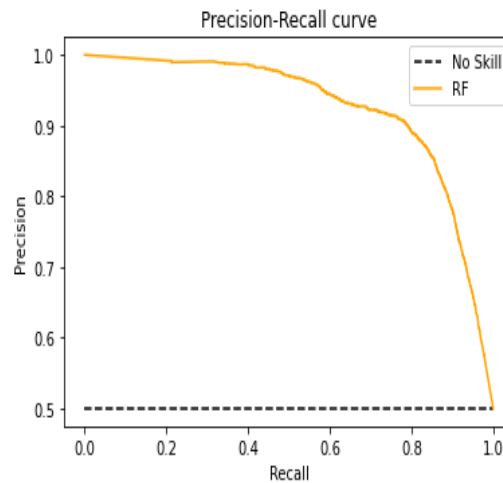
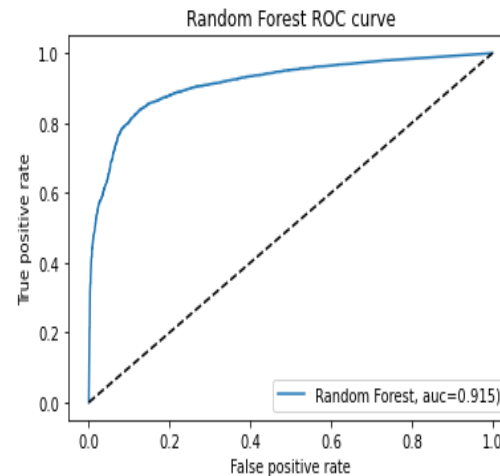
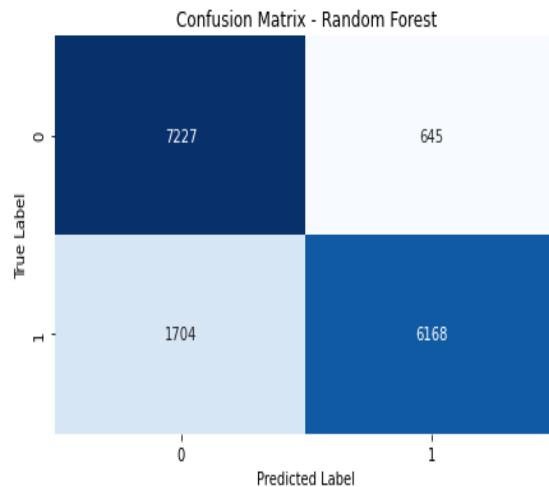
3. Features Engineering

4. Model Selection

5. Hyper-Parameter Tuning

## 4.3 Random Forest

Metrics	Result
Accuracy	0.85
Precision	0.91
Recall	0.78
F1-Score	0.84
AUC	0.91



# Machine Learning Process

1. Data Preparation

2. Data Exploration

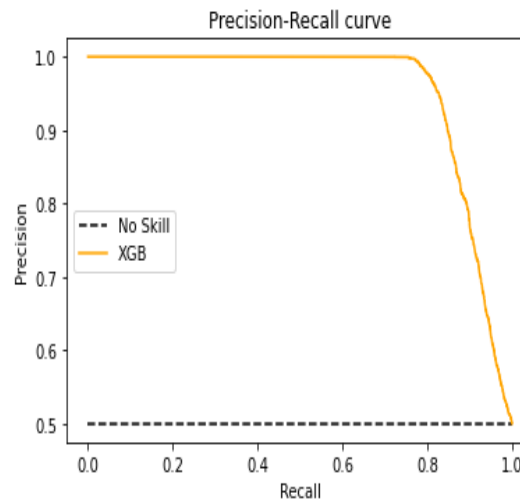
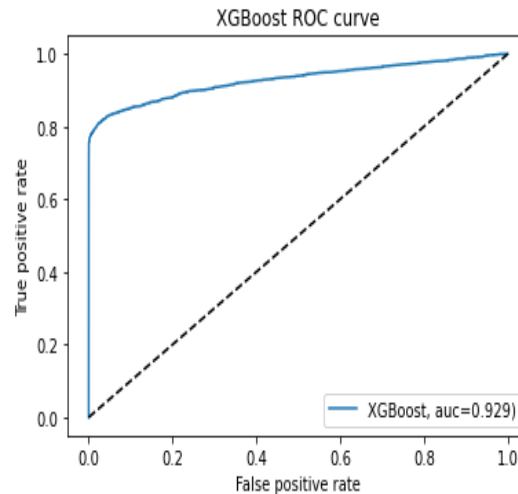
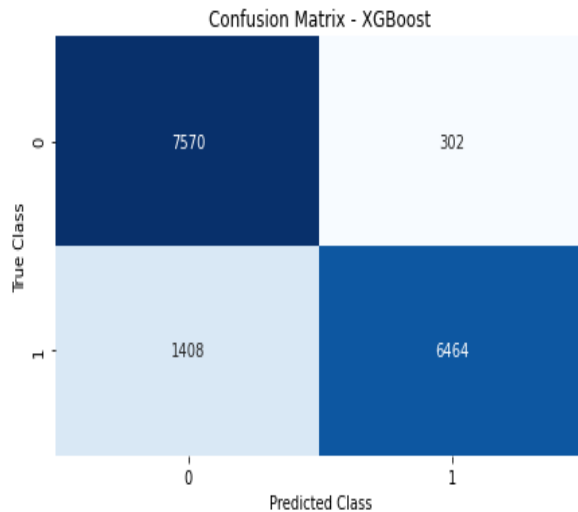
3. Features Engineering

4. Model Selection

5. Hyper-Parameter Tuning

## 4.4 XG Boost

Metrics	Result
Accuracy	0.89
Precision	0.96
Recall	0.82
F1-Score	0.88
AUC	0.93





# Machine Learning Process

1. Data  
Preparation

2. Data  
Exploration

3. Features  
Engineering

4. Model  
Selection

5. Hyper-  
Parameter Tuning

## 4.5 Result Comparison

- Random Forest and XG Boost has the close and best performance according to our classification metrics (Accuracy, F1-score and AUC).
- Next, we can further improved the both models by tuning hyper-parameters.

RESULT COMPARISON TABLE

=====

Test Method:	Log	SVM	RF	XGB
Accuracy	0.527	0.535	0.851	0.891
F1 Score	0.538	0.575	0.840	0.883
AUC	0.531	0.531	0.915	0.929

# Machine Learning Process

1. Data Preparation

2. Data Exploration

3. Features Engineering

4. Model Selection

5. Hyper-Parameter Tuning

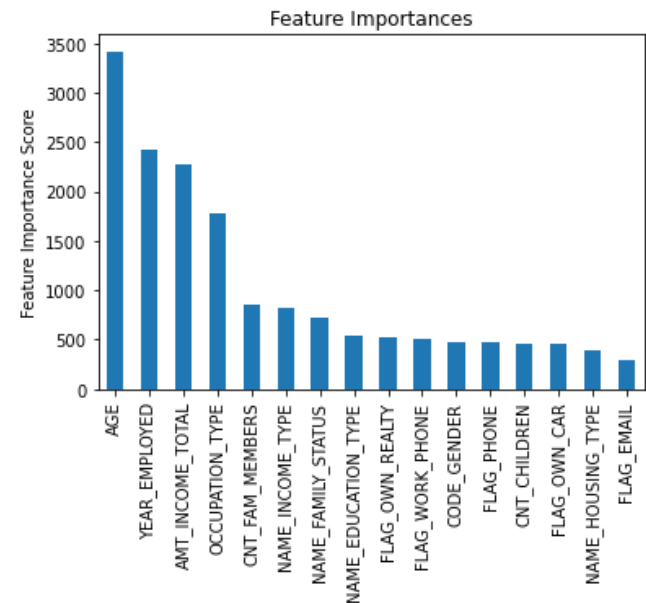
## 5.1 Tune XG Boost with XGB.cv

Hyper-parameters to tune:

- **Min\_child\_weight** - Defines the minimum sum of weights of all observations required in a child.
- **Max\_depth** - The maximum depth of a tree.
- **Subsample** - Denotes the fraction of observations to be randomly samples for each tree.
- **Colsample\_bytree** - Denotes the fraction of columns to be randomly samples for each tree.
- **ETA** - Parameter controls the learning rate. Makes the model more robust by shrinking the weights on each step.

### RANDOM FOREST BEFORE/AFTER TABLE

```
=====
Test Method:   Before  After
Accuracy       0.851   0.854
F1 Score       0.840   0.843
AUC            0.915   0.905
Improvement of 0.00%.
```



- All features have contribution to our target.
- No need to remove any feature to improve the model

# Machine Learning Process

1. Data Preparation

2. Data Exploration

3. Features Engineering

4. Model Selection

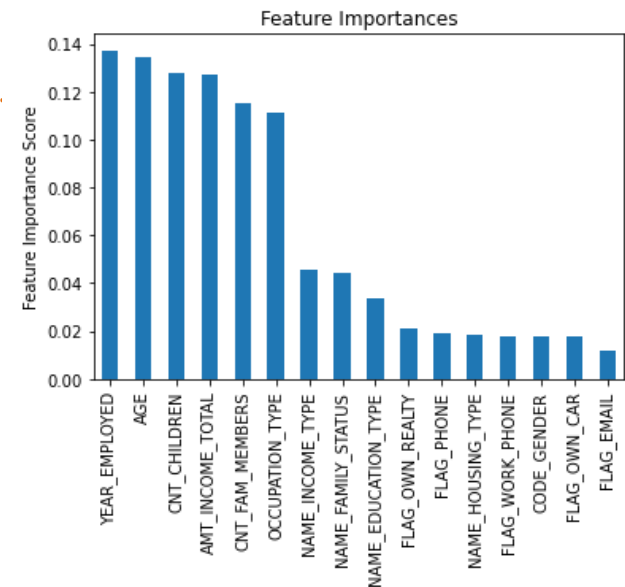
5. Hyper-Parameter Tuning

## 5.2 Tune Random Forest with RandomSearch.cv & GridSearch.cv

We will use Randomsearch.cv to get the parameter range. Then, we will use Gridsearch.cv with cross validation to further select the best parameter.

Hyper-parameters to tune:

- **N\_estimators** - Number of trees in random forest
- **Max\_features** - Number of features to consider at every split
- **Max\_depth** - Maximum number of levels in tree
- **Min\_samples\_split** - Minimum number of samples required to split a node
- **Min\_samples\_leaf** - Minimum number of samples required at each leaf node
- **Bootstrap** - Method of selecting samples for training each tree



- All features have contribution to our target.
- No need to remove any feature to improve the model

### XGBOOST BEFORE/AFTER TABLE

Test Method:	Before	After
Accuracy	0.891	0.891
F1 Score	0.883	0.886
AUC	0.929	0.935

Improvement of 0.48%.

# Machine Learning Process

1. Data  
Preparation

2. Data  
Exploration

3. Features  
Engineering

4. Model  
Selection

5. Hyper-  
Parameter Tuning

## 5.3 Conclusion

- The best model to identify good/bad applicant for credit card approval will be XG Boost model with test of ~89% accuracy.
- We will be using XG Boost to predict our values.
- For future work, we could look into other value range for parameter tuning and we can also look into using ensemble of XG Boost and Random Forest to get a more accurate prediction Hyper-parameters to tune

### RANDOM FOREST BEFORE/AFTER TABLE

=====

Test Method:	RF	XGB
Accuracy	0.854	0.891
F1 Score	0.843	0.886
AUC	0.905	0.935



Thank You

<https://github.com/jiayang243/CreditCardApproval>