LOAN APPROVAL

AZURE ML STUDIO

LI WU

2025 JAN 23

Agenda

- Data Introduction
- Exploratory Data Analysis
- Automated ML
- ML Pipeline Designer
- Business Suggestions

OBJECTIVE

Automate The Loan Eligibility Process

in real-time based on customer details provided in the online application form.

Dataset Introduction

Categorical (7) :

- Gender
- Married
- Dependents
- Education
- Self_Employed
- Credit_History
- Property_Area

Numerical (4):

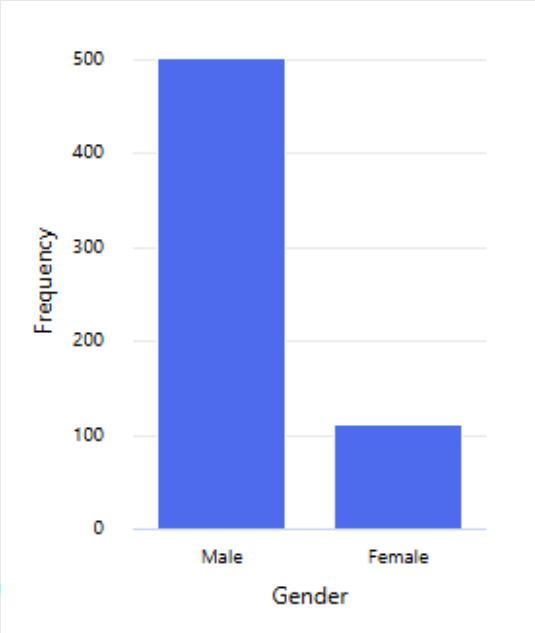
- ApplicantIncome
- CoapplicantIncome
- LoanAmount
- Loan_Amount_Term

Feature	Count	Unique Value Count	Missing Value Count
Gender	601	2	13
Married	611	2	3
Dependents	599	4	15
Education	614	2	0
Self_Employed	582	2	32
ApplicantIncome	614	505	0
CoapplicantIncome	614	287	0
LoanAmount	592	203	22
Loan_Amount_Term	600	10	14
Credit_History	564	2	50
Property_Area	614	3	0
Loan_Status	614	2	0

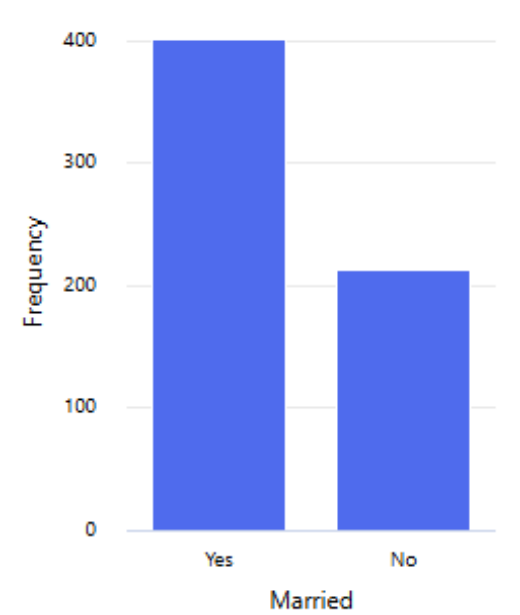
Exploratory Data Analysis

Univariate Analysis - *Categorical*

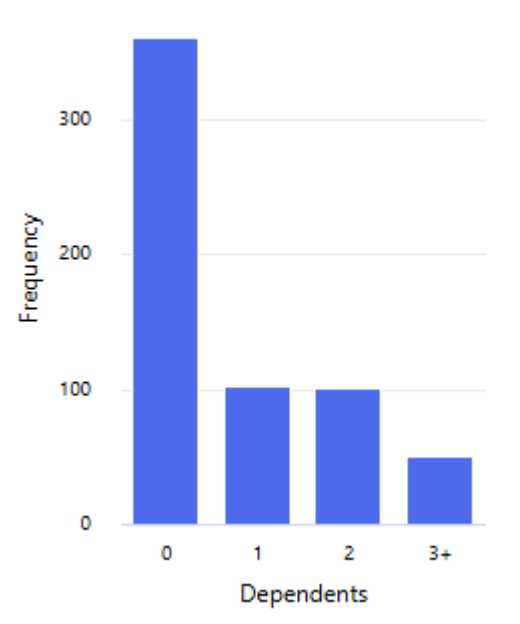
Gender



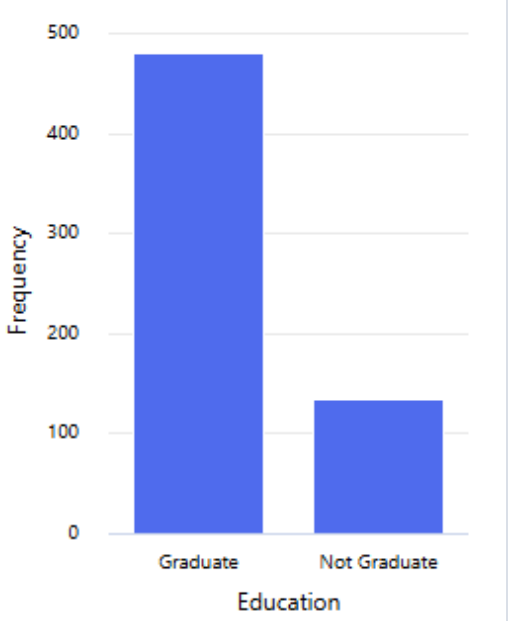
Married



Dependents

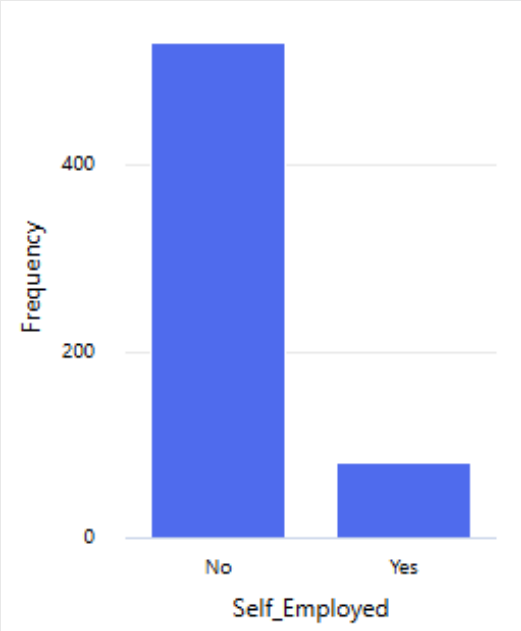


Education

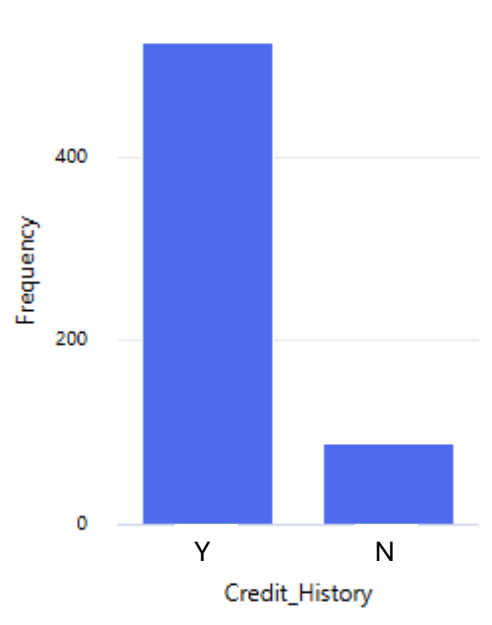


Univariate Analysis - *Categorical*

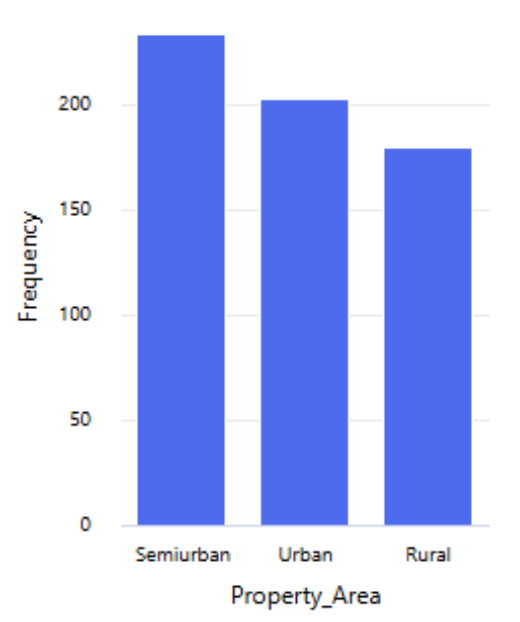
Self_Employed



Credit_History

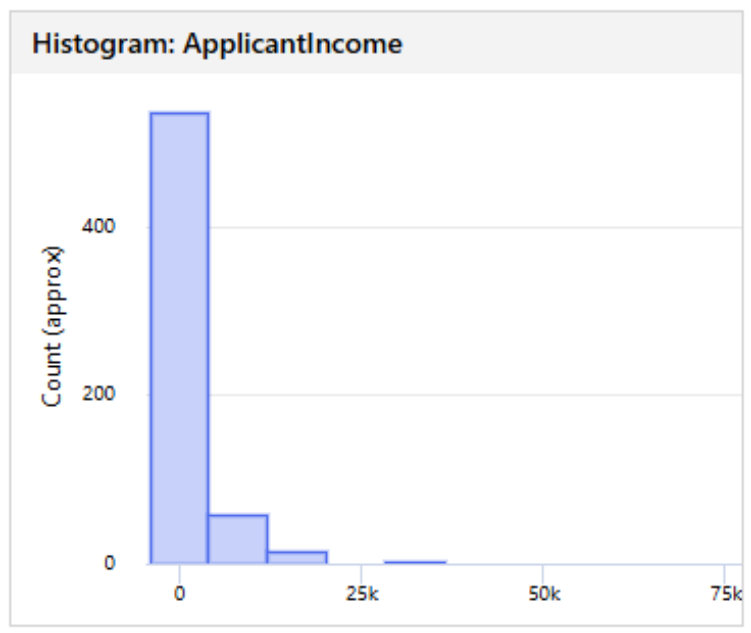


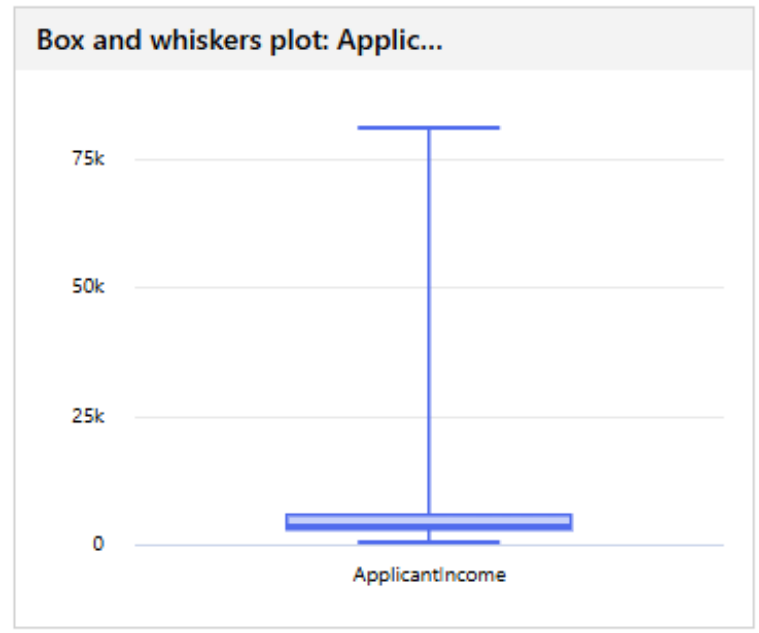
Property_Area



Univariate Analysis - *Numerical*

ApplicantIncome

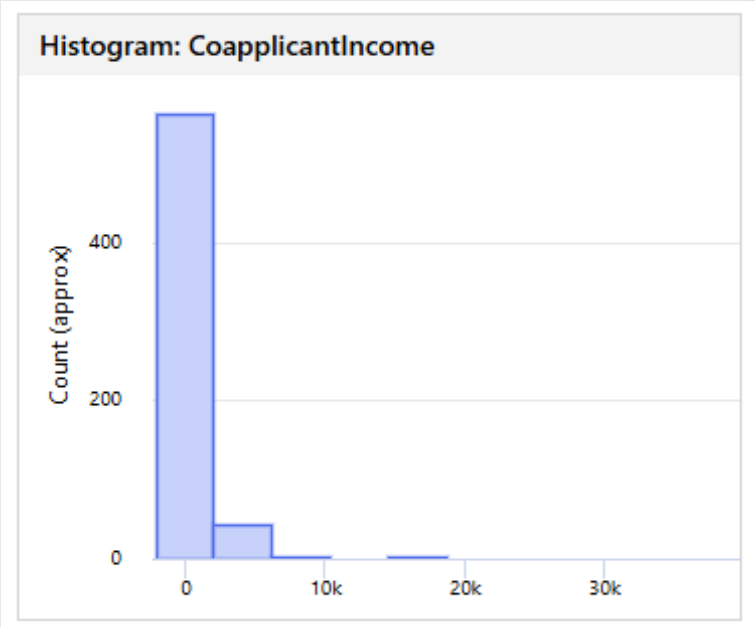


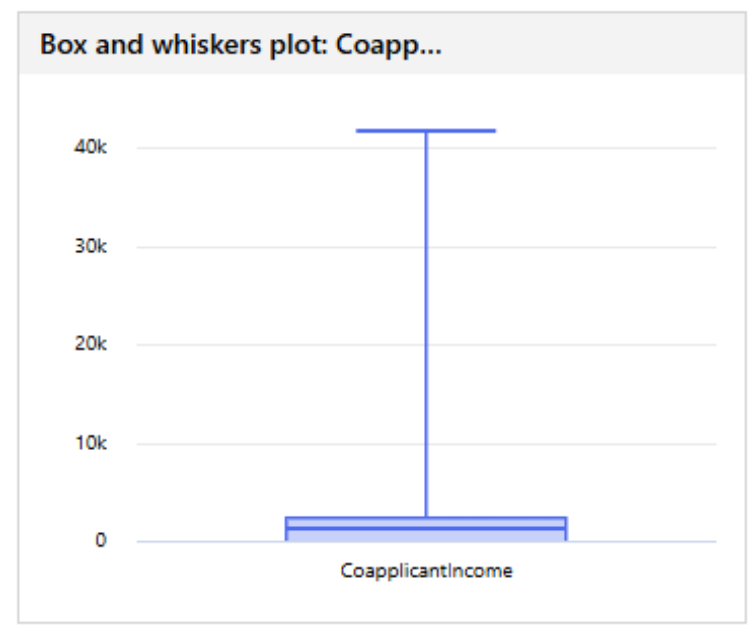


Min	150.00
Q1 (approx)	2869.36
Median (approx)	3804.22
Q3 (approx)	5791.80
Max	81000.00
Moments (2)	
Mean	5403.46
Std deviation	6109.04

Univariate Analysis - *Numerical*

CoapplicantIncome

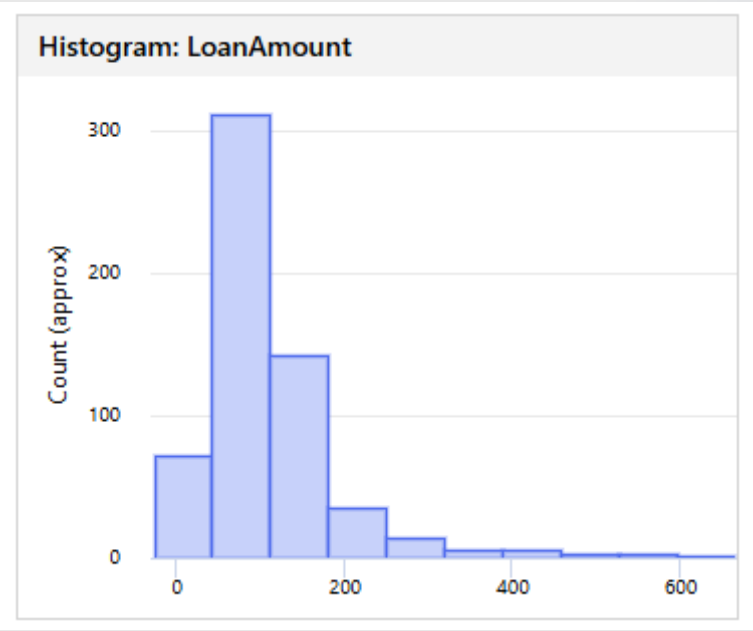


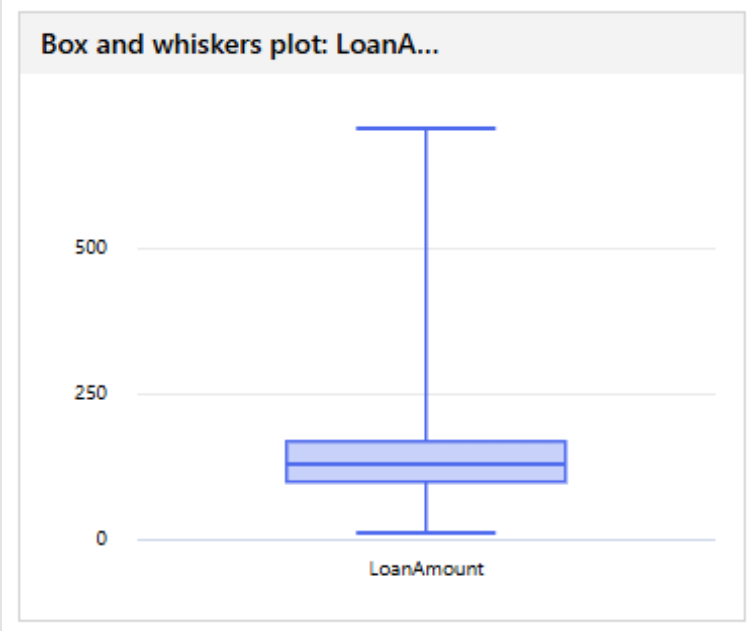


Min	0.00
Q1 (approx)	0.00
Median (approx)	1178.08
Q3 (approx)	2299.20
Max	41667.00
Moments (2)	
Mean	1621.25
Std deviation	2926.25

Univariate Analysis - *Numerical*

LoanAmount (in 1000)

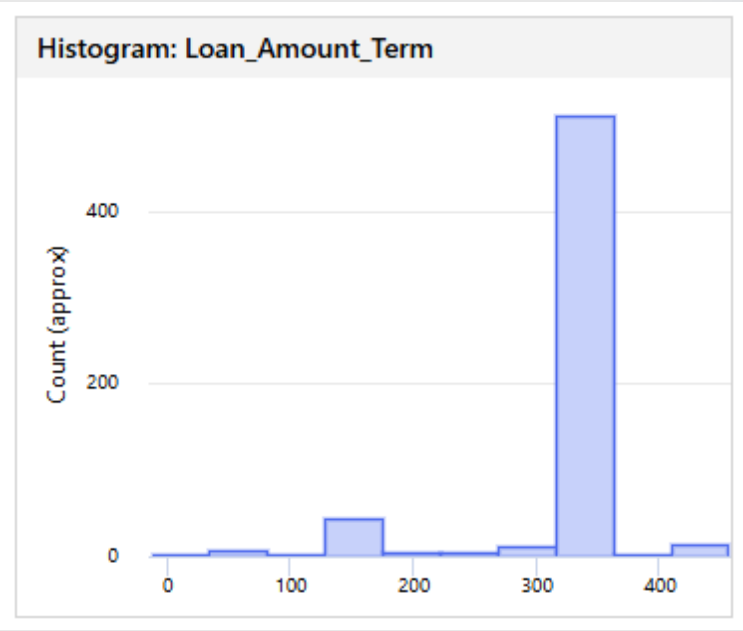


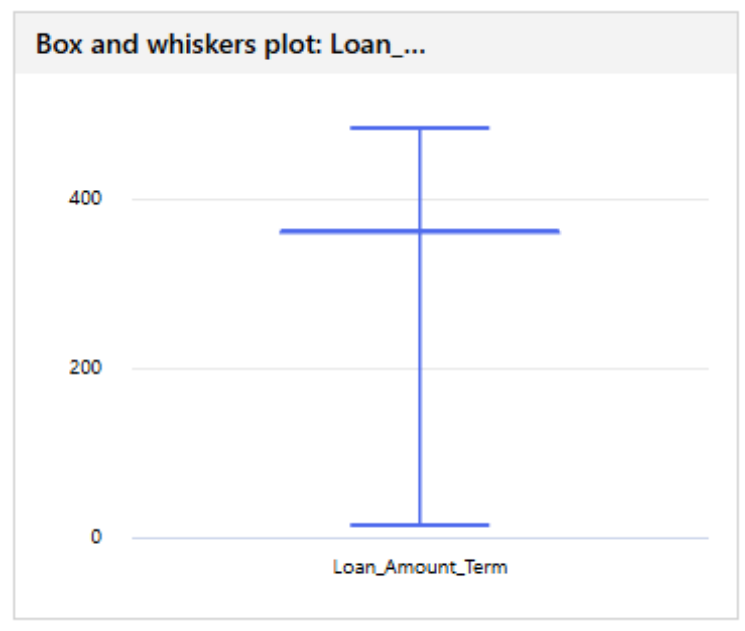


Min	9.00
Q1 (approx)	100.00
Median (approx)	127.50
Q3 (approx)	167.74
Max	700.00
Moments (2)	
Mean	146.41
Std deviation	85.59

Univariate Analysis - *Numerical*

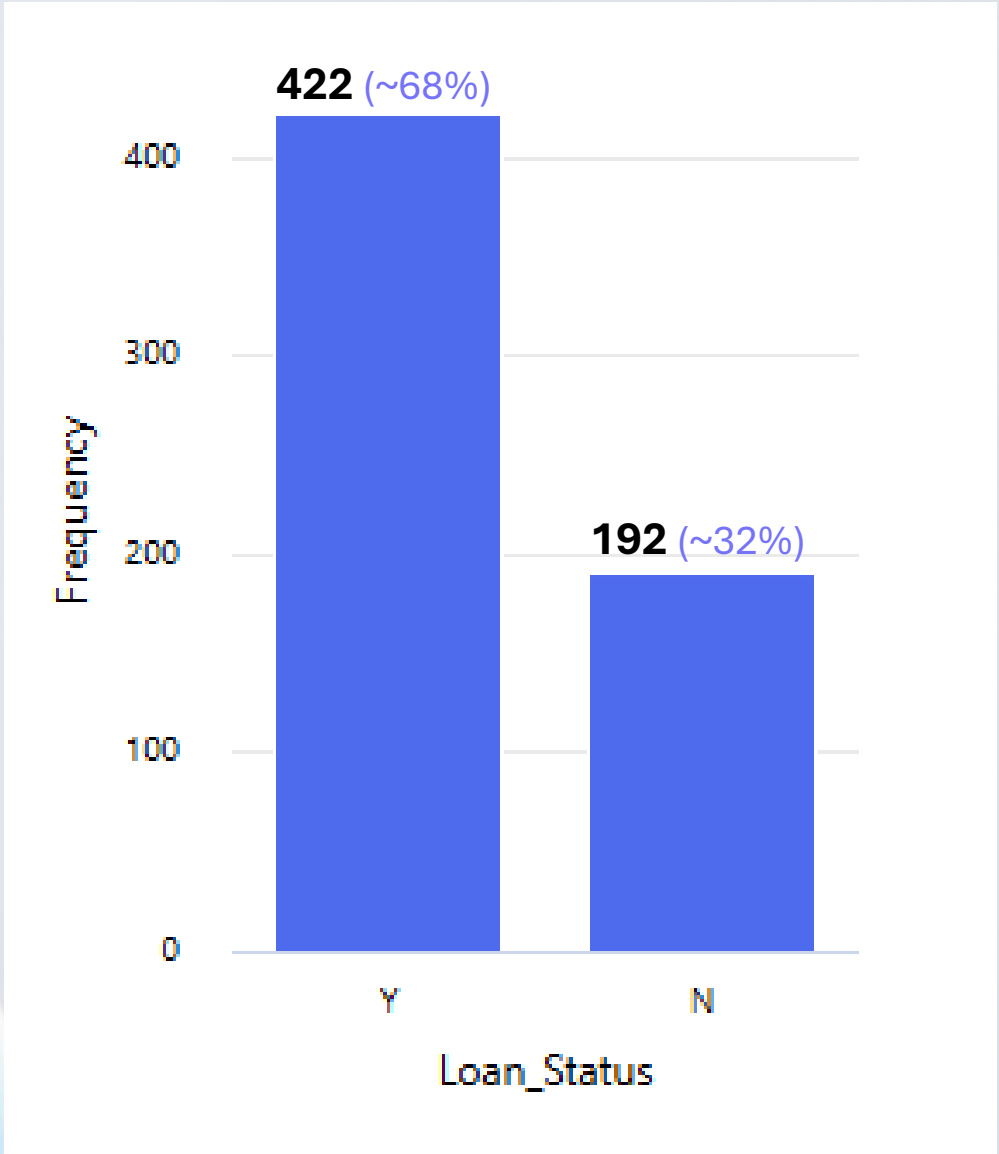
Loan_Amount_Term (months)





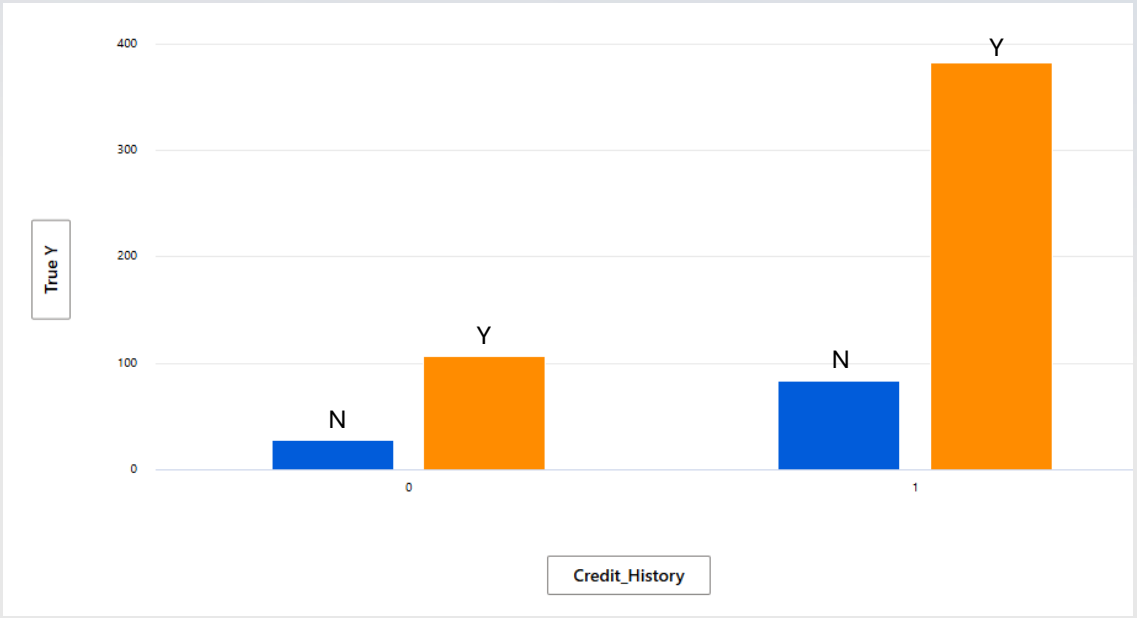
Min	12.00
Q1 (approx)	360.00
Median (approx)	360.00
Q3 (approx)	360.00
Max	480.00
Moments (2)	
Mean	342.00
Std deviation	65.12

Target Variable

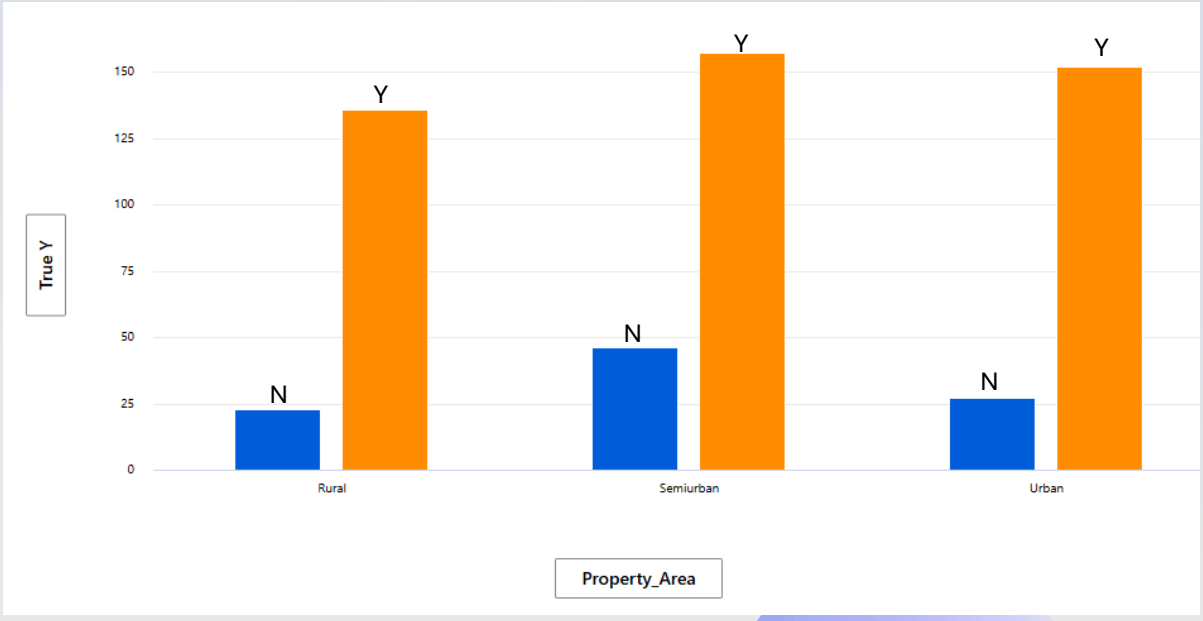


Bivariate Analysis

Credit History

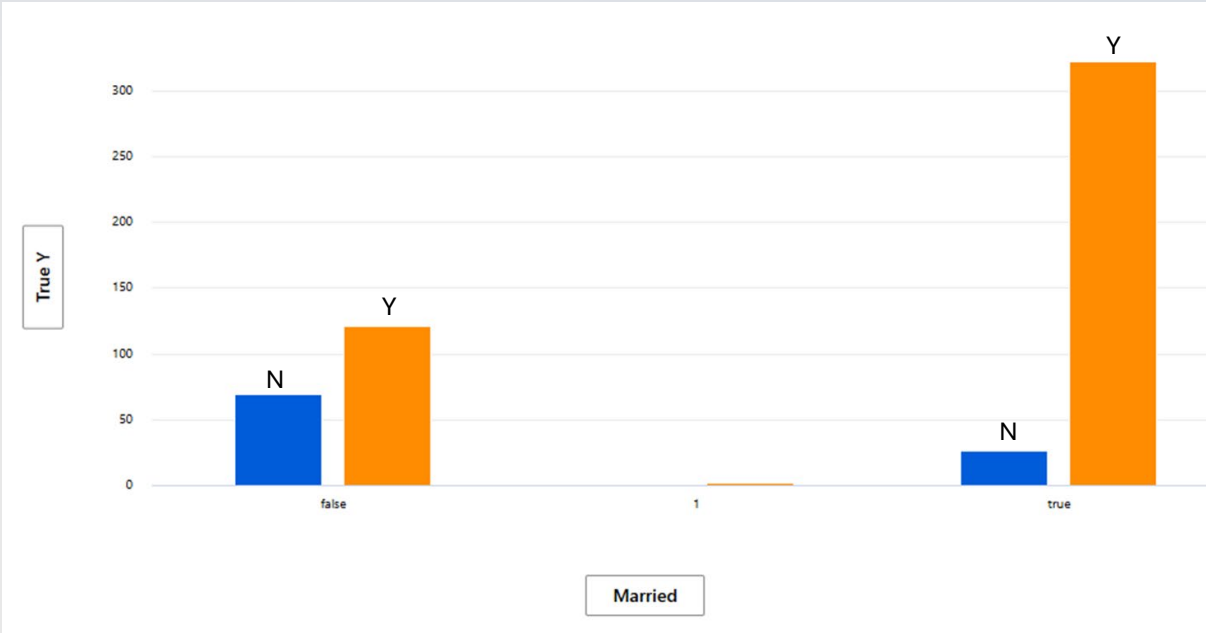


Property Area

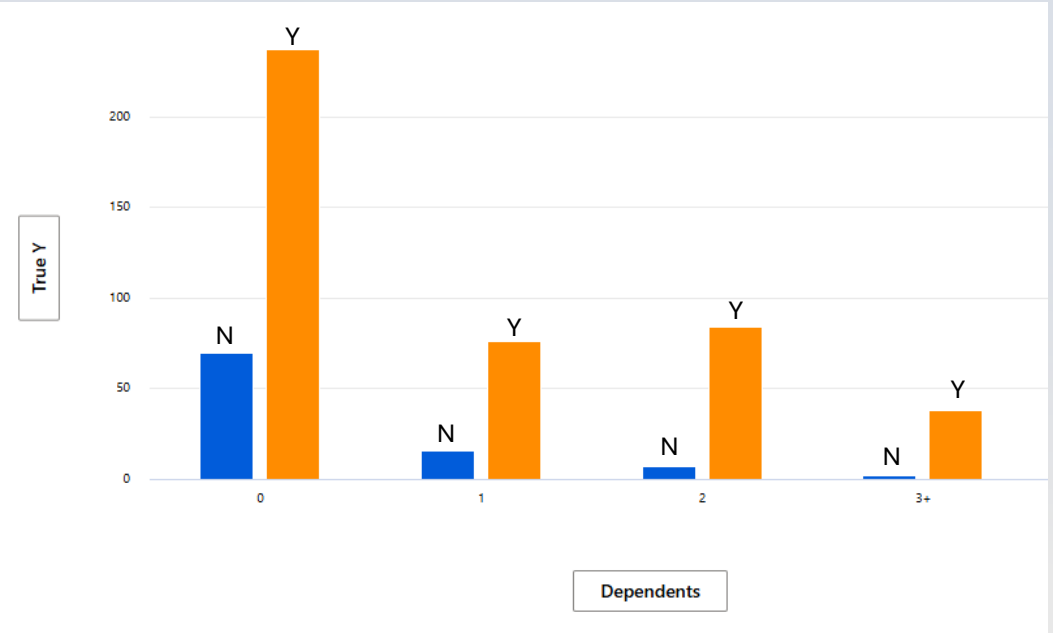


Bivariate Analysis

Married

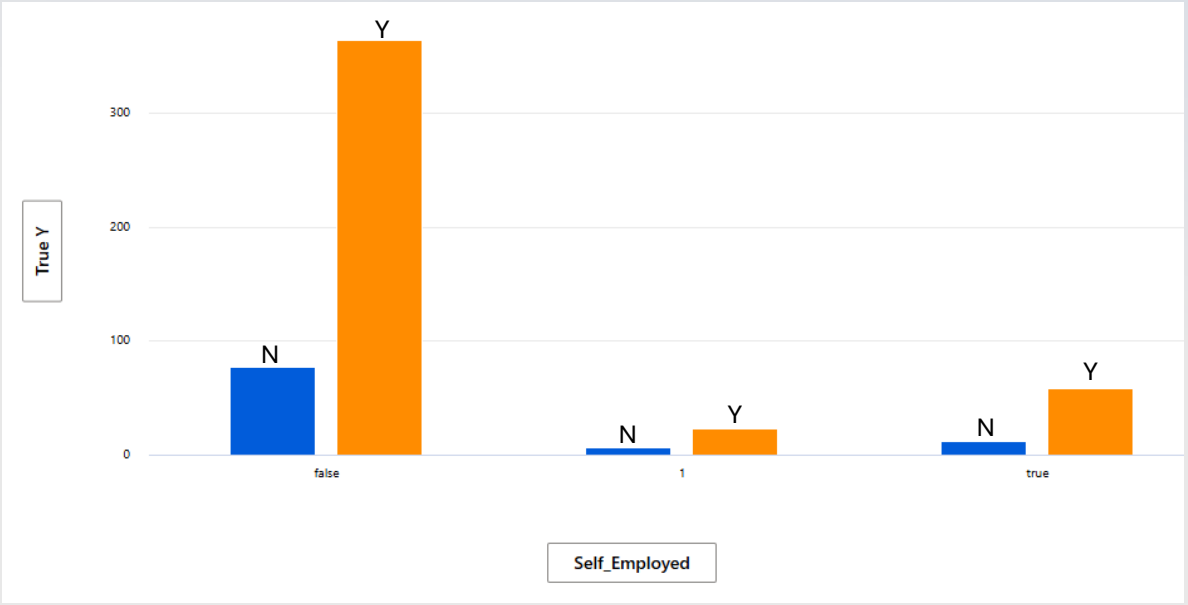


Dependents

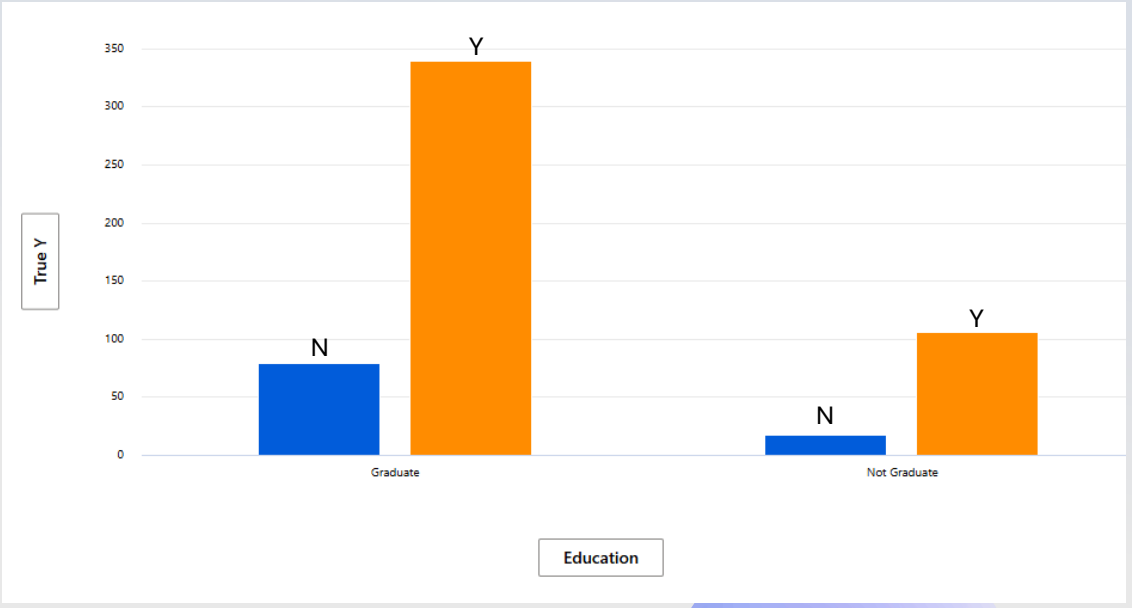


Bivariate Analysis

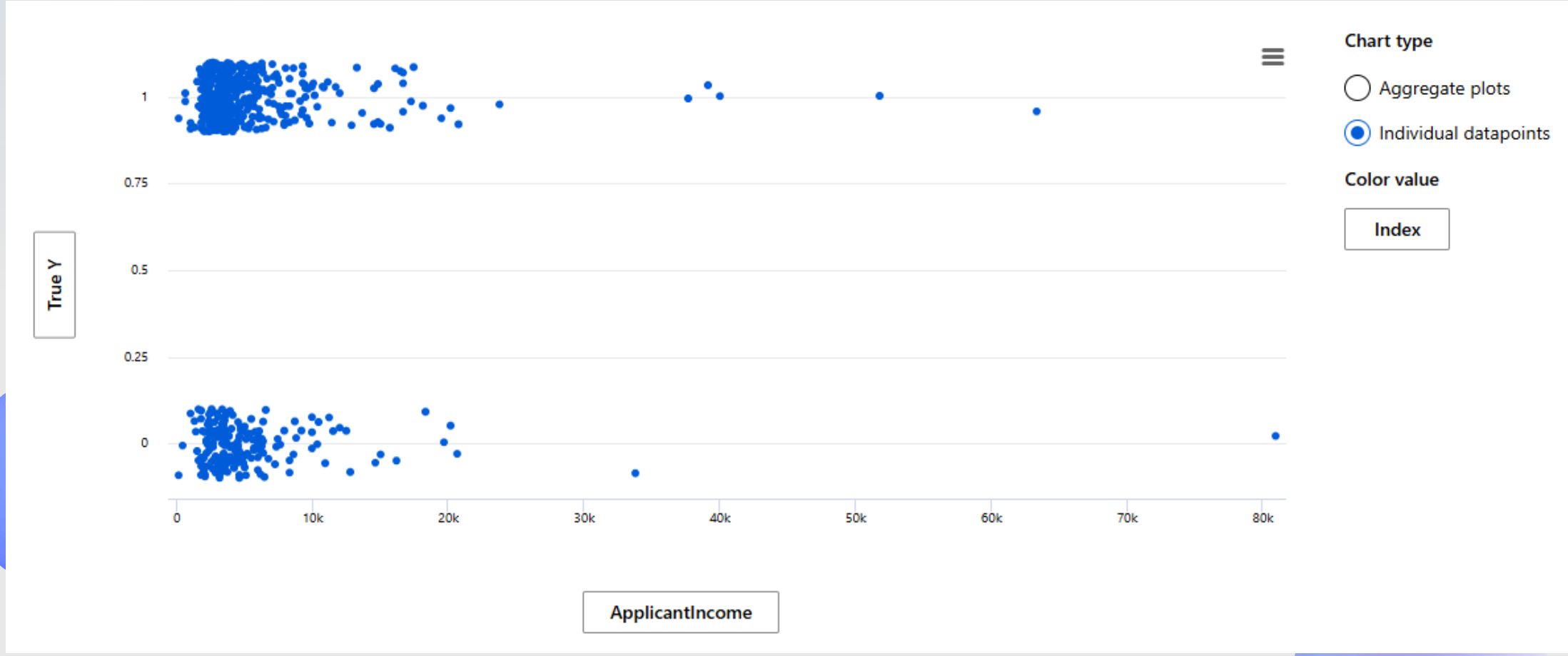
Self Employed



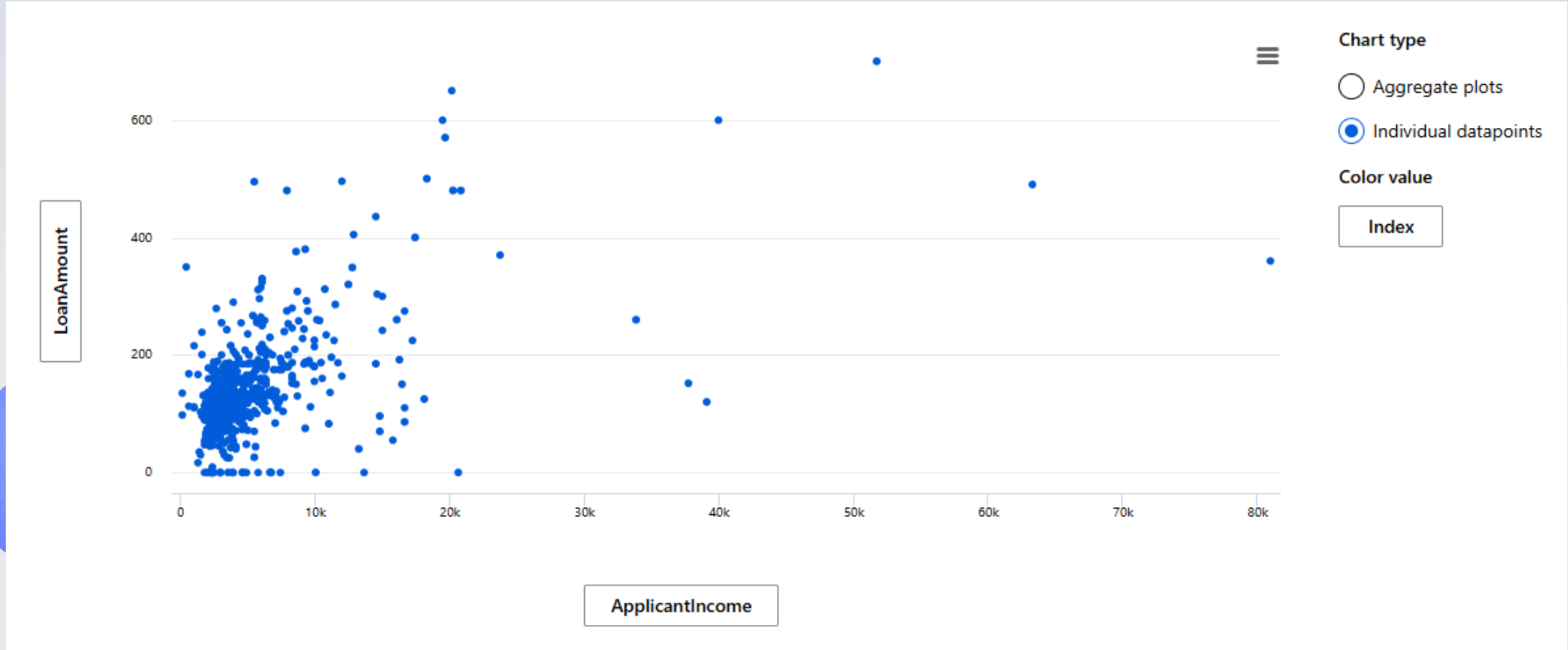
Education



Bivariate Analysis – *Applicant Income vs Loan Approval*



Bivariate Analysis – *Applicant Income vs Loan Amount*



Automated ML

MODEL EVALUATION

Top Performing Model – *Evaluated Across 60 models*

Train 90% | Test 10% | K-Fold Validation: 10

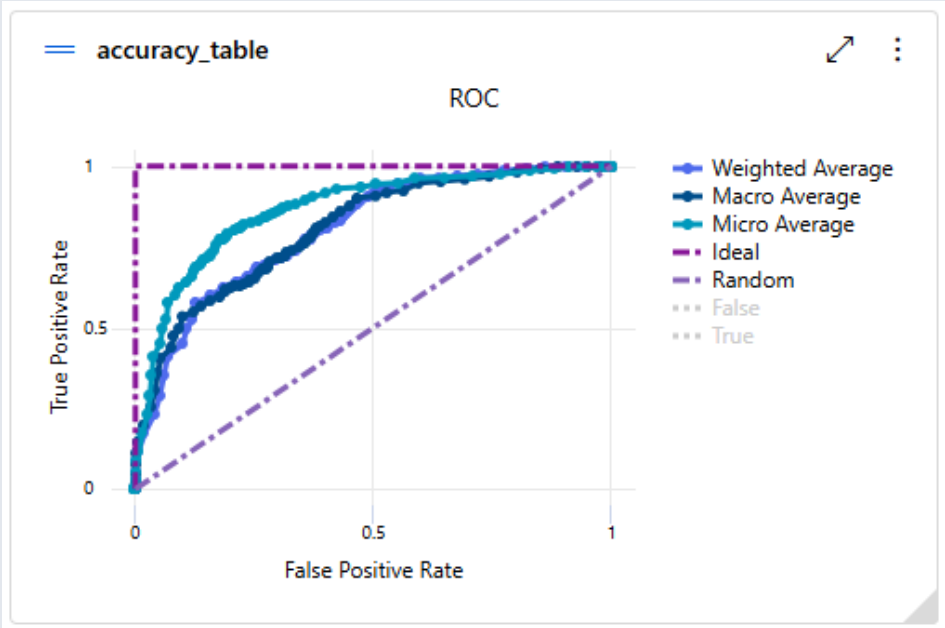
Algorithm name	Explained	Responsible AI	AUC weighted ↓
VotingEnsemble	View explanation		0.80794
SparseNormalizer, XGBoostClassifier			0.77575
SparseNormalizer, RandomForest			0.77526
SparseNormalizer, XGBoostClassifier			0.77416
SparseNormalizer, XGBoostClassifier			0.77208
StandardScalerWrapper, XGBoostClassifier			0.76910
SparseNormalizer, XGBoostClassifier			0.76863
SparseNormalizer, XGBoostClassifier			0.76655
StandardScalerWrapper, RandomForest			0.76649
StandardScalerWrapper, XGBoostClassifier			0.76635
StandardScalerWrapper, XGBoostClassifier			0.76572

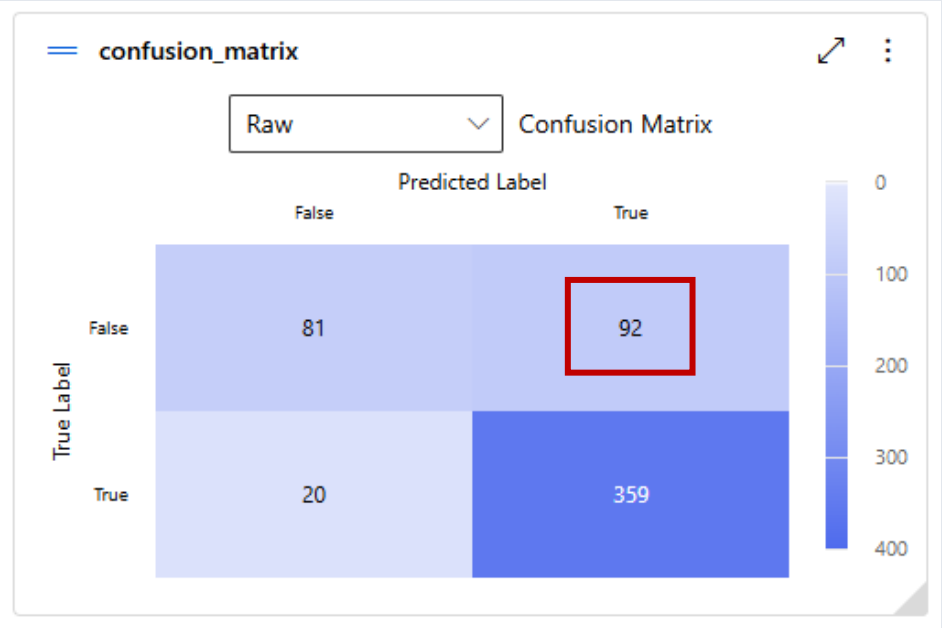
.....

Best Model - *VotingEnsemble*

Train Dataset

AUC_weighted	average_precision_sco...	f1_score_weighted	precision_score_weigh...	recall_score_weighted	weighted_accuracy
0.8079351	0.8375726	0.7781801	0.8003569	0.7969805	0.8591030

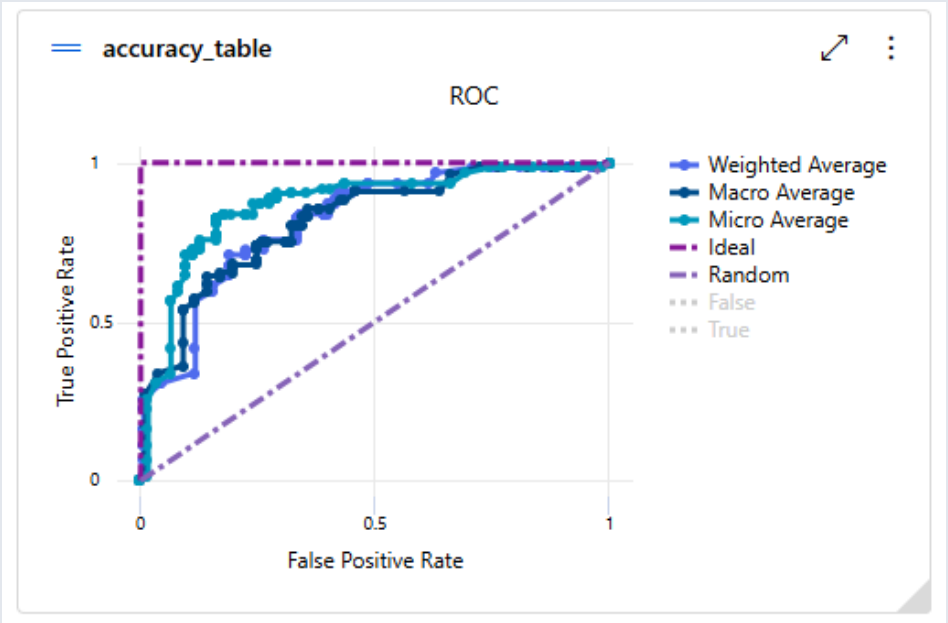


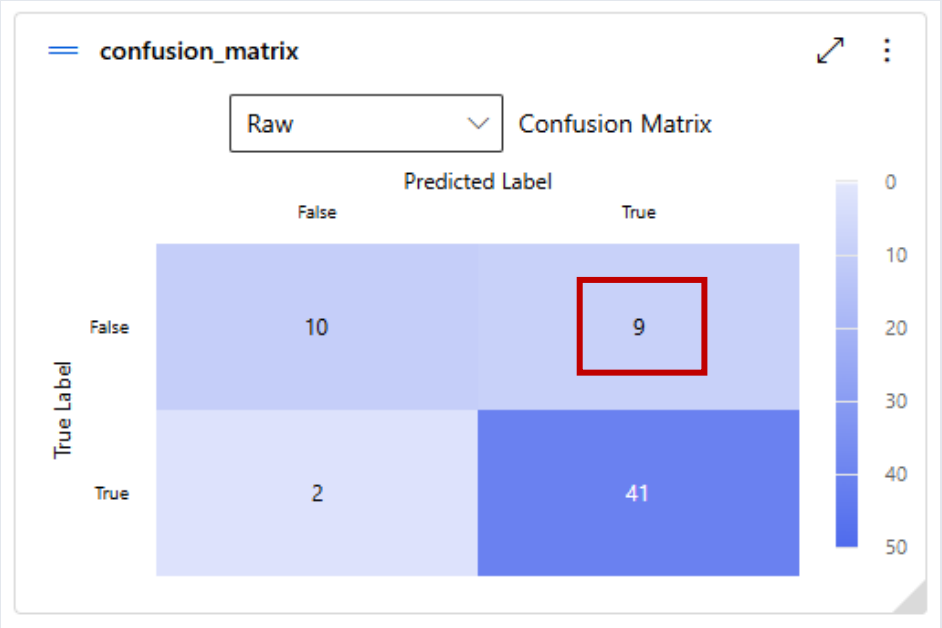


Best Model - *VotingEnsemble*

Test Dataset

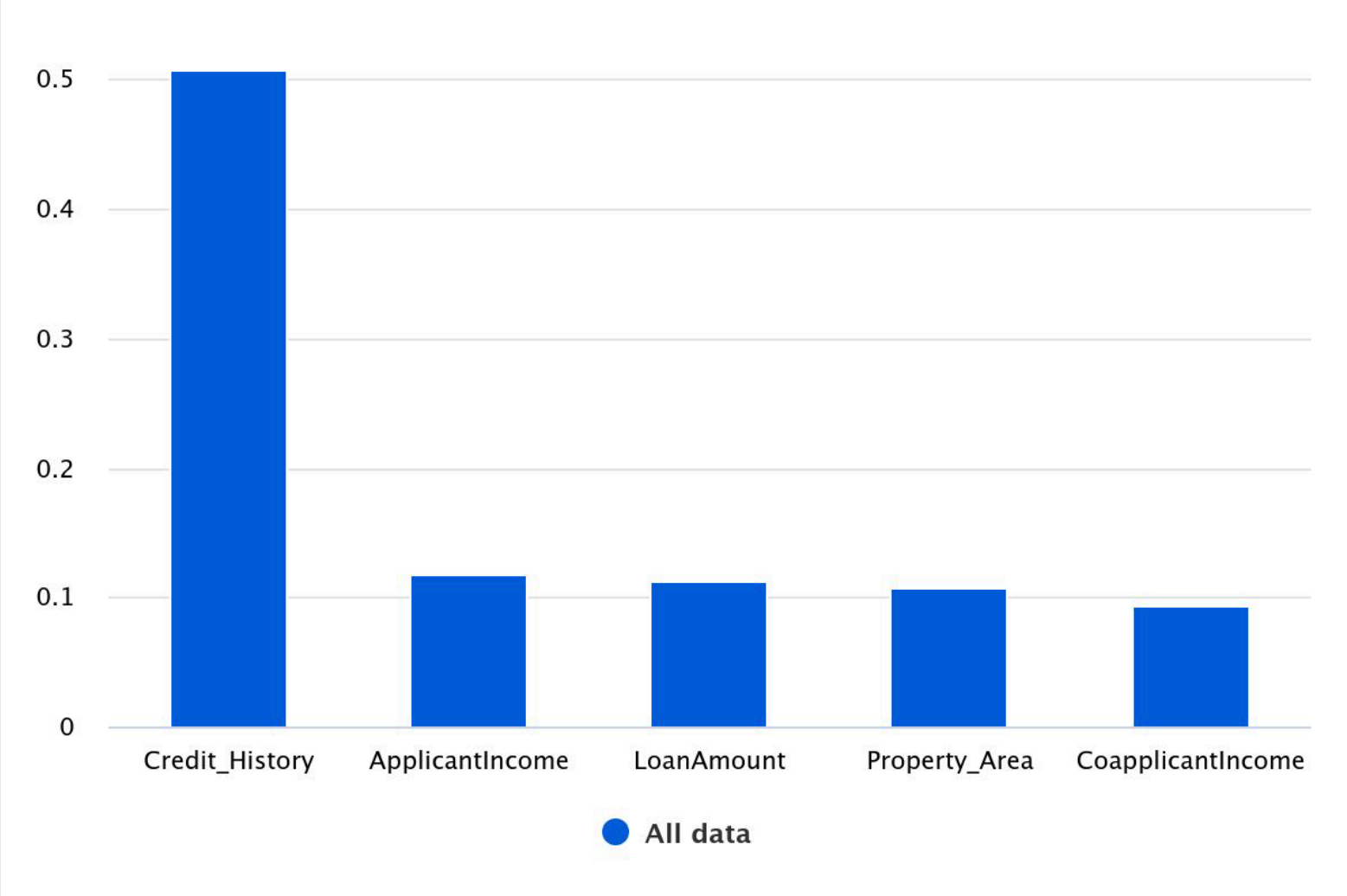
AUC_weighted	average_precision_sco...	f1_score_weighted	precision_score_weigh...	recall_score_weighted	weighted_accuracy
0.8102815	0.8281257	0.8092265	0.8240860	0.8225806	0.8837104





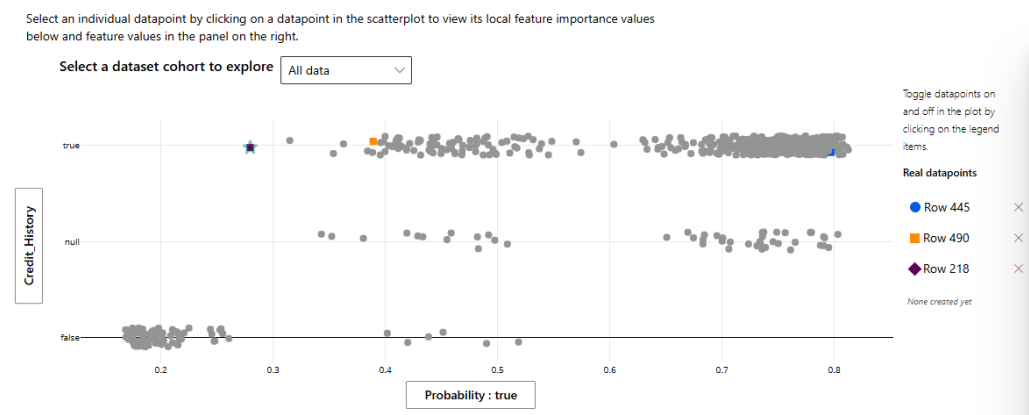
Best Model - *VotingEnsemble*

Top 5 Feature Importance

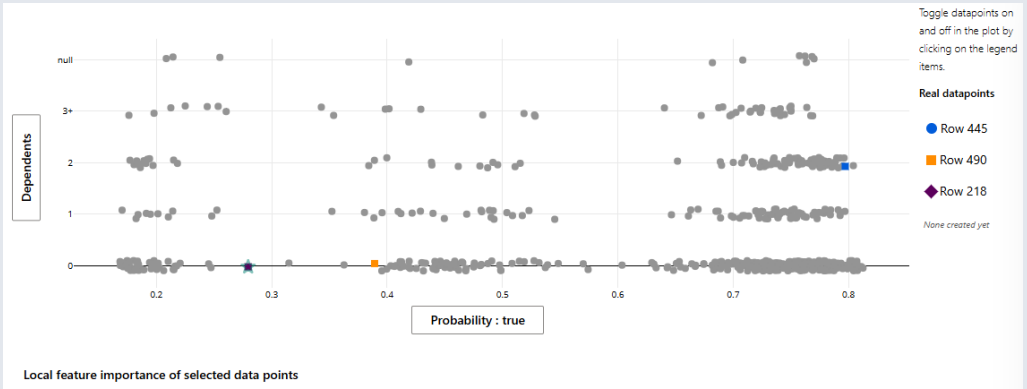


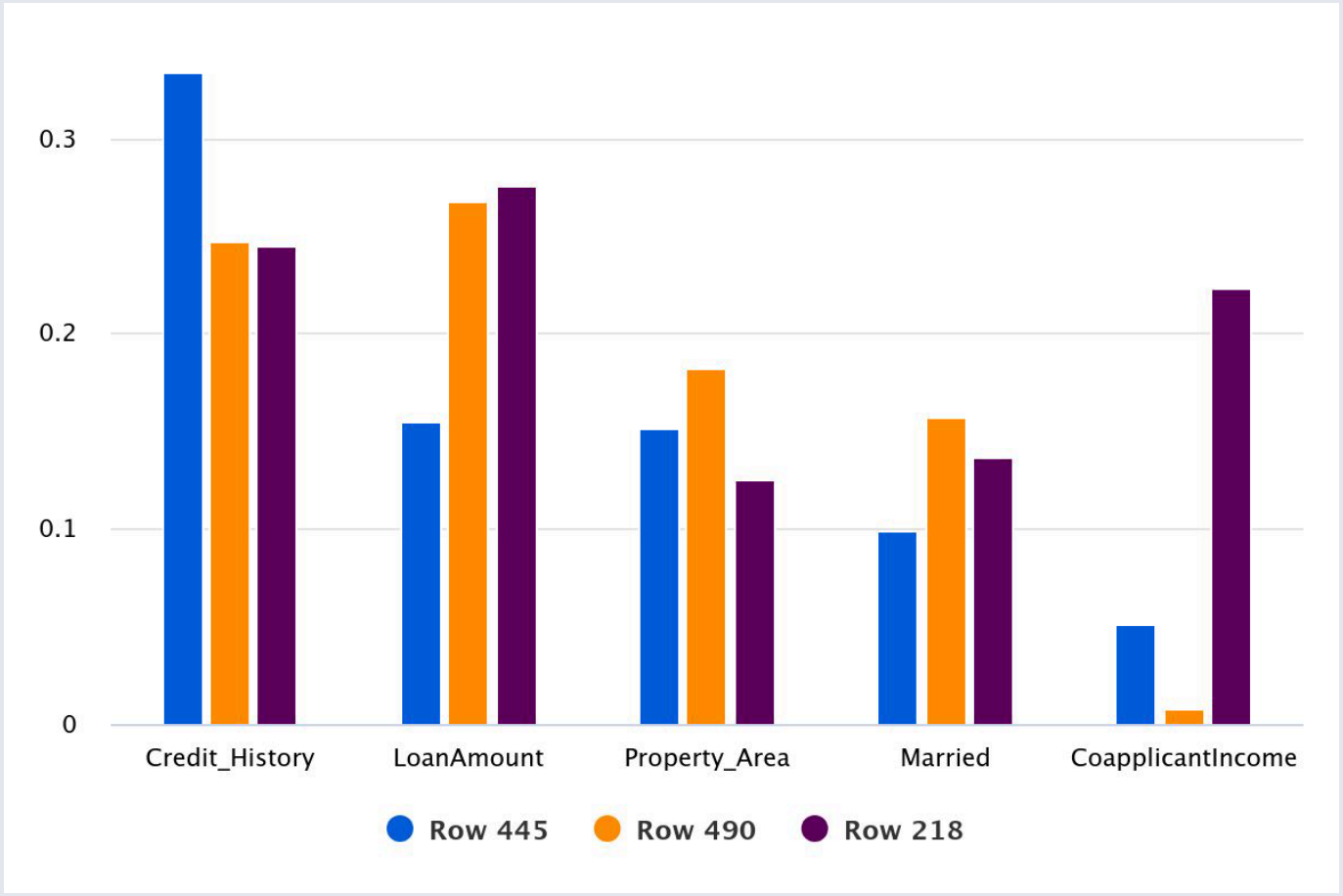
Individual Feature Importance

Credit History vs Loan Approval



Dependents vs Loan Approval





ML Pipeline Designer

Model Pipeline Workflow

Step 1

Data Preprocessing:

Duplicates, Select columns, Missing Values, Normalization, Encoding

Step 2

Model Design:

Split data, Choose Algorithms to train model, Evaluate models

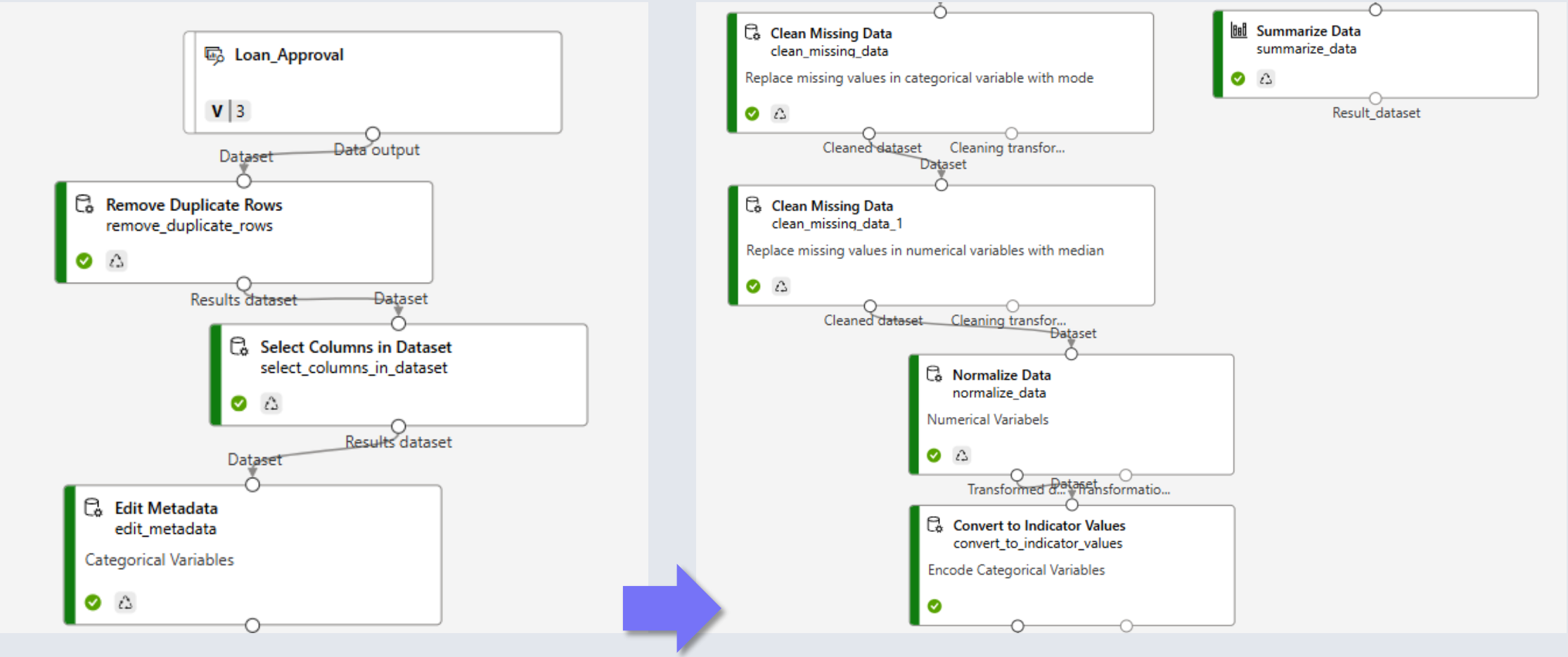
Step 3

Create Inference Pipeline

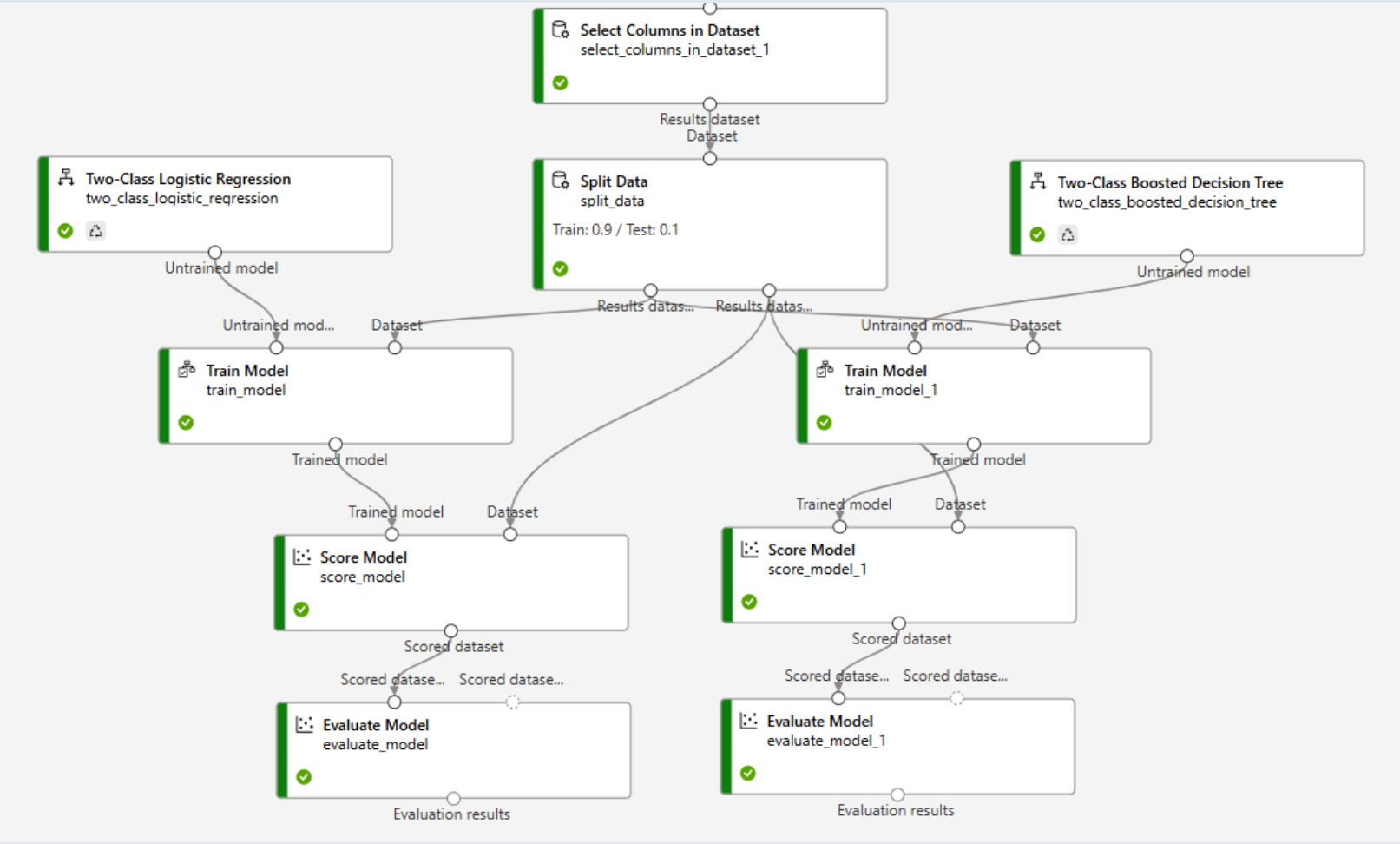
Step 4

Deploy Selected Model

Data Preprocessing Pipeline



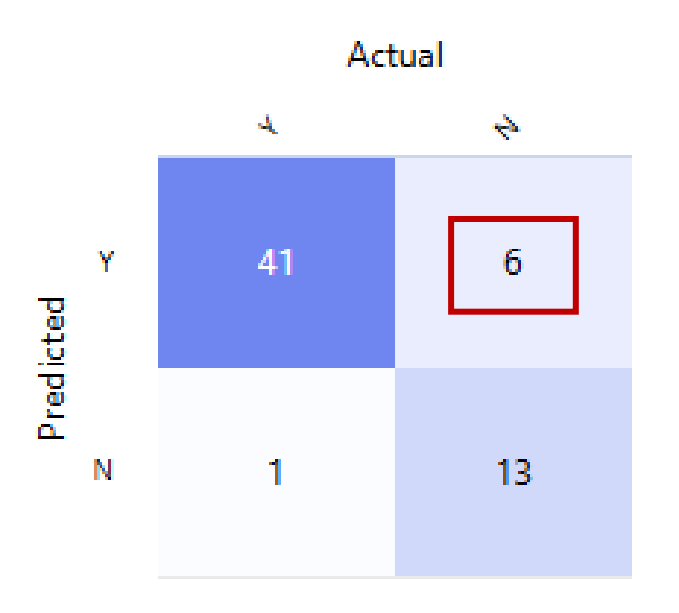
Data Preprocessing Pipeline

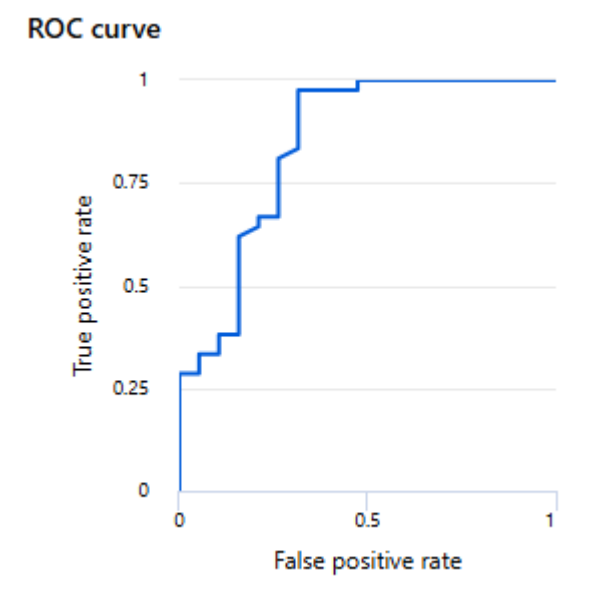


Two-Class Logistic Regression

Model Performance

Accuracy	AUC	F1 Score	Precision	Recall
0.8852459	0.8446115	0.9213483	0.8723404	0.9761905

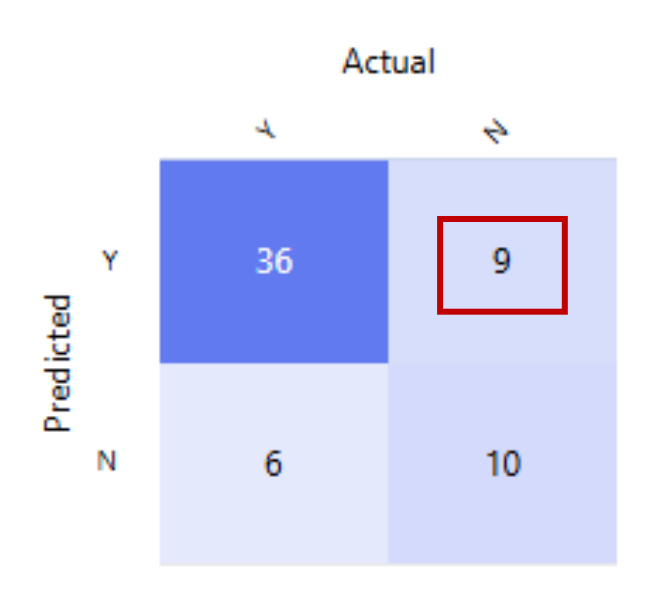


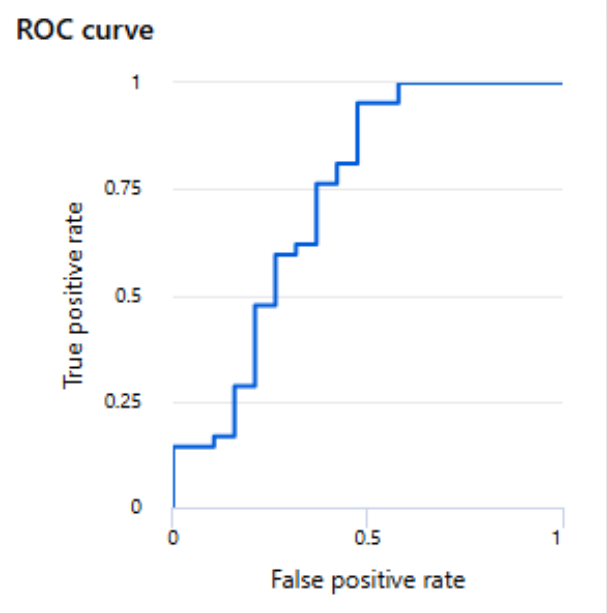


Two-Class Boosted Decision Tree

Model Performance

Accuracy	AUC	F1 Score	Precision	Recall
0.7540984	0.7318296	0.8275862	0.8	0.8571429

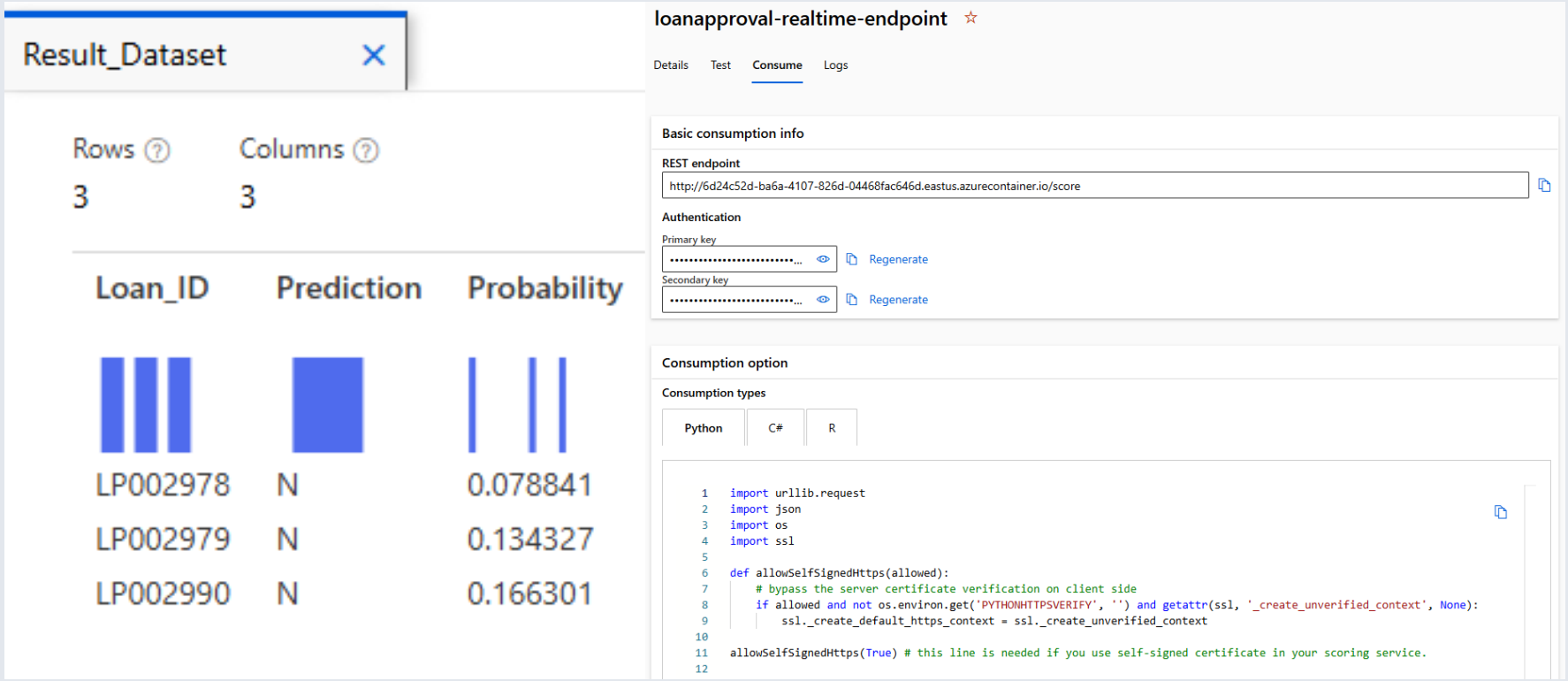




Best Model Deployment

Two-Class Logistic Regression

Accuracy	AUC
AutoML	
0.8837	0.8102
Logistic Regression	
0.8852	0.8446
Boosted Decision Tree	
0.7540	0.7318



Best Model Deployment

Two-Class Logistic Regression

Loan Approval Predictions:

Loan_ID	Prediction	Probability
LP001066	Y	0.899938
LP001068	N	0.444942
LP001109	N	0.250086
LP001112	Y	0.520912
LP001279	N	0.283755
LP001356	Y	0.617396
LP001379	N	0.376486
LP001469	Y	0.984608
LP001915	N	0.326312
LP001917	N	0.157632
LP002031	N	0.209004
LP002053	Y	0.576088
LP002113	N	0.107611
LP002126	N	0.383330
LP002151	N	0.322810
LP002785	N	0.248333
LP002788	N	0.180179
LP002983	Y	0.628469
LP002984	Y	0.719709
LP002990	N	0.496504

Load CSV File to Test

```
import urllib.request
import json
import os
import ssl
import pandas as pd

def allowSelfSignedHttps(allowed):
    # Bypass the server certificate verification on client side
    if allowed and not os.environ.get('PYTHONHTTPSVERIFY', '') and getattr(ssl, '_create_unverified_context', None):
        ssl._create_default_https_context = ssl._create_unverified_context

allowSelfSignedHttps(True)  # This line is needed if you use self-signed certificate in your scoring service.

# Load CSV data
csv_file_path = "loan_data_test.csv"  # Replace with your CSV file path
df = pd.read_csv(csv_file_path)
# Convert dataframe to the required JSON format
input_data = df.to_dict(orient="records")  # Convert rows to list of dictionaries
# Structure it according to the model input schema
data = {"Inputs": {"input1": input_data}, "GlobalParameters": {}}
# Convert to JSON string
body = str.encode(json.dumps(data))

url = 'http://6d24c52d-ba6a-4107-826d-04468fac646d.eastus.azurecontainer.io/score'
api_key = '2wMrAKx6lr5nAMd93WFyZP3ckdNCpa88'
if not api_key:
    raise Exception("A key should be provided to invoke the endpoint")

headers = {'Content-Type': 'application/json', 'Authorization': ('Bearer ' + api_key)}

req = urllib.request.Request(url, body, headers)

try:
    response = urllib.request.urlopen(req)

    # Decode the response and convert to JSON
    result = json.loads(response.read().decode('utf-8'))

    # Extract predictions (adjusting based on API response structure)
    if "Results" in result and "WebServiceOutput0" in result["Results"]:
        predictions = result["Results"]["WebServiceOutput0"]

        # Convert to DataFrame and display the results in a tabular format
        result_df = pd.DataFrame(predictions)
        print("\nLoan Approval Predictions:\n")
        print(result_df.to_string(index=False))  # Print formatted table without index
    else:
        print("Unexpected response format:", result)

except urllib.error.HTTPError as error:
    print("The request failed with status code:", error.code)
    print(error.info())
    print(error.read().decode("utf8", 'ignore'))
```

Input Data Manually

```
import urllib.request
import json
import os
import ssl

def allowSelfSignedHttps(allowed):
    # bypass the server certificate verification on client side
    if allowed and not os.environ.get('PYTHONHTTPSVERIFY', '') and getattr(ssl, '_create_unverified_context', None):
        ssl._create_default_https_context = ssl._create_unverified_context

allowSelfSignedHttps(True) # this line is needed if you use self-signed certificate in your scoring service.

# Request data goes here
# The example below assumes JSON formatting which may be updated
# depending on the format your endpoint expects.
# More information can be found here:
# https://docs.microsoft.com/azure/machine-learning/how-to-deploy-advanced-entry-script
data = {
    "Inputs":{
        "input1": [
            {
                'Loan_ID': 'LP001233',
                'Gender': 'Male',
                'Married': 'Yes',
                'Dependents': 0,
                'Education': 'Graduate',
                'Self_Employed': 'No',
                'ApplicantIncome': 1075000,
                'CoapplicantIncome': 50000,
                'LoanAmount': 20,
                'Loan_Amount_Term': 120,
                'Credit_History': 1,
                'Property_Area': 'Semiurban'

            },
            {
                'Loan_ID': 'LP001036',
                'Gender': 'Female',
                'Married': 'No',
                'Dependents': 0,
                'Education': 'Graduate',
                'Self_Employed': 'No',
                'ApplicantIncome': 3510,
                'CoapplicantIncome': 0,
                'LoanAmount': 76,
                'Loan_Amount_Term': 360,
                'Credit_History': 0,
                'Property_Area': 'Urban'
```

Business Suggestions

- **Expand Data Collection**
  Incorporate employment history, debt-to-income ratio, and credit data to enhance model accuracy.
- **Feature Engineering**
  Develop financial ratios such as loan-to-value to enhance predictive capabilities.
- **Model Optimization**
  Adjust hyperparameters and test advanced algorithms to boost performance.
- **User Feedback Integration**
  Gather insights from loan officers and customers to refine and enhance predictions.
- **Regular Model Updates**
  Continuously retrain the model with updated data to stay aligned with market trends and customer behaviors.

Thank you

Li Wu
Metro College of Technology
Jan 27, 2025