Journal of Artificial Intelligence and Data Science (JAIDA)
e-ISSN: 2791-8335
https://dergipark.org.tr/pub/jaida

Vol. 4, No. 2, pp. 97-103, 2024
Research Article
Received: 16/12/2024 Accepted: 26/12/2024

Artificial Intelligence Based Customer Risk Classification for Receivables Management of Businesses

Şaban Can TİRYAKİ [1], Adnan KAVAK [2*]

Abstract

This study is carried out with the aim of developing and implementing artificial intelligence-based receivables management systems for businesses. A model is created to predict customers' debt payment situations. In the study, invoice data of a company named QF_CARIRAPOR is utilized. The features table is created in Apache druid and risk scoring label is made manually according to set rules. Then, various machine learning models such as XGBoost, Random Forest are implemented on MindsDB platform. The classified risk score is visualized with the Streamlit user interface using the results created in MindsDB. Among the applied models, XGBoost has resulted in the highest classification accuracy of 98.8 %. The findings reveal the potential to increase the effectiveness of receivables management processes by applying machine learning models.

Keywords: Mindsdb; Risk Classification; Receivables Management; Xgboost.

1. Introduction

Receivables management is a critical function for businesses, as it directly impacts cash flow, financial stability, and long-term viability. Despite its importance, many organizations face significant challenges in efficiently managing their accounts receivable. Key issues include delayed payments, increased risk of bad debts, and the inability to accurately predict customer payment behavior. Traditional approaches to receivables management, which often rely on manual processes, are becoming increasingly inadequate. These methods struggle to handle the complexities of modern business environments, which are characterized by high transaction volumes, diverse customer profiles, and rapidly changing market dynamics.

Delayed payments can lead to cash flow shortages, forcing businesses to rely on costly financing options. Unmanaged credit risk increases the likelihood of bad debt write-offs and weakening financial health. For small-sized and medium-sized enterprises (SMEs), in particular, these issues can pose existential threats, as they typically operate with narrower financial margins than larger corporations. Complex network theory has offered an innovative approach to assessing credit risk by examining debt and credit relationships in financial systems [1]. Focusing on the use of complex relationship models in the assessment of credit risk, the study in [2] examined the interactions between customer behavior and future payment habits. It was aimed at improving the credit scoring system by using customer segmentation and behavior analysis. Fuzzy rule-based systems are used to manage uncertainties in credit risk assessment. In [3], it was shown that accurate predictions were made by analyzing customer credit history with fuzzy logic. The study in [4] provides a systemic review of the recent studies, identifying trends in credit scoring using a fixed set of questions.

In [5], a literature survey was conducted to systematically review statistical and machine learning models in credit scoring, to identify limitations in literature, to propose a guiding machine learning framework, and to point to emerging directions. Support Vector Machines (SVM) is applied to predict systemic risk in the complex and interconnected realm of financial markets [6]. Deep neural network model was designed to predict high-risk behaviors in financial traders by analyzing vast amounts of transaction data such as Global Insider Trading data [7]. However, there is a need for advanced, data-driven solutions to enhance the accuracy, efficiency, and adaptability of receivables management systems. Specifically, businesses require tools that can classify customer risk more effectively, enabling proactive strategies to mitigate defaults and optimize cash flow. Leveraging artificial intelligence (AI) and machine learning (ML) offers a promising pathway to address these challenges, providing businesses with the ability to process complex data, predict customer behaviors, and implement dynamic, real-time risk management strategies.

The utilization of artificial intelligence (AI) in receivables management may not only enhance the accuracy of risk assessments but also empower businesses to adopt proactive measures aimed at mitigating financial losses. For example, AI-driven systems can provide tailored recommendations for credit terms or automate follow-up

*Corresponding author

Şaban Can TİRYAKİ; AYASOFYAZILIM Bilişim A.Ş, Bilişim Vadisi, Gebze, Kocaeli, Türkiye & Kocaeli University, Faculty of Engineering, Computer Engineering Department, İzmit, Kocaeli, Türkiye; e-mail: can.tiryaki@ayasofyazilim.com; 0009-0006-2765-5551

Adnan KAVAK*; Kocaeli University, Faculty of Engineering, Computer Engineering Department, İzmit, Kocaeli, Türkiye, e-mail: akavak@kocaeli.edu.tr; 0000-0001-5694-8042

schedules based on the predicted risk profiles of individual customers. The contribution of this study is to develop an AI-based framework for customer risk classification in receivables management of businesses and demonstrate feasibility of implementation of various machine learning models on MindsDB platform based on real data.

2. Dataset

In this study, the invoice information list belonging to a company called QF CARİRAPOR was used (See Table 1). The QF_CARIRAPOR data set contains invoice information of 1000 customers. The data set contains information such as when customers paid their debts, whether they paid late, on time, and how many days it took to pay. There is a number of invoice information for each customer in the data set. This information in QF CARIRAPOR was combined with a dataset called Top Customer, which includes different information about customers belonging to the same company. Both datasets include the following attributes for each customer: *Paid Invoice, Total Paid Invoices, Sum Amount Paid Invoices, Total Invoices Late, Sum Amount Late Invoices, Total Outstanding Invoices, Total Outstanding Late, Sum Total Outstandings, Sum Late Outstanding, Average Days Late, Average Days Outstanding Late.* The merging process was carried out using both data sets. Since the common column in both data sets is the CARDREF column, the merging process was done based on these values. As a result of this merging process, a dataset called Features Table was created. Attributes containing redundant information was eliminated and features were created as shown in the columns of Table 2.

Table 1. A Sample of QF_CARİRAPOR dataset

LOGID	LOGICALREF	CARDREF	PROJECTREF	PROCDATE	DATE_
954689	18021816	91033	2020-08-12 00:00:00	2020-08-12 00:00:00	2020-08-12 00:00:00
954690	20485106	91033	2021-06-09 00:00:00	2021-06-09 00:00:00	2021-06-09 00:00:00
954691	10440444	91033	2018-04-18 00:00:00	2018-04-25 00:00:00	2023-07-05 00:00:00

Table 2. A Sample of Features Table in QF_CARİRAPOR dataset

CARDREF	Paid Invoice	Total Paid Invoices	Sum Amount Paid Invoices	Total Invoice Late
8	1	347	5329577000125605	570
10	1	369	573816600039994867	712
19	1	434	884015000000407552	568
34	1	422	447293200030855488	490
36	1	451	38215310003937	559

After combining the datasets, each customer is labelled manually with a risk score according to

$$S = T_{ADL} + P_{ADOS} + C_{TINV}, \tag{1}$$

where T_{ADL} is the average late day score, P_{ADOS} is the points received for average amount of overdue debt, and C_{TINV} is the score for the number of invoices paid on time, which are expressed as;

$$T_{ADL}(t) = \begin{cases} 627 & t = 0 \\ 470 & 1 \leq t \leq 15 \\ 313.5 & 16 \leq t \leq 30 \\ 156.75 & 31 \leq t \leq 45 \\ 0 & t \geq 46 \end{cases} \tag{2}$$

$$P_{ADOS} = 627\left(1 - \left(\frac{Amount\ of\ unpaid\ invoices}{max(Amount\ of\ unpaid\ invoices)}\right)\right) \tag{3}$$

$$C_{TINV} = \left(\frac{Number\ of\ paid\ invoices\ on\ time}{Total\ number\ of\ invoices}\right) \tag{4}$$

3. Implementation of Customer Risk Classification

Firstly, various Scikit-learn models [8] in Python were applied on the dataset created. While applying the Scikit-learn models, the dataset was divided into training and test sections as 80% and 20%, respectively. A part of the code is given in Figure 1.

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.metrics import classification_report, confusion_matrix


label_encoder = LabelEncoder()
df['Risk Group'] = label_encoder.fit_transform(df['Risk Group'])

X = df.drop(columns=['Risk Group'])
y = df['Risk Group']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Figure 1. Implementation using Scikit-learn models

Simulation screenshots of various models are given in Figures 2 through 6. As seen in Figure 2, the dataset was first labeled according to the Risk group, according to the calculated risk score values. It was labeled with a numerical result called calculated risk score in Apache Druid. Then, according to these numerical results, risk levels were categorized as *very high risk, high risk, medium risk, low risk, and no risk.* The Decision Trees model resulted in classification with 91% accuracy (Figure 2). Random Forest classifier is generally a powerful and flexible ensemble learning method widely used in machine learning. In our problem, it resulted in overall accuracy of 90% (Figure 3). Unfortunately, KNN and SVM classifiers resulted in poor performance with accuracy of 59% and 60%, respectively when our QF_CARİRAPOR dataset was applied (Figure 4 and 5). The poorer performance of KNN and SVM likely stems from their sensitivity to the characteristics of the financial dataset (e.g., imbalance, noise, and high dimensionality) and their reliance on parameter tuning and preprocessing, which tree-based models handle more gracefully. In addition, KNN assumes that similar instances (in terms of feature values) belong to the same class. If the QF_CARRAPOR data has complex patterns or nonlinear relationships between features, KNN might fail to capture them. SVM assumes the existence of a clear margin between classes, which might not hold for financial data. The best classification performance is obtained by XGBoost classifier model resulting in accuracy of 93% (Figure 6). The XGBoost excels at modeling the complex interactions in financial data due to its tree-based structure and gradient boosting framework, which builds successive trees to reduce residual errors from previous ones.

```
Decision Trees- Confusion Matrix:
     31   2    0    0    3
      2  123   6    0    0
      0   3   14    0    0
      0   0    0    8    1
      0   0    0    1    6

Decision Trees- Classification Report:
            precision    recall   F1-score   support

        0       0.94      0.86      0.90        36
        1       0.96      0.94      0.95       131
        2       0.70      0.82      0.76        17
        3       0.89      0.89      0.89         9
        4       0.60      0.86      0.71         7

 accuracy                           0.91       200
 macro avg      0.82      0.87      0.84       200
 weighted avg   0.92      0.91      0.91       200
```

Figure 2. Simulation screenshot of Decision Trees using Scikit-learn models

```
Random Forest - Confusion Matrix:

          32   3   0   0   1
           3 123   5   0   0
           0   6  11   0   0
           0   0   0   9   0
           1   0   0   1   5

Random Forest - Classification Report:
              precision    recall  F1-score   support

           0       0.89      0.89      0.89        36
           1       0.93      0.94      0.94       131
           2       0.69      0.65      0.67        17
           3       0.90      1.00      0.95         9
           4       0.83      0.71      0.77         7

    accuracy                           0.90       200
   macro avg       0.85      0.84      0.84       200
weighted avg       0.90      0.90      0.90       200
```

Figure 3. Simulation screenshot of Random Forest using Scikit-learn models

```
KNN - Confusion Matrix:

   7  28   0   0   1
  13 112   0   3   3
   1  16   0   0   0
   1   8   0   0   0
   0   6   0   1   0

KNN - Classification Report:
              precision    recall  F1-score   support

           0       0.32      0.19      0.24        36
           1       0.66      0.85      0.74       131
           2       0.00      0.00      0.00        17
           3       0.00      0.00      0.00         9
           4       0.00      0.00      0.00         7

    accuracy                           0.59       200
   macro avg       0.20      0.21      0.20       200
weighted avg       0.49      0.59      0.53       200
```

Figure 4. Simulation screenshot of KNN using Scikit-learn models

```
SVM - Confusion Matrix:

 0  36   0   0   0
 0 131   0   0   0
 0  17   0   0   0
 0   9   0   0   0
 0   7   0   0   0

SVM - Classification Report:
              precision    recall  F1-score   support

           0       0.00      0.00      0.00        36
           1       0.66      1.00      0.79       131
           2       0.00      0.00      0.00        17
           3       0.00      0.00      0.00         9
           4       0.00      0.00      0.00         7

    accuracy                           0.66       200
   macro avg       0.13      0.20      0.16       200
weighted avg       0.43      0.66      0.52       200
```

Figure 5. Simulation screenshot of SVM using Scikit-learn models

```
XGBoost - Confusion Matrix:

30   4   0   1   1
 0 127   3   0   1
 0   3  14   0   0
 0   0   0   8   1
 1   0   0   0   6

XGBoost - Classification Report:
              precision    recall  F1-score   support
           0       0.97      0.83      0.90        36
           1       0.95      0.97      0.96       131
           2       0.82      0.82      0.82        17
           3       0.89      0.89      0.89         9
           4       0.67      0.86      0.75         7

accuracy                                  0.93       200
macro avg           0.86      0.87      0.86       200
weighted avg        0.93      0.93      0.93       200
```

Figure 6. Simulation screenshot of XGBoost using Scikit-learn models

Apache Druid [9] and MindsDB [10] have been used for evaluating real-time processing of QF CARİRAPOR dataset. Apache druid is an open-source data storage and analysis platform used for big data analytics. Druid can process and query data in real time. Apache druid also can pull large amounts of data from many different data sources. It can store the retrieved data in a scalable way and then provide rapid access for real-time analysis. Druid's internal architecture is built to provide these fast query and analysis capabilities. MindsDB is an automatic machine learning database that can connect to multiple data sources. It helps to make predictions using the data in the database. The aim of MindsDB is to make data analysis and prediction tasks simple and accessible. The connection process between Apache Druid and MindsDB takes place. In the first stage, a database called "druid_datasource" is created using the dataset described above. It is stated that this database will use the druid data engine. In the next stage, the information required to connect to the druid data engine is given. Which port is on, which path and scheme is used. MindsDB inherently supports various machine learning algorithms, such as XGBoost, Random Forest, Neural Networks, etc. This allows users to choose the most suitable model option. The platform has the ability to automatically select the most suitable machine learning model based on the dataset, making the model selection process easy. As shown in Figure 7, a folder called druid_datasource has been created in MindsDB. In this folder, the tables we uploaded to Apache Druid can be seen in MindsDB.

```sql
CREATE DATABASE druid_datasource
WITH
    engine = 'druid',
    parameters = {
      "host": "localhost",
      "port": 8888,
      "path": "/druid/v2/sql/",
      "scheme": "http"
    };

SELECT * FROM druid_datasource.TopCustomer;
```

Figure 7. Druid data source connection in MindsDB

In MindsDB implementation, the table called FeaturesTable in the files section was selected. Using this table, a model called “tahsilet_sonuc” was created. The desired from this model is to predict the numerical value called TahsiletSkor. It is a command that shows the performance of the applied models. The implementation results on MindsDB real-time database platform is shown in Figure 8. As can be seen, the XGBoost model implementation on MindsDB resulted in the highest accuracy of 98.8%, as in the simulated Scikit-learn models in Python. A sample test case using Streamlit [11] interface using the Collection result model in in MindsDB is given in Figure 9 and 10. As seen in Figure 10, the risk group has been successfully determined based on the information entered by the user.

	name	performance	training_time	selected	accuracy_functions
1	Neural	0.961	55.93	0	['r2_score']
2	XGBoostMixer	0.988	1.01	1	['r2_score']
3	Regression	0.88	0.32	0	['r2_score']
4	RandomForest	0.971	0.52	0	['r2_score']

Figure 8. Classification performance of various models on MindsDB platform applied on QF CARİRAPOR dataset

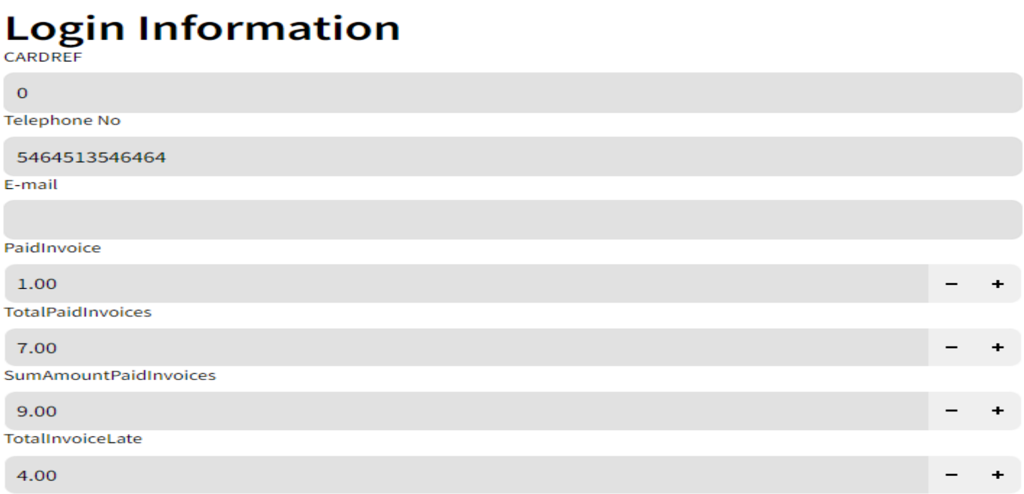

Figure 9. Streamlit user interface applied on QF CARİRAPOR test dataset

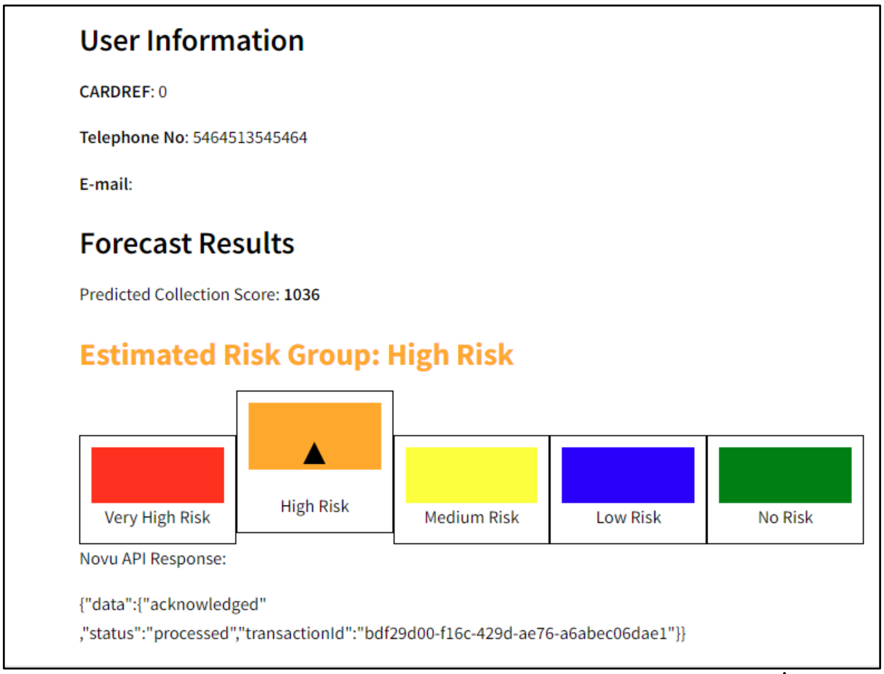

Figure 10. Risk classification when XGBoost is applied on MindsDB using QF CARİRAPOR test dataset.

5. Conclusion

In this study, it was aimed to develop and implement artificial intelligence-based receivables management for businesses. A model was created to determine the risk level of a customer whether this customer may pay debts in time or not. For this purpose, a dataset called QF_CARIRAPOR was utilized in our study, which contains data of 1000 customers. This dataset was then uploaded to the Apache Druid environment. It is labeled with numbers between 0 and 1900 called Apache Druid Collection Score. Then, we applied classification models such as XGBoost, Random Forest, KNN, SVM, and Decision Trees to the dataset. The model with the highest accuracy rate was named Collection Result into the MindsDB environment and created a model named Tahsilet_Sonuc. The XGBoost model resulted in the highest classification accuracy in both the Scikit-learn simulation (93% accuracy) and the MindsDB real-time database implementation (98.8% accuracy).

Acknowledgements

This study was carried out within the scope of TAHSILET project conducted by AYASOFYAZILIM A.Ş. The project is supported by Tübitak Teydeb 1507 program.

References

[1] H.Lam, "Analyzing the Measures of Credit Risk on Financial Corporation and It's Impact on Profitability," International Journal of Research in Vocational Studies (IJRVOCAS), vol. 3, no. 1, pp. 64-70, 2023.

[2] N. Wilson, B. Summers, R. Hope, "Using payment behaviour data for credit risk modelling," International Journal of the Economics of Business; vol. 7, no. 3, pp. 33-346, 2000.

[3] J. Reyes, J. Perez, and S. Ake, "Credit risk management analysis: An application of fuzzy theory to forecast the probability of default in a financial institution," Contaduría y Administración, vol. 69, no. 1, pp. 18 211, 2024.

[4] A. Markov, Z. Seleznyova, and V. Lapshin, "Credit scoring methods: Latest trends and points to consider," The Journal of Finance and Data Science, vol. 8, pp. 180-201, 2022.

[5] X. Dastile, T. Celik, and M. Potsane, "Statistical and machine learning models in credit scoring: A systematic literature survey," Applied Soft Computingt, vol. 91, 106263, 2000.

[6] Q. Zhou, "Predicting Systemic Risk in Financial Markets Using Machine Learning," Transactions on Economics Business and Management Research vol. 8, pp. 455-460, 2024.

[7] K. Xu, Y. Wu, Z. Li, R. Zhang, and Z. Feng, "Investigating Financial Risk Behavior Prediction Using Deep Learning and Big Data," International Journal of Innovative Research in Engineering and Management (IJIREM), vol. 11, no. 3,pp. 77-81, 2024.

[8] Scikit-learn Machine Learning in Python, https://scikit-learn.org/

[9] https://druid.apache.org/

[10] MindsDB-Platform for Building AI, https://docs.mindsdb.com/

[11] https://streamlit.io/