**A PRELIMENERY REPORT ON**

**CENTRALIZED DATA PLATFORM**

SUBMITTED TO THE VISHWAKARMA INSTITUTE OF INFORMATION TECHNOLOGY, PUNE

IN THE PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE AWARD OF THE DEGREE

OF

**BACHELOR OF TECHNOLOGY (COMPUTER ENGINEERING)**

##### SUBMITTED BY

**RUSHIKESH MANTRI Exam Seat No. : 22010703**

**MITALI KATE Exam Seat No. : 22010295**

**DIMPAL BHORKAR Exam Seat No. : 22010435**

**GAURAV PATIL**  **Exam Seat No. : 22120265**



## DEPARTMENT OF COMPUTER ENGINEERING

**BRACT’S**

**VISHWAKARMA INSTITUTE OF INFORMATION TECHNOLOGY**

SURVEY NO. 3/4, KONDHWA (BUDRUK), PUNE – 411048, MAHARASHTRA (INDIA).

**BRACT’S**

**VISHWAKARMA INSTITUTE OF INFORMATION TECHNOLOGY, PUNE**

## 2023 2024

Abstract

The Commodity Price Prediction System is a web application designed to forecast commodity prices using Long Short-Term Memory (LSTM) neural networks. It serves traders, investors, and analysts by providing accurate predictions based on historical price data, real-time news, and currency conversions.

Built with Python, the system utilizes libraries such as Pandas, NumPy, PyTorch, Plotly, and Altair. It fetches historical price data from the Alpha Vantage API, provides real-time news via the NewsData.io API, and converts prices to INR using the ExchangeRate-API. The application interface, developed with Streamlit, allows users to upload CSV files for custom visualizations, access a news dashboard, and convert commodity prices.

The system captures complex patterns in time series data through LSTM models, delivering reliable forecasts and insights through interactive visualizations. By integrating machine learning, real-time data, and user-friendly web technologies, it offers a comprehensive tool for commodity market analysis and decision-making.

**APIs Used** in the Project :

Overview of the APIs Used in the Project : The Commodity Price Prediction System leverages several powerful APIs to gather data, provide realtime news updates, and perform currency conversions. Each API has been carefully selected for its reliability, comprehensive data offerings, and ease of integration.

1. Alpha Vantage API :   
   Alpha Vantage API is employed to fetch historical price data for various commodities. This API offers a wide range of financial data, making it an ideal choice for accessing daily time series data required for predicting commodity prices. The specific endpoint used is `https://www.alphavantage.co/query`, which, combined with parameters like `function` (e.g., `TIME\_SERIES\_DAILY`), `symbol` (representing the commodity, such as `GC=F` for Gold), and the user's `apikey`, provides a seamless method to retrieve detailed historical price data.   
   Alpha Vantage is renowned for its reliability and the breadth of data it covers, including stock prices, forex rates, and cryptocurrencies, making it a versatile tool for financial analysis.
2. NewsData.io API :  
   For news updates, NewsData.io API is integrated into the system. This API is particularly useful for fetching the latest news articles related to commodities, providing realtime insights into market trends and events that may impact commodity prices. The endpoint `https://newsdata.io/api/1/news` is utilized with parameters such as `apikey`, `q` (query string for searching news articles, e.g., `Commodities`), and `language` (e.g., `en` for English).   
   The NewsData.io API supports a wide range of news sources and languages, ensuring comprehensive coverage of relevant news. Its simplicity and generous free tier make it an excellent choice for integrating news features into applications.
3. ExchangeRateAPI :   
   The ExchangeRateAPI is used for currency conversion, specifically to convert commodity prices into INR. This API offers accurate and uptodate exchange rates, essential for financial calculations and conversions. The endpoint `https://v6.exchangerateapi.com/v6/{api\_key}/pair/{base\_currency}/{target\_currency}` is used with parameters like `api\_key`, `base\_currency` (e.g., `USD`), and `target\_currency` (e.g., `INR`).   
   The ExchangeRateAPI supports a wide range of currency pairs, providing reliable data for currency conversion tasks. Its ease of use and robust performance make it a suitable choice for financial applications.

Alternative APIs which can also be used : While the current APIs are wellsuited for the system, several alternative APIs could be considered for additional features or expanded data needs.

* Quandl API :   
  Quandl API is a powerful alternative for fetching historical and realtime financial and economic data. It offers a vast array of datasets, including commodity prices, making it an excellent resource for financial research. The endpoint `https://www.quandl.com/api/v3/datasets/{database\_code}/{dataset\_code}.json` allows users to access specific datasets by providing the `database\_code` and `dataset\_code`, along with an `api\_key`.   
  Quandl is known for its reliability and the breadth of its data offerings, making it a valuable tool for comprehensive financial analysis.
* Yahoo Finance API :   
  Another alternative is the Yahoo Finance API, which provides extensive financial data, including stock prices, commodity prices, and market news. The endpoint `https://query1.finance.yahoo.com/v7/finance/quote?symbols={symbol}` is used to fetch realtime data for specific symbols.   
  Yahoo Finance is widely used for its comprehensive and uptodate financial data, offering an accessible and reliable source for various financial information needs.
* Financial Modeling Prep API :   
  Financial Modeling Prep API is also a robust option, offering realtime and historical stock, commodity, and economic data. The endpoint `https://financialmodelingprep.com/api/v3/historicalpricefull/{symbol}` allows users to fetch detailed historical price data for specific commodities or stocks by providing the `symbol` and an `apikey`.   
  This API is easy to use and provides detailed and accurate financial data, making it a good alternative for expanding the system's data sources.

Reasons for Choosing the Current APIs

1. Alpha Vantage API:

Provides reliable and comprehensive historical price data.

Free tier supports a sufficient number of requests for this application.

Easy integration with Python.

2. NewsData.io API:

Offers access to a wide range of news sources and provides relevant news articles.

Simple and effective API with a generous free tier.

3. ExchangeRateAPI:

Provides accurate and uptodate exchange rates.

Supports a wide range of currency pairs.

Reliable service with easy integration into Python applications.

These APIs were chosen for their reliability, ease of use, and comprehensive data offerings, which are essential for the core functionalities of the Commodity Price Prediction System. The suggested alternatives offer similar benefits and could be considered if additional features or data types are needed in the future.

**Reasons for Using LSTM Model for Commodity Price Prediction :**

1. Handling Sequential Data:

Long ShortTerm Memory (LSTM) networks are a type of Recurrent Neural Network (RNN) specifically designed to handle sequential data and capture longterm dependencies. Commodity prices are inherently sequential, meaning their past values significantly influence future values. LSTM models can effectively capture these dependencies, making them highly suitable for timeseries prediction tasks like commodity price forecasting.

2. Addressing the Vanishing Gradient Problem:

Traditional RNNs often struggle with the vanishing gradient problem, where gradients used in the learning process become too small to allow the network to learn effectively from earlier time steps. LSTMs mitigate this issue through their unique architecture, which includes memory cells and gating mechanisms (input, forget, and output gates). These gates regulate the flow of information, ensuring that important signals are retained over longer periods, thus preserving the network's ability to learn longterm dependencies.

3. Robustness in Capturing Patterns:

Commodity prices can exhibit complex, nonlinear patterns due to various influencing factors like market demand, geopolitical events, and economic indicators. LSTM networks are adept at capturing such intricate patterns within the data. Their ability to maintain a memory of previous states allows them to model these nonlinear relationships more accurately than traditional statistical models or simpler machine learning algorithms.

4. Flexibility and Adaptability:

LSTM models are highly flexible and can be adapted to various types of timeseries data without significant modifications to their core structure. This makes them suitable for a wide range of commodities, each with unique price dynamics and influencing factors. The adaptability of LSTMs allows for their application across different commodities like gold, oil, natural gas, and agricultural products, providing a unified framework for prediction.

**Benefits of Using LSTM Model in This Case :**

1. Scalability:

LSTM models can be scaled to handle large datasets, making them suitable for use with extensive historical price data. This scalability ensures that the model can learn from a vast amount of past data, further enhancing its predictive capabilities.

2. Handling Missing and Noisy Data:

Realworld financial data, including commodity prices, often contain missing or noisy values. LSTM networks are robust to such imperfections in the data, as they can learn to focus on the relevant parts of the sequence and mitigate the impact of noise. This makes LSTMs particularly effective in dealing with realworld financial datasets.

3. Ability to Incorporate Additional Features:

In addition to historical prices, LSTM models can incorporate other relevant features such as trading volumes, economic indicators, and geopolitical events. This multidimensional approach enables the model to capture a broader range of influencing factors, improving the robustness and accuracy of its predictions.

4. Realtime Prediction Capabilities:

LSTM networks are wellsuited for realtime prediction scenarios. Once trained, an LSTM model can quickly generate predictions for new data points, enabling realtime monitoring and forecasting of commodity prices. This capability is essential for traders and investors who need timely insights to make quick decisions.

**Overview of Model** : The system is designed to predict commodity prices, display historical commodity data, visualize data from CSV files, fetch the latest commodity news, and convert commodity prices to INR. It consists of multiple modules each performing specific tasks within a unified Streamlit application.

**Methodology :**

1. LSTM Algorithm for Price Prediction (LSTM\_Algo.py)

Data Fetching:

Uses Alpha Vantage API to fetch daily time series data for commodities.

Converts the data to a Pandas Data Frame and prepares it for analysis.

Data Preparation:

Normalizes the data using MinMaxScaler to scale the prices between 1 and 1.

Converts the time series data into sequences for training the LSTM model.

LSTM Model:

Defines an LSTM model using PyTorch with specified input size, hidden layers, and output size.

Trains the model on the prepared sequences to predict future prices.

Prediction:

Uses the trained model to predict future prices.

Scales the predicted prices back to their original range using the inverse transform of the scaler.

Visualization:

Uses Plotly to visualize actual vs predicted prices and displays the Mean Absolute Error (MAE) on the graph.

2. MultiPage App Structure (main.py)

MultiPage Structure:

Uses a class MultiApp to manage multiple pages within the Streamlit app.

Uses streamlit\_option\_menu to create a sidebar menu for navigating between different app pages.

Page Navigation:

Defines pages for Home, LSTM, CSV, News, and Calculator.

Each menu option loads the corresponding module and displays its content.

3. News Fetching and Display (News.py)

News Fetching : Uses NewsData.io API to fetch the latest news articles related to commodities , Handles API response and error scenarios gracefully.

Display : Displays the news articles using Streamlit, including title, description, source, and publication date. Provides a "Read More" button linking to the full article.

4. CSV Data Visualization (Upload\_CSV.py)

Data Loading : Uses Streamlit’s file uploader to upload CSV files , Loads and preprocesses the data using Pandas.

Data Cleaning : Drops rows with missing values to ensure clean data for visualization.

Visualization : Uses Altair to create line charts for visualizing data , Allows users to select columns for the X and Y axes dynamically.

5. Commodity Price Conversion (calculator.py)

Exchange Rate Fetching : Uses ExchangeRate API to fetch the latest exchange rates for converting commodity prices to INR and Caches the exchange rates to minimize API calls and improve efficiency.

Price Conversion : Converts the user input commodity price to INR using the fetched exchange rate. Also Provides user feedback on the converted price using Streamlit.

Implementation Details

# LSTM\_Algo.py

Data Fetching:

def fetch\_commodity\_data(symbol):

# API call to fetch data

Data Preparation and Model Training:

def train\_and\_predict\_lstm(data):

# Data normalization and LSTM model training

# main.py

Page Setup:

class MultiApp:

def add\_app(self, title, func):

# Add app to list

def run():

# Display sidebar menu and load selected app

# News.py

Fetch and Display News:

def fetch\_news(api\_url):

# API call to fetch news data

def display\_news(articles):

# Display articles in Streamlit

# Upload\_CSV.py

Load and Display CSV Data:

def load\_data(file):

# Load CSV data and preprocess

def app():

# Upload file, select columns, and visualize data

# calculator.py

Fetch Exchange Rate and Convert Price:

def fetch\_exchange\_rate(api\_key, base\_currency, target\_currency="INR"):

# API call to fetch exchange rate

def convert\_price(price, exchange\_rate, conversion\_factor=1.0):

# Convert commodity price to INR

def display\_converter(api\_key):

# User interface for price conversion

This methodology integrates various components seamlessly into a unified Streamlit application, providing functionalities ranging from price prediction using LSTM, data visualization, news fetching, and price conversion. Each component is designed to handle specific tasks with error handling, user friendly interfaces for a smooth user experience.

**Technologies Used** in the Commodity Price Prediction System :

The Commodity Price Prediction System leverages a combination of programming languages, libraries, frameworks, and APIs :

1. Programming Languages:

Python: The primary programming language used for developing the system. Python is chosen for its simplicity, readability, and extensive ecosystem of libraries and frameworks for data analysis, machine learning, and web development.

2. Libraries and Frameworks:

* Streamlit: A framework used for building interactive web applications with Python. Streamlit makes it easy to create and deploy data driven applications quickly, and it is highly suitable for this system due to its simplicity and ability to integrate seamlessly with various data processing and machine learning libraries.
* Pandas: A powerful data manipulation and analysis library for Python. Pandas is used to handle and preprocess the historical price data, manage data frames, and perform various data transformations needed before feeding the data into the machine learning model.
* NumPy: A library for numerical computing in Python. NumPy is used for array operations and efficient numerical calculations, which are essential for data preprocessing and model training.
* Scikit learn: A machine learning library in Python. Scikit learn provides tools for data preprocessing, such as scaling, and includes utilities for splitting data into training and testing sets.
* PyTorch: A deep learning framework that provides flexibility and speed. PyTorch is used to build and train the Long Short Term Memory (LSTM) model for predicting commodity prices. It offers a dynamic computational graph and strong GPU acceleration, making it suitable for training complex neural networks.
* Plotly: A graphing library used to create interactive visualizations. Plotly is integrated with Streamlit to display historical price data and prediction results as interactive charts, enhancing the user experience and providing clear insights into the data.
* Altair: Another visualization library used for generating declarative statistical graphics. Altair is utilized to create line charts for CSV data visualization within the system.

3. Miscellaneous Technologies:

Requests: A Python library used to make HTTP requests. It is employed to interact with the afore mentioned APIs, fetching necessary data such as historical prices, news articles, and exchange rates.

Streamlit Option Menu: A library to create a customizable sidebar menu in the Streamlit application. It helps in organizing the different functionalities of the system (Home, LSTM, CSV, News, Calculator) into a user friendly interface.