End-to-End Credit Risk Scoring

From Data Cleaning to Business Dashboard with a Neural Network Model

Or Ben-Haim

August 2025

Python · PyTorch · Neural Networks · Scikit-learn · Pandas · Tableau

Project Overview Credit Risk Prediction

The goal of this project is to predict the level of credit risk for people applying for loans, using their personal and financial information. The dataset used for this analysis is the publicly available Credit Risk Dataset from Kaggle (link). By analyzing features such as income, age, loan amount, and credit history, we trained a deep learning model capable of estimating the likelihood of repayment for each applicant.

This type of predictive modeling is highly valuable for banks, lenders, and fintech companies that need to make quick and reliable decisions when approving or rejecting loan applications. Instead of relying solely on manual review or rigid rule-based systems, our approach leverages machine learning specifically a neural network to deliver a more flexible and accurate risk assessment.

Predictions were compared with actual loan grades and combined into a clean dataset prepared for visualization in a Tableau dashboard. This dashboard enables decision-makers to explore results interactively, uncover patterns, evaluate misclassification impacts, and monitor model performance across different customer segments.

The project follows a full data science workflow from dataset acquisition, cleaning, and feature engineering, through model training, to the creation of a business-ready analytical tool all aimed at supporting better, data-driven credit decisions.

Step 1: Data Cleaning & Feature Engineering

In this step, I prepared the raw credit risk dataset for modeling. After identifying and handling missing values with median imputation, I transformed key categorical variables such as loan grade, home ownership, loan intent, and credit default history into numerical formats. This streamlined the dataset for machine learning workflows. The cleaned dataset was exported as a new CSV file for use in the next phase.

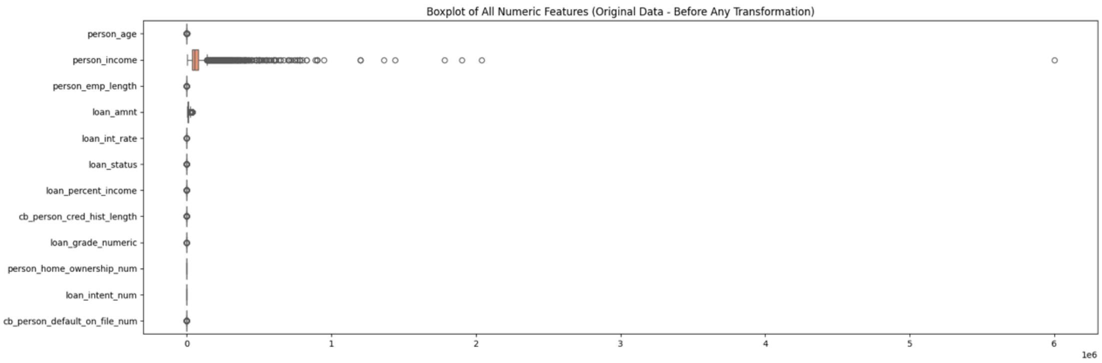
Step 2: Outlier Mitigation & Feature Rescaling

In this step, I visualized all numeric features using boxplots to detect potential outliers. The original data revealed extreme values, especially in person_income and loan_amnt.

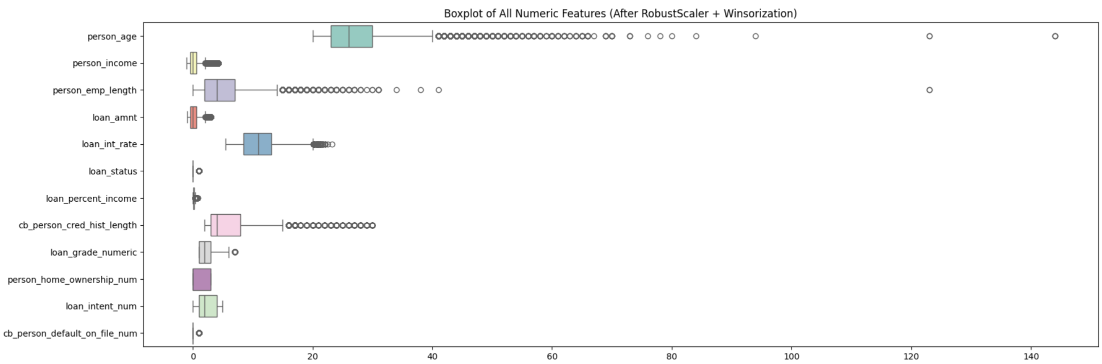
To reduce their impact without distorting the underlying data, I applied RobustScaler, which normalizes features based on the **median** and **interquartile range** — making it naturally robust to outliers.

To further cap the influence of extreme values, I applied **Winsorization** on both features at the 1% level. This clipped the top and bottom 1% of values, affecting approximately 2% of income values and 1% of loan amounts.

The post-transformation boxplot confirms that the distributions are now more compact and ready for modeling.




```
Winsorized 'person_income': 646 rows changed (1.98%)
Winsorized 'loan_amnt': 336 rows changed (1.03%)
```



Step 3: Neural Network Model for Loan Grade Prediction

In this stage, I built a fully connected neural network using PyTorch to predict loan grades (1–7) as a multi-class classification task. The model takes into account features like age, income, employment length, loan characteristics, and credit history.

The data was standardized using StandardScaler and split into training and test sets. The architecture includes two hidden layers with ReLU activations and dropout for regularization.

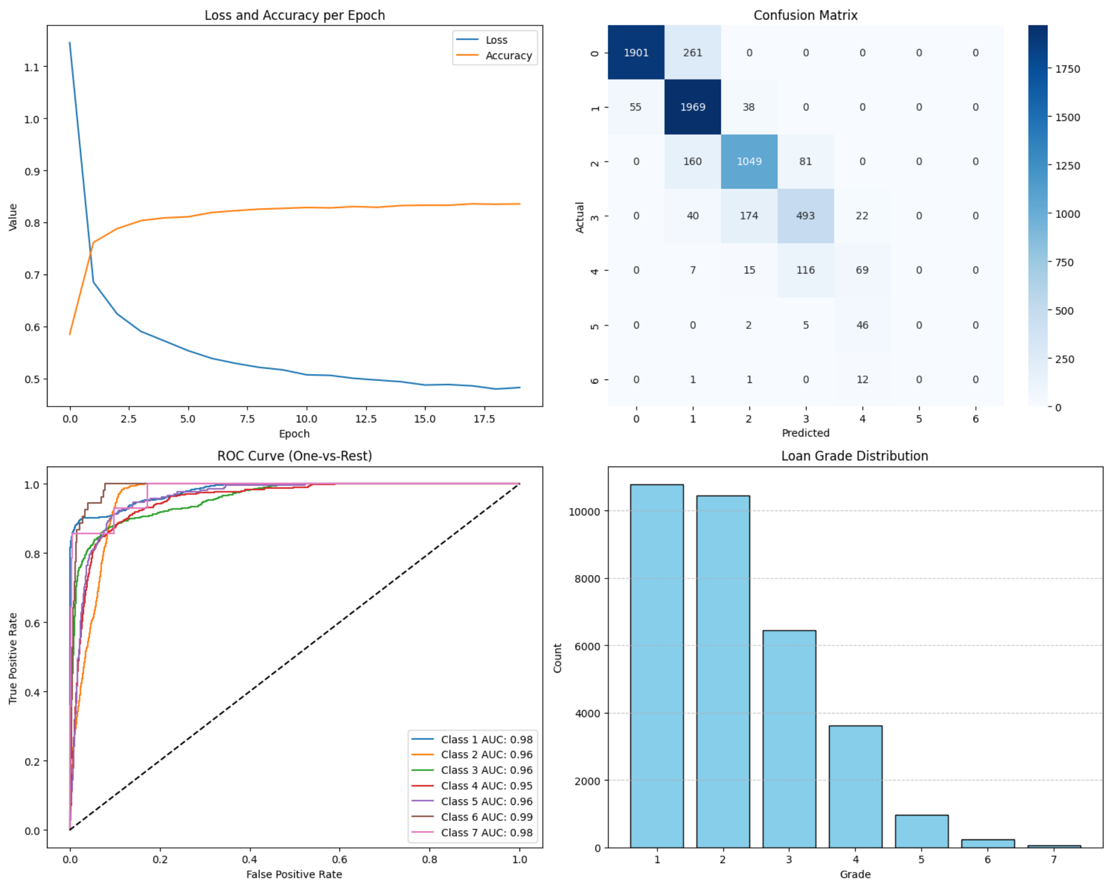
Training ran for 20 epochs, achieving stable convergence with a final test accuracy of **~84%**. Despite class imbalance, the model performed strongly on major classes — reaching AUC scores above 0.95 in most cases. Minority classes (Grades 6–7) were underrepresented, which impacted their performance.

Evaluation included:

- **Loss & Accuracy trends** showing smooth convergence
- **Confusion matrix** revealing strong prediction on Grades 1–3
- **ROC curves (OvR)** confirming robust class separability
- **Class distribution chart** highlighting the imbalance challenge

This step demonstrates the model's capacity to learn complex patterns in credit data and provides a strong foundation for further improvement via balancing techniques or ordinal-specific architectures.

```
Classification Report:
              precision    recall  f1-score   support

           0     0.9719    0.8793    0.9233      2162
           1     0.8076    0.9549    0.8751      2062
           2     0.8202    0.8132    0.8167      1290
           3     0.7094    0.6763    0.6924       729
           4     0.4631    0.3333    0.3876       207
           5     0.0000    0.0000    0.0000        53
           6     0.0000    0.0000    0.0000        14

    accuracy                         0.8410      6517
   macro avg     0.5389    0.5224    0.5279      6517
weighted avg     0.8344    0.8410    0.8346      6517
```



Summary: Neural Network Model Assumptions and Next Steps

We implemented a neural network in PyTorch to predict credit risk ratings across seven ordinal classes (loan_grade_numeric from 1 to 7). The model was trained on standardized numerical features without any dimensionality reduction or modification to the label distribution.

Evaluation of Model Assumptions

1. **Numerical Input and Scaling** All input features are numeric and were standardized using StandardScaler. This supports model convergence and stable optimization.
2. **IID Assumption (Independent and Identically Distributed)** The dataset was randomly split into training and test sets using train_test_split, ensuring that samples are independent and drawn from the same distribution.
3. **Loss Function and Output Configuration** We used CrossEntropyLoss along with integer labels and raw logits. This is the appropriate setup for multiclass classification tasks.

4. **Model Complexity and Overfitting Control** The neural network architecture includes two hidden layers with dropout for regularization. Training over 20 epochs shows consistent improvements in accuracy and loss, with no signs of overfitting.
5. **Class Imbalance** This assumption is currently **not satisfied**. The target variable is highly imbalanced. For example, grade 1 includes over 10,000 samples, while grade 7 has fewer than 100. As a result, the model performs poorly on rare classes, with low recall and precision.
6. **Multicollinearity and Redundant Inputs** While not formally tested, neural networks are generally robust to multicollinearity. Since the goal is prediction rather than interpretation, this is not a concern at this stage.

Conclusion and Next Steps

All formal assumptions of the model are satisfied except for class balance. Performance is strong for the dominant classes but weak for the rare ones. In the next stage, we plan to address this imbalance by collapsing grades 5, 6, and 7 into a single group (grade 4), which will reduce sparsity in the label space and improve classification performance.

Step 4: Label Collapsing & Model Retraining

To address the performance issues caused by extreme class imbalance, I collapsed loan grades 5, 6, and 7 into a single category (Grade 4), reducing the classification task from 7 to 4 classes. This restructuring created a more balanced label distribution while preserving the ordinal structure of the data.

I then retrained the same neural network architecture on the updated labels. The model quickly converged, reaching **~87% accuracy** after 20 epochs. Performance improved across all classes, including the previously underrepresented ones.

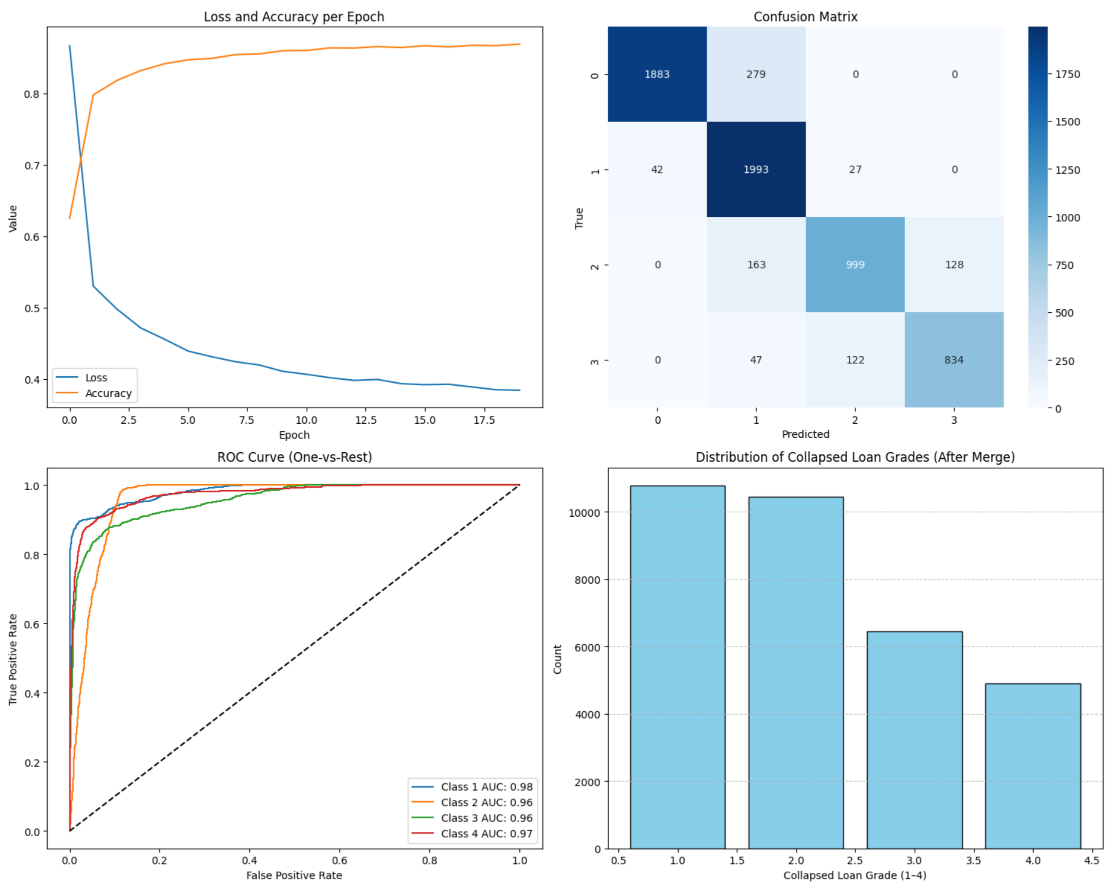
The updated evaluation shows:

- **Consistent loss decrease and accuracy improvement** over epochs
- **Strong classification performance** with precision and recall above 85% for all groups
- **AUC scores between 0.96 and 0.99**, indicating excellent separability
- **Balanced confusion matrix**, showing fewer misclassifications between adjacent grades

- **Improved F1 macro (0.87)** and weighted average metrics

This step demonstrates how strategic label restructuring can significantly boost multi class classification performance in imbalanced ordinal datasets.

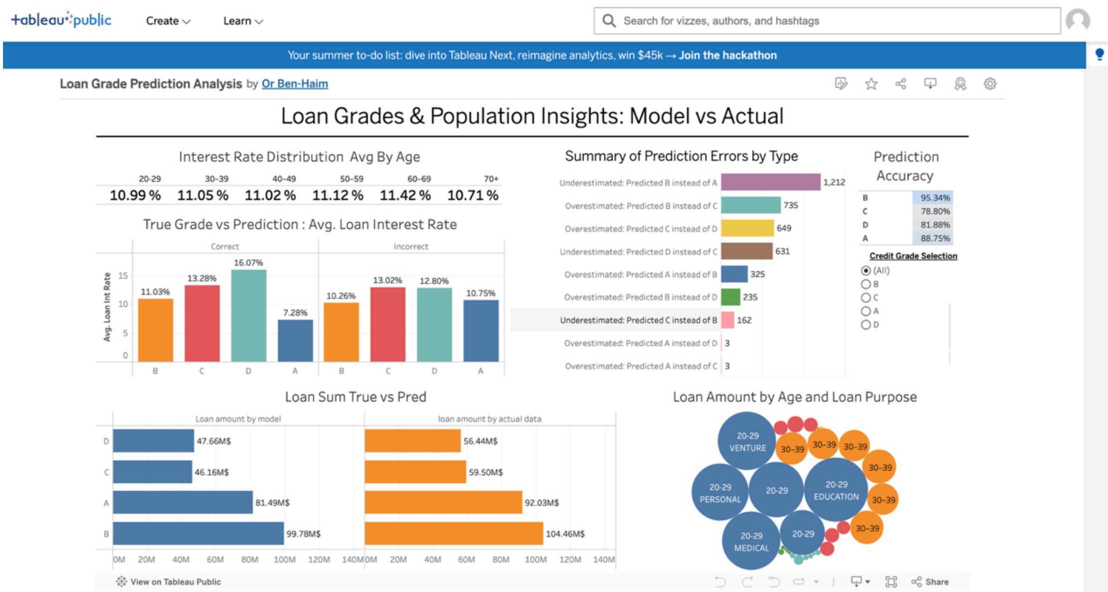
```
Classification Report:
              precision    recall  f1-score   support

           0       0.98      0.87      0.92      2162
           1       0.80      0.97      0.88      2062
           2       0.87      0.77      0.82      1290
           3       0.87      0.83      0.85      1003

    accuracy                           0.88      6517
   macro avg       0.88      0.86      0.87      6517
weighted avg       0.88      0.88      0.88      6517
```



Step 5 Model vs Actual: Analytical Summary

Correct predictions show a clear interest rate hierarchy (D highest, A lowest), but in errors the gap narrows, exposing weak separation at C/D and A/B boundaries. Grade B is most accurate, while Grade C is most error-prone, often confused with neighbors. The model has a conservative bias, notably downgrading A to B, which may limit opportunities for low-risk clients. Loan sums are consistently underestimated across all grades, especially C, risking misinformed capital allocation. Calibration and stronger mid-tier separation are needed.

Interactive Tableau Dashboard: [View Here]



Final Summary & Next Steps

This project built and tested a neural network to predict the credit risk level of loan applicants, using the public Kaggle Credit Risk Dataset. The process covered the full journey from cleaning and preparing the data, through model training, to building an interactive Tableau dashboard for decision-makers.

At first, when training on the original seven loan grades, the model worked well for common grades but struggled with the rare ones. By combining the three rarest grades into a single category, the dataset became more balanced and the model's accuracy, recall, and precision improved significantly, reaching about 88% accuracy . This makes credit approval decisions more reliable and reduces the risk of costly mistakes.

Analysis through the dashboard showed that correct predictions kept the expected link between risk level and interest rate, helping ensure loans are priced in line with risk. However, in some cases, especially between mid tier grades, the differences were blurred, which could lead to pricing that leaves money on the table. The model also tended to underestimate loan amounts, especially for Grade C, meaning capital might not be allocated optimally. Another pattern was a conservative bias that often downgraded top clients from Grade A to B, which could limit opportunities with the most profitable, low risk borrowers.

Moving forward, the model could benefit from adjusting how it separates close grades to improve pricing accuracy, adding penalties for the most costly errors to protect margins, and using algorithms that understand the natural order of grades to keep risk assessments consistent. Expanding the range of borrower information, such as transaction history, could also help increase accuracy without adding risk. Finally, integrating explainability tools would give risk managers and regulators more trust in the model's decisions.

With these refinements, the system has the potential to become a powerful credit risk tool one that not only improves prediction quality but also supports smarter lending strategies, keeps top customers engaged, optimizes the use of available capital, and maximizes profitability while keeping defaults low.