Projects.

cryptocurrency price prediction using Machine learning

Total No of sections are 5.

1) Home
2) About
3) Service
4) News
5) Contact.

Introduction

-> Crypto-currency is a digital currency, enabled by the blockchain technology and allows for peer-to-peer transaction secured by cryptography.

-> We have utilized the variety of machine learning method and consider a comprehensive set of potential market predictive features.

-> The short term predictability of the bitcoin market has not yet been analysed comprehensively.

Machine learning:- Subfield of AI defined as the capability of a machine to imitate (copy). Intelligent human behaviour.

AI:- Make it possible Machine Learning to learn from experience, adjust to new inputs and perform human like-task.

Requirement

Hardware :-

Processor : Pentium IV
System : Intel i3/i5, 2.4 GHz.
RAM : 4 to 8 GB
Harddisk : 40 GB.

Software :-

Python 3.10+ + Ver (Training to ML)
Python (Back-end)
Django (Frontend)
Dataset → Cryptocompare API
OS :- Window7 and above, MacOS, Linux.

System ~~dg~~ diagram/ flow of project.

Bitcoin dataset
↓
visualize the dataset
↓
RNN
↓
LSTM Regression
↓
Linear regression
↓
Extrapolation of result.

Network framework

ICON (International circle of online Neuronians)

→ ICON is blockchain technology and network framework designed to independent blockchains to interact with each other.

→ Communities are connected to ICON network through the decentralized exchange.

Tools used :- Visual studio code.
Model used :- Bitcoin, Ethereum, XRP, +2 more. · dataset-model

Library used :-

(CPP)

1) pandas :- Used ~~to~~ in cryptocurrency price prediction with machine learning for data preprocessing and analysis of historical price data.

2) Numpy :- Used for efficient numerical computation and data manipulation which can be helpful for preprocessing and analysing data used in machine learning models for cryptocurrency price prediction.

3) Scikit-learn :- Can be utilized in CPP by providing a range of ML algorithm for feature extraction, Model training and evaluation.

4) joblib :- Used to efficiently serialize and deserialize ML model in ~~a~~ CPP pipelines.

5) Urllib :- Used to fetch crypto price data from online sources for training ML models in price prediction.

Framework.

Django :- Python web framework used to build rich and secure ~~for~~ web application and it can be employed in cryptocurrency price prediction project for creating user interface to showcase ML-based prediction.

Existing project :- Best for either long term or short term prediction, no proper implementⁿ, low redundancy to perform the predⁿ, long process for filter the data.

Our project :- Best for both long term and short term.

Regression :- In ML it produces the continuous o/p based on dataset present or given by user.

linear regression :- A data analysis technique that predicts the value of unknown data by using another related and known data value.

RNN :- predict the value ~~the~~ by using the nearest value.

LSTM :- predict the value of a variable ~~for~~ by using old data/value (ex 3 yr - - - 5 yrs " ___ etc).

XGBoosts :- Increase training time of dataset.

Advantage

→ Model can be used to predict the crypto future. performance matrix like accuracy, recall and precision can be calculated.

→ Crypto future may be predicted and investment can be made ~~easily~~ wisely.

→ Main focus is on prediction ~~and~~ of currency for intraday trader.

→ It use coinMarkup cap to make prediction.

→ Quandi is used to filter dataset by using Matrix laboratory (MAT lab) properties.

Disadvantage

CRUD operation

REST :- Representational state transfer

Brief overview of Employee crud project in spring boot

Controller layer :-

→ Receive HTTP req. from clients.
→ Routes req. to appropriate method in controller class.

Service layer :-

→ Contains business logic.
→ Receive request from controller.
→ Interact with repository layer to perform CRUD operation on the database.

Repository layer :-

→ Communicate with the database.
→ Perform CRUD operation using Spring data JPA.
→ Translate database operations into Java method call.

Entity layer :-

→ Contains POJOs (plain old Java Objects) that represents database table.

→ Annotated with JPA annotations to map them to database table.

Application properties :-

→ Configure db connection details like URL, username and password.
→ Specifies hibernate properties such as dialect and ddl auto.
→ Sets server port for the Spring Boot application.

Application layer :-

→ Used to initiate / start the springboot application.

→ System generated.

Flow :-

- client sends HTTP request to controller.
- Controller routes request to appropriate method in the service layer.
- Service layer performs business logic and interact with the repository layer.
- Repository layer communicates with the database to perform CRUD operations.
- Data is retrieved or manipulated and returned back through the layers to the client.

Note :-

- I have used the postman to perform/test the data because there is no frontend in this project.

Tech used :-

Java, spring boot, spring data JPA, Hibernate, HTTP (Get, post, put, delete), Tomcat server.

spring data JPA :- simplify db interactn by providing a higher level abstraction over JPA, enabling easier implementn of CRUD operation and dynamic query generation.

- Hibernate query lang. is used internally to create/manipulate the db table's data.

Tools :- Spring Tool suite. (Eclipse 4.17 for macOS)

Db :- MySql (8.0.27)

Intellij :- 2023.3.5

Eclipse :- Eclipse 4.17

Springboot :- 2.6.4

Hibernate :- 5.6.3

JPA :- 2.2

Dependency

- MySql driver
- Spring data JPA
- ~~Spring tool suite~~
- spring dev. tool
- spring web. (to build web app with spring boot)

Sending email using spring boot application

Model :- Holds email related data structure.

Configuration layer :- ~~Setup~~ Sets up email configuration using JavaMailSender.

Service layer :- Handle email sending logic.

Application properties :- Configures email settings like SMTP host and credentials.

Flow :- :-

-> Client (not specified in this project but could be another applⁿ and user interface) interacts with application.

-> Applⁿ uses the configured JavaMailSender to construct and send email messages.

-> Service layer ~~to~~ receives requests to send emails and uses JavaMailSender to send them.

-> JavaMailSender communicates with the SMTP server (in this case, Gmail's SMTP server) to send the email.

-> Email is delivered to the ~~taee~~ recipients specified in the email message.

=> Model and entity are not same, they serve different purpose in ~~vaery~~ various contexts such as software architecture or natural network design.

Dependencies :-

1) Spring data JPA :- Simplify db interaction in spring applⁿ.
2) Spring web :- facilitate web applⁿ by providing HTTP req. handling and MVC architecture.
3) Spring dev Tools :- enables automatic applⁿ restart during development for faster iteration.
4) MySQL driver :- enable Java applⁿ to interact with MySQL database.
5) Spring Boot Starter Mail :- Dependency for Spring Boot auto-configuration of email related components, including JavaMailSender.

6) JavaMail API :- core api for sending email message programmatically in Java appl^n.

7) Spring boot starter Test :- Dependency for testing the spring Boot applications, ensuring sending functionality is robust.

8) Lombok :- Used for boilerplate code in model classes, improving code readability.
(→ It is optional).

How to get/enable SMTP server

1) Go to your Google account setting.
2) Navigate to the Security section.
3) Enable or "Less secure app access".
4) Generate an app specific password for your spring boot application.
5) Use this email address and generated password in your Spring Boot application's JavaMailSender configuration.

Bitcoin Mining Application (in progress.

Tech used :-

- Spring boot
- Hibernate
- JSP + JPA
- Spring Security
- Spring validator
- Restful API
- Java (core + Advance)
- Maven (plugin is also used)
- Bitcoin API (bitcoinJ-core)
- google guava
- Junit Testing.
- Slf4J-simple
- Thymeleaf
- MySQL db.
- Spring MVC
- Spring web
- Lombok
- IntelliJ IDEA + SpringToolSuite.
- HTML
- CSS
- Javascript.
- Annotations
- Dependencies (MySQL driver, spring data JPA, spring dev tools, spring ~~test~~ Junit, spring validator, spring web)

Layers used (impl^n.)

Controller layer :- Handle HTTP requests and deligates them to appropriate services for processing.

- MiningController :- Controller for managing mining related operations.
- WalletController :- Controller for managing wallet-related operations.
- UserController :- Controller for user ~~and~~ reg. & login.
- RegistrationController :- Separate controller for user Registr^n.
- LoginController :- Separate controller for user login.
- TransactionController :- Controller for managing transaction related operations.
- HomeController :- Controller for home page.

Service Layer :- Business logic is written here.

- MiningService :- Interface for mining-related operations
- WalletService :- ---------- wallet- ----------
- UserService :- ---------- User ----------
- TransactionService :- ---------- transaction ----------
- ConfigurationService :- ---------- Configuration ----------
- LogginService :- ---------- Login ----------
- TransactionServiceImp :- Implementation of TransactionService Interface.
- UserServiceImp :- Implementation of UserService interface.

Note

flow of project

-) The project receive HTTP req thr controller which interact with service to perform business logic. Service utilize repository to access data from db. The results are then returned back through controllers to the client.

Model layer :- (Defines data models used by the application to represent entities such as blocks, wallets, users and transactions.)

- → Block.java :- Model class representing a block in blockchain
- → Wallet.java :- Model class representing wallet.
- → User.java :- Model class representing a transaction.
- → Transaction.java :- Model class representing the Transaction.

Repository layer :- (provide ~~access~~ interface for data access operations, allowing to interact with db.)

- → UserRepository.java :- Interface for managing user data in database.
- → TransactionRepository.java :- Interface for managing transaction data in database.

Configuration layer :- (Manages application-level configurations, such as security settings.)

- → SecurityConfig.java :- Manage appln level configuration. (In this project "Security setting").

Resource layer :- (contains static resources and Thymeleaf templates for front-end view.)

static :- Directory for static resource like css, JS etc.

Templates :- Directory for Thymeleaf templates.

- ✓ registration.html :- Registration page template
- ✓ login.html :- Login page template.
- ✓ home.html :- Home page template.

Test layer :-

→ Contains test classes for testing service like MiningService, WalletService, UserService, TransactionService, ConfigurationService and LogginService.

Dependencies ~~us~~ Imported

1) Spring Boot Starter Web:- Provides web development support, including embedded Tomcat server & spring MVC

2) Spring Boot Starter Data JPA:- simplifies implementation of JPA-based data access layers.

3) Spring Boot Starter Security :- provides security features like authentication and authorization.

4) Spring Boot Starter Thymeleaf :- Integrate Thymeleaf as the templating engine for web applications.

5) Spring Boot Starter Test :- Adds testing support for the application including JUnit and Mockito.

6) Spring Boot Dev Tools :- provides additional development-time tools for hot reloading etc.

7) MySQL Connector Java :- JDBC driver for connecting to MySQL database.

8) Hibernate core :- ORM framework for mapping Java objects to relational database tables.

9) Spring Boot Starter Validation:- Adds support for data validation using Hibernate Validator.

10) Spring Boot Starter Actuator :- Adds production-ready features like health check, metrics of API. etc.

11) bitcoinj-core :- used for Bitcoin protocol implementation and provides tools for working with Bitcoin transactions and wallets in Java applications.

12) google guava :- used in this project, likely for its collection utility, caching and other ~~Java~~ common Java utilities provided by guava library.

13) slf4j-simple :- used for simple logging facade binding with the SLF4J API, providing logging functionality for the application.

✓ QR code generator using spring Boot Application (API) ✓

✓ Prabhat Kr. personal website ✓

✓ QR code generator in bytecode using spring boot application. ✓

✓ BookingApp (fligh reservation) (ongoing)
client :- Manjesh Gowda

✓ Bigbasket E-commerse project (spring boot appln) (ongoing).
client :- Anisha Reddy.

✓ Blog Application using spring boot and spring security. ✓

✓ Hotel Management (HTML, css, Bootstrap)

✓ Agriculture crop Management system (HTML, css, JS, DBMS,).

✓ Multiple file upload API using spring boot

✓ Weather-Application using Angular. ✓

✓ Temperature converter. App using Angular.

✓ Body-Mass-Index (BMI) calculator using Java, Android studio (Android project)
"(7 more android project is done).

✓ ~~Aero~~ Aeroplain crash using CG. (C++)

CamScanner

✓ ATM - simulator using Core Java.

✓ Blog-application (simple spring boot).

✓ Employee data Management using spring boot

✓ E-commerce-project 1/2. (spring boot + Angular) (3-Members)

✓ flight Reservation (full stack - using spring boot + Angular) (3 members)

✓ Ebook project (Java).

✓ Farm management system using HTML, CSS, JS

✓ Online ~~Cour~~ Admission Counselling (Java).
(Ongoing)
client :-) Rahul J

✓ Real-Estate project (ongoing)
client :- Aman Raj.