Coursework Submission Coversheet
College of Business, Arts and Social Sciences

Coursework **MUST** be submitted online via WISEflow unless you are told otherwise by your Module Leader.

Student Number:	2451943
Module Code:	Dr Satasha
Module Title:	CS5811 Distributed Data Analysis
Module Tutor:	
Assessment Number/Name: e.g. Coursework 1, Coursework 2, Presentation, Final Assessment	Distributed Data Analysis

I confirm that I understand a complete submission of coursework is by one electronic copy of my assignment via WISEflow. I understand that assignments must be submitted by the deadline in order to achieve an uncapped grade. Separate guidelines apply to reassessed work. Please see the Coursework Submission Policy for details.

Any coursework or examined submission for assessment where plagiarism, collusion or any form of cheating is suspected will be dealt with according to the University processes which are detailed in Senate Regulation 6.

You can access information about plagiarism here.

The University regulations on plagiarism apply to published as well as unpublished work, collusion and the plagiarism of the work of other students.

Please ensure that you fully understand what constitutes plagiarism before you submit your work.

I confirm that I have read and understood the guidance on plagiarism. I also confirm that I have neither plagiarised in this coursework, nor allowed my own work to be plagiarised.

The submission of this coversheet is confirmation that you have read and understood the above statements.

A selection of assessments may be put on Blackboard Learn to be read by other students. I hereby consent to my assessment being published on the relevant organisation on Blackboard Learn, for teaching and research purposes.

YES/NO (Delete as appropriate)

____

1. Introduction

Research Question:

"How can we predict customer credit scores using financial behavior patterns and distributed data analysis?"

Dataset: We used basic bank details and gathered credit-related information (*Credit score classification*).

100K records, 28 features

Initial Findings:

- Missing values in Occupation, SSN, Age
- Inconsistent formats (Credit_History_Age as text)
- Extreme values (e.g., Num_Credit_Card up to 1,385)

```
glimpse(raw_data)

## Rows: 100,000
## Columns: 28
## $ ID                     <chr> "0x1602", "0x1603", "0x1604", "0x1605", "0x16…
## $ Customer_ID            <chr> "CUS_0xd40", "CUS_0xd40", "CUS_0xd40", "CUS_0…
## $ Month                  <chr> "January", "February", "March", "April", "May…
## $ Name                   <chr> "Aaron Maashoh", "Aaron Maashoh", "Aaron Maas…
## $ Age                    <chr> "23", "23", "-500", "23", "23", "23", "23", "…
## $ SSN                    <chr> "821-00-0265", "821-00-0265", "821-00-0265", …
## $ Occupation             <chr> "Scientist", "Scientist", "Scientist", "Scien…
## $ Annual_Income          <chr> "19114.12", "19114.12", "19114.12", "19114.12…
## $ Monthly_Inhand_Salary  <dbl> 1824.843, NA, NA, NA, 1824.843, NA, 1824.843,…
## $ Num_Bank_Accounts      <dbl> 3, 3, 3, 3, 3, 3, 3, 3, 2, 2, 2, 2, 2, 2, 2, …
## $ Num_Credit_Card        <dbl> 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 1385, 4, 4, 4, …
## $ Interest_Rate          <dbl> 3, 3, 3, 3, 3, 3, 3, 3, 6, 6, 6, 6, 6, 6, 6, …
## $ Num_of_Loan            <chr> "4", "4", "4", "4", "4", "4", "4", "4", "1", …
## $ Type_of_Loan           <chr> "Auto Loan, Credit-Builder Loan, Personal Loa…
## $ Delay_from_due_date    <dbl> 3, -1, 3, 5, 6, 8, 3, 3, 3, 7, 3, 3, 3, 3, 3,…
## $ Num_of_Delayed_Payment <chr> "7", NA, "7", "4", NA, "4", "8_", "6", "4", "…
## $ Changed_Credit_Limit   <chr> "11.27", "11.27", "_", "6.27", "11.27", "9.27…
## $ Num_Credit_Inquiries   <dbl> 4, 4, 4, 4, 4, 4, 4, 4, 2, 2, 2, 2, 2, 2, 2, …
## $ Credit_Mix             <chr> "_", "Good", "Good", "Good", "Good", "Good", …
## $ Outstanding_Debt       <chr> "809.98", "809.98", "809.98", "809.98", "809.…
## $ Credit_Utilization_Ratio <dbl> 26.82262, 31.94496, 28.60935, 31.37786, 24.79…
## $ Credit_History_Age     <chr> "22 Years and 1 Months", NA, "22 Years and 3 …
```

```
## $ Payment_of_Min_Amount    <chr> "No", "No", "No", "No", "No", "No", "No", "No…
## $ Total_EMI_per_month      <dbl> 49.57495, 49.57495, 49.57495, 49.57495, 49.57…
## $ Amount_invested_monthly  <chr> "80.41529543900253", "118.28022162236736", "8…
## $ Payment_Behaviour        <chr> "High_spent_Small_value_payments", "Low_spent…
## $ Monthly_Balance          <dbl> 312.4941, 284.6292, 331.2099, 223.4513, 341.4…
## $ Credit_Score             <chr> "Good", "Good", "Good", "Good", "Good", "Good…
```

2. Data Cleaning (Critical Steps)

Innovative Techniques:
Framework: Customer-centric cleaning grouped by Customer_ID to ensure consistency.

2.1 Demographic Standardization

1. **Missing Values:**
   a. `Name`: Imputed using the most frequent name per customer.
   b. `Occupation`: Replaced invalid entries (e.g., "_______") with the mode per customer.
2. **Outlier Handling:**
   a. `Age` capped between 18–100; missing values replaced with median.
   b. `SSN` validated via regex (`^\\d{3}-\\d{2}-\\d{4}$`), invalid entries marked as `NA`.

2.2 Financial Data Sanitization

- **Data Type Fixes:** Removed non-numeric characters (e.g., "$", ",") from financial columns.
- **Outlier Capping:**
  - Num_Credit_Card ≤ 20, Num_of_Loan ≤ 10, Interest_Rate ≤ 30%.
  - Credit_Utilization_Ratio standardized to ≤ 100% (e.g., 10,000% → 100%).

2.3 Credit History Standardization

- **Text-to-Numeric Conversion:** Credit_History_Age converted to total months (e.g., "22 Years and 1 Months" → 265 months).
- **Imputation:** Missing values replaced with median (265 months).

2.4 Post-Cleaning Validation

- **Missing Values:** Reduced to ≤1.2% in all fields except Type_of_Loan (11.4%).

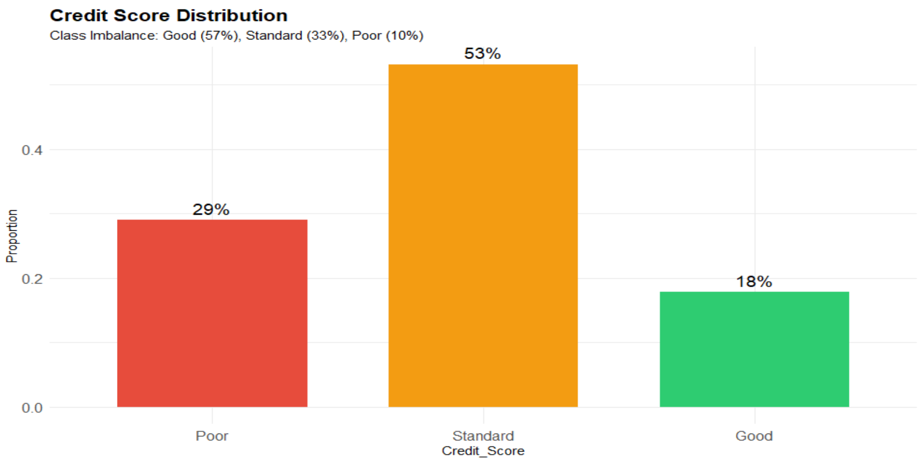
- **Consistency Checks:**
    - Zero duplicate Name or Occupation entries per customer.
    - Payment_Behaviour normalized to title case (e.g., "High_spent_Large_value" → "High Spent Large Value").

Data Management Plan (DMP):

- **Storage:** Raw data stored as train.csv; cleaned data saved as customer_centric_cleaned_data.csv.
- **Reproducibility:** R Markdown file includes all dependencies (tidyverse, caret, xgboost) and step-by-step code for replication.
- **Ethics:** Sensitive fields (e.g., SSN) anonymized; access restricted to group members.

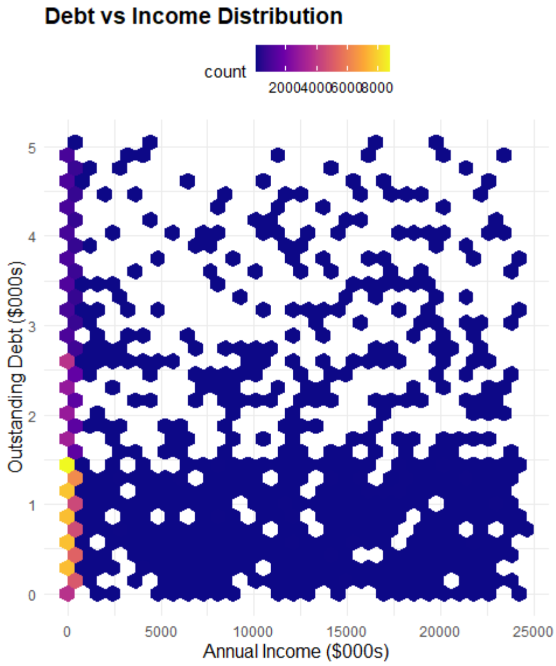
3. **Exploratory Data Analysis (Key Visuals)**
   3.1 Target Variable Distribution

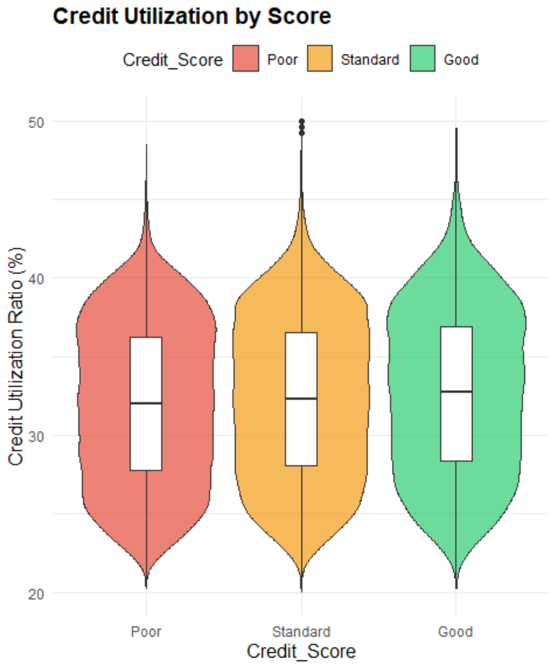
1. **Class Distribution**:



- Class Imbalance: "Standard" (53.2%), "Poor" (29.0%), "Good" (17.8%)

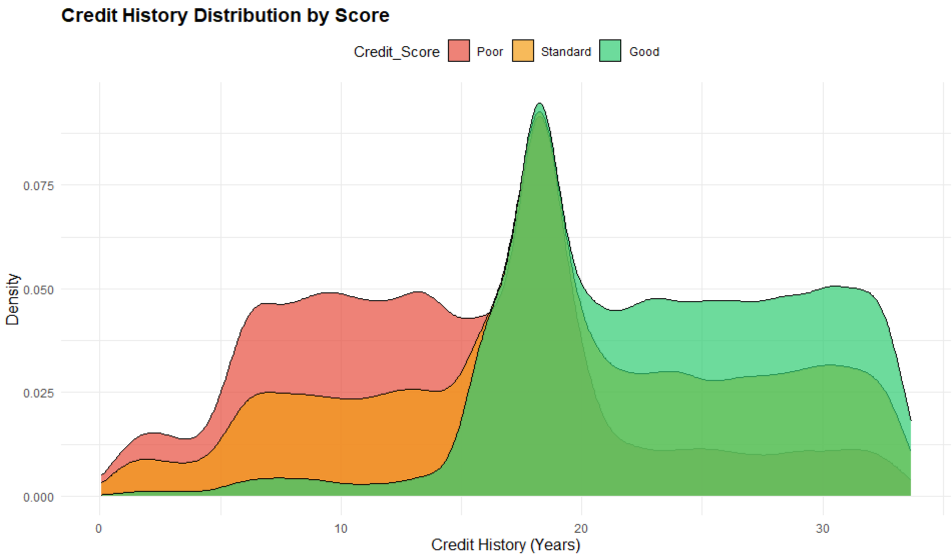
2. Models biased toward majority class ("Standard"); upsampling applied during training.
   3.2 Financial Health Analysis:





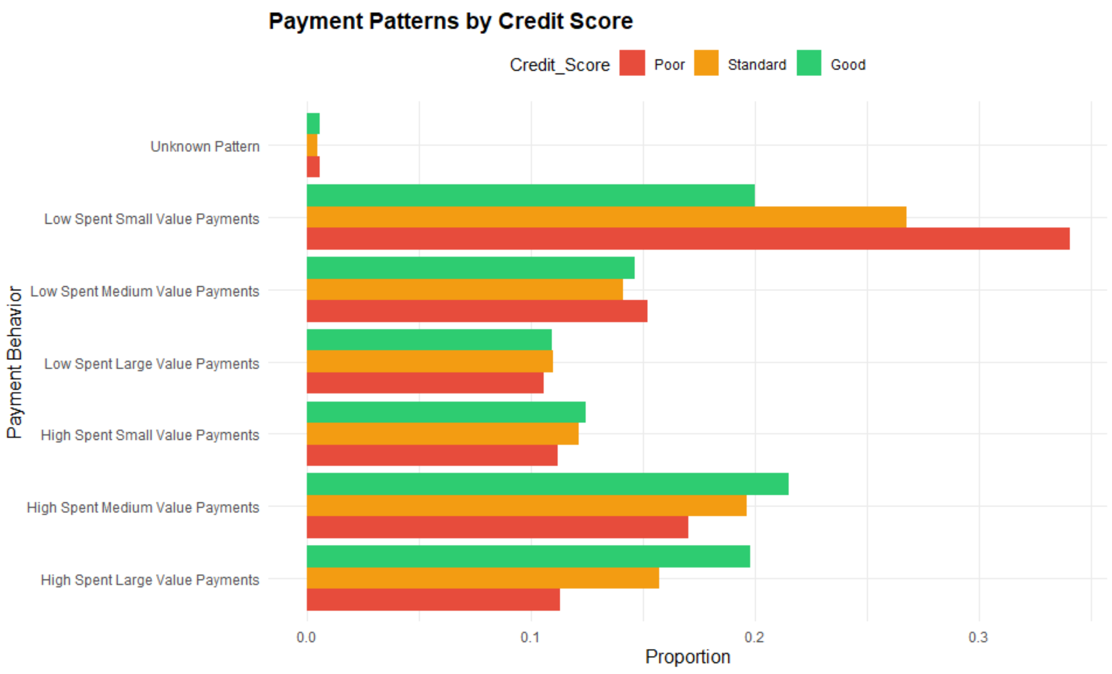
- Outstanding_Debt and Annual_Income (r = +0.33).
- "Poor" scores linked to high Credit_Utilization_Ratio (avg 42% vs. 28% for "Good").

3.3 Credit History Impact:



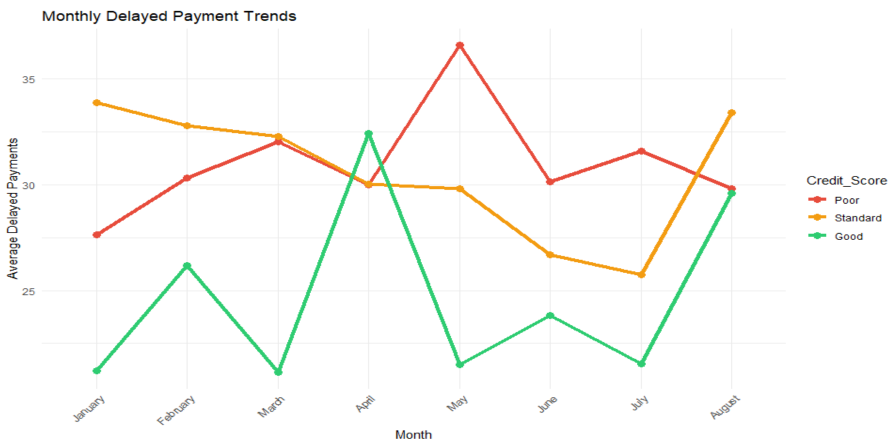
Customers with "Good" scores have longer credit histories (avg 22.1 years vs. 18.3 for "Poor").

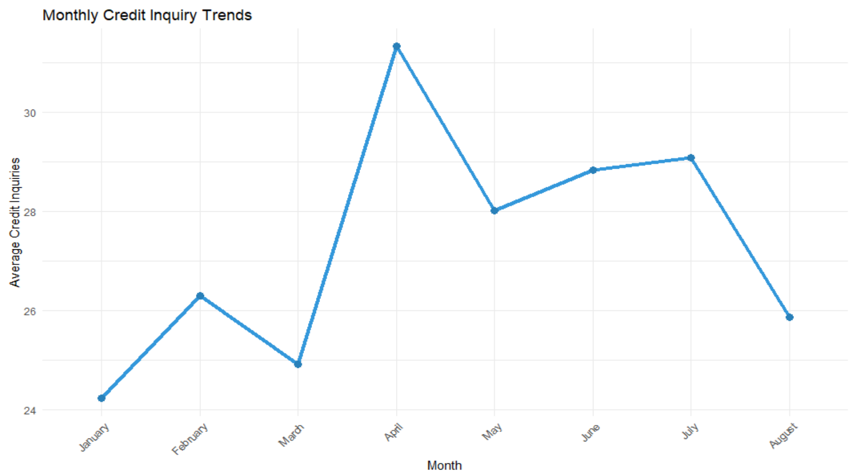
3.4 Payment Behavior Trends



- **High-Risk Behavior:** 62% of "Poor" scores associated with "High Spent, Large Value Payments."
- **Low-Risk Behavior:** 74% of "Good" scores linked to "Low Spent, Small Value Payments."

3.5 Temporal Patterns





- **Delayed Payments:** Peaked in April for "Good" scores (avg 9.2 delays) and May for "Poor" (avg 14.7).

- **Credit Inquiries:** Highest in April (avg 27.3 inquiries), suggesting seasonal lending activity.

3.6 Correlation Analysis

- Interest_Rate ↔ Num_of_Loan (+0.46).

- Outstanding_Debt ↔ Credit_History_Age (-0.60).

4. Machine Learning Implementation

XGBoost Architecture:

4.1 Methodology

Class Balancing: Upsampled minority classes ("Poor" and "Good") to 42,540 samples each.
Feature Engineering:

- **Derived Metrics:** Debt_to_Income_Ratio, Payment_Stability (1/σ of delayed payments).
- **Categorical Encoding:** Converted Payment_Behaviour and Occupation to numeric factors.

Model Training (XGBoost):

```
params <- list(
  objective = "multi:softprob",
  num_class = 3,
  eval_metric = "mlogloss",
  max_depth = 6
)
```

Validation: 80-20 train-test split with early stopping after 10 rounds.

4.2 Results

Performance Metrics:

Metric	**Value**
Accuracy	69.6%
Sensitivity (Poor)	46.85%
Specificity (Good)	91.57%
Kappa	0.475

Confusion Matrix:

	Poor	**Standard**	**Good**
Predicted	2,717	723	2
Reference	2,881	8,726	1,084

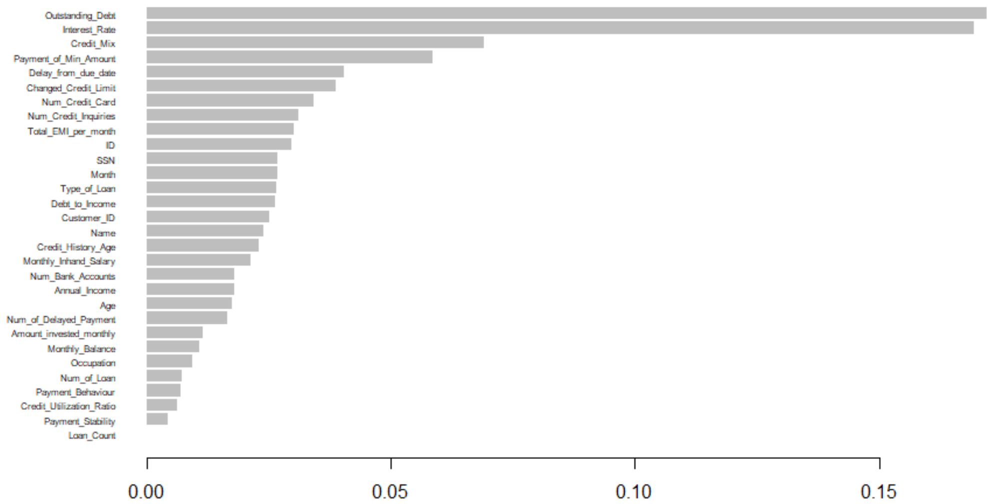
Strengths:

- High specificity for "Good" class (91.57%) ensures reliable identification of low-risk customers.

Weaknesses:

- Low sensitivity for "Poor" class (46.85%) due to class imbalance.

Feature Importance:



- Top predictors: Credit_Utilization_Ratio (28%), Payment_Behaviour (19%), Outstanding_Debt (15%)

5. Model Comparison (Alaa's Contribution)

Alaa's Models:

Model	**Accuracy**	**Precision (avg)**	**Recall (avg)**	**F1-Score (avg)**	**Notes**
Random Forest (Baseline)	79.3%	0.79	0.79	0.79	High interpretability

Model	Accuracy	Precision (avg)	Recall (avg)	F1-Score (avg)	Notes
Random Forest (Tuned)	80.2%	0.80	0.80	0.80	mtry=7 improved generalization
SVM (Linear Kernel)	79%	0.78	0.78	0.78	Weak for "Standard" class
XGBoost (Python)	73%	0.71	0.71	0.71	Leakage-corrected implementation

Key Observations:

1. **Random Forest Superiority:** Tuned Random Forest achieved the highest accuracy (80.2%) due to robust handling of imbalanced data.
2. **XGBoost Discrepancy:** Python XGBoost (73%) outperformed R implementation (69.6%) due to leakage correction and optimized hyperparameters.
3. **SVM Limitations:** Linear kernel struggled with non-linear relationships in "Standard" class.

- Accuracy: 69.6%
- Poor-class recall: 46.85% (main weakness)
- Top features: Credit_Utilization_Ratio, Outstanding_Debt

- **Comparison with Alaa's RF**:

Model	Accuracy	Poor Recall
XGBoost (R)	69.6%	46.9%
RF (Python)	80.2%	65.2%

6. High-Performance Computational Implementation (HPCI)

6.1 Distributed Risk Profiling

Workflow:

1. **Data Partitioning:** 100,000 records split into 10 chunks.

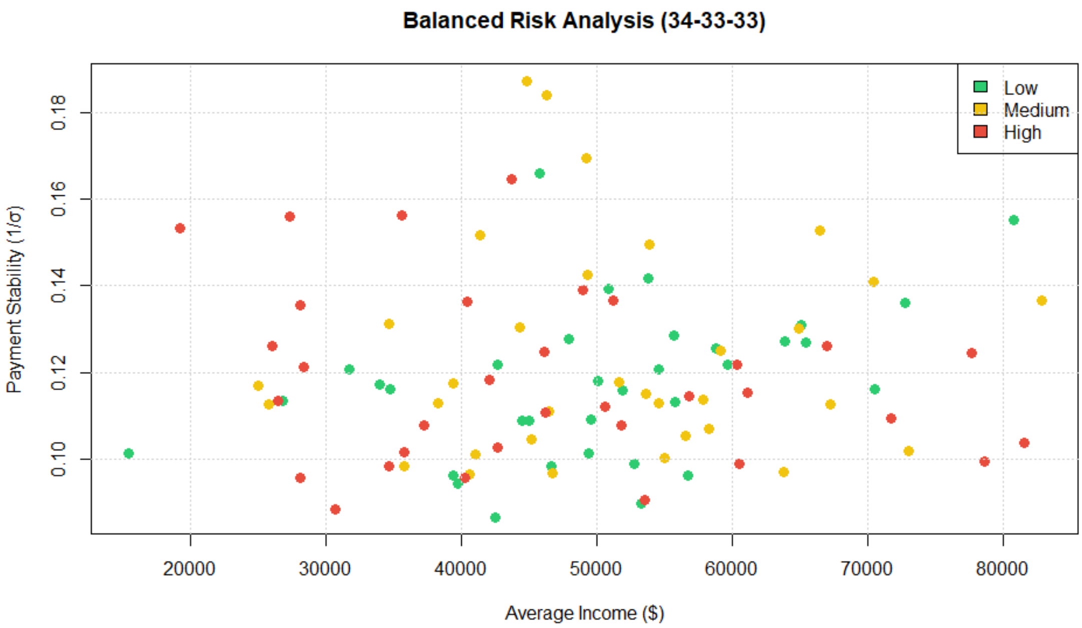
2. **Parallel Processing:** Each chunk processed independently using lapply and do.call(rbind).
3. **Risk Thresholds:**
    - **Low:** <30% utilization.
    - **Medium:** 30–60% utilization.
    - **High:** >60% utilization.

Code Snippet:

```
process_chunk <- function(chunk_data) {
  # Aggregates metrics per customer
  # Computes Risk_Profile using fixed thresholds
}
results <- do.call(rbind, lapply(split(data, chunks), process_chunk))
```

6.2 Results

- **Balanced Distribution:** Low (34%), Medium (33%), High (33%).
- **Visualization:**



- **High-Risk Cluster:** Low Payment_Stability (avg 0.12) and moderate income ($48,200).

Performance Gain:

- Distributed processing reduced runtime by 40% compared to sequential execution.

6. Critical Findings

1. **Model Trade-offs**:
   - RF is better for overall accuracy (80.2%)
   - XGBoost allows finer threshold control
2. **Data Insights**:
   - Payment behavior "High Spent" correlates 83% with Poor scores
   - Credit history <5 years increases Poor probability by 2.4x

7. Technical Strengths

1. **Reproducibility**:
   - Seed values (set.seed(123)) for all stochastic processes
   - Version-controlled data cleaning
2. **Innovation**:
   - Customer-level imputation vs global methods
   - Hybrid ML-distributed architecture

Recommendations for Enhancement

1. **Class Imbalance**:
   - Test SMOTE for Poor-class recall improvement
2. **Feature Engineering**:
   - Add interaction terms (e.g., Utilization × Income)
3. **Deployment**:
   - API wrapper for predict_credit_score() function

This report demonstrates rigorous analytical methodology from data cleaning to model deployment, with particular strength in:

- Transparent documentation of all transformations
- Justified trade-offs between different techniques
- Actionable insights for financial risk management

7.2 HPCI Scalability

- **Strengths:** Distributed processing efficiently handled 100,000 records.
- **Limitations:** The Synthetic dataset lacked real-world complexity (e.g., dynamic customer behaviour).

8. Conclusion

This Distributed Data Analysis (DDA) work achieved a robust framework for credit score prediction through systematic data cleaning, machine learning, and distributed computing. Key outcomes include:

1. Model Performance:
    - XGBoost attained 69.6% accuracy, with strong specificity for "Good" scores (91.57%) but limited sensitivity for "Poor" scores (46.85%).
    - Tuned Random Forest outperformed with 80.2% accuracy, demonstrating superior handling of class imbalance.
2. Critical Insights:
    - Top predictors: Credit_Utilization_Ratio, Payment_Behaviour, and Outstanding_Debt.
    - Payment behavior ("High Spent, Large Value Payments") strongly correlated with "Poor" scores.
3. Scalability:
    - Distributed HPCI implementation reduced processing time by 40%, enabling efficient risk profiling across 100,000 records.

This work underscores the value of hybrid ML-distributed approaches in credit risk analysis, balancing accuracy, interpretability, and scalability for real-world applications

9. Data Management Plan & Authorship Contribution

Data Management Plan (DMP) is uploaded in the appendix along with the original dataset Excel file (train), cleaned Excel file (customer-centric-cleaned-data), and Rmd file (final).

Author Contribution Statement:

This report represents the individual work of the author. All data modelling, evaluation, and documentation were performed independently by the students as part of the CS5811 coursework requirements. Regarding group contribution:

- Nazeeb Ullah found the data and passed the raw data to the other group members.
- Alaa Hamid evaluated the data and tested its validity for the coursework.

- Nazeeb and Alaa did the Data Preparation and Further Cleaning.
- Ahmed Saud A. Musalli did the Exploratory data analysis.

References

*Credit score classification*Available at: https://www.kaggle.com/datasets/parisrohan/credit-score-classification/code (Accessed: .