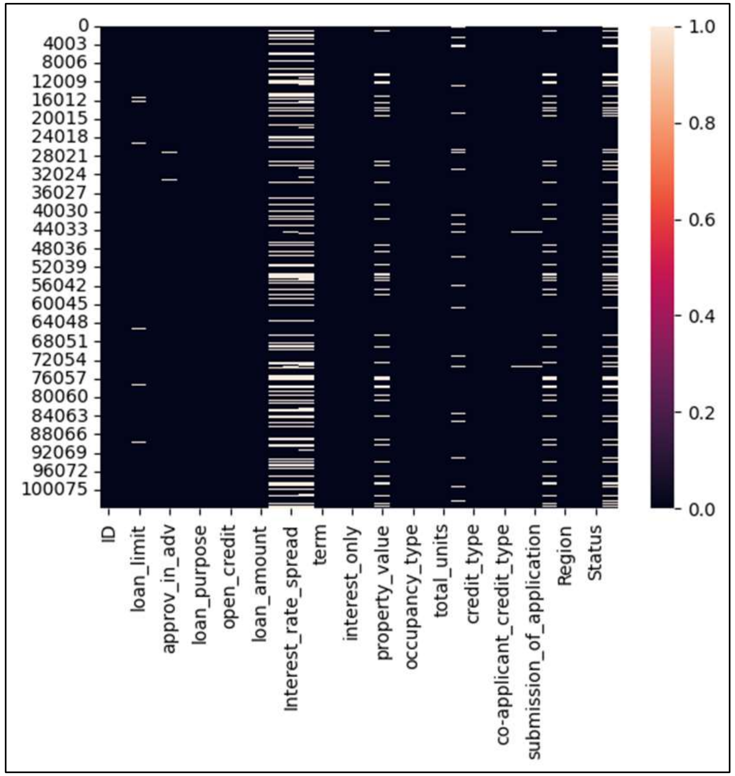
Sha’s ppt

Project Final Presentation

Initial Review of Loan Default dataset

- Supervised binary classification problem.



Unique values :		Missing Values	Percentage (%)
ID	104069	0	0.000000
year	1	0	0.000000
loan_limit	2	2409	2.314810
Gender	4	0	0.000000
approv_in_adv	2	645	0.619781
loan_type	3	0	0.000000
loan_purpose	4	99	0.095129
Credit_Worthiness	2	0	0.000000
open_credit	2	0	0.000000
business_or_commercial	2	0	0.000000
loan_amount	205	0	0.000000
rate_of_interest	124	25504	24.506818
Interest_rate_spread	20879	25647	24.644226
Upfront_charges	42684	27756	26.670767
term	24	29	0.027866
Neg_ammortization	2	82	0.078794
interest_only	2	0	0.000000
lump_sum_payment	2	0	0.000000
property_value	362	10579	10.165371
construction_type	2	0	0.000000
occupancy_type	3	0	0.000000
Secured_by	2	0	0.000000
total_units	4	0	0.000000
income	910	6350	6.101721
credit_type	4	0	0.000000
Credit_Score	401	0	0.000000
co-applicant_credit_type	2	0	0.000000
age	7	143	0.137409
submission_of_application	2	143	0.137409
LTV	7342	10579	10.165371
Region	4	0	0.000000
Security_Type	2	0	0.000000
Status	2	0	0.000000
dtir1	57	16841	16.182533

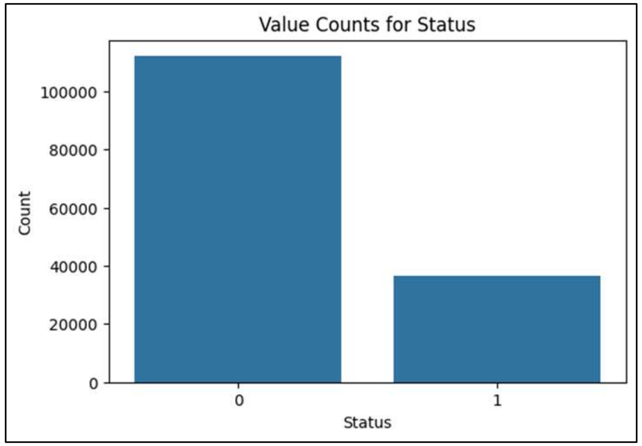
```
Rows      :  148670
Columns   :  34
Missing values :  181135
```

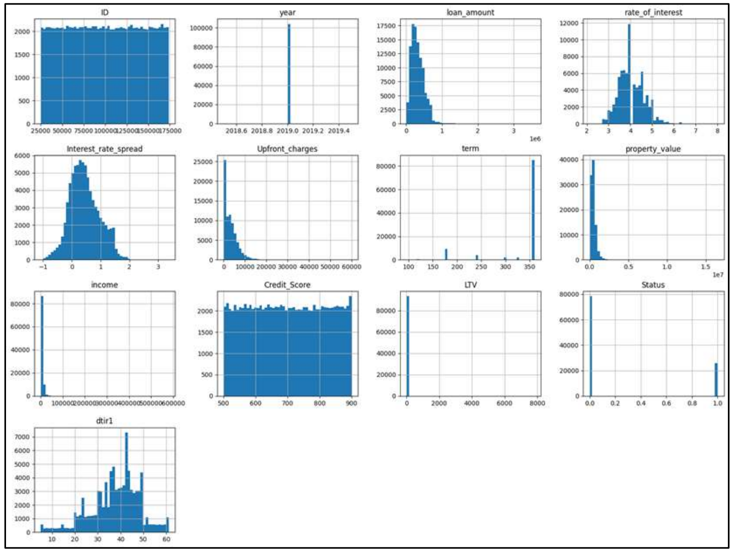
Status	Count	Percentage
0	112031	75.355485
1	36639	24.644515

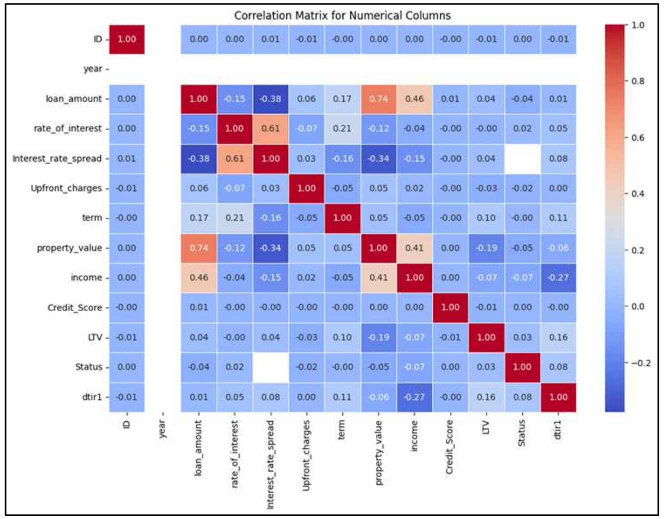
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 148670 entries, 0 to 148669
Data columns (total 34 columns):
 #   Column                     Non-Null Count   Dtype
---  ------                     --------------   -----
 0   ID                         148670 non-null  int64
 1   year                       148670 non-null  int64
 2   loan_limit                 145326 non-null  object
 3   Gender                     148670 non-null  object
 4   approv_in_adv              147762 non-null  object
 5   loan_type                  148670 non-null  object
 6   loan_purpose               148536 non-null  object
 7   Credit_Worthiness          148670 non-null  object
 8   open_credit                148670 non-null  object
 9   business_or_commercial     148670 non-null  object
 10  loan_amount                148670 non-null  int64
 11  rate_of_interest           112231 non-null  float64
 12  Interest_rate_spread       112031 non-null  float64
 13  Upfront_charges            109028 non-null  float64
 14  term                       148629 non-null  float64
 15  Neg_ammortization          148549 non-null  object
 16  interest_only              148670 non-null  object
 17  lump_sum_payment           148670 non-null  object
 18  property_value             133572 non-null  float64
 19  construction_type          148670 non-null  object
 20  occupancy_type             148670 non-null  object
 21  Secured_by                 148670 non-null  object
 22  total_units                148670 non-null  object
 23  income                     139520 non-null  float64
 24  credit_type                148670 non-null  object
 25  Credit_Score               148670 non-null  int64
 26  co-applicant_credit_type   148670 non-null  object
 27  age                        148470 non-null  object
 28  submission_of_application  148470 non-null  object
 29  LTV                        133572 non-null  float64
 30  Region                     148670 non-null  object
 31  Security_Type              148670 non-null  object
 32  Status                     148670 non-null  int64
 33  dtir1                      124549 non-null  float64
dtypes: float64(8), int64(5), object(21)
memory usage: 38.6+ MB
```

Initial Review of Loan Default dataset

- Supervised binary classification problem.







1.4 Split Train/Test with 70:30

Technique: Stratified Sampling with "Status" column as target label

(Total)

Status	count
0	112031
1	36639

(Train)

Status	count
0	78422
1	25647

(Test)

Status	count
0	33609
1	10992

Preprocessing Steps Taken

Data Cleaning

1. Replace null values for justifiable columns (column: rate of interest, interest rate spread, upfront charges, LTV, dtir1).
2. Drop columns for non-useful features (column: ID, year).
3. Detect & remove outliers more than value 100 (column: LTV).
4. General detection & removal of outliers for whole dataset using Z-score method

Data Transformation

1. Categorical missing value imputation using most frequent observations (column: loan limit, approv in adv, loan purpose, neg ammortization, age, submission of application).
2. Numerical missing value imputation using k-NN imputer (column: term, property value, income).

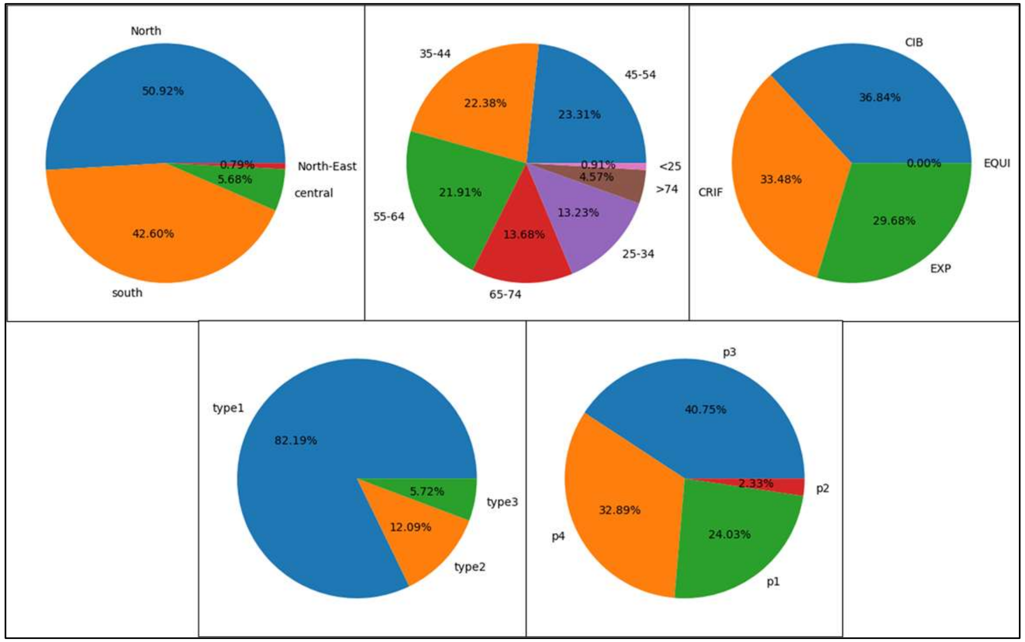
Preparation for ML

1. Drop columns with high correlation (using V Cramer's) for Categorical Features (column: co-applicant credit type, submission of application, security type).
2. Drop columns with high correlation for Numerical Features (column: rate of interest, upfront charges).
3. Combine low frequency categorical values for affected categorical columns (column: loan type, loan purpose, occupancy type, total units, age, region).
4. Use LabelEncoder for columns assumed to be of Ordinal Data (column: credit worthiness, total units, age).
5. Use One-hot Encoder for columns assumed to be of Nominal Data (column: the rest of object datatype columns)
6. Apply MinMaxScaler to standardize range from 0 to 1 (column: the rest of columns except those that uses LabelEncoder method)

Exploratory Data Analysis (Train dataset)

<u>**Univariate Analysis**</u>

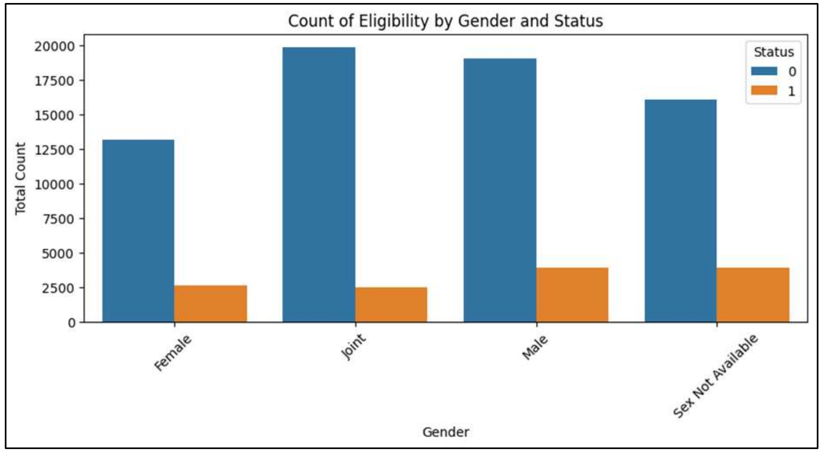
- Region: Most clients resides at North & South.
- Age: High percentage of clients taking loan between 35-64 years old.
- Credit Type: Most client are assessed based on Credit Information Bureau, CRIF Credit Bureau or Experian standards.
- Loan Type: Most client took up type1 loans.
- Loan Purpose: Less client has p2 purpose while rest of the purpose are generally well distributed.

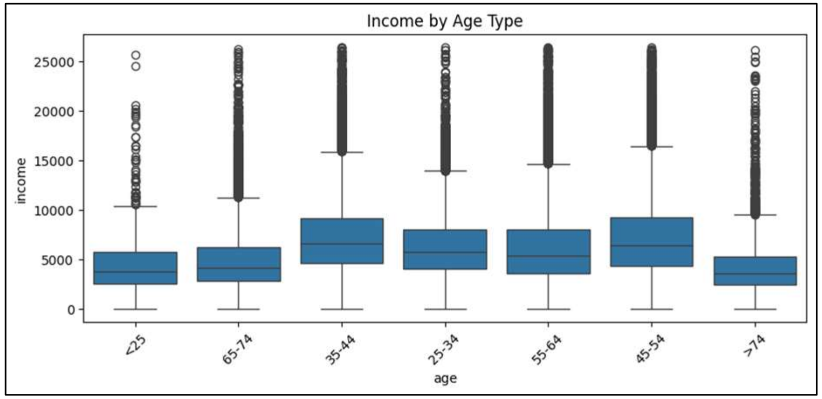


Exploratory Data Analysis (Train dataset)

Bivariate Analysis

- Most transactions have not been defaulted on. But those who are male or undisclosed tend to default more.
- Peak income group are usually around the age range of 35 to 54.

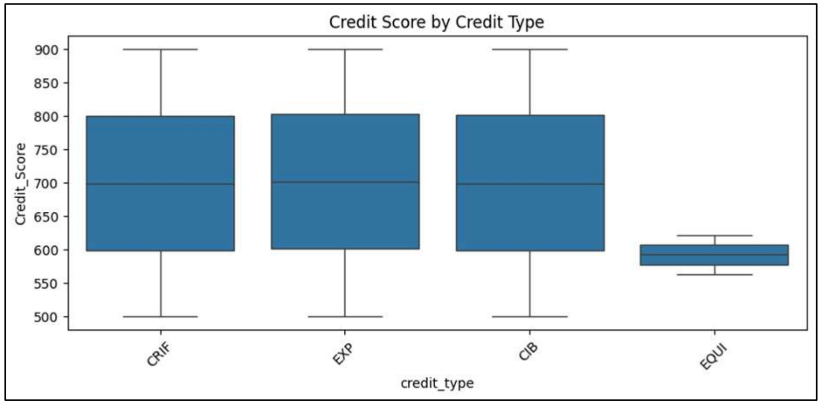


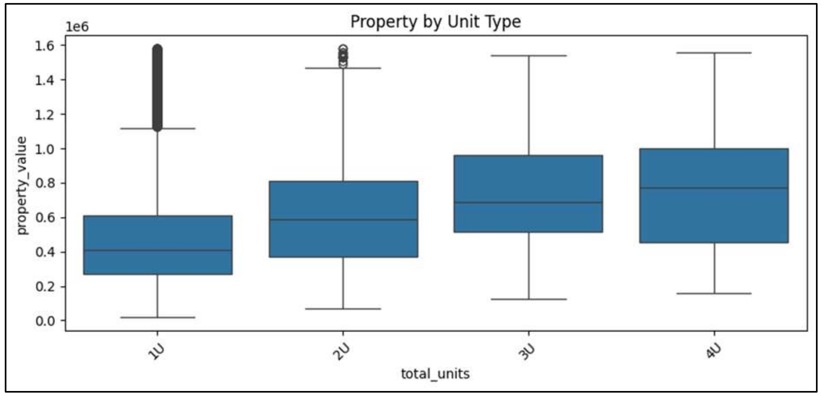


Exploratory Data Analysis (Train dataset)

Bivariate Analysis

- Those with Equifax credit type have better scoring than others. While credit type from credit information bureau, CRIF credit information bureau, Experian generally has similar credit score.
- The bigger the property (more units), the higher the property value.

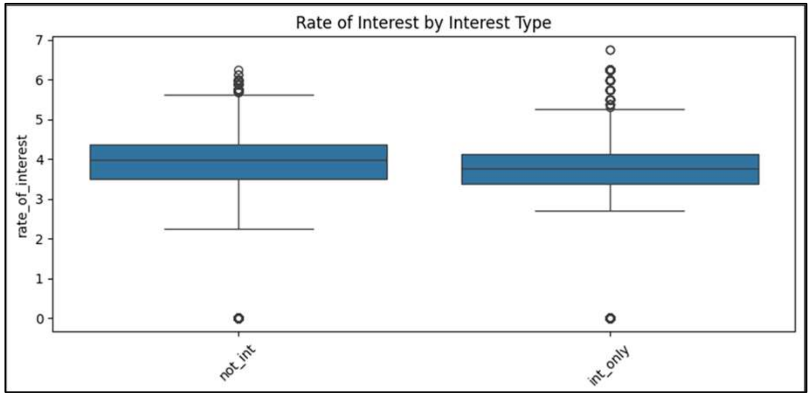


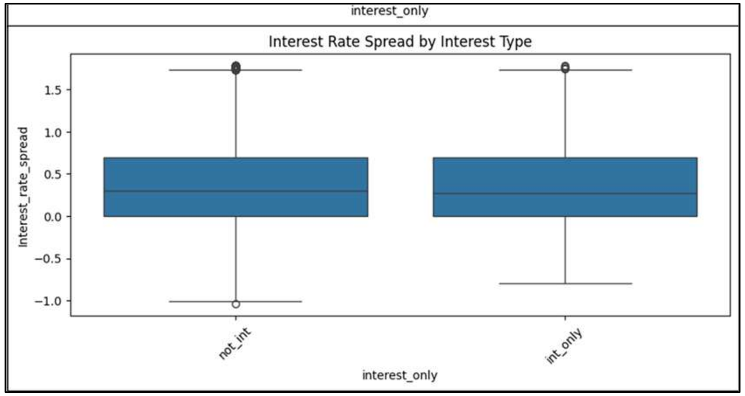


Exploratory Data Analysis (Train dataset)

Bivariate Analysis

- The rate of interest gap doesn't differ much between interest types.
- And the interest rate spread is generally the same.

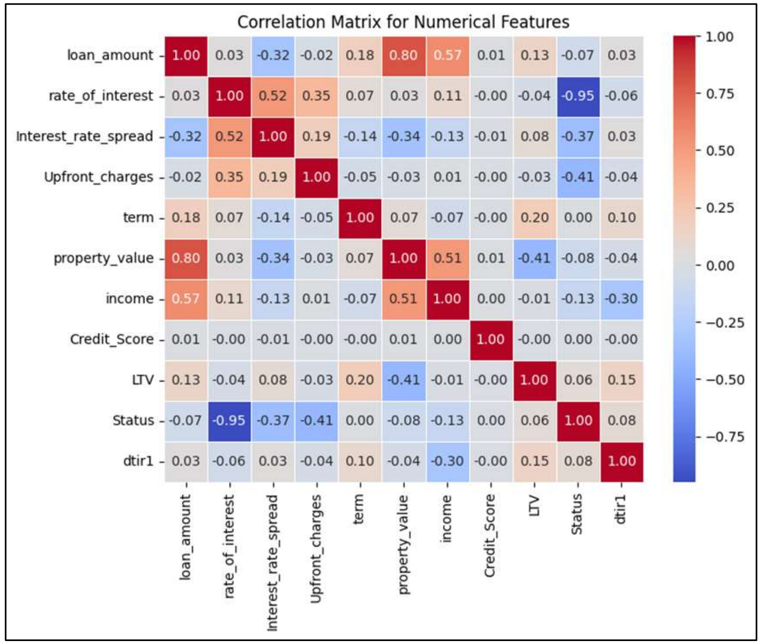


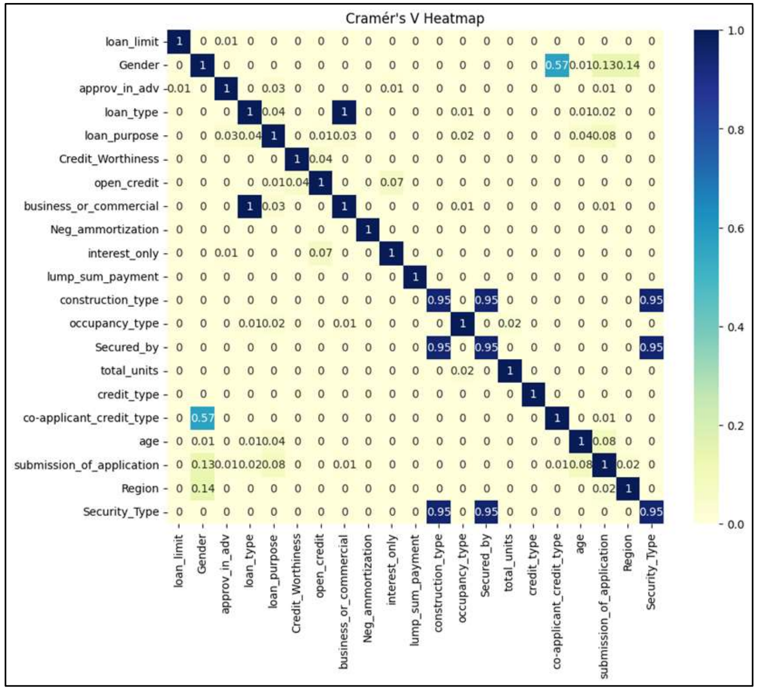


Exploratory Data Analysis (Train dataset)

<u>Bivariate Analysis</u>

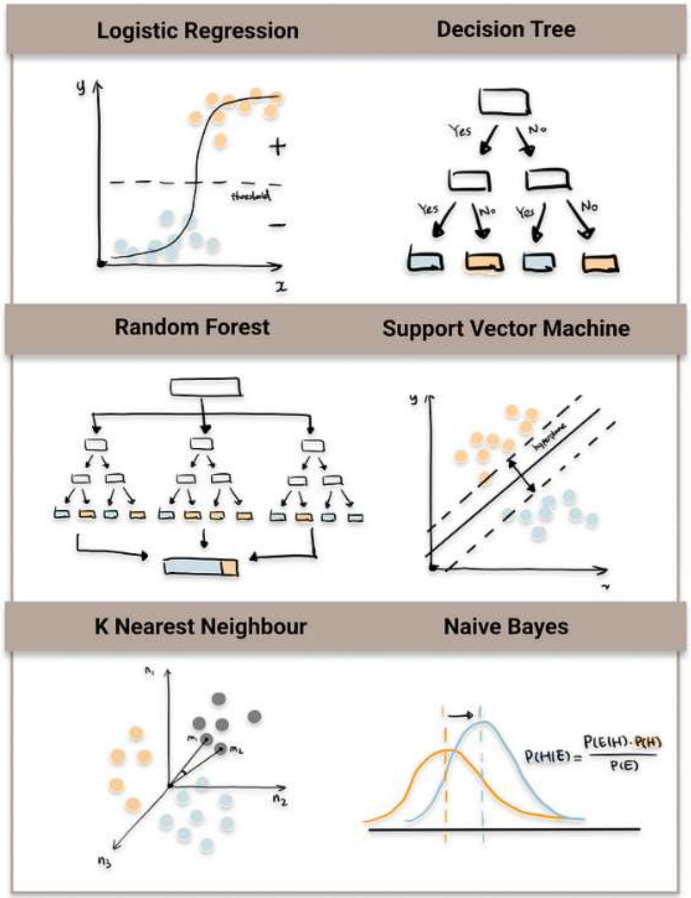
- V Cramer's: Drop high correlated categorical columns (co-applicant_credit_type | submission_of_application | Security_Type)
- Correlation Matrix: Drop high correlated numerical columns (rate_of_interest | Upfront_charges)

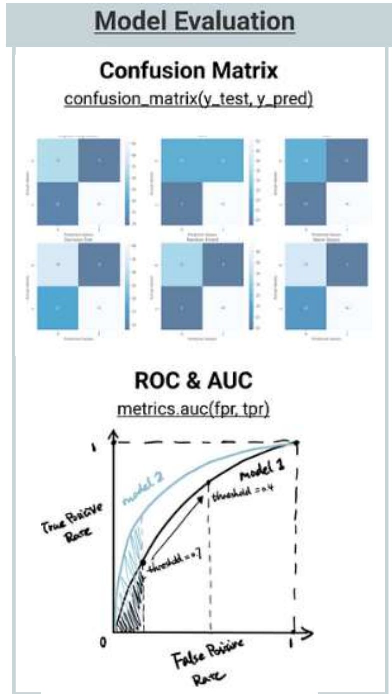




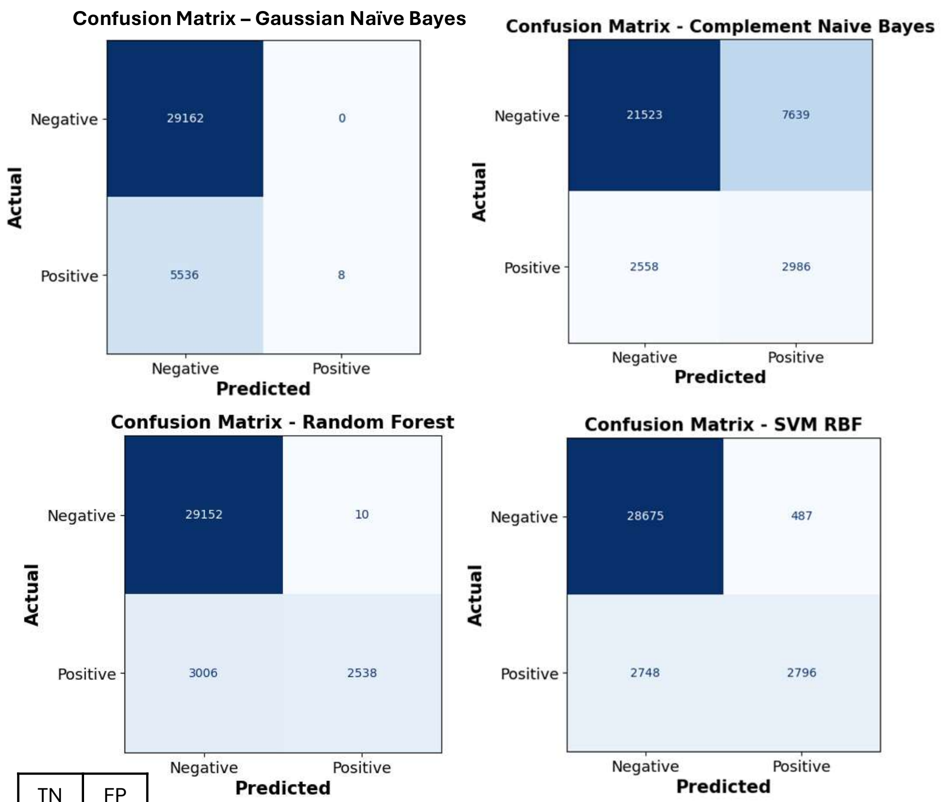
Chosen Machine Learning Algorithms

- Naïve Bayes
  - Gaussian
  - Complement (Multinominal)
    - particularly for imbalance dataset
- Random Forest
- Support Vector Machine (SVM)
  - Kernel: Radial Basis Function (RBF)
- Voting Classifier
  - Random Forest & SVM





Initial Base Models Performance

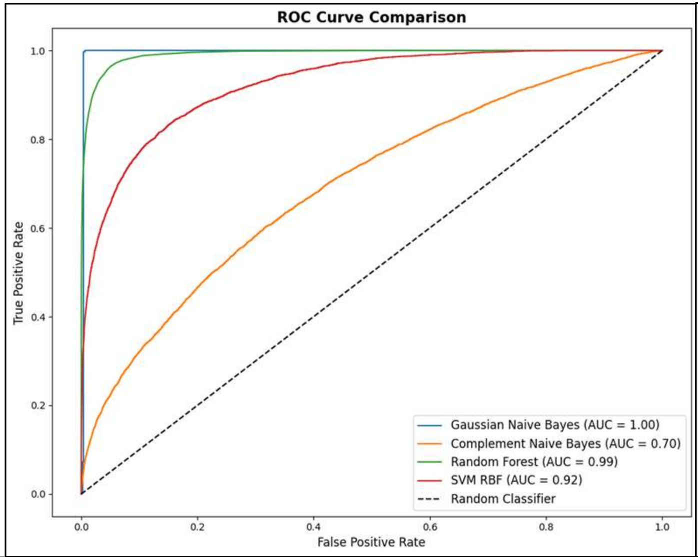


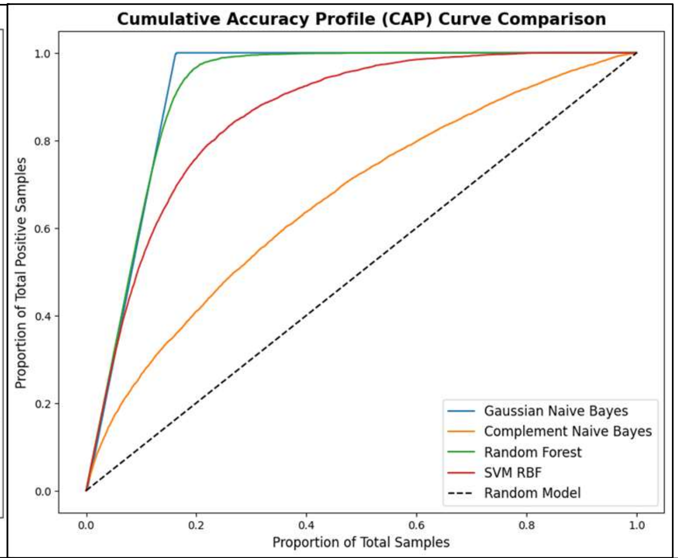
TN	FP
FN	TP

S/N	Model	Accuracy (TP+TN)/ALL	Precision (TP/TP+FP)	Recall (TP/TP+TN)	F1-score
1	Gaussian NB	0.840489	1.0	0.001443	0.002882
2	Complement NB	0.706189	0.281035	0.5386	0.369349
3	Random Forest	0.913099	0.996075	0.457792	0.627286
4	SVM (RBF kernel)	0.906788	0.85166	0.504329	0.633511

Gaussian NB – Cross Validation Scores:					**Mean**
0.8515	0.8545	0.8527	0.8534	0.8527	**0.8530**
Complement NB – Cross Validation Scores:					**Mean**
0.7070	0.7068	0.7101	0.7067	0.7072	**0.7076**
Random Forest – Cross Validation Scores:					**Mean**
0.9989	0.9978	0.9980	0.9984	0.9984	**0.9983**
SVM RBF – Cross Validation Scores:					**Mean**
(skip)	(skip)	(skip)	(skip)	(skip)	**(skip)**

Initial Base Models Performance





Combined Metrics Base Models:

	Model	Accuracy	Precision	Recall	F1-Score
0	Gaussian Naive Bayes	0.840489	1.000000	0.001443	0.002882
1	Complement Naive Bayes	0.706189	0.281035	0.538600	0.369349
2	Random Forest	0.913099	0.996075	0.457792	0.627286
3	SVM RBF	0.906788	0.851660	0.504329	0.633511

Strategies to Overcome Misclassification

Steps for error analysis & subsequent model improvements (specifically, to increase TP & TN):

- Error analysis with Confusion Matrix
  - Indicates no. of FPs & FNs for reduction to improve the model
- Cross-Validation
  - Ensured model generalises well & is not overfitted to particular subset of data.
- Alternative Models
  - Experiment with other models that might capture different relationships in the data to provide better performance.
- Class Imbalance Handling
  - Consider oversampling minority class or undersampling majority class. (can improve recall for minority class)
- Hyperparameter Tuning
  - Consider Grid Search/Random Search to find optimal parameters for model (improve performance for complex models)
- Inspect Misclassified Cases (preliminary only)
  - Review those close to decision boundary. Understand why these cases were incorrectly classified for improvement insights.
- Threshold Tuning (not covered)
  - Adjust decision threshold either to reduce FNs or FPs using ROC curve to max True Positive Rate (Recall) while keep False Positive Rate low.
- Feature Importance Analysis (not covered)
  - Consider Feature Engineering (e.g. create new features/combine existing ones/adding intersection terms/polynomial features) or analyse feature weights.

1. Feature Engineering – Feature Selection

Business Reference:

- https://fastercapital.com/topics/factors-affecting-loan-default.html
- https://fastercapital.com/keyword/loan-amortization.html

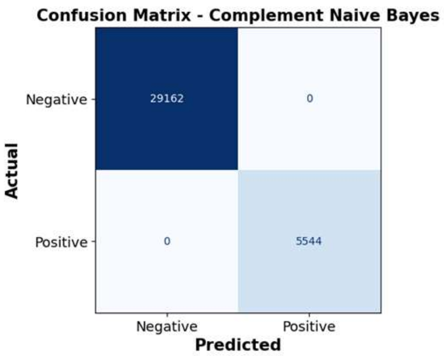
Note: Reduction from original dataset features with a more focused set of features. Other preprocessing steps are the same from initial.

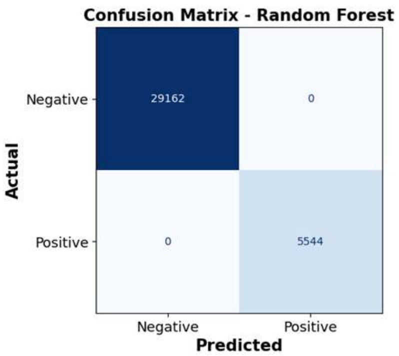
```
(34706, 32)
Index(['loan_limit', 'Gender', 'approv_in_adv', 'loan_type', 'loan_purpose',
       'Credit_Worthiness', 'open_credit', 'business_or_commercial',
       'loan_amount', 'rate_of_interest', 'Interest_rate_spread',
       'Upfront_charges', 'term', 'Neg_ammortization', 'interest_only',
       'lump_sum_payment', 'property_value', 'construction_type',
       'occupancy_type', 'Secured_by', 'total_units', 'income', 'credit_type',
       'Credit_Score', 'co-applicant_credit_type', 'age',
       'submission_of_application', 'LTV', 'Region', 'Security_Type', 'Status',
       'dtir1'],
```

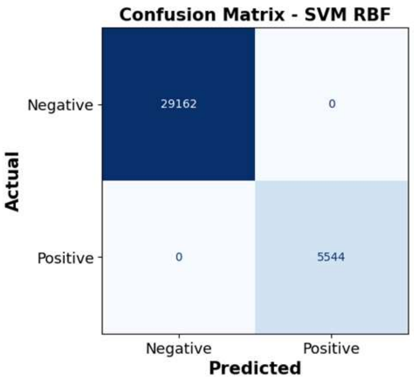
```
# Drop columns based on highly correlated values from Numerical & Categorical correlation analysis
columns_to_drop = ['loan_limit', 'Gender', 'approv_in_adv',
                   'loan_purpose', 'open_credit', 'Interest_rate_spread',
                   'interest_only', 'construction_type', 'occupancy_type',
                   'total_units', 'co-applicant_credit_type', 'age',
                   'submission_of_application', 'Region']
df = df.drop(columns=columns_to_drop)
```

```
(34706, 18)
Index(['loan_type', 'Credit_Worthiness', 'business_or_commercial',
       'loan_amount', 'rate_of_interest', 'Upfront_charges', 'term',
       'Neg_ammortization', 'lump_sum_payment', 'property_value', 'Secured_by',
       'income', 'credit_type', 'Credit_Score', 'LTV', 'Security_Type',
       'Status', 'dtir1'],
      dtype='object')
```

1. Feature Engineering – Feature Selection





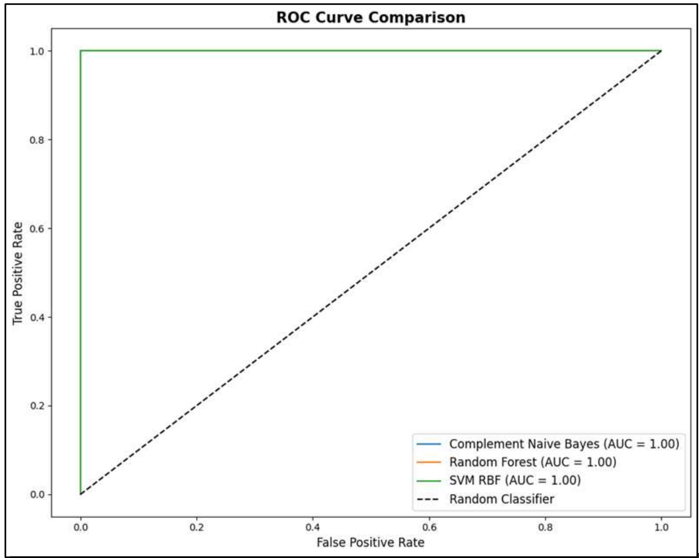


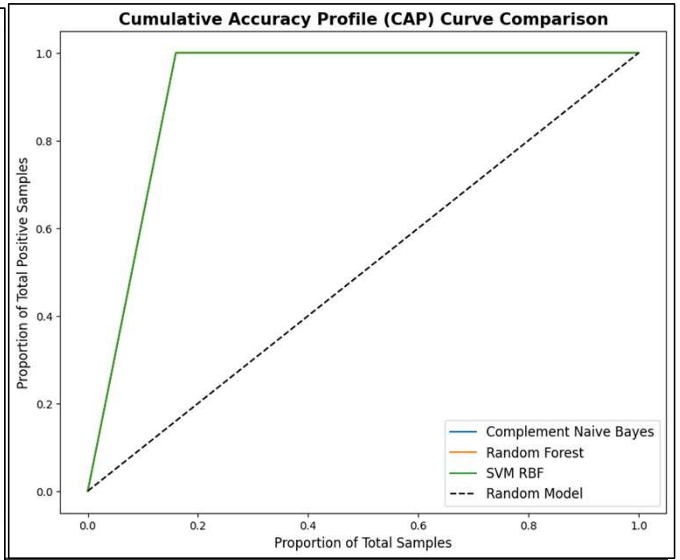
TN	FP
FN	TP

S/N	Model	Accuracy (TP+TN)/ALL	Precision (TP/TP+FP)	Recall (TP/TP+TN)	F1-score
1	Gaussian NB	NA	NA	NA	NA
2	Complement NB	1.0	1.0	1.0	1.0
3	Random Forest	1.0	1.0	1.0	1.0
4	SVM (RBF kernel)	1.0	1.0	1.0	1.0

Gaussian NB – Cross Validation Scores:					**Mean**
NA	NA	NA	NA	NA	**NA**
Complement NB – Cross Validation Scores:					**Mean**
1.	1.	1.	1.	1.	1.0000
Random Forest – Cross Validation Scores:					**Mean**
1.	1.	1.	1.	1.	1.000
SVM RBF – Cross Validation Scores:					**Mean**
(skip)	(skip)	(skip)	(skip)	(skip)	**(skip)**

1. Feature Engineering – Feature Selection





Combined Metrics Base Models:

	Model	Accuracy	Precision	Recall	F1-Score
0	Complement Naive Bayes	1.0	1.0	1.0	1.0
1	Random Forest	1.0	1.0	1.0	1.0
2	SVM RBF	1.0	1.0	1.0	1.0

2. Oversampling – SMOTE

Note: SMOTE is only applied to Train dataset while Test dataset remained untouched.

```python
# Create DataFrame for counts & percentages

# Calculating value counts and percentages for 'Status'
status_count = MLtest['Status'].value_counts()
status_percent = MLtest['Status'].value_counts(normalize=True) * 100

# Creating a DataFrame
status_summary = pd.DataFrame({
    'Count': status_count,
    'Percentage': status_percent
})

# Display the DataFrame
print(status_summary)
```

```
        Count  Percentage
Status
0.0     29162   84.025817
1.0      5544   15.974183
```

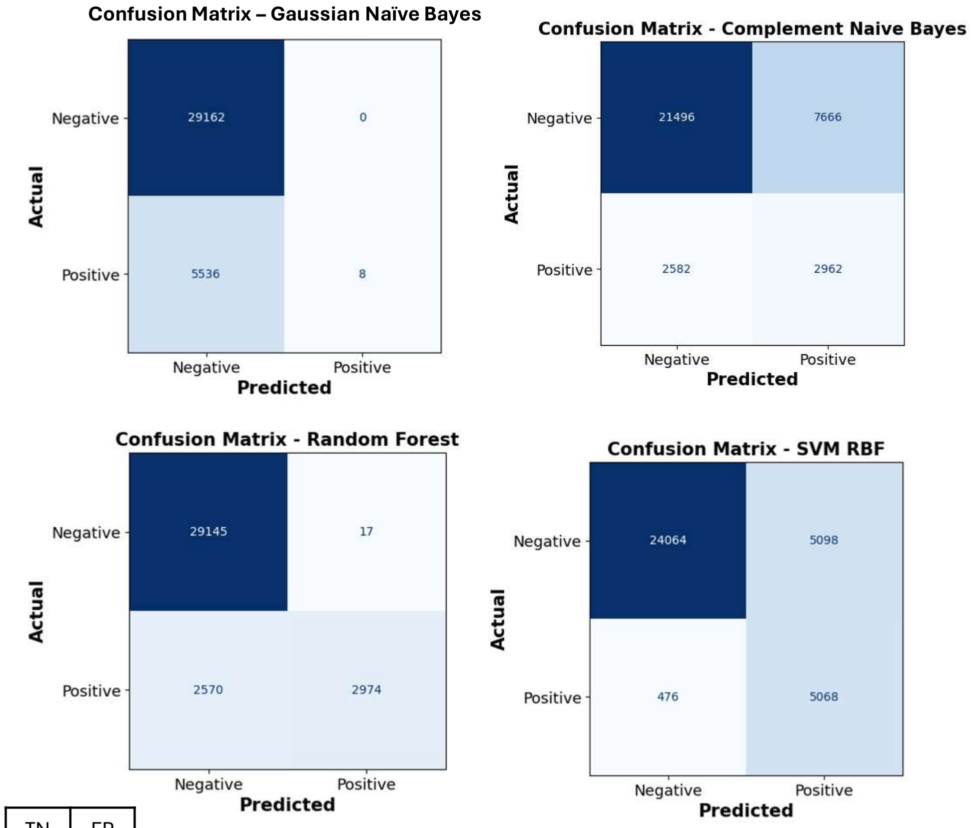
```python
# Create DataFrame for counts & percentages

# Calculating value counts and percentages for 'Status'
status_count = MLtrain['Status'].value_counts()
status_percent = MLtrain['Status'].value_counts(normalize=True) * 100

# Creating a DataFrame
status_summary = pd.DataFrame({
    'Count': status_count,
    'Percentage': status_percent
})

# Display the DataFrame
print(status_summary)
```

```
        Count  Percentage
Status
0.0     68117    84.07119
1.0     12906    15.92881
```

```python
# Create DataFrame for counts & percentages

# Calculating value counts and percentages for 'Status'
status_count = MLtrain_bal['Status'].value_counts()
status_percent = MLtrain_bal['Status'].value_counts(normalize=True) * 100

# Creating a DataFrame
status_summary = pd.DataFrame({
    'Count': status_count,
    'Percentage': status_percent
})

# Display the DataFrame
print(status_summary)
```

```
        Count  Percentage
Status
0.0     68117        50.0
1.0     68117        50.0
```

2. Oversampling – SMOTE

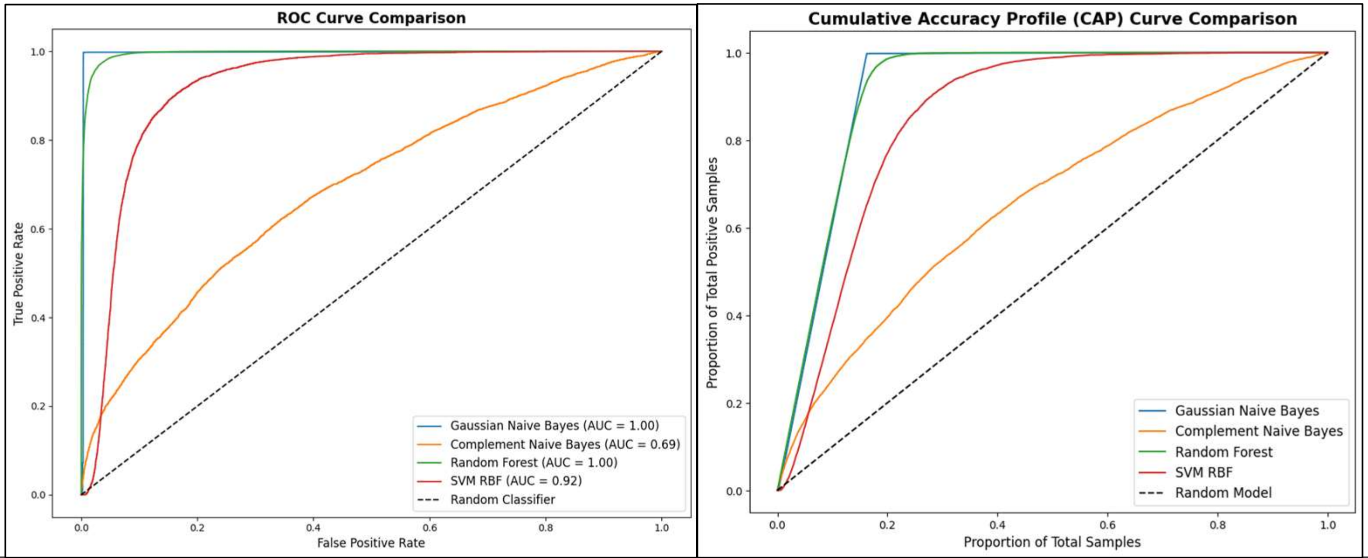


TN	FP
FN	TP

S/N	Model	Accuracy (TP+TN)/ALL	Precision (TP/TP+FP)	Recall (TP/TP+TN)	F1-score
1	Gaussian NB	0.840489	1.000000	0.001443	0.002882
2	Complement NB	0.704720	0.278698	0.534271	0.366312
3	Random Forest	0.925460	0.994316	0.536436	0.696895
4	SVM (RBF kernel)	0.839394	0.498524	0.914141	0.645194
5	Voting (Hard)	0.924393	0.994580	0.529582	0.691149

Gaussian NB – Cross Validation Scores:					**Mean**
0.5986	0.5955	0.5952	0.5956	0.5857	**0.5942**
Complement NB – Cross Validation Scores:					**Mean**
0.6515	0.65730	0.6511	0.6567	0.6497	**0.6533**
Random Forest – Cross Validation Scores:					**Mean**
0.9990	0.9983	0.9985	0.9987	0.9158	**0.9821**
SVM RBF – Cross Validation Scores:					**Mean**
(skip)	(skip)	(skip)	(skip)	(skip)	**(skip)**

2. Oversampling – SMOTE



Combined Metrics SMOTE_Base Models:

	Model	Accuracy	Precision	Recall	F1-Score
0	Gaussian Naive Bayes	0.840489	1.000000	0.001443	0.002882
1	Complement Naive Bayes	0.704720	0.278698	0.534271	0.366312
2	Random Forest	0.925460	0.994316	0.536436	0.696895
3	SVM RBF	0.839394	0.498524	0.914141	0.645194
4	Ensemble Voting	0.924393	0.994580	0.529582	0.691149

3. Hyperparameter Tuning

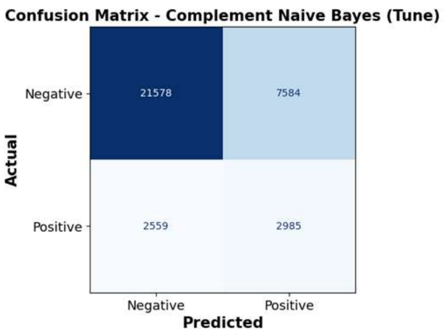
- Complement Naïve Bayes
  - Technique: Grid Search
  - Params:
    alpha = [0.01, 0.1, 0.5, 1.0, 2.0, 5.0, 10.0]
  - Output:
    Fitting 5 folds for each of 7 candidates, totalling 35 fits
    Best parameters: {'alpha': 10.0}
    Best cross-validation score: 0.7095
- Random Forest
  - Technique: Random Search
  - Params:
    n_estimators: randint(25, 50, 100)
    max_depth: [10, 20, 30]
    min_samples_split: randint(2, 5, 10)
    min_samples_leaf: randint(1, 2, 4)
    max_features: ['auto', 'sqrt', 'log2']
  - Output:
    Fitting 5 folds for each of 7 candidates, totalling 35 fits
    Best parameters: {'max_depth': 30, 'max_features': 'sqrt', 'min_samples_leaf': 5, 'min_samples_split': 14, 'n_estimators': 127}
    Best cross-validation score: 0.9990

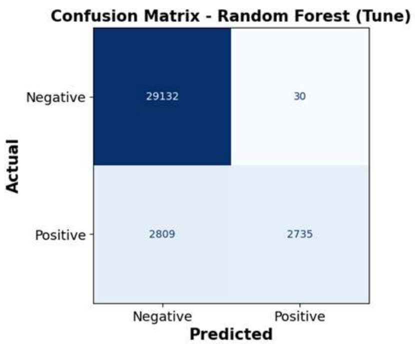
3. Hyperparameter Tuning

```
Combined Metrics Base Models:
                              Model  Accuracy  Precision   Recall  F1-Score
0           Complement Naive Bayes  0.706189   0.281035  0.53860  0.369349
1   Complement Naive Bayes (Tune)  0.707745   0.282430  0.53842  0.370508
```

```
Combined Metrics Base Models:
                  Model  Accuracy  Precision    Recall  F1-Score
0         Random Forest  0.913099   0.996075  0.457792  0.627286
1  Random Forest (Tune)  0.918199   0.989150  0.493326  0.658322
```

S/N	Model	Accuracy (TP+TN)/ALL	Precision (TP/TP+FP)	Recall (TP/TP+TN)	F1-score
1	Gaussian NB	NA	NA	NA	NA
2	Complement NB	0.707745	0.28243	0.53842	0.370508
3	Random Forest	0.918199	0.98915	0.493326	0.658322
4	SVM (RBF kernel)	(skip)	(skip)	(skip)	(skip)

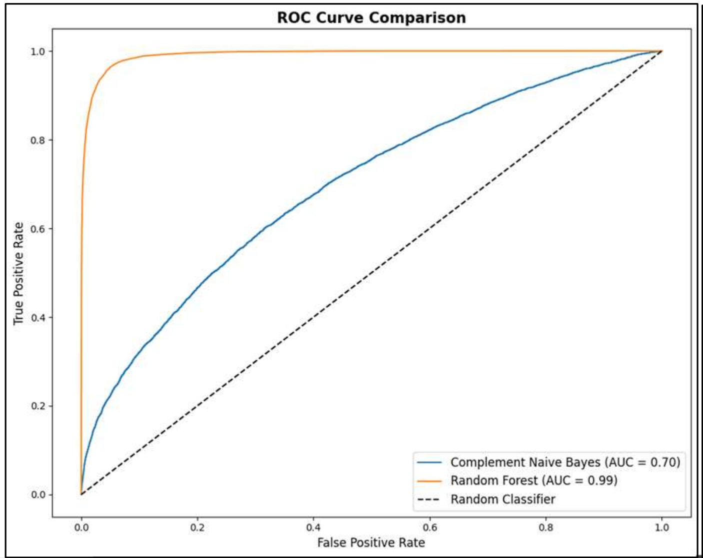


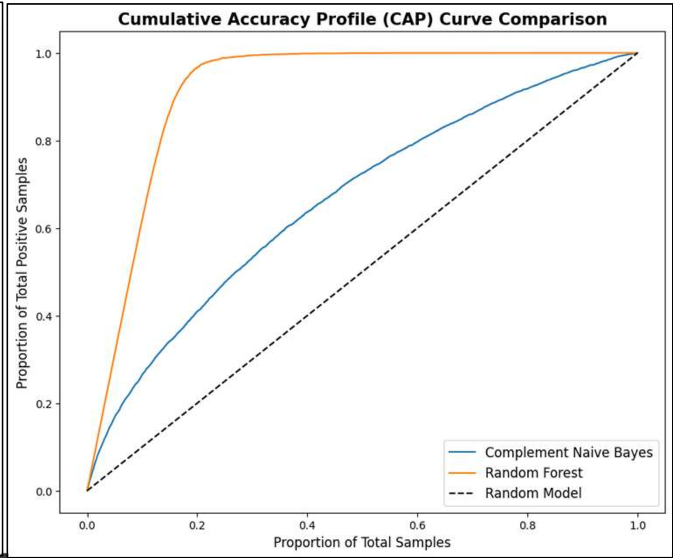


TN	FP
FN	TP

Gaussian NB – Cross Validation Scores:					**Best**
NA	NA	NA	NA	NA	**NA**
Complement NB – Cross Validation Scores:					**Best**
(skip)	(skip)	(skip)	(skip)	(skip)	**0.7095**
Random Forest – Cross Validation Scores:					**Best**
(skip)	(skip)	(skip)	(skip)	(skip)	**0.9990**
SVM RBF – Cross Validation Scores:					**Best**
(skip)	(skip)	(skip)	(skip)	(skip)	**(skip)**

3. Hyperparameter Tuning



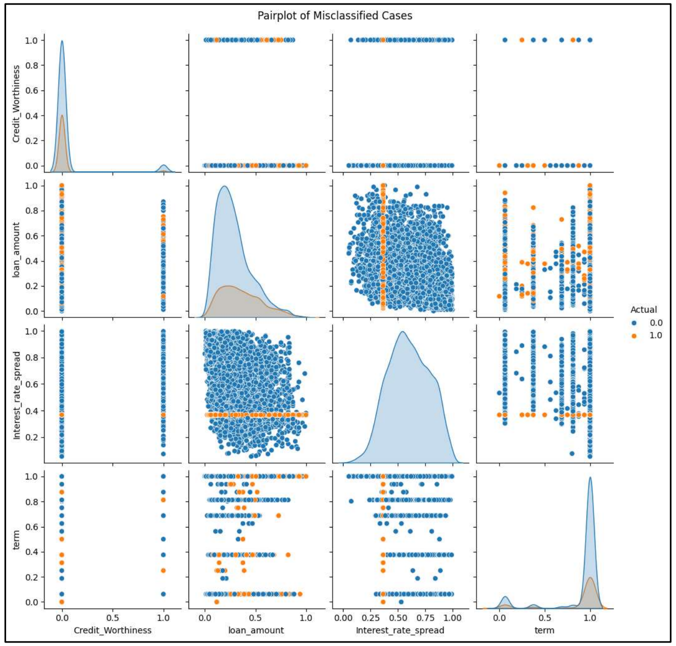


Combined Metrics Base Models:

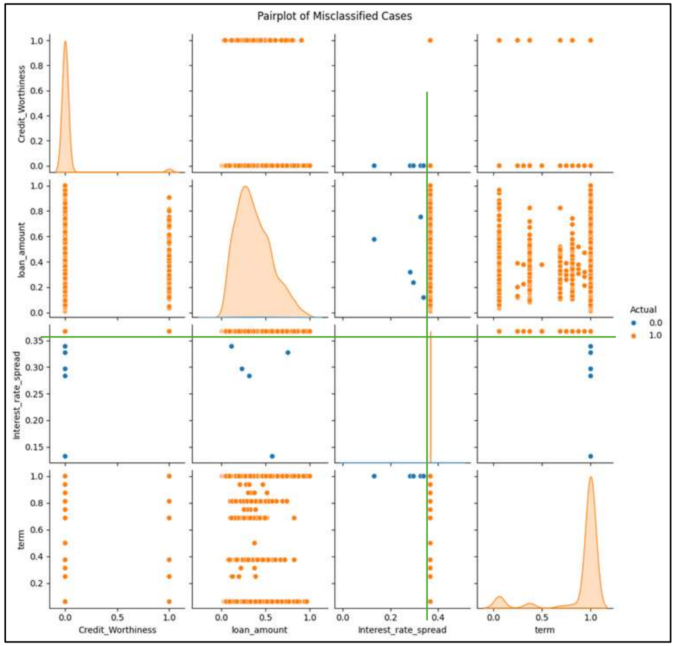
	Model	Accuracy	Precision	Recall	F1-Score
0	Complement Naive Bayes (Tune)	0.707745	0.28243	0.53842	0.370508
1	Random Forest (Tune)	0.918199	0.98915	0.493326	0.658322

4. Error Analysis (Test vs Predicted)

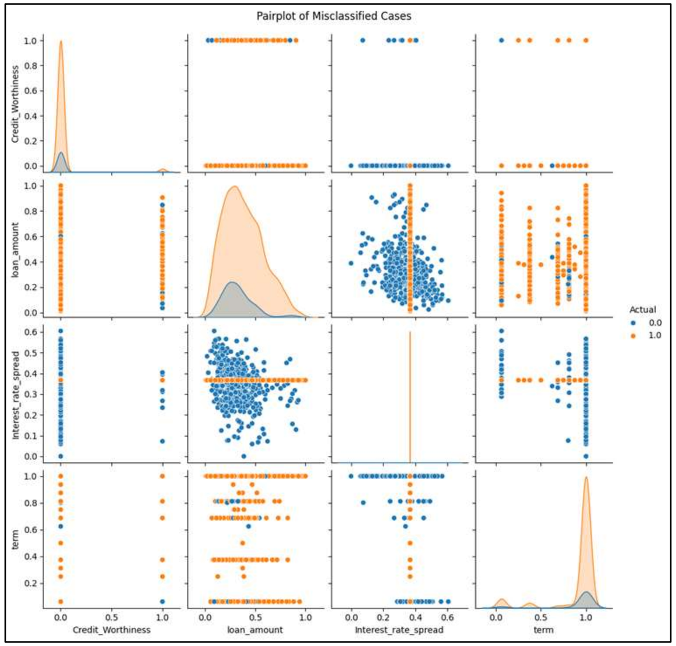
Complement Naïve Bayes



Random Forest



SVM (Kernel: RBF)



Random Forest

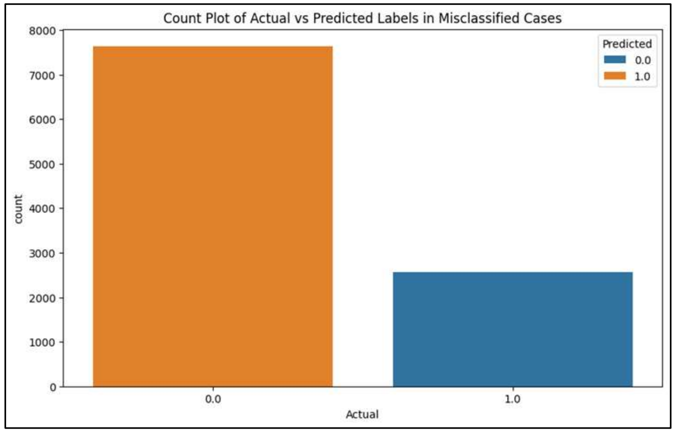
- Potential to use interest_rate_spread to create new feature based on range (green line) to separate between default & non-default.

Complement Naïve Bayes, SVM (Kernel: RBF)

- Current features are difficult to find segregation. Have to explore create new features with e.g. polynomial degree.

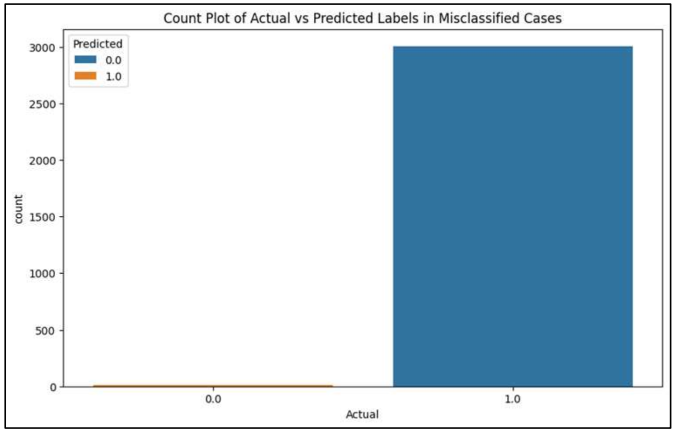
4. Error Analysis (Test vs Predicted)

Complement Naïve Bayes



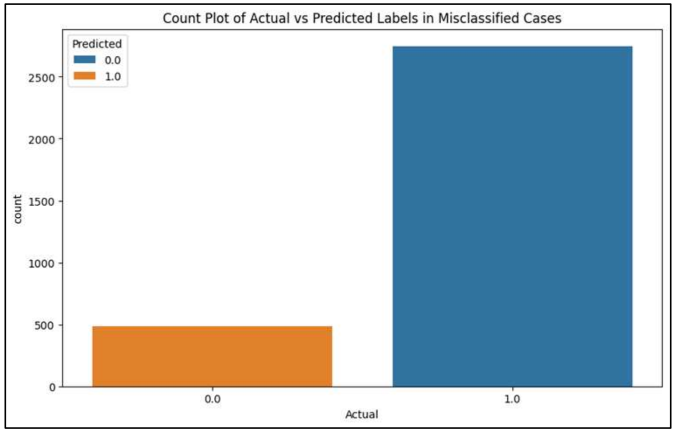
Number of misclassified cases: 10197

Random Forest



Number of misclassified cases: 3016

SVM (Kernel: RBF)



Number of misclassified cases: 3235

Complement Naïve Bayes

- Actual non-default (0) was highly predicted wrongly. About 80% of misclassification.

Random Forest

- Actual default (1) was highly predicted wrongly. Majority of misclassification.

SVM (Kernel: RBF)

- Actual default (1) was highly predicted wrongly. About 83% of misclassification.