An Industry-Oriented Mini Project Report

On

"Machine Learning based Analysis of Cryptocurrency Market Financial Risk Management"

Submitted in Partial Fulfillment of the Academic Requirement for the Award of Degree

BACHELOR OF TECHNOLOGY

in

Computer Science and Engineering (Artificial Intelligence and Machine learning)

Submitted By:

B. Sai Charan	**(21R01A6678)**
K. Ashwini	**(21R01A6689)**
KB. Aakash	**(21R01A6690)**

Under the esteemed guidance of

Dr. L. Arokia Jesu Prabhu

(Assoc.Prof. CSE(AI&ML))

CMR INSTITUTE OF TECHNOLOGY

(UGC AUTONOMOUS)

Approved by AICTE, Affiliated to JNTUH, Accredited by NAAC with A+ Grade, NBA Accredited

Kandlakoya(V), Medchal Dist - 501 401

www.cmrithyderabad.edu.in

2024-25

CMR INSTITUTE OF TECHNOLOGY

(UGC AUTONOMOUS)

Approved by AICTE, Affiliated to JNTUH, Accredited by NAAC with A+ Grade, NBA Accredited

Kandlakoya (V), Medchal Dist - 501 401

www.cmrithyderabad.edu.in

CERTIFICATE

This is to certify that an Industry oriented Mini Project entitled with "Machine Learning based analysis of Cryptocurrency Market Financial Risk management" is being submitted by:

B. Sai Charan	**(21R01A6678)**
K. Ashwini	**(21R01A6689)**
KB. Aakash	**(21R01A6690)**

To JNTUH, Hyderabad, in partial fulfillment of the requirement for award of the degree of B. Tech in CSE (AI&ML) and is a record of a bonafide work carried out under our guidance and supervision. The results in this project have been verified and are found to be satisfactory. The results embodied in this work have not been submitted to have any other University for award of any other degree or diploma.

Signature of Guide	Signature of Project Coordinator	Signature of HOD
Dr. L. Arokia Jesu Prabhu	**Dr. L. Arokia Jesu Prabhu**	**Prof. P. Pavan Kumar**

ACKNOWLEDGEMENT

We are extremely grateful to **Dr. M Janga Reddy**, Director, **Dr. G. Madhusudhana Rao**, Principal and **Prof. P. Pavan Kumar**, Head of Department, Dept of Computer Science and Engineering(Artificial Intelligence and Machine Learning), CMR Institute of Technology for their inspiration and valuable guidance during entire duration.

We are extremely thankful to**, Dr. L. Arokia Jesu Prabhu** Mini Project Coordinator and internal guide **Dr. L. Arokia Jesu Prabhu** Dept of Computer Science and Engineering (Artificial Intelligence and Machine Learning), CMR Institute of Technology for their constant guidance, encouragement and moral support throughout the project.

We will be failing in duty if we do not acknowledge with grateful thanks to the authors ofthe references and other literatures referred in this Project.

We express our thanks to all staff members and friends for all the help and coordination extended in bringing out this Project successfully in time.

Finally, we are very much thankful to our parents and relatives who guided directly or indirectly for every step towards success.

B. Sai Charan **(21R01A6678)**
K. Ashwini **(21R01A6689)**
KB. Aakash **(21R01A6690)**

ABSTRACT

With huge popularity gained in the past few years, the cryptocurrency market is highly volatile in nature, posing great challenges to financial risk management. Its unpredictability in price, lack of regulatory frameworks, and susceptibility to speculative trading create big problems for traditional risk assessment. Issues thus call for better and advanced techniques to improve the quality of decision-making and reducing losses that result from financial risks.

With the latest advancements in machine learning, it has become possible to process large datasets and look for patterns or predict market trends or classify risks. The best classification models are Support Vector Machines (SVM), Random Forest, and Neural Networks. The integration of sentiment analysis and real-time market data has added further value to predictions and given actionable insights to investors.

However, the existing approaches have several limitations, like overfitting due to noisy data, difficulty in handling rapid market dynamics, and a lack of exploration of the impact of external factors like geopolitical events or blockchain-specific metrics. In addition, most models are not scalable and fail to adapt quickly to evolving market trends, thus limiting their practical application in real-time trading scenarios.

This paper tries to bridge this gap by developing a strong classification model that combines historical market data, sentiment analysis, and blockchain-specific indicators for predicting financial risks in cryptocurrency trading. The proposed model is focused on adaptive learning techniques and rigorous feature selection to ensure scalability, accuracy, and reliability in managing market risks.

The proposed approach draws from several key studies in the field of portfolio theory approach, by Planakis and A. Urquat (2020) . Another high frequency multiscale relationship among major cryptocurrencies portfolio management implications proposed by Financial innovations of Rehman Sha_ullah (2021). These studies collectively inform the design of our machine learning based frameworks, that are aimed at improving the detection of inappropriate content on platforms like YouTube.

INDEX

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1 ABOUT PROJECT

It emerged as a revolutionary financial ecosystem, offering decentralized digital assets and innovative technologies such as blockchain. Although very potential, the market features extreme volatility, rapid fluctuations of prices, and a vulnerability to speculative trading; it is a significant challenge to financial risk management in front of investors and stakeholders. Different from traditional financial markets, cryptocurrencies lack historical regulatory frameworks and exhibit unique patterns in terms of technology adoption, market sentiment, and other external events.

ML has promise in its use of data-driven approaches for understanding underlying patterns, which makes outcomes-based predictions feasible and allows the creation of algorithms to automate decision-making processes. In the realm of cryptocurrencies, that same technology could be utilized using methods in ML as robustly designed methodologies that assess voluminous data for possible categorization of market condition related to financial risks. Classification models, for example, can categorize market states into pre-defined risk groups, like high-risk, medium-risk, or low-risk categories, allowing for proactive strategies to avoid loss and maximize returns.

This addresses the development and evaluation of classification-based machine learning models for cryptocurrency financial risk management. The proposed models will integrate historical market data, trading volumes, sentiment analysis, and other critical indicators to classify the risk levels of the market accurately. The research not only emphasizes the potential of ML classification models in risk management but also addresses the unique challenges of cryptocurrency data, including noise, sparsity, and dynamic behavior. This section of the paper discusses the methodology, key features, machine learning models applied, and performance evaluation. It demonstrates that classification models may be useful for improving the management of risks and that this work forms a base for further exploration of applications of ML in the ever-changing cryptocurrency market development.

1.2 EXISTING SYSTEM

In previous developments, traditional methods for assessing risks such as mean-variance optimization and inverse volatility allocation are used for cryptocurrencies. These techniques have their own sets of limitations in dealing with the risks associated with the cryptocurrency market, including its volatile nature and changing unpredictably very often. In the existing system, Hierarchical Risk Parity is used as an advancement of traditional risk-based allocation methods. HRP would look to maximize portfolios by aligning it with asset correlation in order to limit tails risk. Anti-money laundering is also part of the existing systems within financial institutions. However, blockchain transactions lack anonymity, making fraud and money laundering with cryptocurrencies impossible to trace and prevent easily. The systems employ techniques in clustering for identifying unusual patterns or correlations of assets as signs of risks or fraudulent activity. Traditional models do not precisely address the uniqueness of cryptocurrencies in terms of decentralization, regulatory uncertainties, and extreme volatility. Robust correlation matrices are not developed for such assets; rather, data quality and coverage pose a great issue.

Disadvantages

- Choosing the exchange of cryptocurrency based on the entity contains no control on transactions and its overbalanced for the maintained account of the entity.
- Cryptocurrency wallet which is belonging to the entity has no account.
- Its not possible to access to cryptocurrency by loosing the private key.
- If an unauthorized party get any access to the private key then all the cryptocurrency stolen.
- Sending the incorrect address from entity which is not possible of recovery from cryptocurrency.

1.3 PROPOSED SYSTEM

Application with advanced machine learning techniques and models to predict and manage risks related to high volatility of digital assets: historical data gathered from varied sources, including Kaggle and GitHub, cleaned on missing values and outliers by features: measures of price change percentage and volatility measures as ways for improving model performance. Some techniques employed include exploratory data analysis of patterns and correlations mainly related to price trends and volatility factors that are relevant in the context of cryptocurrency markets. Model acquired through multiple machine learning algorithms that were selected for training such include Naïve Bayes, SVM, logistic regression, and decision trees. Such models are suited as applied to financial data often found noisy and unpredictable. The system makes use of these models for predicting risk at financial levels. In fact, it does so in distinguishing between the periods that are high-risk and those that are low-risk.

All the models are assessed in terms of performance metrics of accuracy, precision, recall, and F1 score. The model's predictions are plotted against actual data, so that when graphed and charted, their trends as well as findings are clearer to the users. Cross-validation methods are applied to check for the strength of the models. Their results are benchmarked and compared against similar reports. Its advanced machine learning models, wider scope of risk metrics, real-time processing of data, and a set of additional security features can be considered as the potential improvements.

Advantages

- The proposed system implements a graph-based theory and using the machine learning techniques, the proposed system is processing in the following way.
- Clustering datasets.
- Recursive bisection on datasets.
- Quasi-diagonalization on datasets.

2. LITERATURE SURVEY

2.1 Effects of Cryptocurrency in Tail Risk Network during COVID-19

AUTHORS: Rui Ren (2020)
Rui Ren and published by the **International Research Training Group**, investigates the impact of cryptocurrency markets on tail risk networking during the COVID-19 pandemic. The research leverages the duality problem in cryptocurrency markets, an approach that considers the bidirectional relationship between market behavior and risk dynamics. The research provides a framework to determine the network topology by examining spillover effects—instances where risk from one cryptocurrency influences others within the market. This spillover analysis uncovers the structural characteristics of risk transmission and highlights the interdependencies among various cryptocurrencies.

2.2 Cryptocurrency Regularity Risk Index Based on Machine Learning

AUTHORS: Xinwen (2021)
Wolfgang Karl Härdle in collaboration with **Xinwen**, was published in **2021**. The research employs machine learning techniques to quantify **market risks** arising from changes in regulatory frameworks. The solution involves a comprehensive analysis of how the introduction of new regulations or changes to existing ones originates market risks. Using the machine learning model, the study identifies the critical factors influencing risk and evaluates their impact on market stability. This approach highlights the dynamic relationship between regulatory environments and cryptocurrency market behavior.

2.3 Investing the risks and returns of cryptocurrencies

AUTHORS: Debi Eka (2021)
Debi Eka and published in **2021** in *The Review of Financial Studies*, provides a comprehensive analysis of the risks and returns associated with cryptocurrencies. The study focuses exclusively on legal statements and their implications for cryptocurrency markets. It employs a **heteroscedastic model** to measure risks, capturing the variability in market returns under changing legal and regulatory conditions. The solution involves the quantification of market risks through the heteroscedastic model, enabling a nuanced understanding of how legal factors contribute to market volatility. The model assesses the dynamic risk-return trade-offs, reflecting the sensitivity of cryptocurrencies to legal developments.

2.4 Return and Risk Analysis on Cryptocurrency Assets

AUTHORS: Sakina Ichsani (2022)
Sakina Ichsani and published in **2022** in the *Kontigensi Scientific Journal of Management*, examines the risk-return dynamics of cryptocurrency assets. The study adopts an analytical approach to assess the **returns and risks** of individual cryptocurrency assets. By examining historical price data and market performance, the research quantifies the average returns and volatilities of selected cryptocurrencies. The solution involves summarizing the **average return** and **volatility** of each cryptocurrency to provide a clear understanding of their investment profiles. This analysis identifies patterns in performance and risk, enabling investors to make informed decisions.

These studies explore the multifaceted dynamics of cryptocurrency markets, focusing on risks, returns, regulatory impacts, and network spillovers. Analyzes spillover effects in cryptocurrency networks during extreme events. Uses machine learning to assess market risks from regulatory changes. Evaluates legal impacts on risk-return profiles using heteroscedastic models. Summarizes average returns and volatilities of cryptocurrencies for investment insights. Together, these works highlight the intricate balance between opportunity and risk in the cryptocurrency ecosystem, shaped by market behavior, legal frameworks, and systemic interconnections.

3. SYSTEM DESIGN

3.1 ARCHITECTURE / BLOCK DIAGRAM

The architecture has incorporated the element of the service provider, whereby data is processed, trained, and tested to yield insights. The web server provides user requests, in this case, seeing the risk types and ratios, with proper handling in retrieval of data storage. Remote users log in to predict the potential financial risk that may result from the cryptocurrency and view their results, which are visualized on charts to make it clearer. Its applicability lies in the prediction of correct values by processing a large dataset, hence suitable for investors and analysts who assess cryptocurrency risks (Architecture).

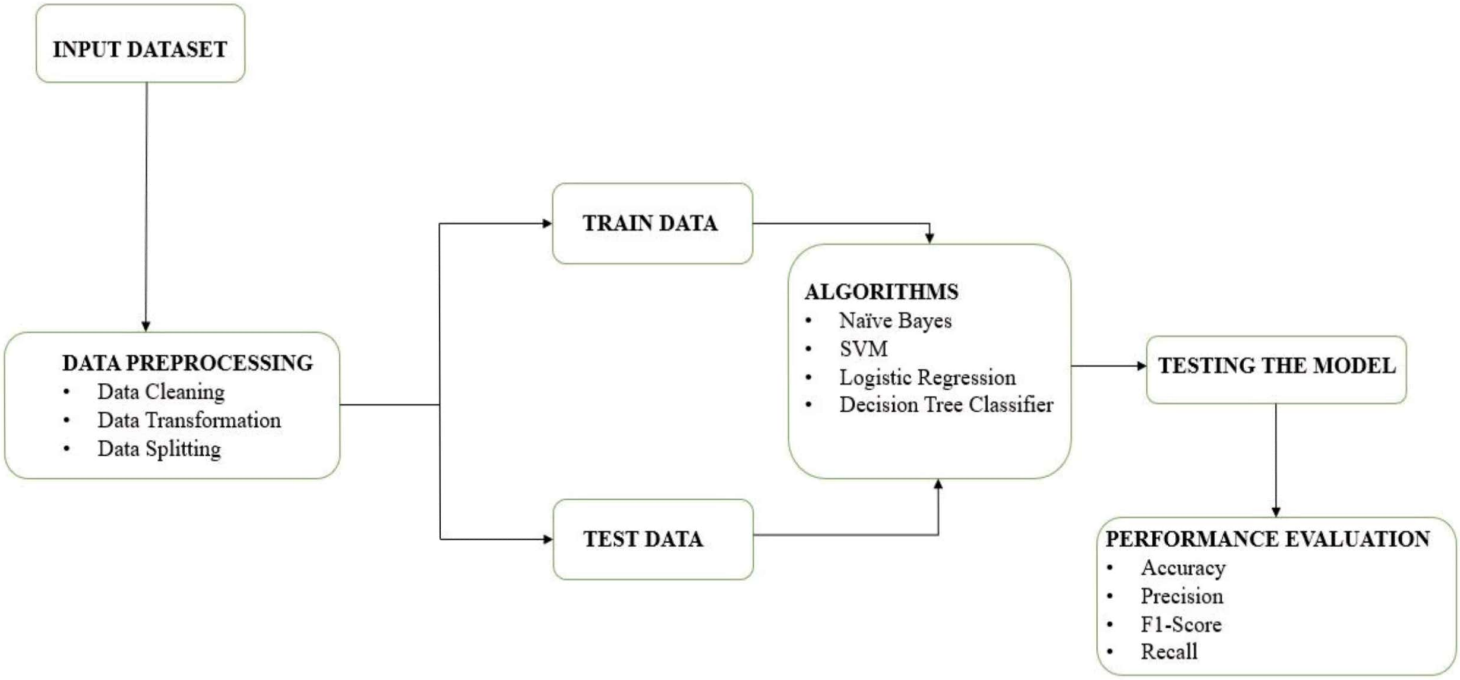

Fig 3.1.1: Architecture Diagram

The architecture diagram represents a systematic workflow for implementing machine learning-based classification models to analyze and evaluate financial risks in the cryptocurrency market.

1. Input Dataset: The process begins with raw data, which serves as the foundation for model training and testing.

2. Data Preprocessing: The dataset undergoes cleaning, transformation, and splitting to prepare it for machine learning algorithms. This ensures data quality and readiness for training.

3. Train Data and Test Data: The preprocessed data is divided into training and testing sets to train models and evaluate their performance, respectively.

4. Algorithms: Classification models, including Naïve Bayes, SVM, Logistic Regression, and Decision Tree Classifier, are applied to the training data for risk classification.

5. Testing the Model: The trained models are tested on unseen test data to assess their generalization capabilities.

6. Performance Evaluation: Finally, the models are evaluated using key metrics such as accuracy, precision, F1-score, and recall to determine their effectiveness in financial risk classification.

This architecture ensures a structured and efficient approach to developing and evaluating machine learning models for cryptocurrency risk management.

3.2 Data Flow Diagram

A Data Flow Diagram (DFD) is a visual representation that illustrates how data moves through a system. It's a valuable tool used in software engineering and systems analysis to understand, analyze, and design information systems. Below is the data flow diagram where the key components are service provider, user, web server and web storage. By understanding the flow of data within a system, DFDs help in making informed decisions about system design.

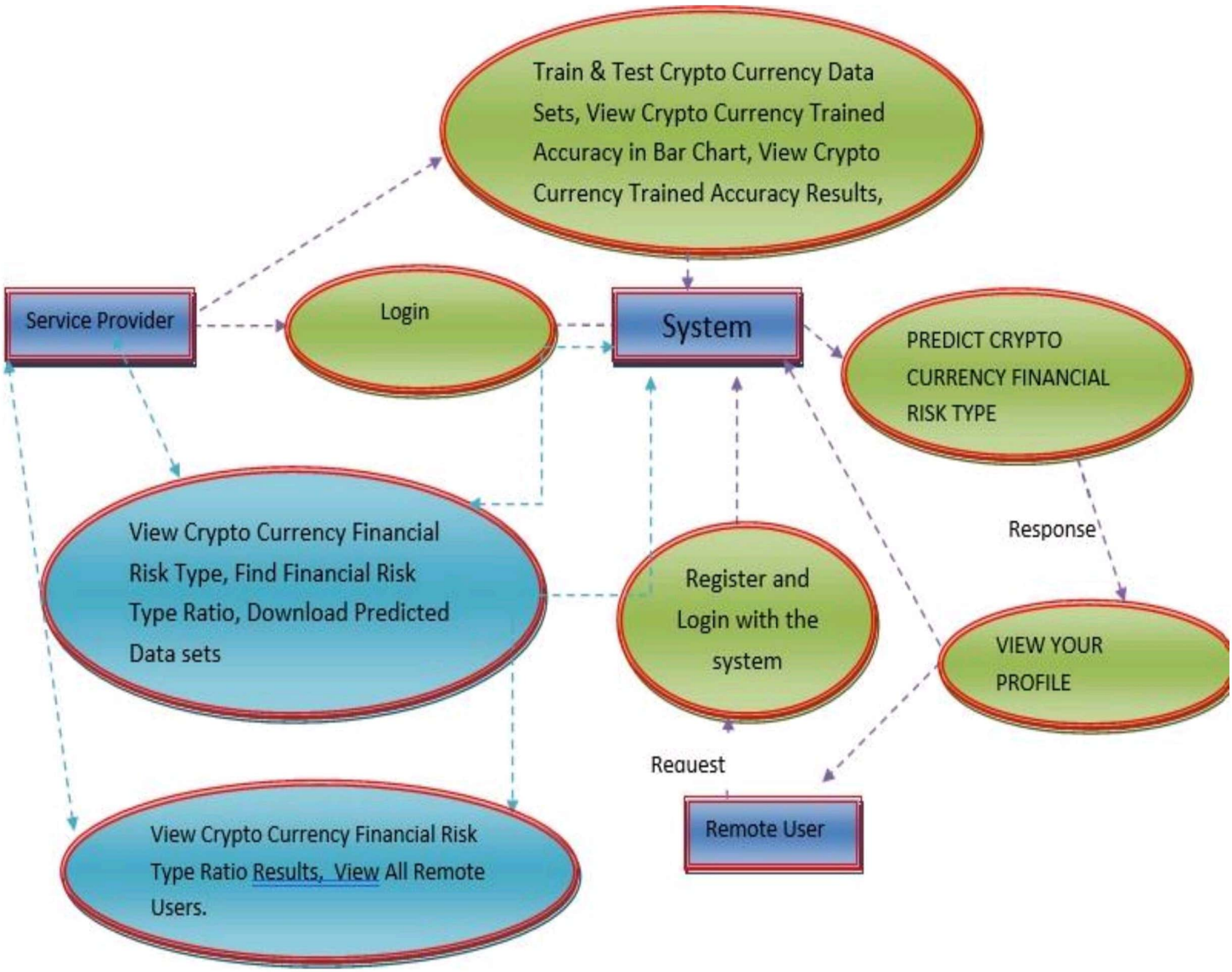

Fig: 3.2.1 Data Flow Diagram

User Interaction:

The remote user initiates the data flow by interacting with the system through the "Login" or "Register and Login with the system" options.

Request:

The user's request, which could be to predict a used car price type, view their profile, or access other functionalities, is sent to the system.

System Processing:
The system receives the request and processes it based on the user's input.

Prediction or Calculation:
The system performs the required calculations or predictions, such as determining the used car price type based on the input data.

Response Generation:
The system generates a response to the user's request, which could be a predicted car price type, a profile view, or other relevant information.

Response Delivery:
The system sends the generated response back to the remote user.

3.3 CLASS DIAGRAM

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

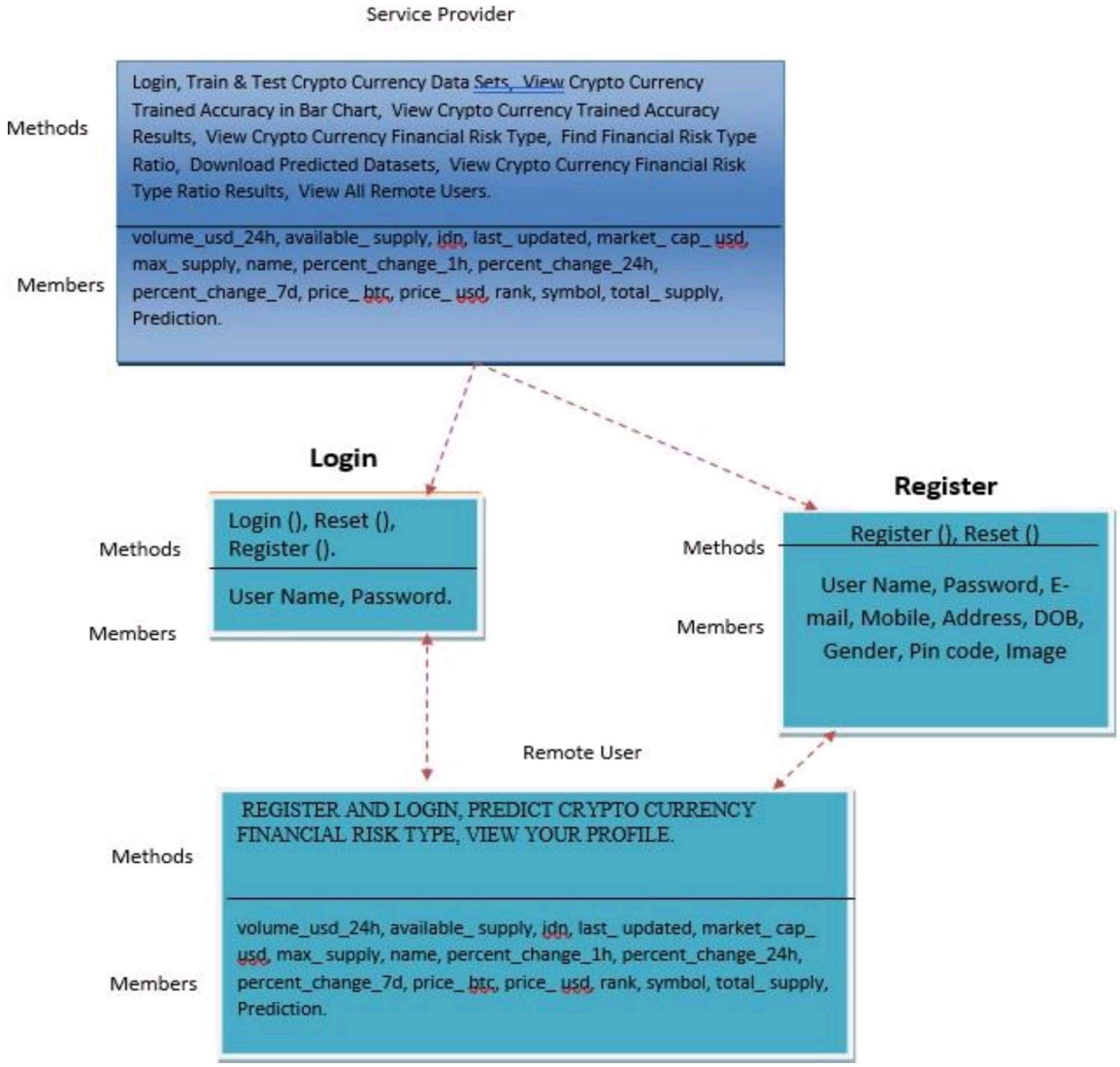

Fig 3.3.1 Class diagram

Service Provider:

This class is responsible for handling various operations related to used car data, including training, testing, prediction, and data analysis.

User:

This class represents a user of the system, who can register, login, predict used car prices, and view their profile.

Register:

This class is a specialized class for handling user registration and login functionalities.

4. IMPLEMENTATION

4.1 PROJECT MODULES

1. Data Collection and Preprocessing

- Data was gathered from websites and newspaper advertisements comprising 200 records with attributes such as year, make, engine capacity, mileage, and price.
- Preprocessing involved handling missing values, normalizing features, and encoding categorical data.

2. Model Development

- Algorithms used:
- Support Vector Machine (SVM)
- Logistic Regression
- Naïve Bayes
- Decision Tree Classifier

3. Evaluation and Results

- Performance Metrics: Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE).
- Results: SVM outperformed others, with SVM showing the highest accuracy.

4. User Interaction

- Service Provider Module: For training, testing models, and analyzing results.
- Remote User Module: Allowed users to predict car prices based on input attributes.

5. Deployment

- A user-friendly interface enables real-time predictions and model management, ensuring the system is accessible and effective

4.2 ALGORITHMS

The core of the project relies on machine learning algorithms to predict the risks of cryptocurrency market. The following algorithms were implemented and tested:

1. Naive Bayes

Purpose: Naive Bayes is used for **classification tasks**. It is particularly effective for text classification, spam detection, and sentiment analysis. It assumes independence among predictors, making it computationally efficient.

Implementation:

- Calculate prior probabilities of each class.
- Compute the likelihood of each feature given the class.
- Use Bayes' Theorem to calculate the posterior probability for each class.
- Assign the class with the highest posterior probability.

2. Support Vector Machine (SVM)

Purpose: SVM is a supervised learning algorithm used for **classification** and **regression**. It is ideal for datasets with clear margins of separation and is effective in high-dimensional spaces.

Implementation:

- Transform data to a higher dimension using a kernel function if necessary.
- Find a hyperplane that maximizes the margin between classes.
- Use the hyperplane to classify new data points.

3. Logistic Regression

Purpose: Logistic regression is used for **binary classification** (or multiclass using extensions). It models the probability of a target class using a sigmoid function.

Implementation:

- Calculate the weighted sum of input features.
- Apply the sigmoid function to map outputs to a probability range (0, 1).
- Classify based on a threshold (e.g., 0.5).

4. Decision Tree Classifier

Purpose: Decision trees are used for **classification** and **regression** tasks. They model decisions and their possible consequences as a tree structure, making them interpretable.

Implementation:

- Start with the entire dataset at the root.
- Split the data based on the feature that provides the highest information gain or lowest Gini impurity.
- Repeat splits recursively until stopping criteria are met.

Advantages

1. **Automation of Complex Tasks**
   Machine learning (ML) automates tasks that are too complex for traditional programming, such as image recognition, speech processing, and anomaly detection.

2. **Improved Accuracy and Predictions**
   ML algorithms improve with data. They can provide highly accurate predictions and classifications, especially in tasks like fraud detection, recommendation systems, or medical diagnostics.

3. **Scalability**
   Once trained, ML models can handle vast amounts of data, making them suitable for projects requiring real-time processing or large-scale deployment.

4. **Versatility**
   ML algorithms are versatile and applicable to diverse fields like finance, healthcare, marketing, and robotics.

5. **Data Insights**
   ML helps uncover hidden patterns and relationships in data, enabling informed decision-making and innovation.

6. **Customization**
   Algorithms can be tailored to specific project needs, ensuring adaptability to varying scenarios and data types.

Disadvantages

1. **High Data Dependency**
   ML algorithms require large, high-quality datasets for training. Poor or insufficient data can lead to unreliable models.

2. **Complexity and Interpretability**
   Many ML models, like deep learning, are black-box models, making them difficult to interpret and explain, especially in critical applications like healthcare or law.

3. **Resource Intensive**
   Training ML models can be computationally expensive and time-consuming, requiring powerful hardware and optimization.

4. **Overfitting or Underfitting**
   If not tuned properly, models may overfit (memorize the training data) or underfit (fail to learn adequately), leading to poor generalization.

5. **Maintenance Challenges**
   ML models require regular updates and retraining as data and conditions evolve, increasing maintenance overhead.

6. **Ethical and Bias Issues**
   ML models can perpetuate biases present in the training data, leading to unfair outcomes and ethical concerns in sensitive domains.

7. **Dependency on Expertise**
   Implementing and optimizing ML algorithms requires specialized knowledge, which can limit accessibility for non-experts.

5. TESTING

5.1 TESTING METHODS

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.
The following are the Testing Methodologies:

- **Unit Testing.**
- **Integration Testing.**
- **User Acceptance Testing.**
- **Output Testing.**
- **Validation Testing.**

5.1.1 Unit Testing

Unit testing focuses verification effort on the smallest unit of Software design that is the module. Unit testing exercises specific paths in a module's control structure to ensure complete coverage and maximum error detection. This test focuses on each module individually, ensuring that it functions properly as a unit. Hence, the naming is Unit Testing.

During this testing, each module is tested individually and the module interfaces are verified for the consistency with design specification. All important processing path are tested for the expected results. All error handling paths are also tested.

5.1.2 Integration Testing

Integration testing addresses the issues associated with the dual problems of verification and program construction. After the software has been integrated a set of high order tests are conducted. The main objective in this testing process is to take unit tested modules and builds a program structure that has been dictated by design.

The following are the types of Integration Testing:

- **Top Down Integration**

This method is an incremental approach to the construction of program structure. Modules are integrated by moving downward through the control hierarchy, beginning with the main

program module. The module subordinates to the main program module are incorporated into the structure in either a depth first or breadth first manner.

In this method, the software is tested from main module and individual stubs are replaced when the test proceeds downwards.

- **Bottom-up Integration**

This method begins the construction and testing with the modules at the lowest level in the program structure. Since the modules are integrated from the bottom up, processing required for modules subordinate to a given level is always available and the need for stubs is eliminated. The bottom up integration strategy may be implemented with the following steps:

- The low-level modules are combined into clusters into clusters that perform a specific Software sub-function.
- A driver (i.e.) the control program for testing is written to coordinate test case input and output.
- The cluster is tested.
- Drivers are removed and clusters are combined moving upward in the program structure

The bottom up approaches tests each module individually and then each module is module is integrated with a main module and tested for functionality.

5.1.3 User Acceptance Testing

User Acceptance of a system is the key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with the prospective system users at the time of developing and making changes wherever required. The system developed provides a friendly user interface that can easily be understood even by a person who is new to the system.

5.1.4 Output Testing

After performing the validation testing, the next step is output testing of the proposed system, since no system could be useful if it does not produce the required output in the specified format. Asking the users about the format required by them tests the outputs generated or displayed by the system under consideration. Hence the output format is considered in 2 ways – one is on screen and another in printed format.

5.1.5 Validation Testing

Validation checks are performed on the following fields:

Text Field:

The text field can contain only the number of characters lesser than or equal to its size. The text fields are alphanumeric in some tables and alphabetic in other tables. Incorrect entry always flashes and error message.

Numeric Field:

The numeric field can contain only numbers from 0 to 9. An entry of any character flashes an error messages. The individual modules are checked for accuracy and what it has to perform. Each module is subjected to test run along with sample data. The individually tested modules are integrated into a single system. Testing involves executing the real data information is used in the program the existence of any program defect is inferred from the output. The testing should be planned so that all the requirements are individually tested.

A successful test is one that gives out the defects for the inappropriate data and produces and output revealing the errors in the system.

Preparation of Test Data

Taking various kinds of test data does the above testing. Preparation of test data plays a vital role in the system testing. After preparing the test data the system under study is tested using that test data. While testing the system by using test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future use.

Using Live Test Data:

Live test data are those that are actually extracted from organization files. After a system is partially constructed, programmers or analysts often ask users to key in a set of data from their normal activities. Then, the systems person uses this data as a way to partially test the system. In other instances, programmers or analysts extract a set of live data from the files and have them entered themselves.

It is difficult to obtain live data in sufficient amounts to conduct extensive testing. And, although it is realistic data that will show how the system will perform for the typical processing requirement, assuming that the live data entered are in fact typical, such data generally will not test all combinations or formats that can enter the system. This bias toward typical values then does not provide a true systems test and in fact ignores the cases most

likely to cause system failure.

Using Artificial Test Data:

Artificial test data are created solely for test purposes, since they can be generated to test all combinations of formats and values. In other words, the artificial data, which can quickly be prepared by a data generating utility program in the information systems department, make possible the testing of all login and control paths through the program.

The most effective test programs use artificial test data generated by persons other than those who wrote the programs. Often, an independent team of testers formulates a testing plan, using the systems specifications.

The package "Virtual Private Network" has satisfied all the requirements specified as per software requirement specification and was accepted.

5.2 USER TRAINING

Whenever a new system is developed, user training is required to educate them about the working of the system so that it can be put to efficient use by those for whom the system has been primarily designed. For this purpose, the normal working of the project was demonstrated to the prospective users. Its working is easily understandable and since the expected users are people who have good knowledge of computers, the use of this system is very easy.

5.3 MAINTAINENCE

This covers a wide range of activities including correcting code and design errors. To reduce the need for maintenance in the long run, we have more accurately defined the user's requirements during the process of system development. Depending on the requirements, this system has been developed to satisfy the needs to the largest possible extent. With development in technology, it may be possible to add many more features based on the requirements in future. The coding and designing is simple and easy to understand which will make maintenance easier.

TESTING STRATEGY:

A strategy for system testing integrates system test cases and design techniques into a well planned series of steps that results in the successful construction of software. The testing strategy must co-operate test planning, test case design, test execution, and the resultant data

collection and evaluation. A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high level tests that validate major system functions against user requirements.

Software testing is a critical element of software quality assurance and represents the ultimate review of specification design and coding. Testing represents an interesting anomaly for the software. Thus, a series of testing are performed for the proposed system before the system is ready for user acceptance testing.

SYSTEM TESTING:

Software once validated must be combined with other system elements (e.g. Hardware, people, database). System testing verifies that all the elements are proper and that overall system function performance is achieved. It also tests to find discrepancies between the system and its original objective, current specifications and system documentation.

UNIT TESTING:

In unit testing different are modules are tested against the specifications produced during the design for the modules. Unit testing is essential for verification of the code produced during the coding phase, and hence the goals to test the internal logic of the modules. Using the detailed design description as a guide, important Conrail paths are tested to uncover errors within the boundary of the modules. This testing is carried out during the programming stage itself. In this type of testing step, each module was found to be working satisfactorily as regards to the expected output from the module.

In Due Course, latest technology advancements will be taken into consideration. As part of technical build-up many components of the networking system will be generic in nature so that future projects can either use or interact with this. The future holds a lot to offer to the development and refinement of this project.

6. RESULTS

A large number of experiments have been conducted in order to find the best network structure and the best parameters for the neural network. We found that a neural network with 1 hidden layer and 2 nodes produced the smallest mean absolute error among various neural network structures that were experimented with. However, we found that Support Vector Regression and a multilayer perceptron with back-propagation produced slightly better predictions than linear regression while the k-Nearest Neighbor algorithm had the worst accuracy among these four approaches. All experiments were performed with a cross-validation value of 10 folds. The results are summarized in Table 6.1 below.

Sl.no	Model Type	Accuracy
1	Naïve Bayes	50.751879
2	Support Vector Machine	53.383458
3	Logistic Regression	51.127819
4	Decision Tree Classifier	53.383458

Table 6.1: Model Types and Accuracy

Table 6.1 presents the accuracy scores of four different machine learning models applied to a specific dataset. Support Vector Machine (SVM) achieved the highest accuracy of 53.38%, followed by Decision Tree Classifier (53.38%), Logistic Regression (51.12%), and Naïve Bayes (50.75%). These results suggest that SVM is the most suitable model for this particular task, but further analysis may be necessary to confirm its superiority and identify potential areas for improvement.

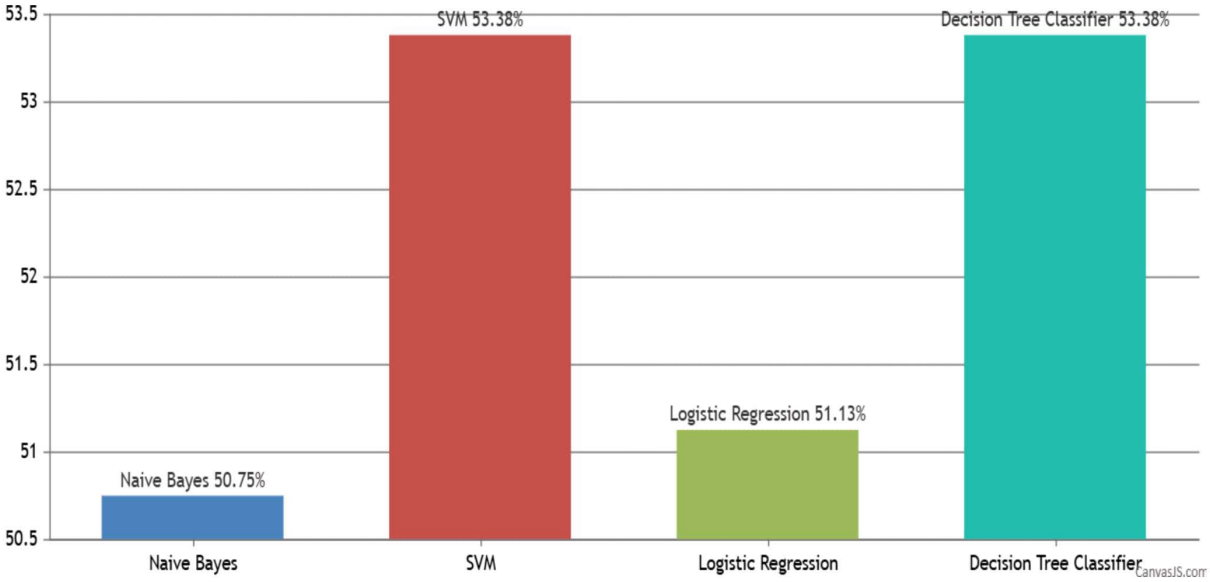

Fig 6.1 Model vs Accuracy Graph

The figure shows the variation in the accuracies in different machine learning models. As we can see the support vector machine model performed well against various models. The data in graph is fairly accurate and can relied upon in many cases.

7. CONCLUSION

The risk management of crypto currency network analysed using Naive bayes, Support vector machine, Logistic regression and Decision tree classification model technique and asset allocation method named as Hierarchical Risk Parity (HRP) that applied in crypto currencies portfolio. Naive bayes and Support vector machine gives a high performance evaluation results as compare to other machine learning techniques have been used in this area. The main reason of applying Naive bayes and Support vector machine in this process because of their different strengths in handling financial data, which can be noisy, unpredictable, and nonlinear. It gives the opportunity to system structure to get the high accuracy in term of providing the quality datasets.

8. REFERENCES

[1] C. Y. Kim and K. Lee, ``Risk management to cryptocurrency exchange and investors guidelines to prevent potential threats," in *Proc. Int. Conf. Platform Technol. Service (PlatCon)*, Jan. 2018, pp. 1_6.

[2] I. U. Haq, A. Maneengam, S. Chupradit, W. Suksatan, and C. Huo, ``Economic policy uncertainty and cryptocurrency market as a risk management avenue: A systematic review," *Risks*, vol. 9, no. 9, p. 163, Sep. 2021.

[3] J. Gold and S. D. Palley, ``Protecting cryptocurrency assets," *Risk Man- age.*, vol. 68, no. 3, pp. 12_13, 2021.

[4] I. Barkai, T. Shushi, and R. Yosef, ``A cryptocurrency risk_return analysis for bull and bear regimes," *J. Alternative Investments*, vol. 24, no. 1, pp. 95_118, Jun. 2021.

[5] V. Boiko, Y. Tymoshenko, R. Y. Kononenko, and D. Goncharov, ``The optimization of the cryptocurrency portfolio in view of the risks," *J. Manage. Inf. Decis. Sci.*, vol. 24, pp. 1_9, Sep. 2021.

[6] G. Köchling, ``Essays in _nance: Corporate hedging, mutual fund managers' behavior, and cryptocurrency markets," M.S. thesis, Universitätsbibliothek Dortmund, Germany, 2021.

[7] Z. Umar, N. Trabelsi, and F. Alqahtani, ``Connectedness between cryptocurrency and technology sectors: International evidence," *Int. Rev. Econ. Finance*, vol. 71, pp. 910_922, Jan. 2021.

[8] T. Kurosaki and Y. S. Kim, ``Cryptocurrency portfolio optimization with multivariate normal tempered stable processes and foster-hart risk," *Finance Res. Lett.*, vol. 45, Mar. 2022, Art. no. 102143.

[9] A. Masharsky and I. Skvortsov, ``Cryptocurrency market development in Latvia and the Baltic states," *Eur. Cooperation*, vol. 1, no. 49, pp. 7_22, 2021.

[10] S. Bhattacharya and K. Rana, ``A case study on cryptocurrency driven euphoria in 2020-21," *Int. J. Res. Eng., Sci. Manage.*, vol. 4, no. 3, pp. 9_11, 2021.

[11] H. Lohre, C. Rother, and K. A. Schäfer, ``Hierarchical risk parity: Accounting for tail dependencies in multi-asset multi-factor allocations," in *Machine Learning for Asset Management: New Developments and Financial Applications*. 2020, pp. 329_368.

[12] P. Jain and S. Jain, ``Can machine learning-based portfolios outperform traditional risk-based portfolios? The need to account for covariance misspeci _cation," *Risks*, vol. 7, no. 3, Jul,2019.

[13] T. Raf_not, ``Hierarchical clustering-based asset allocation," *J. Portfolio Manage.*, vol. 44, no. 2, pp. 89_99, Dec. 2017.

[14] T. Burggraf, ``Risk-based portfolio optimization in the cryptocurrency world," Available at SSRN 3454764, Tech. Rep., 2019.

[15] W. Mensi, M. U. Rehman, M. Sha_ullah, K. H. Al-Yahyaee, and A. Sensoy, ``High frequency multiscale relationships among major cryptocurrencies: Portfolio management implications," *Financial Innov.*, vol. 7, no. 1, pp. 1_21, Dec. 2021.

[16] E. Platanakis, C. Sutcliffe, and A. Urquhart, ``Optimal vs naïve diversi_- cation in cryptocurrencies," *Econ. Lett.*, vol. 171, pp. 93_96, 2018.

[17] E. Platanakis and A. Urquhart, ``Should investors include bitcoin in their portfolios? A portfolio theory approach," *Brit. Accounting Rev.*, vol. 52, no. 4, Jul. 2020, Art. no. 100837.

[18] S. Qureshi, M. Aftab, E. Bouri, and T. Saeed, ``Dynamic interdependence of cryptocurrency markets: An analysis across time and frequency," *Phys. A, Stat. Mech. Appl.*, vol. 559, Dec. 2020, Art. no. 125077.

[19] S. Corbet, V. Eraslan, B. Lucey, and A. Sensoy, ``The effectiveness of technical trading rules in cryptocurrency markets," *Finance Res. Lett.*, vol. 31, pp. 32_37, Dec. 2019.

[20] S. Kethineni and Y. Cao, ``The rise in popularity of cryptocurrency and associated criminal activity," *Int. Criminal Justice Rev.*, vol. 30, no. 3, pp. 325_344, Sep. 2020.

[21] B. Silahli, K. D. Dingec, A. Cifter, and N. Aydin, ``Portfolio value-atrisk with two-sided Weibull distribution: Evidence from cryptocurrency markets," *Finance Res. Lett.*, vol. 38, Jan. 2021, Art. no. 101425.

[22] A. Kaplan, ``Cryptocurrency and corruption: Auditing with blockchain," in *Auditing Ecosystem and Strategic Accounting in the Digital Era*. Springer, 2021, pp. 325_338.

[23] Y. Yang and Z. Zhao, ``Large cryptocurrency-portfolios: Ef_cient sorting with leverage constraints," *Appl. Econ.*, vol. 53, no. 21, pp. 2398_2411, May 2021.

[24] S. Fan, S. Fu, H. Xu, and X. Cheng, ``Al-SPSD: Anti-leakage smart Ponzi schemes detection in blockchain," *Inf. Process. Manage.*, vol. 58, no. 4, Jul. 2021, Art. no. 102587.

[25] U. Hacioglu, D. Chlyeh, M. K.Yilmaz, E. Tatoglu, and D. Delen, ``Crafting performance-based cryptocurrency mining strategies using a hybrid analytics approach," *Decis. Support Syst.*, vol. 142, Mar. 2021, Art. no. 113473.

APPENDIX I – SCREENSHOTS

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	idn	24h_volun	available_:	id	last_updat	market_ca	max_supp	name	percent_cl	percent_cl	percent_cl	price_btc	price_usd	rank	symbol	total_supp	Label
2	0	9.01E+09	16723525	bitcoin	1.51E+09	2.13E+11	21000000	Bitcoin	0.12	7.33	17.45	1	12739.5	1	BTC	16723525	1
3	1	1.55E+09	96165368	ethereum	1.51E+09	4.35E+10		Ethereum	-0.18	-3.93	-7.33	0.036177	452.652	2	ETH	96165368	0
4	2	1.11E+09	16840438	bitcoin-ca:	1.51E+09	2.53E+10	21000000	Bitcoin Ca:	1.65	-5.51	-4.75	0.12005	1502.09	3	BCH	16840438	1
5	3	2.94E+09	2.78E+09	iota	1.51E+09	1.48E+10	2.78E+09	IOTA	-2.38	83.35	255.82	0.000424	5.30746	4	MIOTA	2.78E+09	0
6	4	2.32E+08	3.87E+10	ripple	1.51E+09	9.37E+09	1.00E+11	Ripple	0.56	-3.7	-14.79	1.93E-05	0.241754	5	XRP	1E+11	1
7	5	2.29E+08	7736420	dash	1.51E+09	5.79E+09	18900000	Dash	1.22	-3.31	10.64	0.059856	748.935	6	DASH	7736420	0
8	6	4.09E+08	54153908	litecoin	1.51E+09	5.63E+09	84000000	Litecoin	-0.17	0.8	3.68	0.008316	104.046	7	LTC	54153908	1
9	7	1.38E+08	16690974	bitcoin-go	1.51E+09	4.92E+09	21000000	Bitcoin Go	-0.86	-8.65	-11.24	0.023559	294.774	8	BTG	16790974	1
10	8	5.5E+08	15442957	monero	1.51E+09	4.33E+09		Monero	-2	25.65	41.23	0.022418	280.496	9	XMR	15442957	0
11	9	61647500	2.59E+10	cardano	1.51E+09	3.23E+09	4.5E+10	Cardano	-0.28	-5.8	-8.25	9.96E-06	0.124635	10	ADA	3.11E+10	1
12	10	4.02E+08	98125659	ethereum-	1.51E+09	2.87E+09		Ethereum	-0.2	-3.47	-7.7	0.002335	29.2131	11	ETC	98125659	0
13	11	31728500	9E+09	nem	1.51E+09	2.58E+09		NEM	-0.7	-0.39	14.56	2.30E-05	0.287103	12	XEM	9E+09	0
14	12	2.49E+08	5.2E+08	eos	1.51E+09	2.57E+09		EOS	2.26	29.56	69.66	0.000395	4.9381	13	EOS	1E+09	1
15	13	1.13E+08	65000000	neo	1.51E+09	2.45E+09		NEO	1.24	-7.28	0.2	0.003016	37.7369	14	NEO	1E+08	1
16	14	2.58E+08	1.78E+10	stellar	1.51E+09	2.41E+09		Stellar Lun	-2.71	39.48	51.09	1.08E-05	0.134972	15	XLM	1.03E+11	0
17	15	1.26E+08	55817225	monacoin	1.51E+09	1.12E+09		MonaCoin	17.1	109.98	251.44	0.001603	20.0578	16	MONA	55817225	1
18	16	24598100	3185692	bitconnect	1.51E+09	1.11E+09	28000000	BitConnec	0.49	3.7	15.49	0.02776	347.342	17	BCC	8392580	0
19	17	69659800	1.16E+08	lisk	1.51E+09	1.05E+09		Lisk	1.06	-2.52	15.79	0.000723	9.0525	18	LSK	1.16E+08	1
20	18	1.9E+08	2774181	zcash	1.51E+09	9.8E+08		Zcash	-0.99	9	-2.2	0.02824	353.351	19	ZEC	2774181	0
21	19	60038600	1.02E+08	omisego	1.51E+09	9.58E+08		OmiseGO	-0.72	-8.06	3.93	0.00075	9.38659	20	OMG	1.4E+08	1
22	20	1.42E+08	73696328	qtum	1.51E+09	9.15E+08		Qtum	0.36	-9.85	-16.12	0.000992	12.4131	21	QTUM	1E+08	0
23	21	1.08E+09	8.14E+08	tether	1.51E+09	8.15E+08		Tether	0.17	-0.07	-0.06	8.00E-05	1.00128	22	USDT	8.45E+08	0
24	22	46766600	1E+08	waves	1.51E+09	7.06E+08		Waves	0.97	-4.33	22.21	0.000565	7.06319	23	WAVES	1E+08	0
25	23	37334000	98649745	stratis	1.51E+09	6.96E+08		Stratis	1.48	-6.64	16.87	0.000564	7.0544	24	STRAT	98649745	0
26	24	1567790	41252246	populous	1.51E+09	6.54E+08	53252246	Populous	-0.39	16.05	48.46	0.001266	15.8434	25	PPT	53252246	0

Fig 1: The dataset of Cryptocurrency

The dataset which consists of information about cryptocurrencies collected from various sources, including websites and newspapers. It contains key features such as the cryptocurrency id, last updated, market cap used, max supply, name, price, rank, symbol, total supply. This dataset serves as the foundation for building and training the machine learning models used in the project to predict risks accurately.

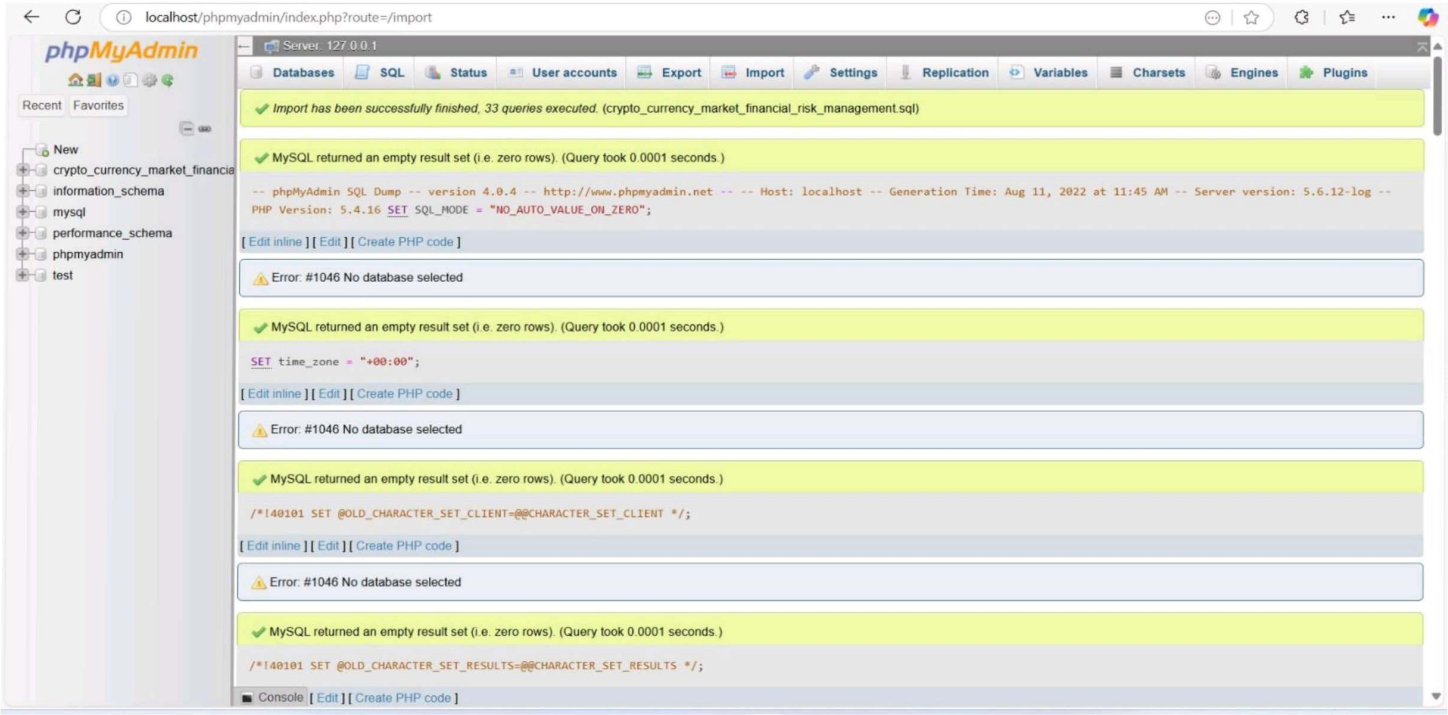

Fig 2: Upload the Database

XAMPP is used to upload the car dataset into phpMyAdmin, a web-based tool for managing MySQL databases. The data is stored and organized within a MySQL database, allowing efficient access and management. This setup enables seamless integration between the database and the machine learning models used for price prediction.

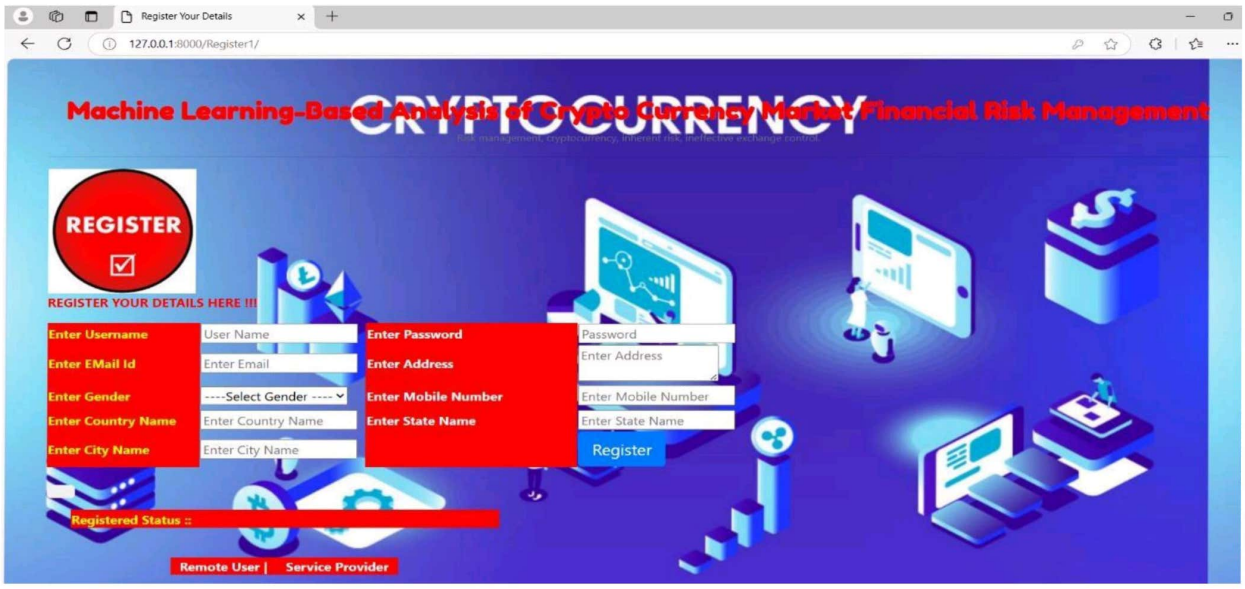

Fig 3: User Registration page

The User Registration Page allows new users to create an account by providing details such as their name, email, and password. This ensures secure access to the system, enabling users to log in and utilize features like entering car details and predicting prices. All user information is securely stored in the database for authentication and profile management.

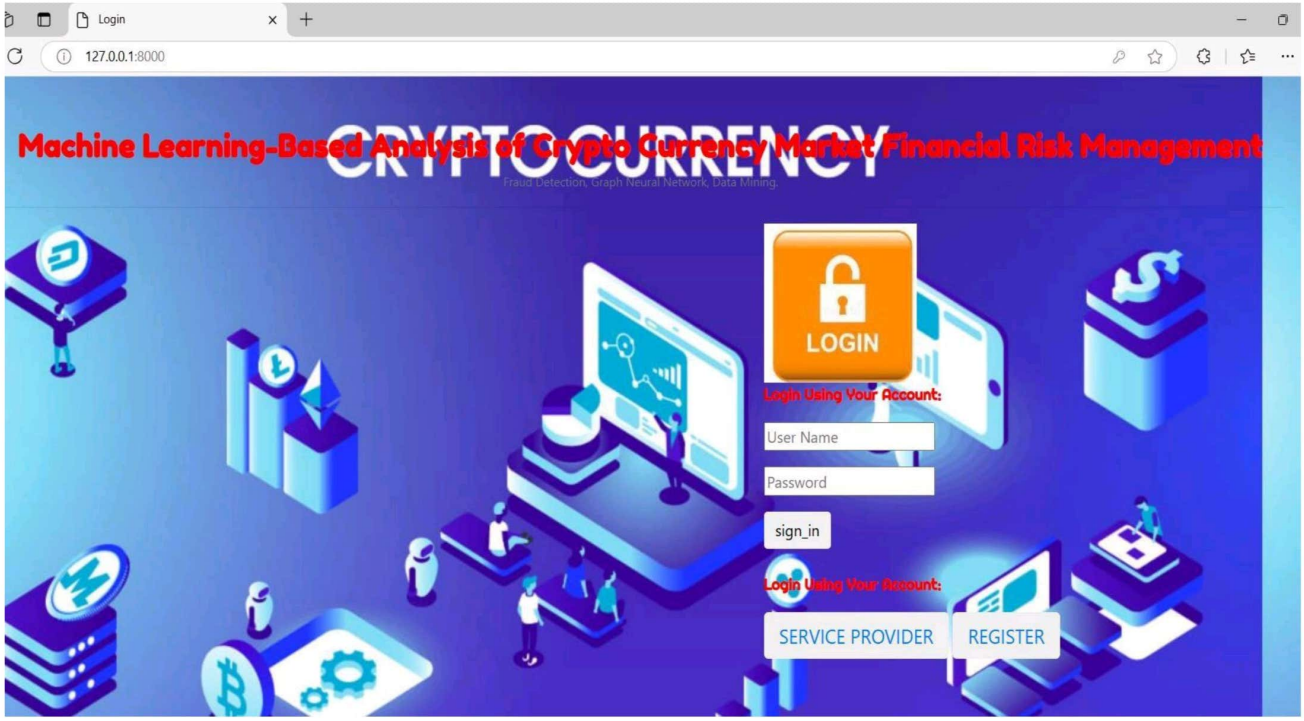

Fig 4: User Login Page

The user logins into the profile using his credentials. The service provider has to choose the service provider option before login to access the admin profile.

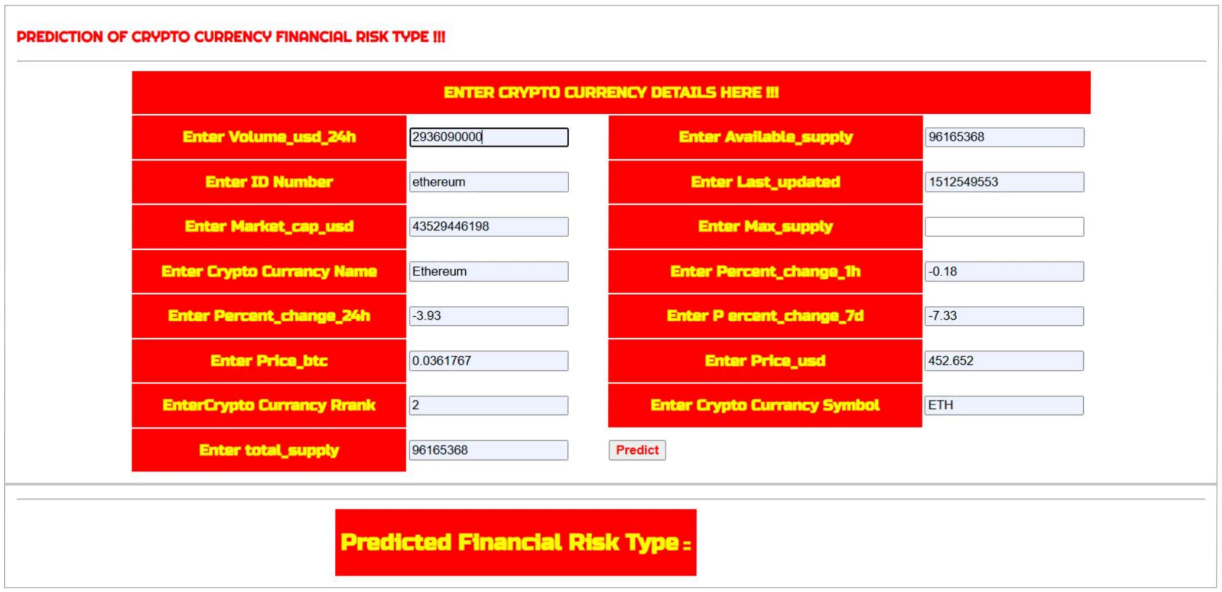

Fig 5: Details of Cryptocurrency

To predict the risk of a cryptocurrencies users must provide key details about the one cryptocurrency, including year, id number, market cap used, available supply, max supply, cryptocurrency name, price, rank, total supply. These features are essential inputs for the machine learning model to generate an accurate risk prediction tailored to the cryptocurrency specifications.

Fig 6: Predicted risk of cryptocurrency

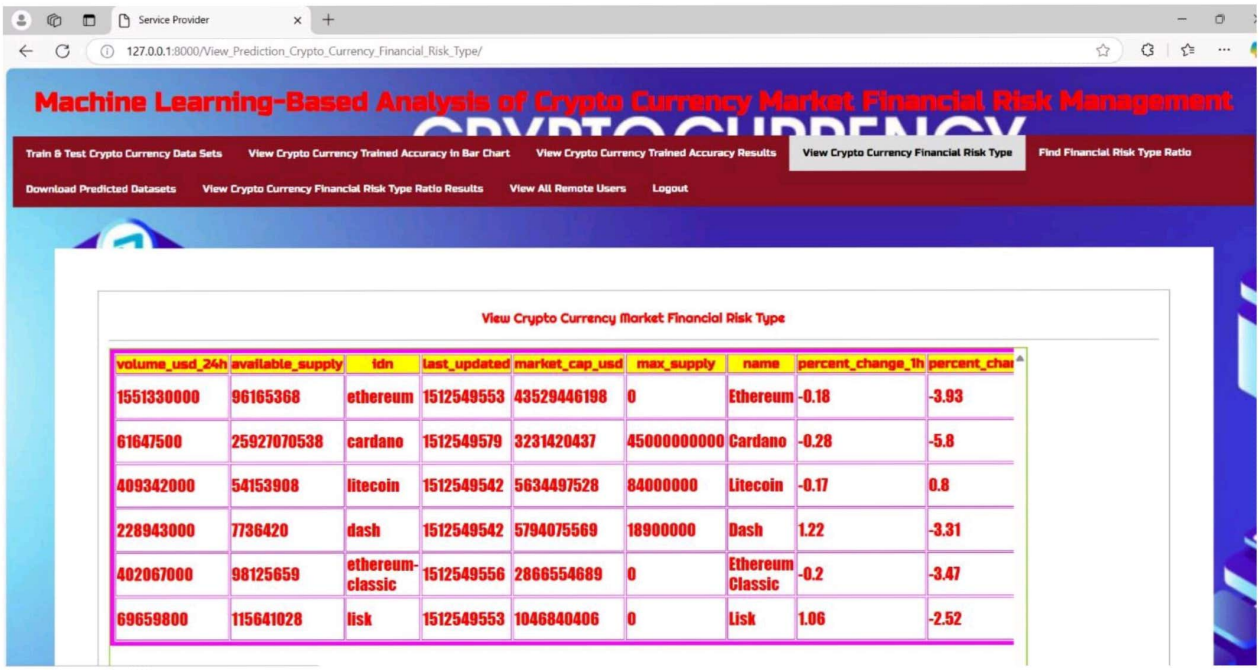

Fig 7: View Cryptocurrency Financial Risk Type

The service provider or Admin can view all the history of the cars the users have predicted using the model. They can view the details about the cars users have entered and also view the predicted price of the model.

APPENDIX II – SAMPLE CODE

```
# Importing necessary libraries
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, confusion_matrix

# Load the dataset (Replace 'your_dataset.csv' with the actual dataset file)
# Ensure the dataset has features and a target column
data = pd.read_csv('your_dataset.csv')

# Display first 5 rows
print("Dataset Sample:")
print(data.head())

# ========== DATA PREPROCESSING ==========
# Handling missing values (if any)
data = data.dropna()

# Splitting data into features (X) and target (y)
```

```
X = data.drop('target', axis=1)  # Replace 'target' with the name of the label column
y = data['target']

# Data Transformation - Standardizing the features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Splitting data into Train and Test sets (80% Train, 20% Test)
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)

# ========================== ALGORITHMS ==========================
# Initializing the classification models
models = {
    "Naive Bayes": GaussianNB(),
    "SVM": SVC(),
    "Logistic Regression": LogisticRegression(),
    "Decision Tree": DecisionTreeClassifier()
}

# Dictionary to store model performance
performance = {}

# ========================== TRAINING, TESTING, AND EVALUATION ==========================
for model_name, model in models.items():
    # Training the model
    model.fit(X_train, y_train)
```

```
# Predicting on the test set
y_pred = model.predict(X_test)

# Calculating performance metrics
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred, average='weighted')
recall = recall_score(y_test, y_pred, average='weighted')
f1 = f1_score(y_test, y_pred, average='weighted')

# Storing results
performance[model_name] = {
    "Accuracy": accuracy,
    "Precision": precision,
    "Recall": recall,
    "F1-Score": f1
}

# Printing model results
print(f"\n{model_name} Performance:")
print(f"Accuracy: {accuracy:.4f}")
print(f"Precision: {precision:.4f}")
print(f"Recall: {recall:.4f}")
print(f"F1-Score: {f1:.4f}")
print("Confusion Matrix:")
print(confusion_matrix(y_test, y_pred))
```

```
# ============================ PERFORMANCE SUMMARY ============================
print("\nFinal Performance Summary:")
for model_name, metrics in performance.items():
    print(f"\n{model_name}:")
    for metric, value in metrics.items():
        print(f"{metric}: {value:.4f}")
```