ECE 568 PROJECT PROPOSAL

ROBOSTOCK

RoboStock is a web application that leverages machine learning techniques to predict stock prices. It aims to provide users with accurate forecasts for specific stocks. The web application consists of the following services:

- **Data Retrieval Service**: This service fetches historical stock data for the specified stock symbol and date range from reliable sources, such as Yahoo Finance or Alpha Vantage.
- **Machine Learning Service**: This service employs a machine learning model, such as LSTM or RNN, trained on historical stock data to predict future prices.
- **Prediction Visualization Service**: This service displays predicted stock prices plotted alongside actual historical data using interactive plots and graphs.

STOCK PREDICTION WEB APPLICATION

- **Data Source**: The data source for this service is the historical stock data fetched by the Data Retrieval Service. The data is automatically updated every 24 hours or whenever the user requests a new prediction.
- **Target Data Set**: The target data set for this service is the daily closing prices of the specified stock. The service uses this data to train and evaluate the machine learning model and generate predictions.
- **Data Summary**: The data summary for this service is as follows:
    - The data consists of daily closing prices of the specified stock for the specified date range.
    - The data is numeric and continuous.
    - The data has temporal patterns and dependencies, meaning that the stock prices are influenced by the previous prices and the time of the year.
    - The data may have noise, outliers, or missing values, which need to be handled by data preprocessing techniques.
- **Service Description**: The service description for this service is as follows:
    - The service uses a recurrent neural network (RNN) with long short-term memory (LSTM) cells to learn the temporal patterns and dependencies in the stock data.
    - The service trains the model using the training subset of the data and evaluates the model performance using the validation subset of the data.
    - The service tunes the model hyperparameters, such as learning rate, number of epochs, batch size, etc., using cross-validation and regularization techniques.
    - The service generates predictions for the test subset of the data and compares them with the actual stock prices using error metrics, such as mean absolute error (MAE), root mean squared error (RMSE), or mean absolute percentage error (MAPE).

EXECUTIVE SUMMARY

A stock prediction web application service is worth undertaking because it can provide users with valuable insights into the future performance of stocks. By using artificial intelligence techniques, such as machine learning and natural language processing, the service can analyze large amounts of financial data and generate accurate and reliable forecasts. This can help users make informed investment decisions, optimize their portfolios, and gain a competitive edge in the dynamic stock market.

PROJECT BACKGROUND

The service of stock prediction web application is motivated by the following issues or problems:

- Stock market is a complex and dynamic system that involves many factors, such as economy, news, sentiment, etc. It is challenging to analyze and predict the future performance of stocks using traditional methods or human intuition.
- Investors and traders need reliable and timely information to make informed decisions and optimize their portfolios. However, they may not have enough time, experience, or resources to conduct extensive research and analysis on their own.
- Existing stock prediction tools may not be user-friendly, accurate, or comprehensive enough to meet the diverse needs and preferences of different users.

The service of stock prediction web application is desirable because it offers the following features:

- It uses advanced artificial intelligence techniques, such as machine learning to analyze large amounts of financial data and generate accurate and reliable forecasts.
- It provides users with a web-based interface that is easy to use and interactive. Users can input a stock symbol and a date range and view the predicted stock prices plotted alongside the historical data.
- It allows users to track their investment portfolios, implement real-time predictions, and add user authentication features.

PROPOSED SOLUTION

My hypothesis is that machine learning techniques can be used to predict stock prices based on historical data and sentiment analysis. I created a machine learning model that uses a recurrent neural network (RNN) with long short-term memory (LSTM) cells to learn the temporal patterns and dependencies in the stock data. I expect that my model can generate accurate and reliable predictions that can help users make informed investment decisions.

PROJECT DELIVERABLES

Following is a complete list of all project deliverables:

Deliverable	Description
Data Collection and Preprocessing	Describes the data sources, data quality, data cleaning, and data transformation processes. It also provides descriptive statistics and visualizations of the data.
Machine Learning Model Development	Explains the machine learning model selection, training, evaluation, and optimization processes. It also provides performance metrics and validation results of the model.
Web Application Prototype	A functional prototype of the web application that demonstrates the user interface, user input, data retrieval, machine learning integration, and prediction visualization features.
Web Application Testing	Summarizes the web application testing processes, methods, tools, and results. It also identifies and resolves any bugs, errors, or issues in the web application.
Project Closure	The completion of the project and evaluates the project performance, outcomes, and benefits. It also provides feedback, lessons learned, and recommendations for future improvements.

Timeline for Execution

Summarize the timeline of project-related events from start to finish.

Key project dates are outlined below. Dates are best-guess estimates and are subject to change until a contract is executed.

In the table that follows, include all important dates related to the project, broken down by date and duration. The descriptions shown are for illustration purposes only: replace them with meaningful descriptions related to your project. Items can include such things as payment and project milestones, installation schedules, meetings, or reviews.

Description	Start Date	End Date	Duration
Project Start	3/10/2024	3/17/2024	1 week
Milestone 1 (Data Collection)	3/10/2024	3/12/2024	2 days
Milestone 2 (Data Pre-Processing)	3/13/2024	3/17/2024	5 days
Phase 1 Complete	3/17/2024	4/1/2024	2 weeks
Milestone 3 (Model Selection)	3/17/2024	3/24/2024	1 week
Milestone 4 (Model Validation)	3/25/2024	4/1/2024	1 week
Phase 2 Complete	4/1/2024	5/1/2024	1 month
Milestone 5 (Frontend)	4/1/2024	4/7/2024	1 week
Milestone 6 (Backend/ Database)	4/8/2024	4/15/2024	1 week
Milestone 7 (Website Prototype)	4/16/2024	4/23/2024	1 week
Milestone 8 (Website Testing)	4/24/2024	5/1/2024	1 week

REQUIREMENTS

Required Libraries: I will use the following Python libraries for my web application:

- **yfinance**: A library for fetching stock data from Yahoo Finance.
- **TensorFlow**: A library for developing and training machine learning models.
- **Plotly**: A library for creating interactive plots and graphs.

Data Requirements: I will use the following data sources and parameters for my web application:

- **Stock Data**: I will use the historical stock data from Yahoo Finance for the specified stock symbol and date range. I will fetch the data using the yfinance library and store it in a pandas dataframe. I will use the daily closing prices as the target variable for prediction.
- **Polling Cycle**: I will update the data and predictions every 24 hours or whenever the user requests a new prediction.

SUCCESS METRICS & CONCLUSION

- I use the training data to train and tune my machine learning model. I split the training data into two subsets: one for training and one for validation. I use the training subset to fit the model parameters and the validation subset to evaluate the model performance and adjust the hyperparameters, such as learning rate, number of epochs, batch size, etc. I use cross-validation techniques, such as k-fold or leave-one-out, to reduce the variance and bias of the model. I also use regularization techniques, such as dropout or weight decay, to prevent overfitting and improve generalization.

- I use the test data to measure the final performance and accuracy of my model. I do not use the test data for training or tuning the model. I compare the predicted stock prices with the actual stock prices in the test data and calculate the error metrics, such as mean absolute error (MAE), root mean squared error (RMSE), or mean absolute percentage error (MAPE). I also plot the predictions and the actual data to visualize the results and identify any trends or patterns.

- I use the following metrics to measure the service performance and successfulness:
  - **Accuracy**: The degree to which the predicted stock prices match the actual stock prices. I use error metrics, such as MAE, RMSE, or MAPE, to quantify the accuracy of the predictions. The lower the error, the higher the accuracy.
  - **Reliability**: The consistency and stability of the predictions over time and across different stocks. I use statistical tests, such as t-test or ANOVA, to compare the predictions with the historical data and check for any significant differences or outliers. I also use confidence intervals or error bars to indicate the uncertainty or variability of the predictions.