A black background with white text

Description automatically generated

**IE – 7275**

**DATA MINING FOR ENGINEERING**

**ACADEMIC PROJECT REPORT**

**TOPIC: METAVERSE FINANCIAL RISK ASSESSMENT**

**DATE: 04/10/2024**

**SUBMITTED BY**

MOHEESH KAVITHA ARUMUGAM

002296201

**TABLE OF CONTENTS**

1. Objective
2. Introduction
3. Data Loading and Preparation
4. Exploratory Data Analysis (EDA)
5. Feature Engineering
6. Logistic Regression
7. K-Nearest Neighbors (KNN)
8. Decision Tree
9. Support Vector Machines (SVM)
10. Model Comparison and Conclusion

**OBJECTIVE**

The primary goal of this project is to construct a robust classification model tailored for forecasting the risk levels linked with financial transactions within the Open Metaverse blockchain environment. This model will effectively categorize transactions into distinct risk tiers, namely low, moderate, or high, by leveraging a comprehensive set of criteria including transaction types, monetary amounts, user behavioral patterns, geographical locations, and IP addresses.

By harnessing the power of classification analysis on this dataset, we aim to establish a predictive framework capable of significantly enhancing fraud detection capabilities, risk evaluation processes, and overall security management within virtual ecosystems. This framework will provide automated decision-making capabilities, allowing for swift and accurate classification of transactions based on their associated risk levels.

Ultimately, the development and deployment of such a classification model will contribute to fostering a safer and more trustworthy financial landscape within the Open Metaverse. It will empower stakeholders to make informed decisions, mitigate potential risks, and bolster the integrity of financial transactions, thereby fostering greater confidence and reliability in virtual economic interactions.

**INTRODUCTION**

**Blockchain**

In the evolving landscape of the Open Metaverse, blockchain technology stands as a cornerstone for enabling secure, transparent, and decentralized financial transactions within virtual environments. Leveraging blockchain technology ensures that users can engage in digital commerce, exchange virtual assets, and partake in financial activities with a heightened sense of trust and security. This is achieved through the fundamental principle of blockchain, which involves the recording of transactions in an immutable and transparent manner on a distributed ledger.

By utilizing blockchain in the context of the Open Metaverse, users benefit from a system where transactional data is securely stored across a network of nodes, preventing any single entity from exerting control or tampering with the transaction history. This decentralized architecture fosters integrity, as transactions cannot be altered retroactively, thereby reducing the risk of fraud and ensuring the authenticity of digital interactions.

Furthermore, blockchain technology facilitates seamless cross-platform transactions across the diverse virtual worlds that comprise the Open Metaverse. This interoperability enables users to transact with ease, regardless of the specific virtual environment they inhabit. Whether purchasing virtual goods, transferring digital currencies, or engaging in financial exchanges, users can do so confidently, knowing that blockchain ensures the integrity and security of their transactions.

**Working of Blockchain Transaction**

A diagram of blockchain technology

Description automatically generated

* Users send transactions, specifying recipient, amount, and details.
* Nodes verify transaction authenticity and compliance with network rules.
* Valid transactions are bundled into blocks by miners.
* Miners solve puzzles to append blocks to the blockchain.
* Network achieves consensus on block validity.
* Valid blocks are added to the blockchain ledger.
* Transactions become immutable within the distributed ledger system.

**Dataset Source**

This dataset provides blockchain financial transactions within the Open Metaverse, aiming to provide a rich, diverse, and realistic set of data for developing and testing anomaly detection models, fraud analysis, and predictive analytics in virtual environments. With a focus on applicability, this dataset captures various transaction types, user behaviors, and risk profiles across a global network.

Transactions were extracted using a sophisticated model that incorporates distributions, behavioral patterns, and risk assessments. The model ensures a diverse representation of activities, from typical transactions to potential fraudulent activities, across different user groups and global regions.

Acknowledgments: This dataset is shared by the Open Metaverse, a collaborative initiative dedicated to the advancement and democratization of virtual worlds. For more information, visit <https://www.openmv.org/>.

**Dataset Overview**

* Timestamp: Date and time of the transaction.
* Hour of Day: Hour part of the transaction timestamp.
* Sending Address: Blockchain address of the sender.
* Receiving Address: Blockchain address of the receiver.
* Amount: Transaction amount in a simulated currency.
* Transaction Type: Categorization of the transaction (e.g., transfer, sale, scam, phishing).
* Location Region: Simulated geographical region of the transaction.
* IP Prefix: Simulated IP address prefix for the transaction.
* Login Frequency: Frequency of login sessions by the user, varying by age group.
* Session Duration: Duration of activity sessions in minutes.
* Purchase Pattern: Behavioral pattern of purchases (e.g., focused, random, high-value).
* Age Group: Categorization of users into new, established, and veteran.
* Risk Score: A calculated risk score based on transaction characteristics and user behavior.
* Anomaly: Risk level assessment (e.g., high\_risk, moderate\_risk, low\_risk).

**BASIC METHODOLOGY**

The methodology for developing a robust classification model tailored to forecasting financial transaction risk levels within the Open Metaverse blockchain environment involves several key stages. Initially, data collection ensues, where a diverse dataset capturing various transaction types, user behaviors, and geographical profiles is acquired. This dataset forms the foundation for subsequent analysis. Following data collection, thorough data preprocessing procedures are implemented, encompassing tasks such as data cleaning, categorical variable encoding, and numerical feature scaling. These steps ensure that the dataset is suitably prepared for analysis, addressing any inconsistencies or missing values that may affect model performance.

In the subsequent stage, exploratory data analysis (EDA) is conducted to glean insights into the underlying patterns and distributions within the dataset. Through visualization techniques and statistical analysis, the EDA phase uncovers correlations between features, identifies potential anomalies or outliers, and provides a comprehensive understanding of the dataset's characteristics. Armed with insights from EDA, the feature engineering process commences, where new features are created and existing ones are transformed to capture relevant information for risk classification. This step leverages domain knowledge and data-driven techniques to extract meaningful signals from the dataset, enhancing the predictive power of the ensuing models.

Moving forward, model building constitutes a pivotal phase in the methodology, wherein various classification algorithms are trained and evaluated on the prepared dataset. Algorithms such as logistic regression, K-nearest neighbors (KNN), decision trees, and support vector machines (SVM) are considered, each offering distinct strengths in handling classification tasks. Through rigorous evaluation using appropriate metrics, including accuracy, precision, recall, and F1-score, the performance of each model is assessed. Furthermore, model optimization techniques, such as hyperparameter tuning and ensemble learning, are employed to fine-tune the selected models and maximize their predictive performance. This iterative process of model building and refinement culminates in the development of a robust classification framework capable of effectively categorizing financial transactions based on their associated risk levels within the Open Metaverse blockchain ecosystem.

A diagram of a company

Description automatically generated

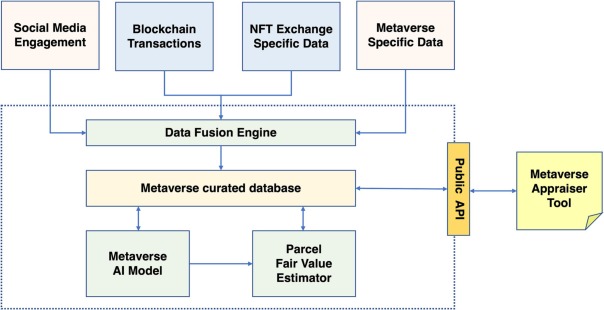
**DATA LOADING AND PREPARATION**

I initiated the analysis by loading the dataset from the provided CSV file using the Pandas library. A preliminary examination of the dataset was conducted by displaying the first few rows, revealing its structure and content. Subsequently, I determined the dataset's dimensions to comprehend the extent of the available data. To ensure data integrity, I meticulously checked for any duplicate rows within the dataset. Additionally, I assessed the number of unique values in each column to gauge data variability and identify potential categorical variables.

**Data Cleaning and Preprocessing**

During the data cleaning phase, I identified columns with a small number of unique values, suggesting potential categorical variables with low variability. These columns were scrutinized for any inconsistencies or data quality issues to ensure the reliability of subsequent analyses. Moreover, I examined the presence of missing values in each column and devised appropriate strategies for handling them, such as imputation or removal. Furthermore, I converted the 'timestamp' column to DateTime format to facilitate temporal analysis and enhance interpretability.

**Handling Missing Data**



Data can be classified into two categories: on-chain data, which refers to financial transactions involving NFTs of parcels and are stored on the blockchain, and off-chain data such as parcel descriptions (e.g., location, size) and utilization (e.g., traffic patterns) that are generally not persistent and available from centralized servers. These data must be aggregated and carefully organized in order to be analyzed effectively. In the following sections, we will explore the main types of data that make up these metaverses and how we have implemented these data in our daily data acquisition and analysis tool

With this information there are no missing data, however, there are chances that this API can have duplicate data concerning different timeframes. For modeling purposes these data are considered to be duplicates, so those data are removed. Compared to the entire dataset, these data are very low and have no importance.

**Outlier Detection and Treatment**

My analysis encompassed the detection and treatment of outliers within the dataset. Numeric columns are relevant for outlier detection, including 'amount', 'login\_frequency', and 'session\_duration', were selected. Summary statistics, such as mean, median, and quartiles, were computed for these numeric columns to establish a basis for outlier detection. Subsequently, I employed the interquartile range (IQR) method to determine outlier thresholds, enabling the identification of potential anomalies. Any outliers detected within the dataset were flagged and subsequently filtered to maintain data quality and ensure the robustness of subsequent analyses.

**EXPLORATORY DATA ANALYSIS (EDA)**

Exploratory Data Analysis (EDA) serves as a crucial preliminary step in understanding the underlying structure and patterns within a dataset. By visually inspecting and summarizing the data, EDA helps identify outliers, trends, and relationships between variables. This process aids in making informed decisions regarding data preprocessing, feature selection, and modeling strategies. EDA also enables the detection of potential data quality issues, such as missing values or inconsistencies, ensuring the reliability and integrity of subsequent analyses. Overall, EDA plays a fundamental role in uncovering insights, guiding further investigation, and ultimately enhancing the effectiveness of data-driven decision-making processes.

**A screenshot of a computer

Description automatically generated**

A graph of a graph of a number of bars

Description automatically generated with medium confidenceThe histogram appears to be skewed to the right, which means that there are more data points with lower risk scores than there are data points with higher risk scores. This suggests that most of the data points have relatively low risk scores.

A graph of login frequency

Description automatically generatedIt is also important to note that the histogram does not show the actual values of the data points. It only shows the frequency of each range of scores. This means that it is possible that there are a few data points with very high risk scores, even though they are not visible in the histogram.

The login frequency appears to be relatively low, with most users logging in 1-2 times per day. There is a slight increase in the number of logins for users logging in 7-8 times per day. The login frequency appears to be higher than age group 1, with a peak around 4 logins per day. There is a steeper decline in the number of logins for users logging in more than 4 times per day compared to age group 1.

A graph of a graph

Description automatically generatedIn this specific histogram, the density line shows that there are more accounts with lower amounts than there are accounts with higher amounts. The density curve is highest around $2000 and tapers off towards both the lower and higher ends of the x-axis. This suggests that the majority of the accounts in this dataset have amounts around $2000.

A screenshot of a heatmap

Description automatically generatedThe largest portion of the pie chart, labeled "low\_risk" in blue, represents 83.8% of the population. This suggests that the majority of transactions are considered to be low risk. The orange slice labeled "moderate\_risk" accounts for 10.9% of the population. This indicates a smaller portion of the population falls under the moderate risk category. The smallest slice, labeled "high\_risk" in red, represents 5.3% of the population. This suggests that a relatively small portion of the population is considered high risk. A pie chart with different colored triangles

Description automatically generated

This correlation matrix is plotted for all the numerical features of the dataset of metaverse financial transactions. This shows that except one of the feature all the other feature are not correlated with each other. From this heatmap, login\_frequency and sessopm\_duration are heavily correlated with each other with the value of 0.87. So, for the model, one of the feature should be removed.

**FEATURE ENGINEERING**

**Removing Unwanted Features**

The timestamp column was converted into separate columns for month and day to extract temporal information. Additionally, irrelevant features such as sending and receiving addresses were removed from the dataset to streamline subsequent analyses. The features, sending\_address and receiving\_address are just from the id of the user so it needs to be removed. Also the correlated features are also removed such as session\_duration.

**Extracting Target Variable and Features**

The target variable 'anomaly' was extracted into a separate DataFrame 'Y' to facilitate supervised learning tasks. Categorical and numerical features were also separated into 'X\_Categorical' and 'X\_Numerical' DataFrames, respectively, to prepare for feature encoding and scaling.

**Handling Class Imbalance**

To mitigate class imbalance in the dataset with outcome variables 0, 1, and 2, various resampling methods can be utilized. Oversampling, a common approach, involves duplicating instances from minority classes to match the majority class's size. This method, often referred to as Random Oversampling, aims to artificially increase minority class representation, ensuring they receive adequate attention during model training. Conversely, undersampling, such as Random Undersampling, entails randomly removing instances from the majority class to balance class distribution. While effective in reducing imbalance, undersampling may lead to information loss from the majority class. Hybrid methods, like SMOTE (Synthetic Minority Over-sampling Technique), blend oversampling and undersampling to maintain class balance while minimizing data loss. These strategies, tailored to the dataset's characteristics, enable the development of more robust predictive models capable of handling imbalanced data distributions.

**One-Hot Encoding for Categorical Features**

Categorical features in 'X\_Categorical' were one-hot encoded using a ColumnTransformer to convert them into a suitable format for machine learning algorithms. This transformation ensures that categorical variables are represented as binary vectors, preserving their information without imposing any ordinality.

In this section, categorical features such as transaction type, location region, purchase pattern, and age group were encoded using one-hot encoding to prepare them for machine learning analysis. The ColumnTransformer was employed to apply the OneHotEncoder to the specified categorical features while preserving other non-categorical features using the remainder='passthrough' parameter. After encoding, the resulting binary vectors representing each categorical variable were concatenated with the original DataFrame, forming the updated DataFrame X\_Categorical\_encoded. This DataFrame was then assigned back to X\_Categorical, replacing the original categorical features with their one-hot encoded counterparts. The updated DataFrame now contains a set of binary-encoded columns corresponding to each unique category within the original categorical features, ready for further analysis and modeling.

**Standard Scaling for Numerical Features**

Numerical features in 'X\_Numerical' were standardized using a StandardScaler to ensure that their values have a mean of 0 and a standard deviation of 1. Standardization improves the convergence and performance of certain machine learning algorithms by bringing all features to a similar scale.

**Mapping Risk Levels to Numerical Labels**

A dictionary mapping was applied to the 'anomaly' column in 'Y' to convert risk levels into numerical labels. This transformation enables the use of classification algorithms that require numerical target variables, facilitating the modeling of risk prediction tasks.

0 – low\_risk

1 – moderate\_risk

2 – high\_risk

**Combining Categorical and Numerical Features**

Finally, the encoded categorical features were concatenated with the scaled numerical features to create the combined feature matrix 'X\_Combined'. This integrated dataset incorporates both categorical and numerical information, providing a comprehensive input for subsequent machine learning models.

**LOGISTIC REGRESSION**

A computer screen shot of a number

Description automatically generatedA screenshot of a computer

Description automatically generatedThe process began with training logistic regression models using different sets of features: categorical values only, numerical values only, and both categorical and numerical values combined. For each model, the training accuracy was computed, providing insight into how well the model learned from the training data. Subsequently, the trained models were evaluated using the test dataset, where predictions were made and various classification metrics were calculated.

A screenshot of a computer

Description automatically generated

The logistic regression model trained solely on categorical values achieved a training accuracy of approximately 89.25%. Upon evaluation, it exhibited an overall accuracy of around 88.52%, with precision, recall, and F1-score values averaging at 78.47%, 88.52%, and 83.16%, respectively. The confusion matrix revealed correct predictions for class 0, but none for classes 1 and 2 due to their minimal representation.

Similarly, the model trained solely on numerical values attained a training accuracy of approximately 82.98%. When evaluated, it demonstrated an overall accuracy of around 82.45%, with precision, recall, and F1-score values averaging at 74.76%, 82.45%, and 77.92%, respectively. The confusion matrix depicted correct predictions for class 0 and class 1, but not for class 2.

The logistic regression model utilizing both categorical and numerical values achieved the highest training accuracy of approximately 95.64%. During evaluation, it exhibited an overall accuracy of around 95.37%, with precision, recall, and F1-score values averaging at 95.25%, 95.37%, and 95.29%, respectively. The confusion matrix illustrated accurate predictions across all classes.

A graph with a line

Description automatically generated

Following this, hyperparameter tuning was performed using GridSearchCV to optimize the logistic regression model's performance. Various combinations of hyperparameters were explored, and the best-performing model achieved an accuracy of approximately 95.36% on the test dataset. These results demonstrate the effectiveness of logistic regression in predicting outcome variables across different feature sets, with the combined feature set yielding the most promising results.

**K-NEAREST NEIGHBOURS (KNN)**

A screenshot of a computer screen

Description automatically generatedA screenshot of a computer screen

Description automatically generatedThe process began by training and evaluating KNN (K-Nearest Neighbors) classifiers using different sets of features: numerical features only, categorical features only, and combined features. For each dataset, the data was split into training and testing sets, and the KNN model was trained with a default K value of 5. After predictions were made on the test data, accuracy, precision, and recall scores were calculated to assess the model's performance. The classification report for each dataset provided additional insights into the model's performance across different outcome variables.

A screenshot of a computer screen

Description automatically generated

The KNN model trained on numerical features achieved an accuracy of approximately 83.47%, with precision and recall scores averaging at 80.82% and 83.47%, respectively. The classification report revealed varying performance across different outcome variables, with class 2 exhibiting the lowest precision and recall.

Similarly, the KNN model trained on categorical features attained an accuracy of approximately 87.52%, with precision and recall scores averaging at 86.39% and 87.52%, respectively. The classification report highlighted challenges in predicting class 1, with lower precision and recall compared to the other classes.

A graph of different colored bars

Description automatically generated

The KNN model utilizing combined features achieved the highest accuracy of approximately 94.48%, with precision and recall scores averaging at 94.45% and 94.48%, respectively. The classification report demonstrated improved performance across all outcome variables compared to models trained on individual feature sets.

A graph with a line drawn on it

Description automatically generated

Additionally, the impact of different K values on model performance was explored using the combined feature set. By varying the K values (3, 5, 7, and 9), the accuracy scores were plotted to identify the optimal K value. The results indicated that K = 5 yielded the highest accuracy among the tested values.

Overall, these findings highlight the effectiveness of KNN classification in predicting outcome variables based on various sets of features, with combined features providing the most robust performance.

**DECISION TREE**

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedThe decision tree classifier was trained and evaluated using three different sets of features: numerical features only (X\_Numerical), categorical features only (X\_Categorical), and combined features (X\_Combined). For each dataset, the data was split into training and testing sets, and the decision tree classifier was trained with a maximum depth of 4. The evaluation metrics including accuracy, precision, recall, and F1 score were calculated for each dataset to assess the model's performance.

A screenshot of a computer

Description automatically generated

For the X\_Numerical dataset, the decision tree classifier achieved an accuracy of approximately 86.95%, with precision and recall scores averaging at 80.27% and 86.95%, respectively. However, the model struggled to predict class 2, resulting in lower precision, recall, and F1 score for this class.

Similarly, for the X\_Categorical dataset, the decision tree classifier attained an accuracy of approximately 88.52%, with precision and recall scores averaging at 78.47% and 88.52%, respectively. The classification report highlighted challenges in predicting class 1, with precision and recall scores close to zero for this class.

In contrast, the decision tree classifier trained on the combined features (X\_Combined) achieved the highest accuracy of approximately 94.47%, with precision and recall scores averaging at 94.19% and 94.47%, respectively. The model exhibited improved performance across all outcome variables compared to models trained on individual feature sets.

A graph with blue and orange lines

Description automatically generated Additionally, the impact of different maximum depth values on model performance was explored by varying the maximum depth parameter from 1 to 5. The evaluation metrics for each parameter value were recorded, and the results indicated that a maximum depth of 4 yielded the highest accuracy among the tested values.

Overall, these findings underscore the importance of feature selection and parameter tuning in improving the performance of decision tree classifiers for multi-class classification tasks. The combined feature set demonstrated superior performance compared to using individual feature sets, highlighting the benefits of leveraging diverse sources of information for classification.

**SUPPORT VECTOR MACHINES (SVM)**

The SVM classifier was trained and evaluated using three different sets of features: numerical features only (X\_Numerical), categorical features only (X\_Categorical), and combined features (X\_Combined). Each dataset was split into training and testing sets, and the SVM classifier was trained using the default parameters.

For the X\_Numerical dataset, the SVM classifier achieved a test accuracy of approximately 85.93%, with precision and recall scores averaging at 79.01% and 85.93%, respectively. The F1 score was approximately 82.31%. The classification report indicated challenges in predicting class 2, with precision, recall, and F1 score close to zero for this class.

A screenshot of a computer

Description automatically generatedA screenshot of a computer screen

Description automatically generatedA screenshot of a computer

Description automatically generated Similarly, for the X\_Categorical dataset, the SVM classifier attained a test accuracy of approximately 88.52%, with precision and recall scores averaging at 78.47% and 88.52%, respectively. The F1 score was approximately 83.16%. The classification report highlighted difficulties in predicting class 1, with precision and recall scores close to zero for this class.

In contrast, the SVM classifier trained on the combined features (X\_Combined) achieved the highest test accuracy of approximately 95.87%, with precision and recall scores averaging at 95.73% and 95.87%, respectively. The F1 score was approximately 95.77%. The model demonstrated improved performance across all outcome variables compared to models trained on individual feature sets.

Additionally, a grouped bar chart was plotted to visualize the performance metrics (accuracy, precision, recall, and F1 score) for each dataset. The chart indicated that the SVM classifier trained on the combined features outperformed the classifiers trained on individual feature sets across all metrics.

A graph of different colored bars

Description automatically generated

These results underscore the importance of feature engineering and combining diverse sources of information to enhance the performance of SVM classifiers for multi-class classification tasks. The combined feature set yielded the most robust and accurate predictions, highlighting the effectiveness of leveraging both numerical and categorical features in the classification process.

**MODEL COMPARISON AND CONCLUSION**

First, define the models and their corresponding performance metrics (accuracy, precision, recall) as lists. Then, specify the colors for each metric. Next, create a figure and define the parameters for the grouped bar chart, such as figure size, bar width, and index. Plot the bars for accuracy, precision, and recall using plt.bar(), adjusting the x-coordinates for each metric to ensure they are grouped together. Add labels, title, and legend to the plot. Annotate the bars with their values using plt.text(). Finally, adjust the y-axis limits based on the metric values and display the plot.

A graph of different colored bars

Description automatically generated

The evaluation of classification models, including Decision Tree, Logistic Regression, SVM, and KNN, based on accuracy, precision, and recall metrics reveals valuable insights into their performance in categorizing financial transactions within the Open Metaverse dataset.

The SVM model emerged as the top performer in terms of accuracy, achieving an impressive score of approximately 95.87%. This indicates its superior ability to accurately classify transactions into low, moderate, or high-risk tiers. Following closely behind, Logistic Regression and KNN demonstrated strong performances, with accuracies around 95.37% and 94.48%, respectively. Decision Tree also exhibited commendable accuracy, scoring above 94%.

Precision analysis further illuminates the models' capabilities in minimizing false positives. SVM and Logistic Regression stood out with precision scores of approximately 95.73% and 95.25%, respectively, showcasing their effectiveness in correctly identifying high-risk transactions while minimizing misclassifications. Decision Tree and KNN also demonstrated strong precision scores, hovering around 94.19% and 94.45%, respectively.

When considering recall, which measures the models' ability to capture all actual positive instances, SVM once again excelled with a score of approximately 95.87%. Logistic Regression closely trailed with a recall score of around 95.37%, indicating its capability to identify a significant portion of high-risk transactions. Additionally, Decision Tree and KNN exhibited robust recall scores, both surpassing the 94% mark.

In summary, while all models showcased commendable performances, SVM emerged as the most suitable choice for this classification task within the Open Metaverse dataset. Its high accuracy, precision, and recall scores highlight its effectiveness in accurately categorizing financial transactions and assessing associated risk levels. These findings underscore the importance of leveraging machine learning techniques to enhance fraud detection, risk assessment, and security management within virtual ecosystems, ultimately fostering a safer and more trustworthy financial landscape within the Open Metaverse.