**project**

TITLE :

Cracking the market code with

AI-driven

Stock price prediction using time series analysis

***SOURCE CODE*** :

Upload the Dataset python Copy Edit

from google.colab import files

uploaded = files.upload()

Load the Dataset

import pandas as pd

import io

# Display uploaded file names

print(uploaded.keys())  # This will help you find the actual filename

# Then load the dataset using the correct name

df = pd.read\_csv(io.BytesIO(uploaded['Data set']))  # No .csv if the filename is "Data set"

df.head()

Data Exploration

df.info()

df.describe(include='all')

Check for Missing Values

df.isnull().sum()

Visualize a Few Features python Copy Edit

import matplotlib.pyplot as plt

import seaborn as sns

# Distribution of opening prices

plt.figure(figsize=(8, 4))

sns.histplot(df['Open'], kde=True)

plt.title("Distribution of Opening Prices")

plt.show()

# Correlation heatmap

plt.figure(figsize=(10, 6))

sns.heatmap(df.select\_dtypes(include='number').corr(), annot=True, cmap="coolwarm")

plt.title("Feature Correlation")

plt.show()

Identify Target and Features

# Predicting 'Last Traded Price'

target = 'Last Traded Price'

features = df.select\_dtypes(include=['number']).drop(columns=[target]).columns.tolist()

X = df[features]

y = df[target]

One Hot Encoding

df\_encoded = pd.get\_dummies(df, columns=categorical, drop\_first=True)

Feature Scaling

target = 'Last Traded Price'

Train Test Split python Copy Edit

y = y.reset\_index(drop=True)

Model Building

from sklearn.ensemble import RandomForestRegressor

model = RandomForestRegressor()

Evaluation

import numpy as np

# Fix percentage/numeric columns: remove commas, %, and replace '-' with NaN

for col in ['Change', 'Percentage Change', '365 Day Percentage Change', '30 Day Percentage Change', 'Last Traded Price']:

  df[col] = df[col].astype(str).str.replace(',', '', regex=False)  # remove commas

    df[col] = df[col].str.replace('%', '', regex=False)              # remove %

    df[col] = df[col].replace('-', np.nan)                           # replace '-' with NaN

    df[col] = df[col].astype(float)                                  # convert to float

Convert to DataFrame and Encode

# Already done in Step 8 using pd.get\_dummies

# df\_encoded is your working DataFrame

Predict the Final Grade (in this context, the price)

[3153.61, 440.90, 635.23, 396.65, 596.84]

Build an Interactive App (Streamlit alternative using Gradio)

!pip install gradio

Create a Prediction Function

def predict\_price(Open, High, Low, Previous\_Close, Volume):

    input\_data = pd.DataFrame([[Open, High, Low, Previous\_Close, Volume]],

    columns=['Open', 'High', 'Low', 'Previous Close', 'Share Volume'])

    input\_scaled = scaler.transform(input\_data)

    prediction = model.predict(input\_scaled)

    return f"Predicted Last Traded Price: ₹{prediction[0]:.2f}"

Create a Gradio Interface

import gradio as gr

interface = gr.Interface(

    fn=predict\_price,

    inputs=[

gr.Number(label="Open"),

        gr.Number(label="High"),

        gr.Number(label="Low"),

        gr.Number(label="Previous Close"),

        gr.Number(label="Share Volume")

    ],

    outputs="text",

    title="Stock Price Prediction"

)

interface.launch()