**GOLD PRICE PREDICTION USING MACHINE LEARNING**

1. Problem Formulation
2. Data preprocessing
3. Data wrangling
4. Model Development
5. Model Explainability
6. Model Deployment

**Overview:**

**Importing Libraries**

import numpy as np

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

**Loading the Dataset**

**Data preprocessing  – Missing Values/Null Values**

Missing values have a very drastic effect on our model training. some of the models like LinearRegression do not fit the dataset which has missing values in it. However, there are some models which work well even with a missing dataset like RandomForest. But it is always a good practice to handle missing values first when working with the dataset. Also, one thing to note is that when we load the data using pandas it automatically detects null values and replaces them with NAN.

**Correlation Between Columns**

**Data Wrangling**

To gain insight and knowledge from the data. We see data through every aspect and try to fetch most of the information from the dataframe.

**Trend in Gold Prices Using Moving Averages**

To visualize the trend in the data we have to apply a smoothing process on this line which looks very noisy. There are several ways to apply to smooth. In our project, we will take an average of 20 previous data points using the pandas rolling functions This is also known as the moving average

**Handling Outliers**

Outliers can have a very bad effect on our model like in linear regression if a data point is an outlier then it can add a very large mean square error. Removing outliers is a good process in EDA. Some models like Decision tree and ensemble methods like Random Forests are not that much by outliers.

**Plotting Boxplot to Visualize the Outliers**

Boxplots are very useful in plotting the spread and skewness of the data, it is also useful in plotting the individual’s outlier data points, they consist of the box which represents points in the range of 25% to 75% quantiles.

**Modeling the Data**

Before We start modeling the data must divide the data into train-test, so that after training the data We can see how much our data is learning the pattern and generalizing on new data points. it  is also a way to see that our model is not learning the noise in the data or say it is not overfitting the dataset

**Scaling the Data**

Before we train the model on our data we should perform scaling on our data to normalize. After scaling the data our mean of each column becomes zero and their standard deviation becomes 1. It is also called z-score normalization since we subtract the mean of the column from each element and divide it by the standard deviation of the column. It brings all the columns to the same scale and directly comparable with one another.

**Lasso Regression**

In this model, we have used linear regression with L1 regularisation, also with help of the make\_pipeline object, we will use lasso regression with 2 degrees. We will also use the gridline  object in every model to get the best-performing hyperlink and lower the variance.

**RandomForestRegressor for Regression**

In the second model, we will use the  method to fit our training data. like in Random Forest it uses several decision trees to fit on the data, one thing to note is that in random forest regressor m number of rows are used for training which is always less than n (m<n). where n is the total number of original columns present in the training dataset, also for row points random forest

**Model Explainability**

In the black box model Boosting and Bagging, we will not be able to see the actual weights given to these columns hoWever there are some libraries that we can use to the fraction of Weight out of 1 given to a particular column when we predict on a single vector. We will be using eli5 package to demonstrate the model explainability. You can install this package by running the following command in the terminal.

**Model Deployment using Pickle**

To deploy the model We will use the pickle library from the Python language. We will deploy our best-performing model which is XGBoost. Pickle is a Python module that is used for serializing and deserializing the model i.e saving and loading the model. It stores Python objects which can be moved to a disk(serializing) and then again from disk to memory(deserialize).