Kaggle Competition

Mobina Kargar
dymamsijhidjj@gmail.com

Abstract

In this assignment, you will build and compare a variety of classification models to pre- dict whether an employee will leave the company ('left_company'), using the provided HR dataset.

1 Introduction

There are 3 csv files. One of them is the sample file where we will store the results. And 2 other csv file one of them is because test set and other one is train set. Our target is to make model to learn our model from tree based , SVM, LDR or other method. But before starting we should see the whole data and if necessary remove or handle missing value or outliers or anything else to clean our data.

2 Features

The data is composed of 35 columns and 1341 entries (Full train dataset shape is (1341, 35)). We can see all 35 dimensions of our dataset by printing out the first 3 entries:

Table 1: train dataset (3 rows x 35 columns)

	ID	Unnamed: 0	age_years	travel_freq	...	years_current_role	years_post_promotion	years_with_manager	left_company
0		-1.0420390017173113	-1.7307596774546234	-1.9291271172143192	...	0.8112898580024629	-0.6434148811276065	-0.8962088681587661	0
1		-0.4568738839187711	-1.7281764540554374	0.16557690236819267	...	-0.8684935175734035	-0.3181908496190802	-0.3417377760245537	0

2		0.9475223987977254	-1.7255932306562514	0.16557690236819267	...	1.091253753931774	-0.3181908496190802	0.767204408243871	0

We can inspect the types of feature columns:

Table 2: Data columns:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1341 entries, 0 to 1340
Data columns (total 35 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Unnamed: 0            1341 non-null   int64
 1   age_years             1341 non-null   int64
 2   travel_freq           1341 non-null   object
 3   daily_salary          1341 non-null   int64
 4   work_division         1341 non-null   object
 5   commute_distance      1341 non-null   int64
:
:
:
 29  work_life_score       1341 non-null   int64
 30  tenure_years          1341 non-null   int64
 31  years_current_role    1341 non-null   int64
 32  years_post_promotion  1341 non-null   int64
 33  years_with_manager    1341 non-null   int64
 34  left_company          1341 non-null   int64
dtypes: int64(27), object(8)
memory usage: 366.8+ KB
```

3 Distribution

3.1 Preparing Data:

_ Step 1: Remove duplicate or irrelevant observations

Duplicate columns: []

_ Step 2: Fix structural errors

… Hopefully in this code it doesn't need to use this approach

_ Step 3: Filter unwanted outliers
Detect all rows that have outliers in at least one column and treat them

```
      Unnamed: 0  age_years  daily_salary  commute_distance  education_level  \
44           NaN         59           NaN               NaN              NaN
89           NaN         59           NaN               NaN              NaN
111          NaN         58           NaN               NaN              NaN
313          NaN         58           NaN               NaN              NaN
318          NaN         59           NaN               NaN              NaN
345          NaN         60           NaN               NaN              NaN
583          NaN         59           NaN               NaN              NaN
629          NaN         60           NaN               NaN              NaN
681          NaN         59           NaN               NaN              NaN
702          NaN         58           NaN               NaN              NaN
705          NaN         58           NaN               NaN              NaN
717          NaN         59           NaN               NaN              NaN
814          NaN         60           NaN               NaN              NaN
838          NaN         58           NaN               NaN              NaN
843          NaN         58           NaN               NaN              NaN
946          NaN         59           NaN               NaN              NaN
986          NaN         58           NaN               NaN              NaN
1003         NaN         58           NaN               NaN              NaN
1098         NaN         59           NaN               NaN              NaN
1131         NaN         59           NaN               NaN              NaN
1322         NaN         59           NaN               NaN              NaN

      headcount  env_satisfaction  hourly_wage  job_engagement  \
44          NaN               NaN          NaN             NaN
...
1339                  0.0               2.0          0.0
1340                  2.0               1.0          0.0

[1341 rows x 35 columns]
```

_ Step 4: Handle missing data by inplace them with true or zero

	# 0
Unnamed: 0	0
age_years	0
travel_freq	0
daily_salary	0
work_division	0
commute_dist	0
education_lev	0
degree_field	0
headcount	0
env_satisfactio	0

35 rows x 1 cols 10 per page

3.2 Distribution for Numerical Data


Unnamed: 0
age_years
travel_freq
daily_salary
work_division
commute_distance
education_level
degree_field
headcount
env_satisfaction
sex
hourly_wage
job_engagement
position_level
job_title
job_satisfaction
marital_state
monthly_earnings
monthly_salary
companies_count
is_adult
overtime_status
salary_increase_pct
performance_score
relation_satisfaction
std_work_hours
stock_level
career_years
trainings_last_year
work_life_score
tenure_years
years_current_role
years_post_promotion
years_with_manager
left_company

Figure1: [Grid of histograms] each representing the distribution of values in one of the numeric columns from dataset

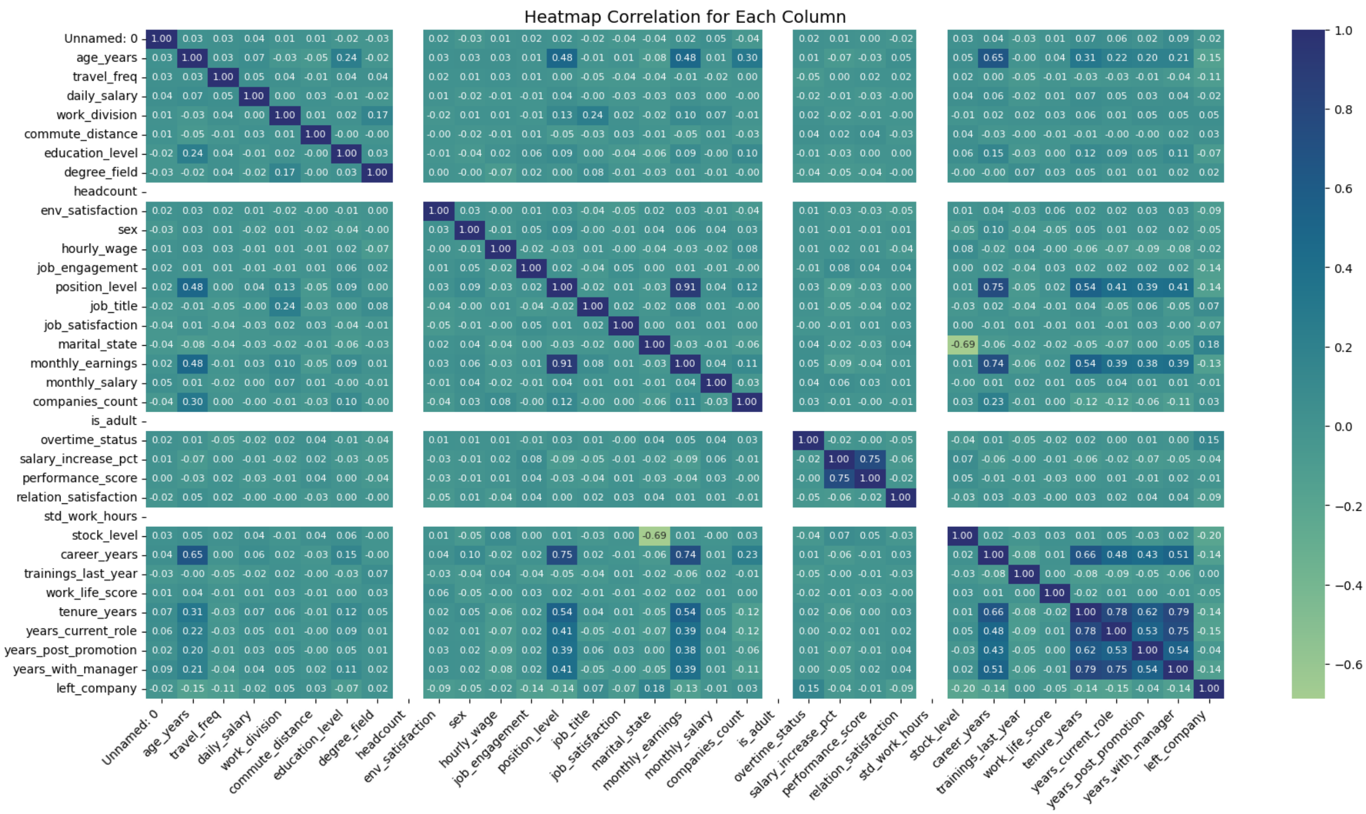
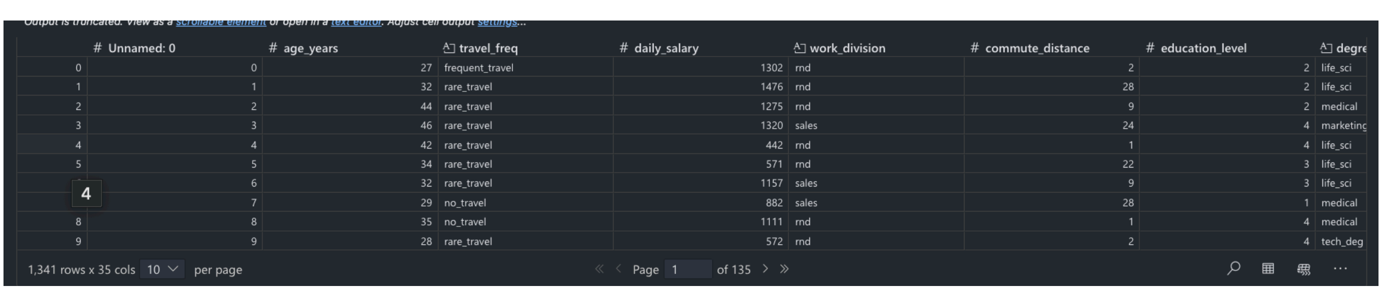


Figure2: [Heatmap] Correlation For Each Column to have more insight about data

look at how this data is distributed.



	# Unnamed: 0	# age_years	travel_freq	# daily_salary	work_division	# commute_distance	# education_level	degre
0	0	27	frequent_travel	1302	rnd	2	2	life_sci
1	1	32	rare_travel	1476	rnd	28	2	life_sci
2	2	44	rare_travel	1275	rnd	9	2	medical
3	3	46	rare_travel	1320	sales	24	4	marketing
4	4	42	rare_travel	442	rnd	1	4	life_sci
5	5	34	rare_travel	571	rnd	22	3	life_sci
[illegible]	6	32	rare_travel	1157	sales	9	3	life_sci
[illegible]	7	29	no_travel	882	sales	28	1	medical
8	8	35	no_travel	1111	rnd	1	4	medical
9	9	28	rare_travel	572	rnd	2	4	tech_deg

1,341 rows x 35 cols 10 per page Page 1 of 135

4 BaseLine Model (Perform Exploratory Analysis with Statistics)

The test file provided is really validation data for competition submission. So, we will use sklearn function to split the training data in two datasets; 75/25 split. This is important, so we don't overfit our model. Meaning, the algorithm is so specific to a given subset, it cannot accurately generalize another subset, from the same dataset.

```
[      Unnamed: 0  age_years  travel_freq  daily_salary  work_division  \
 0      -1.730760  -1.042039    -1.929127      1.085068      -0.642201
 1      -1.728176  -0.456874     0.165577      1.548976      -0.642201
 2      -1.725593   0.947522     0.165577      1.013083      -0.642201
 3      -1.723010   1.181588     0.165577      1.133059       1.288721
 4      -1.720427   0.713456     0.165577     -1.207810      -0.642201
 ...          ...        ...          ...           ...            ...
 1336    1.720427  -0.105775     0.165577     -0.829219      -0.642201
 1337    1.723010  -0.573907     0.165577      0.498518       1.288721
 1338    1.725593   0.011258     2.260281     -0.106695      -0.642201
 1339    1.728176  -1.042039    -1.929127      0.879776       1.288721
 1340    1.730760   1.415654     2.260281      1.229040       1.288721

       commute_distance  education_level  degree_field  headcount  \
 0            -0.865562        -0.884256     -0.803222        0.0
 1             2.494720        -0.884256     -0.803222        0.0
 2             0.039129        -0.884256      0.055042        0.0
 3             1.977753         0.997946      0.913305        0.0
 4            -0.994804         0.997946     -0.803222        0.0
 ...                ...              ...           ...        ...
 1336          0.297612        -0.884256     -0.803222        0.0
 1337         -0.348596         0.997946      2.629833        0.0
 1338         -0.994804         0.997946     -0.803222        0.0
 1339         -0.477837        -0.884256     -0.803222        0.0
 1340         -0.477837        -0.884256      0.913305        0.0
...
 333            0.212733
 334           -0.618973
 335           -1.173444

 [336 rows x 34 columns]]
```

4.1 Feature Engineering

The way we can significantly improve our machine learning model is through feature engineering. Feature engineering is the process of transforming raw data into features that better represent the underlying problem that one is trying to solve. There's no specific way to go about this step, which is what makes data science as much of an art as it as a science. That being said, here are some things that you can consider:
Converting a DateTime variable to extract just the day of the week, the month of the year, etc…
Creating bins or buckets for a variable. (eg. for a height variable, can have 100–149cm, 150–199cm, 200–249cm, etc.)
Combining multiple features and/or values to create a new one. For example, one of the most accurate models for the titanic challenge engineered a new variable called "Is_women_or_child" which was True if the person was a woman or a child and false otherwise.

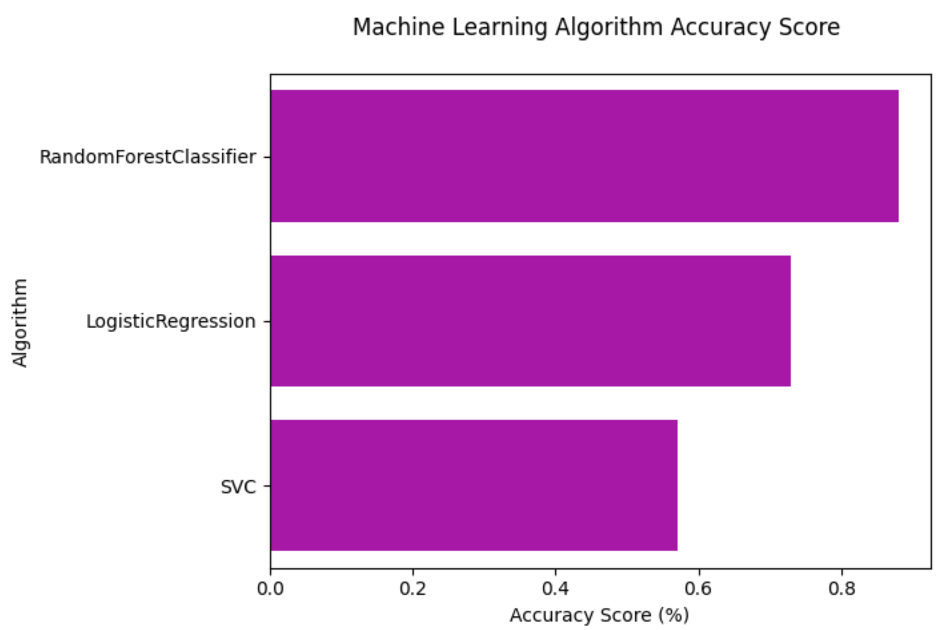
```
position_level       monthly_earnings      0.906910
tenure_years         years_with_manager    0.792115
                     years_current_role    0.775484
years_current_role   years_with_manager    0.754404
position_level       career_years          0.753900
salary_increase_pct  performance_score     0.746755
monthly_earnings     career_years          0.735746
marital_state        stock_level           0.686920
career_years         tenure_years          0.660944
age_years            career_years          0.651054
dtype: float64
```

X_train shape: (1072, 33), X_val shape: (269, 33), X_sample shape: (336, 33)

5 Model Data

the No Free Lunch Theorem (NFLT) of Machine Learning. In short, NFLT states, there is no super algorithm, that works best in all situations, for all datasets. So the best approach is to try multiple MLAs, tune them, and compare them for your specific scenario.

Machine Learning Algorithm (MLA) Selection and Initialization



6 Logistic Regression (Without & With Class Weights)

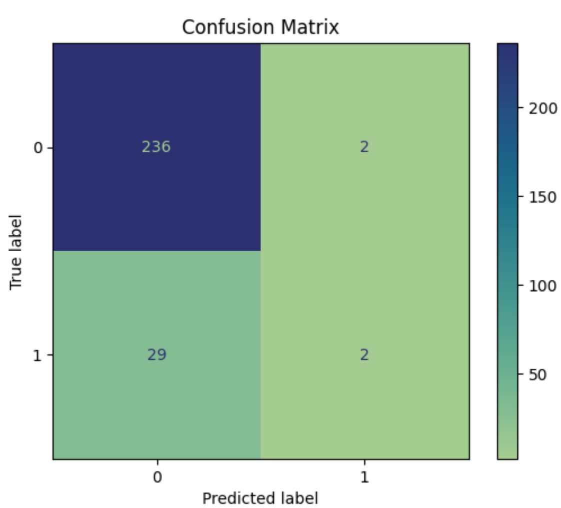
Without Class Weights

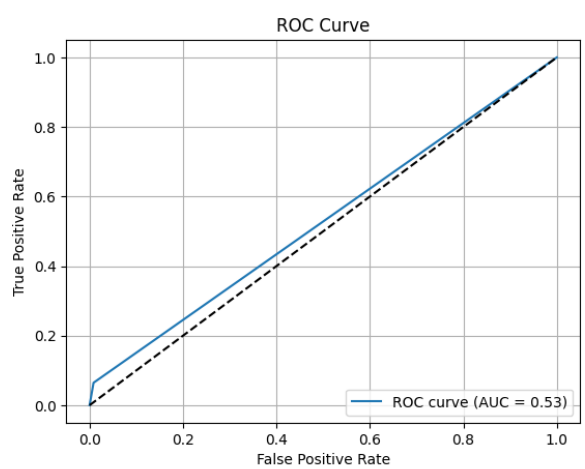
The model's performance is evaluated using various metrics, including precision, recall, f1-score, and support. The accuracy of the model is also reported. The confusion matrix is provided, which shows the number of true positives, true negatives, false positives, and false negatives for the model's predictions. The ROC (Receiver Operating Characteristic) curve is displayed, which is a common way to evaluate the performance of a binary classification model. The area under the ROC curve (AUC) is 0.53, indicating moderate performance. The report mentions that the author used StandardScaler to standardize the features by removing the mean and scaling to unit variance. They also used the fit_transform method to transform the training data and the transform method to scale the validation and sample data.The author states that they used feature selection, but does not provide details on the specific methods used.

[[236   2]

[29   2]]

	precision	recall	f1-score	support
0	0.89	0.99	0.94	238
1	0.50	0.06	0.11	31
accuracy			0.88	269
macro avg	0.70	0.53	0.53	269
weighted avg	0.85	0.88	0.84	269



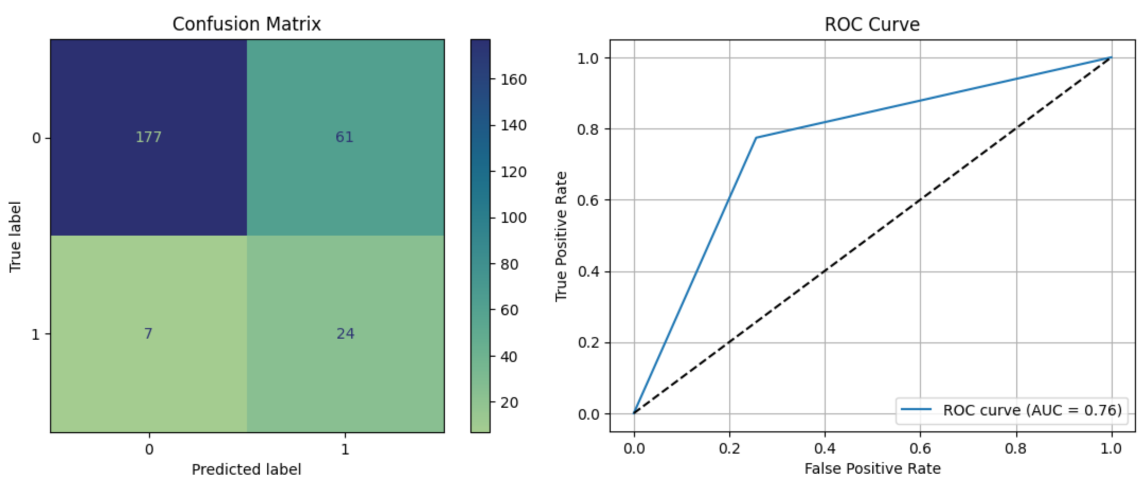


With Class Weights

The model's performance is evaluated using various metrics, including precision, recall, f1-score, and support. The accuracy of the model is also reported.The confusion matrix is provided, which shows the number of true positives, true negatives, false positives, and false negatives for the model's predictions.The ROC (Receiver Operating Characteristic) curve is displayed, which is a common way to evaluate the performance of a binary classification model. The area under the ROC curve (AUC) is 0.76, indicating moderate performance.The report mentions that the author used StandardScaler to standardize the features by removing the mean and scaling to unit variance. They also used the fit_transform method to transform the training data and the transform method to scale the validation and sample data.The author states that they used feature selection, but does not provide details on the specific methods used.

```
[[177  61]
 [  7  24]]
```

	precision	recall	f1-score	support
0	0.96	0.74	0.84	238
1	0.28	0.77	0.41	31
accuracy			0.75	269
macro avg	0.62	0.76	0.63	269
weighted avg	0.88	0.75	0.79	269

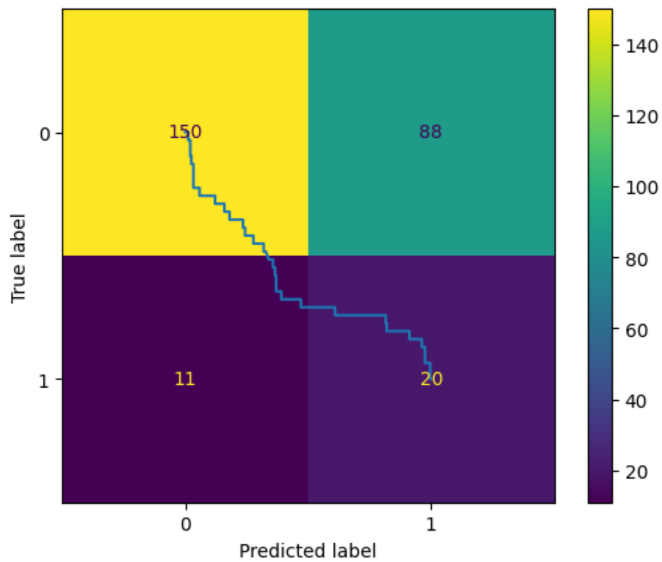


7 Advanced Models

=== SVM RBF ===

F1 Score: 0.28776978417266186

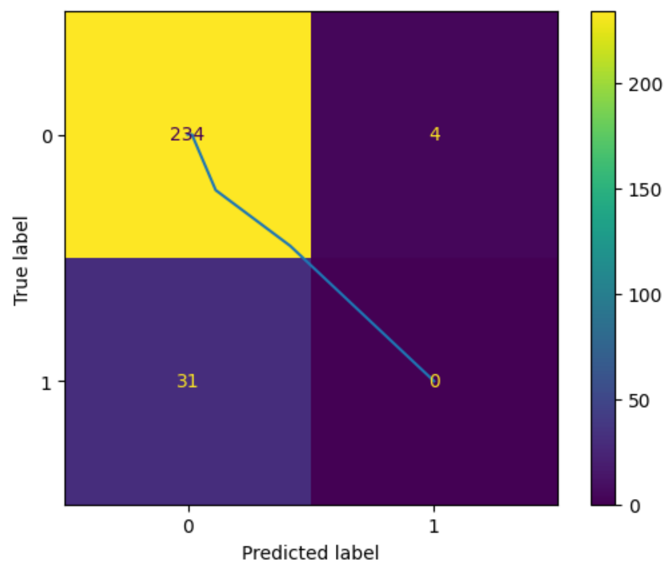
ROC AUC Score: 0.5969775006776904



=== KNN ===

F1 Score: 0.1889763779527559

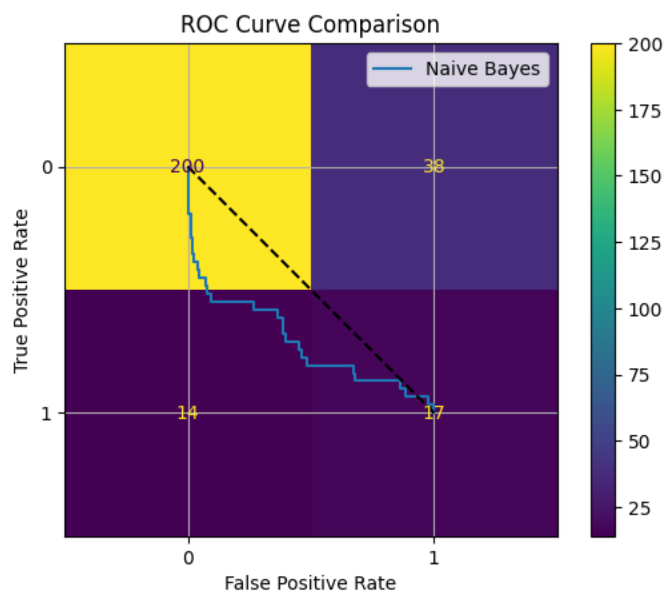
ROC AUC Score: 0.5040661425860666



=== Naive Bayes ===

F1 Score: 0.34234234234234234

ROC AUC Score: 0.701409596096503

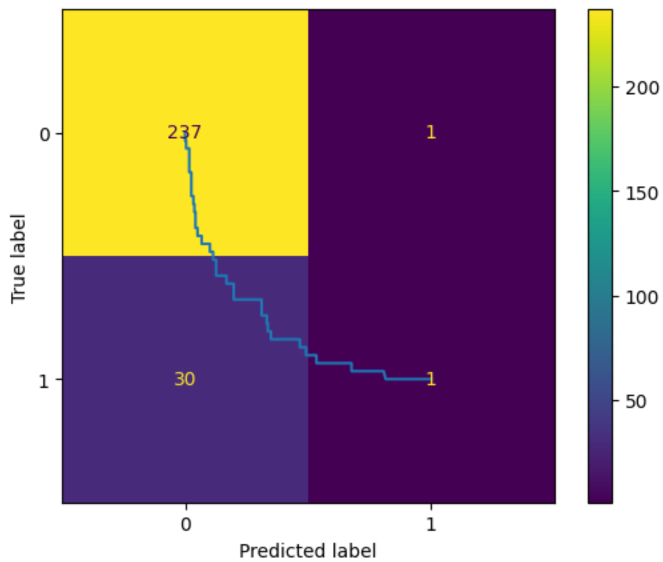


7.B BONUS

=== Simple RF (from scratch) ===

F1 Score: 0.06060606060606061

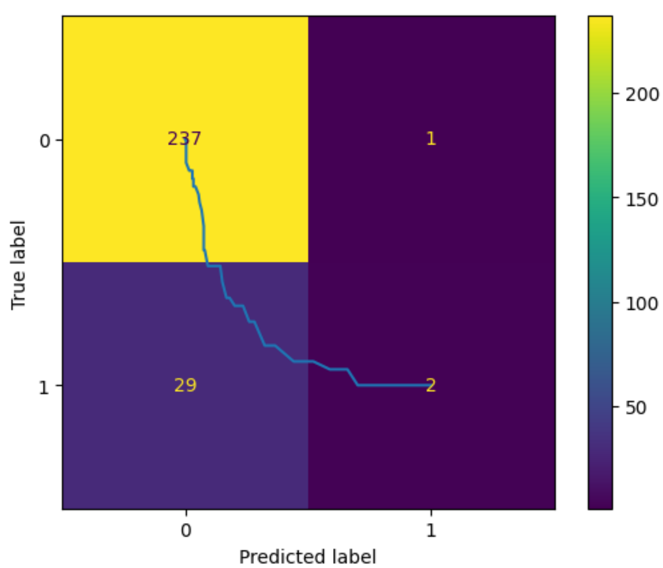
ROC AUC Score: 0.8048251558687991



=== Library RF ===

F1 Score: 0.11764705882352941

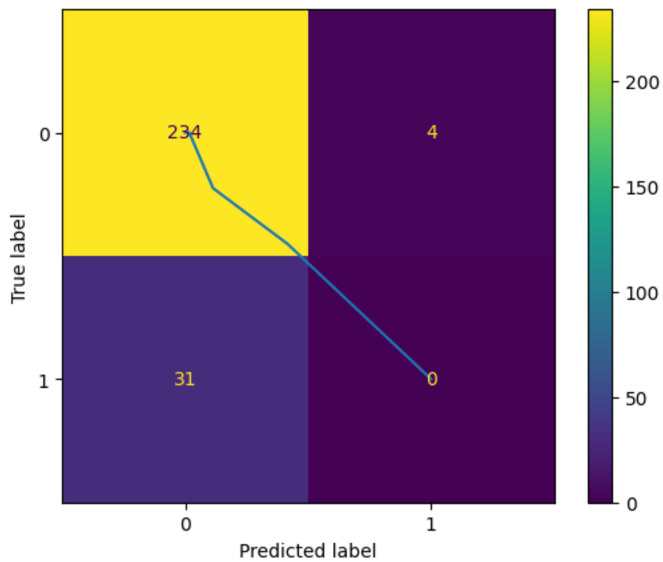
ROC AUC Score: 0.8152615885063702



=== KNN ===

F1 Score: 0.0

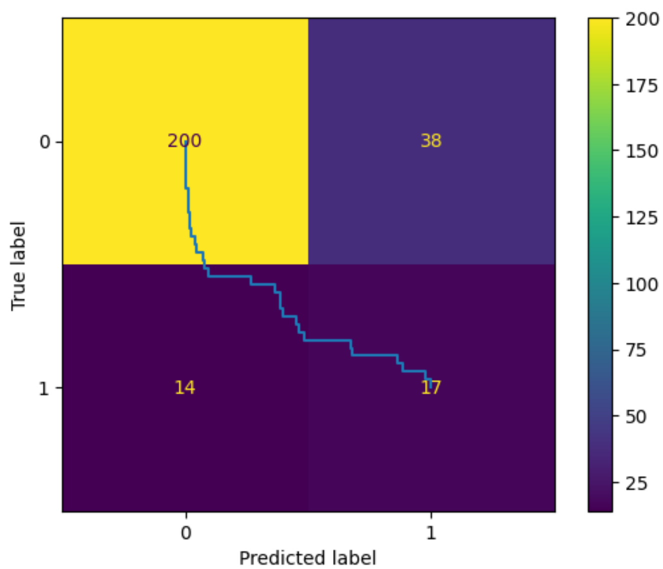
ROC AUC Score: 0.5372729737056112



=== Naive Bayes ===

F1 Score: 0.3953488372093023

ROC AUC Score: 0.7191650853889943

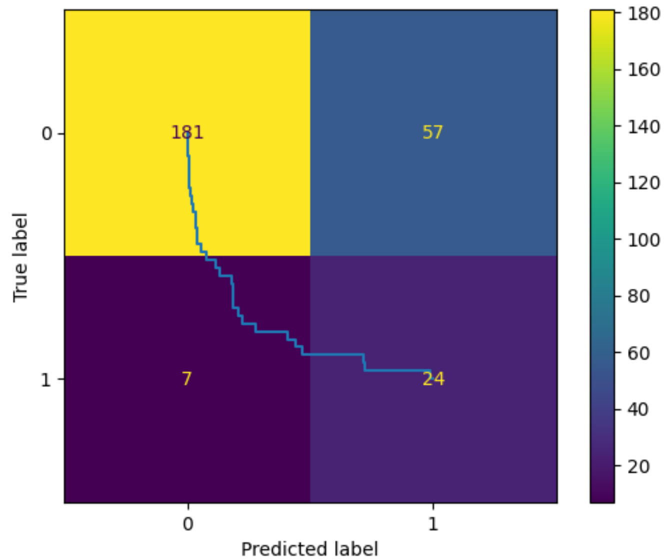


8 Handling Imbalanced Data

=== Logistic Regression (class_weight) ===

F1 Score: 0.42857142857142855

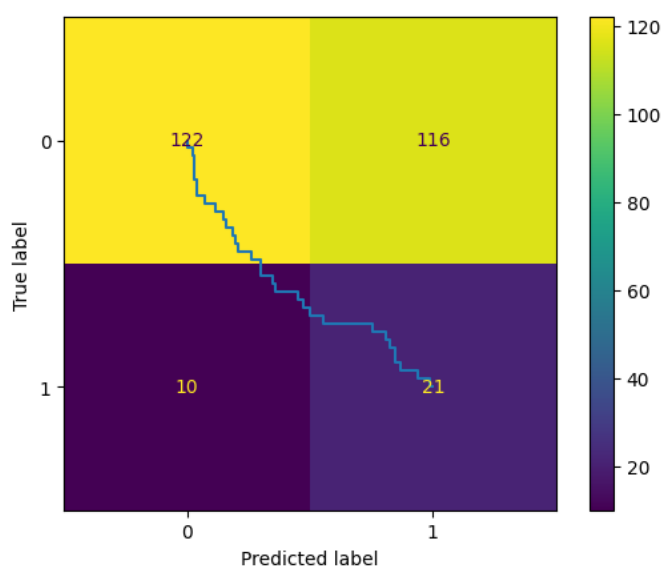
ROC AUC Score: 0.8125508267823258



=== SVM RBF (class_weight) ===

F1 Score: 0.25

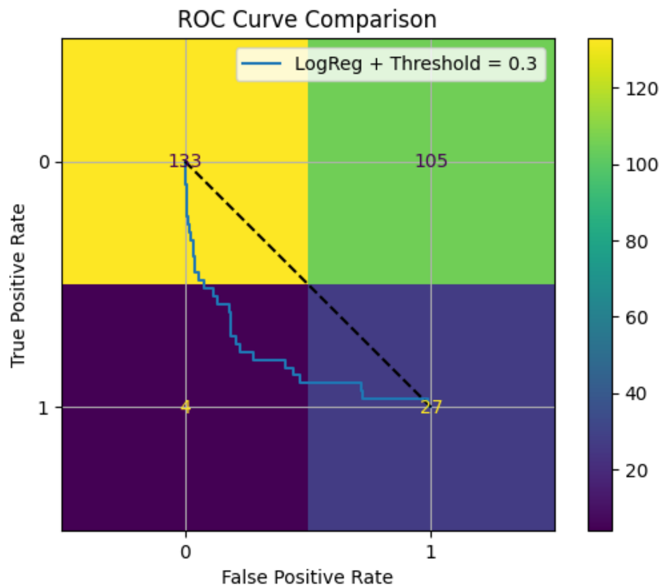
ROC AUC Score: 0.6223908918406071



=== LogReg + Threshold = 0.3 ===

F1 Score: 0.3312883435582822

ROC AUC Score: 0.8125508267823258

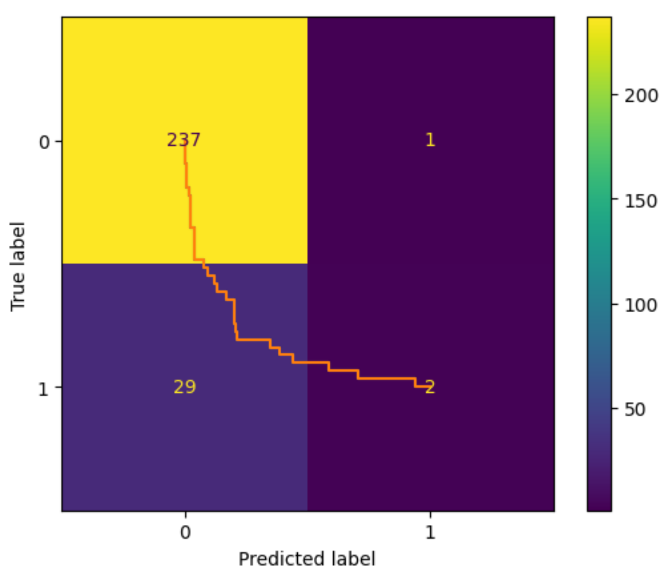


9 Model Stacking (Optional)

=== Stacked Model ===

F1 Score: 0.11764705882352941

ROC AUC Score: 0.8277310924369747



10 Prediction in Kaggle

A soft-voting ensemble classifier was built using Logistic Regression, Random Forest, XGBoost, and SVC, with model weights reflecting relative performance. Recursive Feature Elimination (RFE) selected the top 30 features, and the ensemble was calibrated using 5-fold cross-validation. The optimal decision threshold was determined by maximizing the F1 score on the validation set. Final predictions were generated on the test set using this threshold and exported to newML.csv.

For this data I used this model based on my heatmap to increase my accuracy :

Before that, I scaling features:

- StandardScaler: This is used to standardize features by removing the mean and scaling to unit variance.

- fit_transform: This method fits the scaler to the training data (X_train) and transforms it.

transform: This method is applied to validation (X_val) and sample data (X_sample) to ensure they are scaled in the same way as the training data.

In feature selection I used:

- RFE (Recursive Feature Elimination): This technique selects important features by recursively removing the least important features based on the estimator's performance.

- RandomForestClassifier: Used as the estimator for RFE, it helps identify the top 30 features to retain.

- fit_transform: Applies RFE to the scaled training data and target labels (y_train).

- transform: Applies the same feature selection to validation and sample datasets.

And after that I applied smote because this technique is used to balance the dataset by generating synthetic samples for the minority class.

- fit_resample: This method is used to create a balanced version of the training dataset.

Then, defined base models such as:

- LogisticRegression: A linear model for binary classification with L2 regularization.

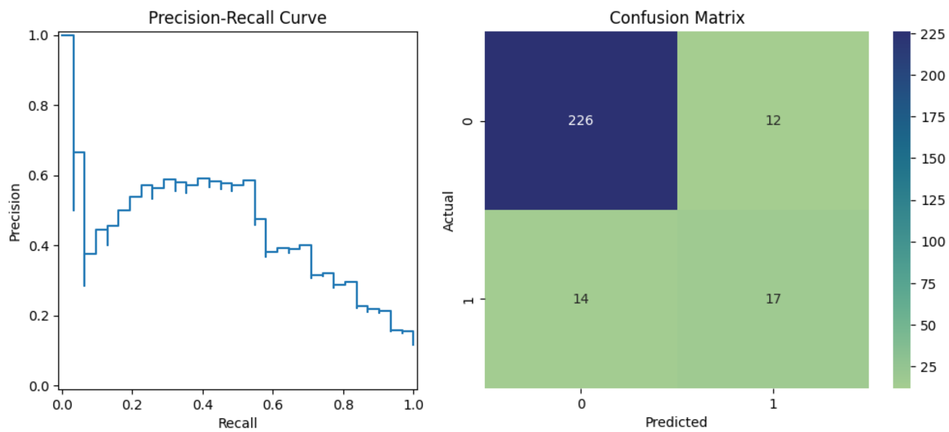
- SVC (Support Vector Classifier): A classifier that finds the optimal hyperplane for classification, with probability=True allowing for probabilistic predictions.
in random forest I used :

- fit: Trains the model on the balanced training data.

And exactly in xgboost mplementation of gradient boosting for classification tasks.

- Similar to the Random Forest tuning, it uses GridSearchCV to find the best parameters for XGBoost.

This part effectively prepares data for machine learning by scaling features, selecting important features, balancing classes, tuning models, and creating an ensemble classifier. The final predictions are saved for further analysis or use.



10 Conclusion

I built a classification pipeline that begins with feature scaling via StandardScaler, followed by dimensionality reduction using Recursive Feature Elimination (RFE) with a RandomForestClassifier, selecting the top 30 features. A soft-voting ensemble was constructed using Logistic Regression, Random Forest, XGBoost, and SVC, with model-specific weights to emphasize stronger learners. To improve probability calibration and enable effective thresholding, the ensemble was wrapped with CalibratedClassifierCV using 5-fold cross-validation. The decision threshold was optimized based on F1 score derived from the precision-recall curve. Final predictions were generated on the test set using this calibrated, threshold-optimized ensemble and exported to newML.csv. While the ensemble leverages diverse model strengths, future improvements could include exploring stacking, fine-tuning hyperparameters for better generalization, and evaluating feature selection alternatives to RFE.

References

[1] M.A , “ML_4_SBU” 2025

https://www.kaggle.com/competitions/ml4sbu/ [2] LD Freeman “A Data Science Framework: To Achieve 99% Accuracy”2018

https://www.kaggle.com/code/ldfreeman3/a-data-science-framework-to-achieve-99-accuracy/notebook#Step-4:-Perform-Exploratory-Analysis-with-Statistics