University of Bolton

Teaching Intensive, Research Informed

Exploring Borrower Reliability in Predicting Loan Repayment Ability

Vassileios Gousetis - 2332861
CLD6001: Undergraduate Research Project

Supervisor: Mr. George Prokopakis

2nd Reader: Dr. Anastasios Liapakis

Problem Statement: Challenges in Credit Risk Prediction

- Slow computation in large financial datasets.
- Low accuracy of established techniques.
- Resource-intensive models.
- Limited use of advanced feature engineering methods (e.g., feature creation, scaling, normalization).

Current Solutions in Credit Risk Prediction

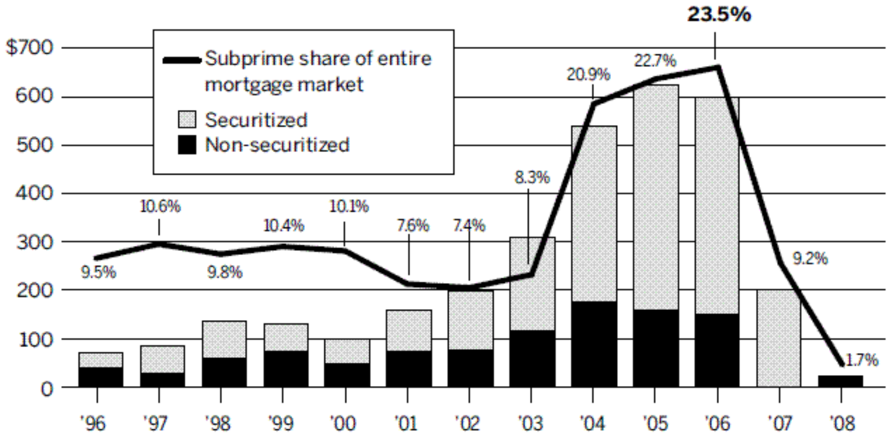
- Machine learning models in use: Support Vector Machines, Naïve Bayes, KNN.
- Emerging approaches: Deep Learning and Neural Networks (moderate success).
- Goal: Improve prediction accuracy, minimize credit losses, and optimize lending.

The 2008 Financial Crisis: Lessons for Credit Risk Modeling

Subprime Mortgage Originations

In 2006, $600 billion of subprime loans were originated, most of which were securitized. That year, subprime lending accounted for 23.5% of all mortgage originations.

IN BILLIONS OF DOLLARS



NOTE: Percent securitized is defined as subprime securities issued divided by originations in a given year. In 2007, securities issued exceeded originations.

SOURCE: Inside Mortgage Finance

Lessons Learnt

- Failures of risk models.
- Need for advanced credit risk tools and regulatory reforms.

The Crisis

- Collapse of major institutions and banks.
- Housing bubble and risky derivatives.

Research Objective: Advancing Credit Risk Models

Focus Areas

- Reduce **computation** time.
- Improve prediction **accuracy**.
- Utilize advanced **feature engineering** techniques."

Literature Review: Machine Learning in Risk Modeling (1/3)

Ji (2023): Proposed a risk rating system for evaluating loan repayment outcomes in commercial banks.

- Machine learning models Used: **XGBoost**, **LightGBM**, **Trees**
- Performance Metrics: **F1 score**, **accuracy**, **ROC AUC**
- The Accuracy Ji reported reached 75%

Literature Review: Advances in Credit Risk Prediction (2/3)

Smith et al. (2022): Investigated deep learning approaches for credit scorin.

- The research Focused on ANN, CNN, and hybrid models
- Results: ANN achieved 87.2% accuracy, outperforming traditional ML models

Literature Review: Different Methods to Predict Loan Default (3/3)

Bhandart, T. et al, researched the use of Different machine learning techniques to predict loan default.

The algorithms and their prediction rate are:

- ANN (Artificial Neural Network) –85.88%
- Support Vector Machines –85%
- Random Forest -86.32%

Proposed Solution: Advanced Credit Risk Modeling

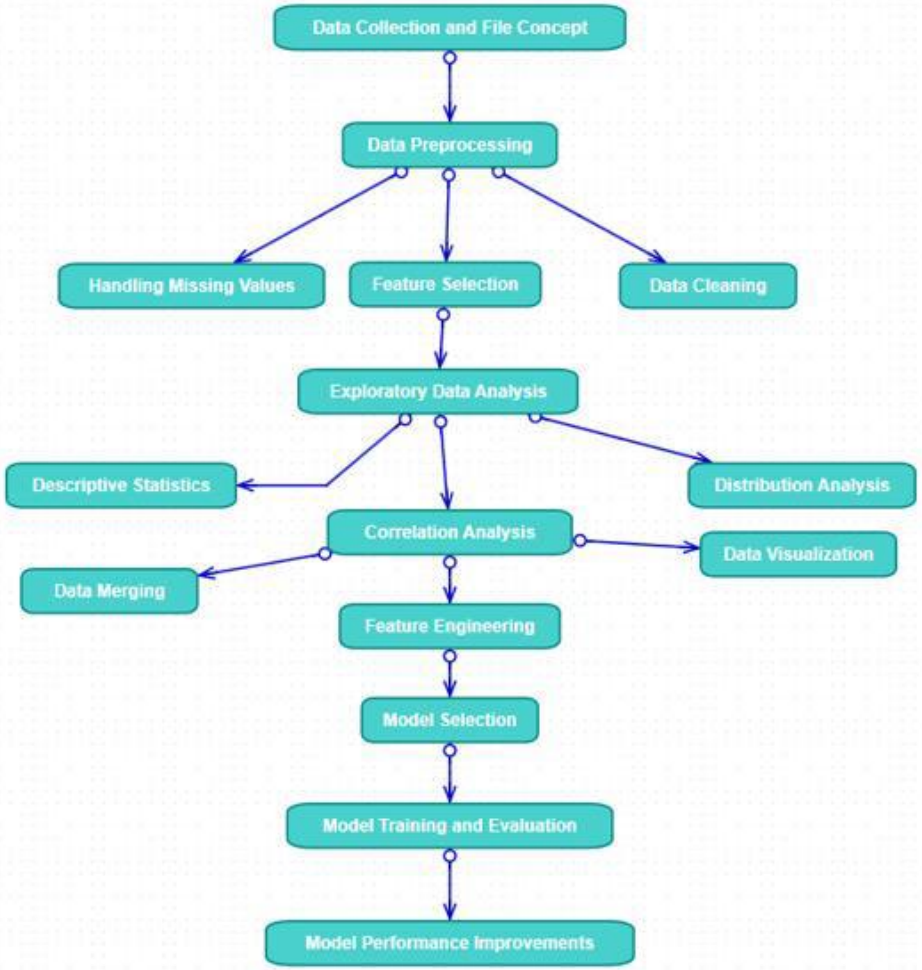
- Develop a neural network-based model for credit risk prediction
- Enhance predictive accuracy with advanced feature engineering (feature aggregation, scaling, and normalization)
- Optimize computational efficiency using big data technologies

Technology Stack

- **Database**: MySQL -> To store the data avoiding the use of CSV
- **Data Analytics**: Python -> To perform exploratory data analysis and create different charts
- **Model Develpoment**: PySpark -> Using Distributed technology to reduce computational resource consumption and reduce running time

Project Methodology: Steps to Model Development

- **Dataset Exploration**: Analyze and preprocess raw data from database.
- **Feature Engineering**: Create, scale, and normalize features.
- **Model Design**: Develop a multilayer perceptron neural network.
- **Training & Evaluation**: Opimize performance with metrics like ROC AUC and accuracy.


Data Collection and File Concept
Data Preprocessing
Handling Missing Values
Feature Selection
Data Cleaning
Exploratory Data Analysis
Descriptive Statistics
Distribution Analysis
Correlation Analysis
Data Visualization
Data Merging
Feature Engineering
Model Selection
Model Training and Evaluation
Model Performance Improvements

Dataset Exploration: Analyzing Given-Input Data

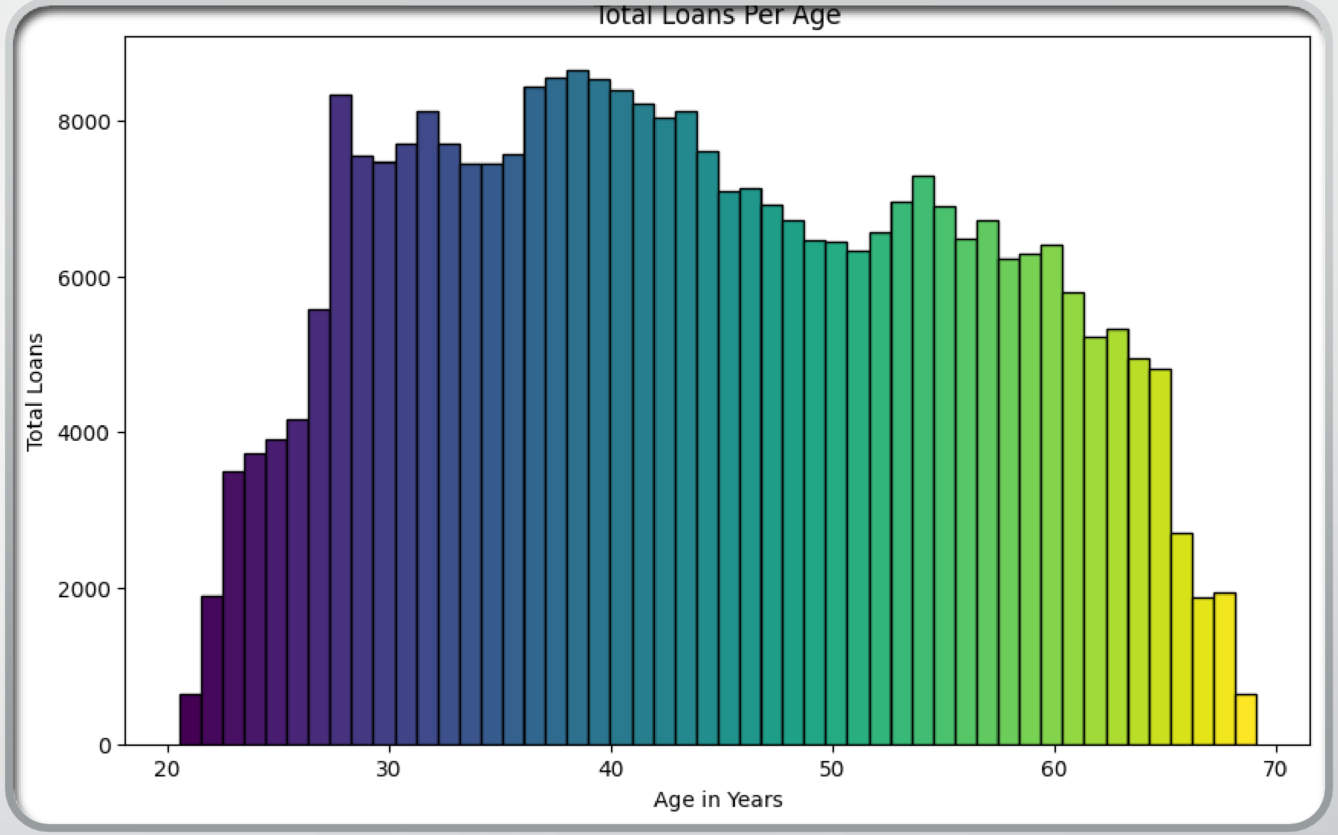
The data is sourced from **Kaggle**, and specifically from a Contest, organized by **Home Credit Group**

Multiple files containing loan and borrower information, such as:

- Loan applications (Both Train and Test are given)
- Bureau records and balance
- POS cash balance, previous applications
- Installment payment history

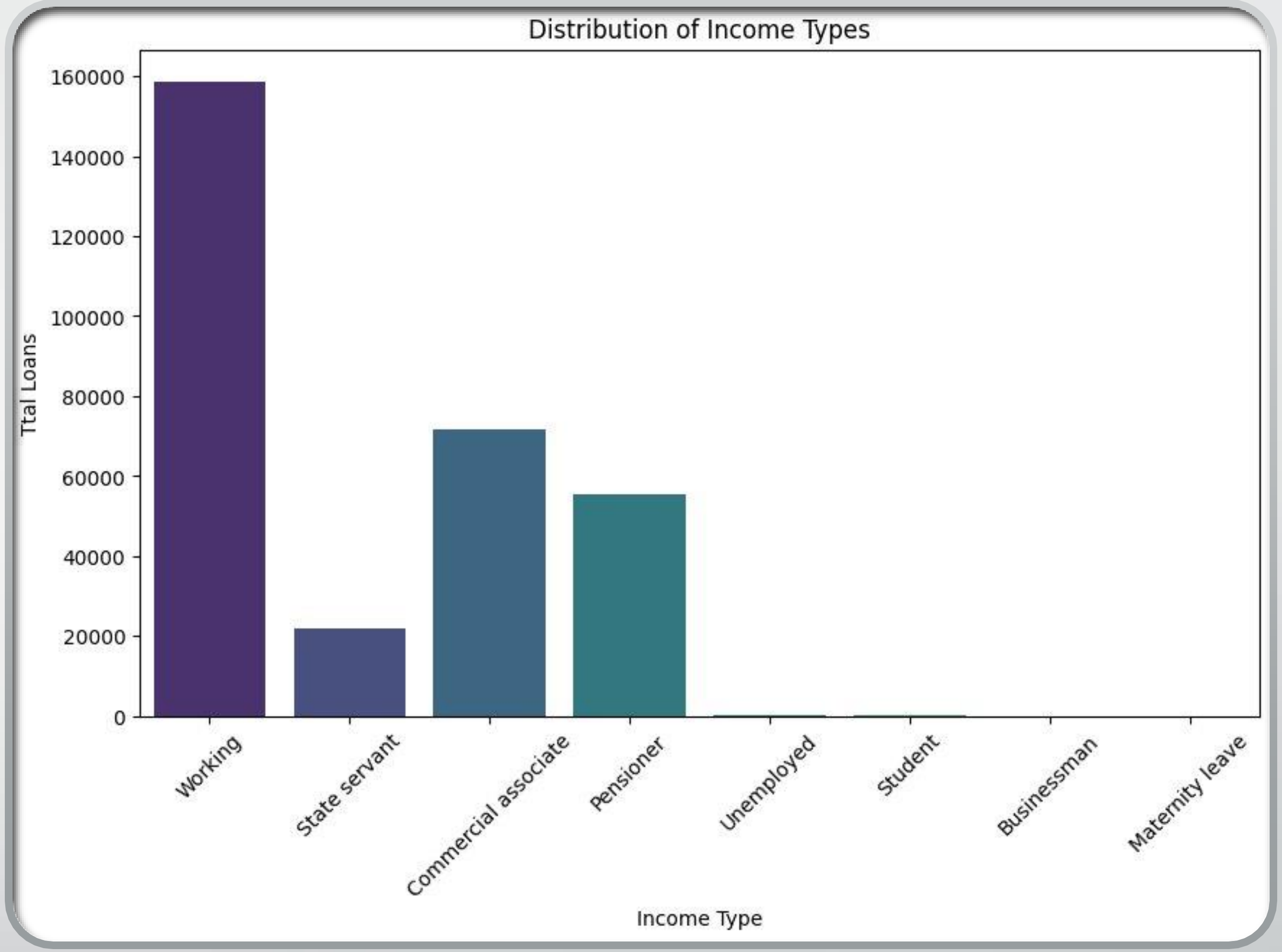
Exploratory Data Analysis: Loan Distribution by Age

- Loans peak between ages 38–43
- Younger and older borrowers have lower loan densities



Exploratory Data Analysis: Income Types of Borrowers

- Most of the Borrowers belong to the working class, commercial associates, and pensioners
- Minimal loan demand from unemployed individuals, students, and maternity leave cases

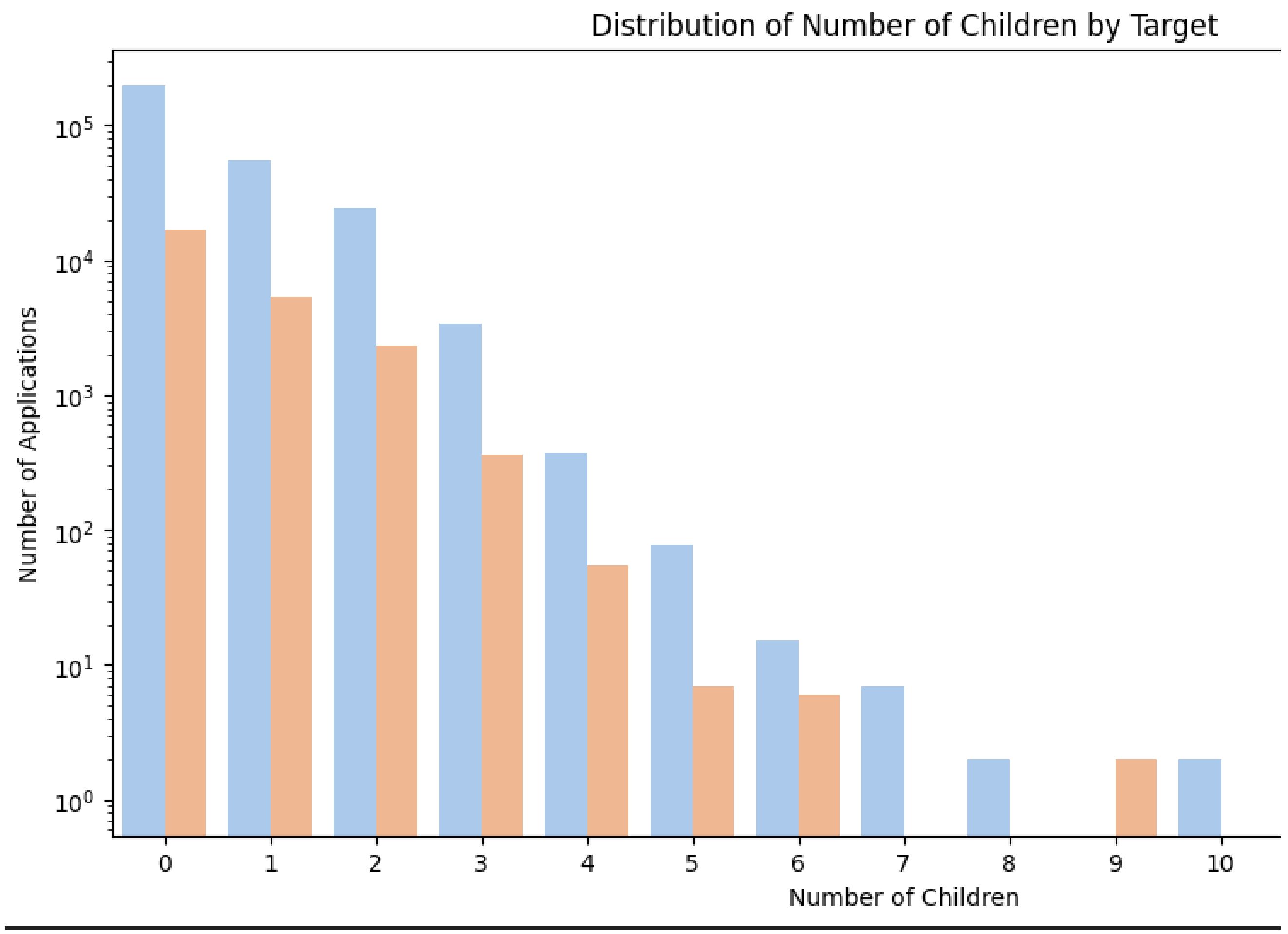


Exploratory Data Analysis: Loan Default by Number of Children

- Default risk increases slightly with the number of children
- Households without financial problems are more common across all child categories

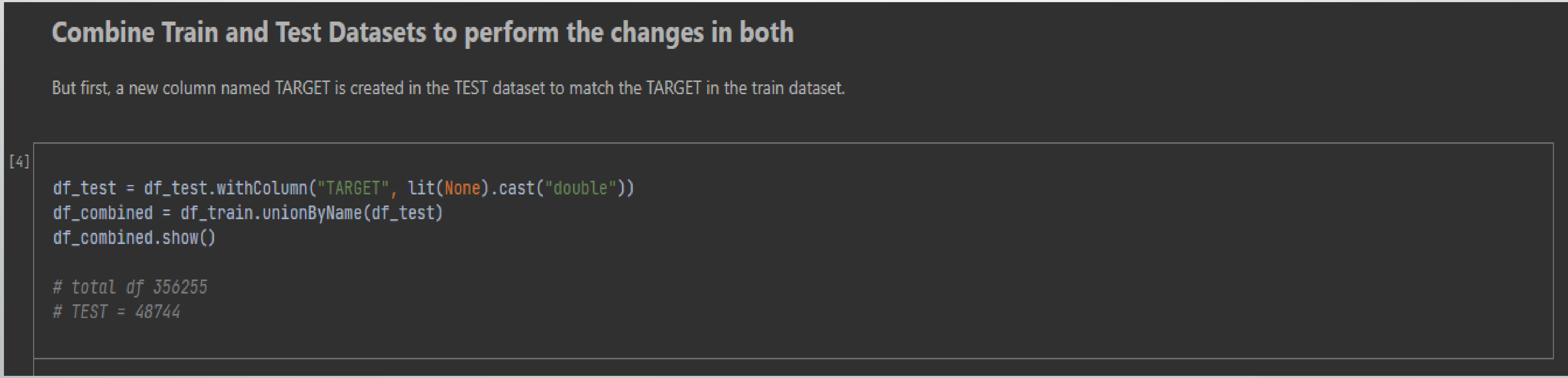
Blue = No Financial Problems

Orange = All Other



Feature Engineering: Data Combination

- Unified multiple datasets for comprehensive analysis
- Linked datasets based on borrower IDs to create a consolidated dataset



Combine Train and Test Datasets to perform the changes in both

But first, a new column named TARGET is created in the TEST dataset to match the TARGET in the train dataset.

[4]

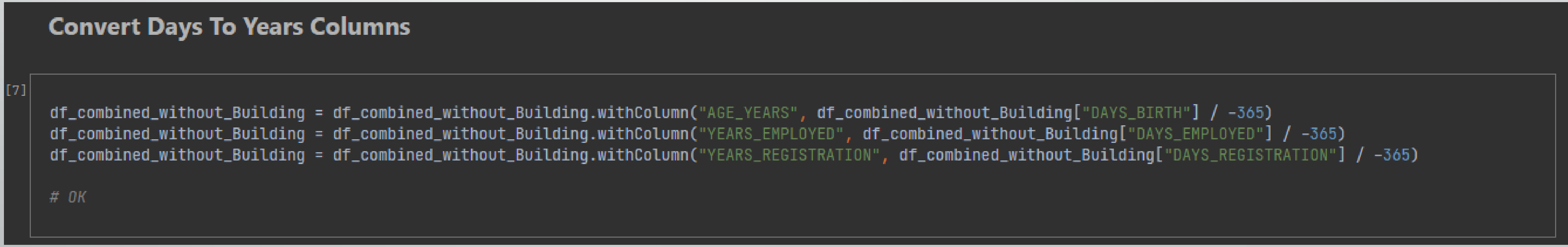
```
df_test = df_test.withColumn("TARGET", lit(None).cast("double"))
df_combined = df_train.unionByName(df_test)
df_combined.show()

# total df 356255
# TEST = 48744
```

Feature Engineering: Handling Missing Values

- Identified missing data across key columns, chosen by literature and common logic
- Replaced missing numeric values with their mean
- Filled missing categorical values with their mode

Feature Engineering: Data Conversion and Discrepancy Fixes

- Replaced negative values using Equatations
  - $X_2 = -(-X_1)$
- Converted days into years
  - $X_2 = X_1 / (-365)$
- Ensured final null handling using mean/mode imputation again



Convert Days To Years Columns

[7]

```
df_combined_without_Building = df_combined_without_Building.withColumn("AGE_YEARS", df_combined_without_Building["DAYS_BIRTH"] / -365)
df_combined_without_Building = df_combined_without_Building.withColumn("YEARS_EMPLOYED", df_combined_without_Building["DAYS_EMPLOYED"] / -365)
df_combined_without_Building = df_combined_without_Building.withColumn("YEARS_REGISTRATION", df_combined_without_Building["DAYS_REGISTRATION"] / -365)

# OK
```

Deep Learning Implementation: Encoding Categorical Values

- Converted categorical variables into numerical format to be compatible with neural network structure/requirements
- Used encoding techniques

  - One-hot encoder for non-ordinal categories

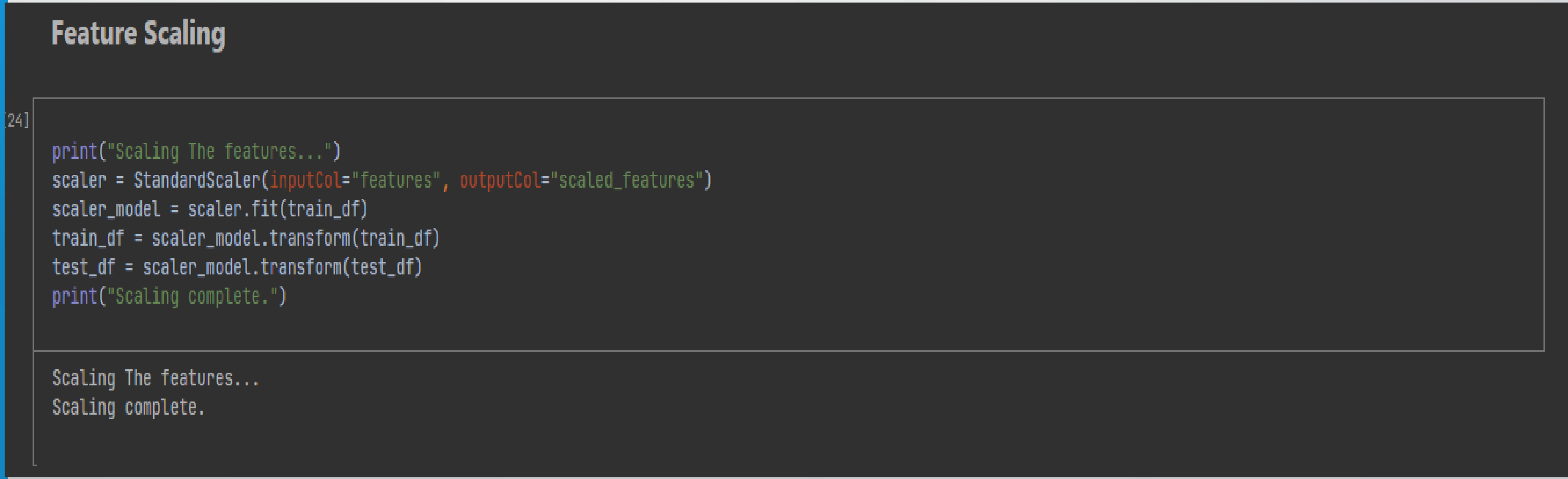
  - Label encoder for ordinal categories

Deep Learning Implementation: Feature Vector and Scaling

- Created feature vectors to represent independent variables for the model
- Normalized data using scaling techniques to standardize value ranges

Techniques Used:

- VectorAssembler for feature combination
- StandardScaler for normalization



Feature Scaling

[24]

```
print("Scaling The features...")
scaler = StandardScaler(inputCol="features", outputCol="scaled_features")
scaler_model = scaler.fit(train_df)
train_df = scaler_model.transform(train_df)
test_df = scaler_model.transform(test_df)
print("Scaling complete.")
```

```
Scaling The features...
Scaling complete.
```

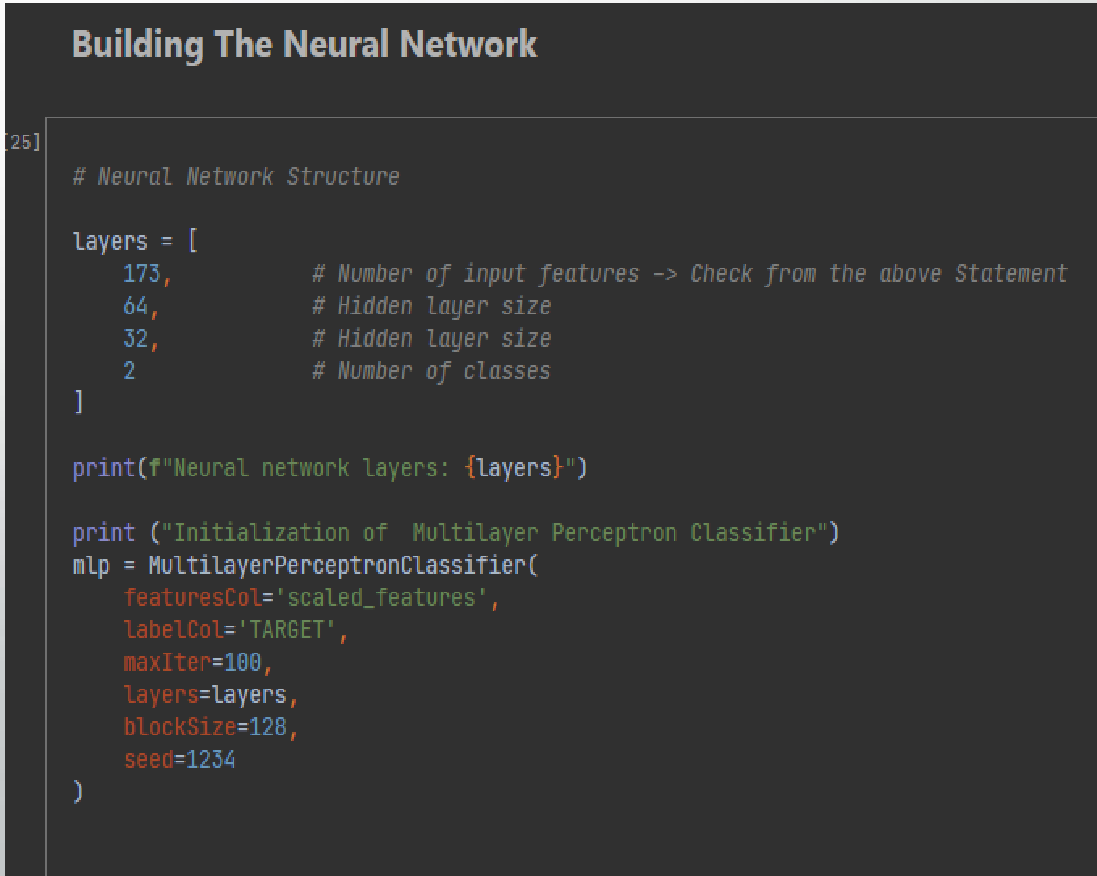
Neural Network: Multilayer Perceptron (MLP)

Structure:

- 4 layers
- 2 hidden layers
- Input: 173 features
- Output: 2 classes (0 and 1)

Parameters:

- MaxIter: Maximum iterations through train dataaset
- BlockSize: Batch size per iteration
- Seed: Ensures consistent randomization



Building The Neural Network

[25]

```
# Neural Network Structure

layers = [
    173,            # Number of input features -> Check from the above Statement
    64,             # Hidden layer size
    32,             # Hidden layer size
    2               # Number of classes
]

print(f"Neural network layers: {layers}")

print ("Initialization of  Multilayer Perceptron Classifier")
mlp = MultilayerPerceptronClassifier(
    featuresCol='scaled_features',
    labelCol='TARGET',
    maxIter=100,
    layers=layers,
    blockSize=128,
    seed=1234
)
```

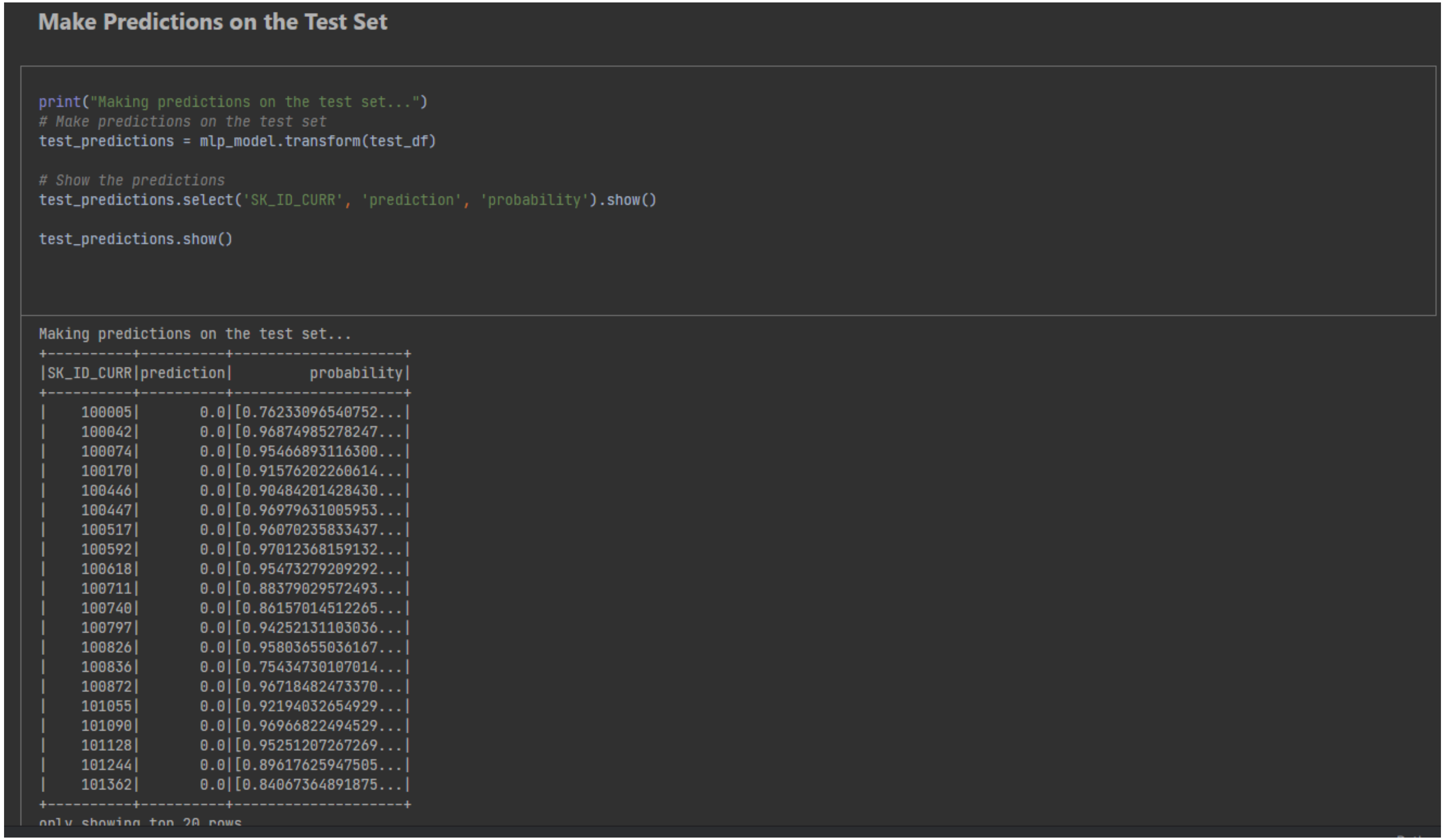
Model Training and Evaluation: Metrics and Discrepancy

Performance Metrics:

- Accuracy: 0.5 (affected by imbalance)
- ROC AUC: 0.78 (indicates good distinction between classes)

Scenarios of why Accuracy metric is not as expected:

- Highly imbalanced dataset (**10x** more non-defaulting cases)
- Accuracy metric not ideal for imbalanced data (Found through literature
- Dependence on threshold adjustments affects accuracy values



Make Predictions on the Test Set

```
print("Making predictions on the test set...")
# Make predictions on the test set
test_predictions = mlp_model.transform(test_df)

# Show the predictions
test_predictions.select('SK_ID_CURR', 'prediction', 'probability').show()

test_predictions.show()
```

```
Making predictions on the test set...
+----------+----------+--------------------+
|SK_ID_CURR|prediction|         probability|
+----------+----------+--------------------+
|    100005|       0.0|[0.76233096540752...|
|    100042|       0.0|[0.96874985278247...|
|    100074|       0.0|[0.95466893116300...|
|    100170|       0.0|[0.91576202260614...|
|    100446|       0.0|[0.90484201428430...|
|    100447|       0.0|[0.96979631005953...|
|    100517|       0.0|[0.96070235833437...|
|    100592|       0.0|[0.97012368159132...|
|    100618|       0.0|[0.95473279209292...|
|    100711|       0.0|[0.88379029572493...|
|    100740|       0.0|[0.86157014512265...|
|    100797|       0.0|[0.94252131103036...|
|    100826|       0.0|[0.95803655036167...|
|    100836|       0.0|[0.75434730107014...|
|    100872|       0.0|[0.96718482473370...|
|    101055|       0.0|[0.92194032654929...|
|    101090|       0.0|[0.96966822494529...|
|    101128|       0.0|[0.95251207267269...|
|    101244|       0.0|[0.89617625947505...|
|    101362|       0.0|[0.84067364891875...|
+----------+----------+--------------------+
only showing top 20 rows
```

Find the Data, Code and the Presentation

Data: https://www.kaggle.com/competitions/home-credit-default-risk/data

Code and Presentation: https://github.com/Erevos-IV/Predicting-Loan-Outcome?tab=readme-ov-file

Conclusion: Results of Research and Future Work

Key Takeaways and Results:

- Advanced credit risk models, like neural networks, show potential for improved predictions
- Feature engineering enhances data representation and model performance
- ROC AUC is a more reliable metric for imbalanced datasets

Future Work and Improvements:

- Address class imbalance through sampling techniques (e.g. SMOTE)
- Test additional neural network algorithms for comparison (e.g Convolutional neural networks)
- Optimize (or add more) hyperparameters for better performance

Acknowledgements

- Supervisor: Mr. George Prokopakis
- Second Reader: Dr. Anastasios Liapakis
- Special Person: Ms Diareme Konstantina
- Support from family, friends, and colleagues

About the Presenter

- Name: Vassileios Gousetis
- Degree: Bachelor of Science in Data Analytics
- Current Professional Role: Data Engineer and Private in Hellenic Military Units Administration Office of Research and Informations in Cyprus
- Contact: +30 6983227655, Vasilhsgxr5000@gmail.com

  - Linkedin: Vasileios Gousetis

Time for Questions

Thank you for your Attention