PROJECT TITLE- STOCK MARKET PREDICTION .

KHUSHI PAL (2240401108)

PROJECT MENTOR – DR. C.K VERMA

Date – 9 february 2024

Introduction to Stock market Prediction

Predicting stock prices is a complex challenge, but advancements in data analysis and machine learning have made it more achievable. This presentation will explore the process of prediction stock performance using the powerful Random Forest algorithm.

Libraries Used

1. **NumPy**

   With NumPy, you can perform operations on large multidimensional arrays and matrices efficiently.

2. **Pandas**

   It provides data structures and functions designed to make working with structured (tabular) data fast, easy, and expressive.

3. **Scikit-learn**

   It provides a simple and efficient toolkit for various machine learning tasks, including classification, regression, clustering, dimensionality reduction, and preprocessing.

4. **yfinance**

   It provides a simple interface for accessing a wide range of financial data, including historical stock prices, dividends, splits, and more.

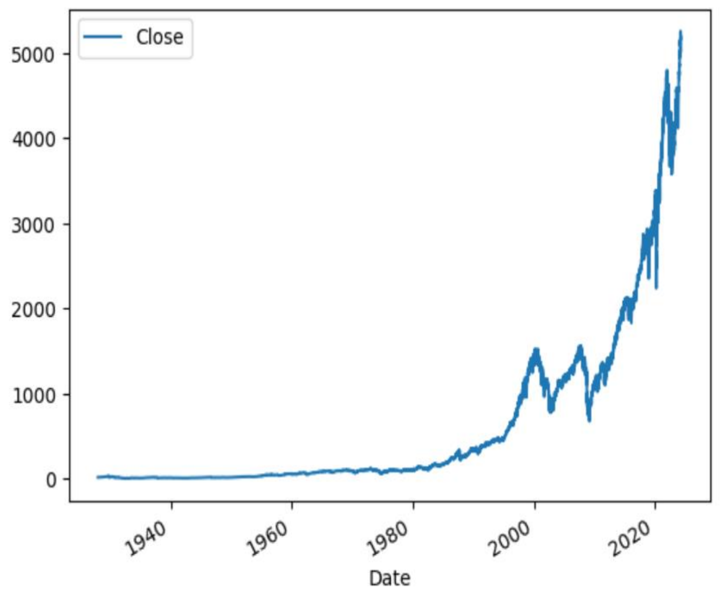
5. **Matplotlib**

   it is a comprehensive library for creating static, animated, and interactive visualizations in Python.

Data Collection and Preprocessing

```
sp500.plot.line(y="Close", use_index=True)
```

```
<Axes: xlabel='Date'>
```



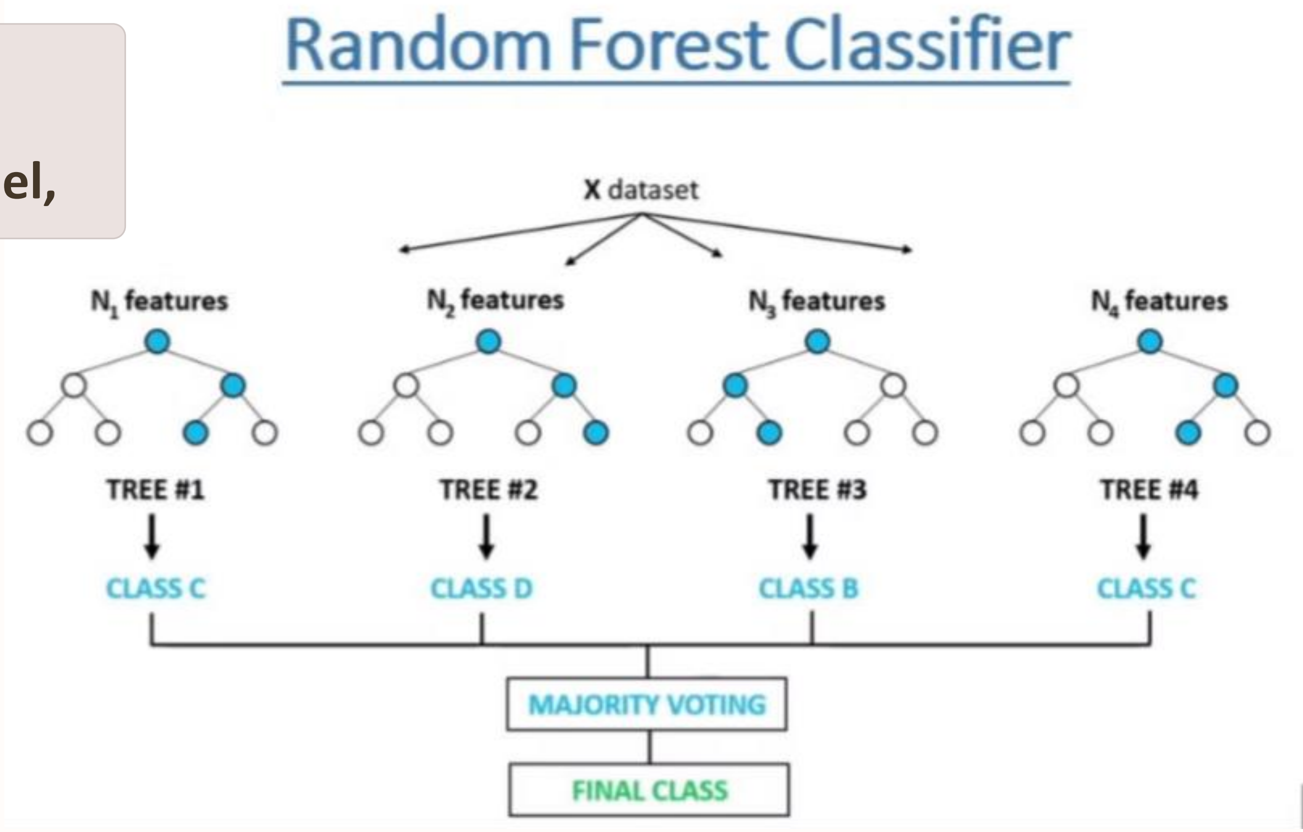
- **Data Sources**
  - Historical S&P 500 data was obtained using the Yahoo Finance API, a widely used and reliable source for financial data.

- **What is S&P 500?**
  - The S&P 500 is a stock market index that measures the stock performance of 500 large companies on stock exchanges in the U.S.
  - One of the best indicators of the U.S. stock market's health and is often used as a benchmark for the overall performance of the U.S. equity market.

Model Selection

Random Forest

Our chosen model,



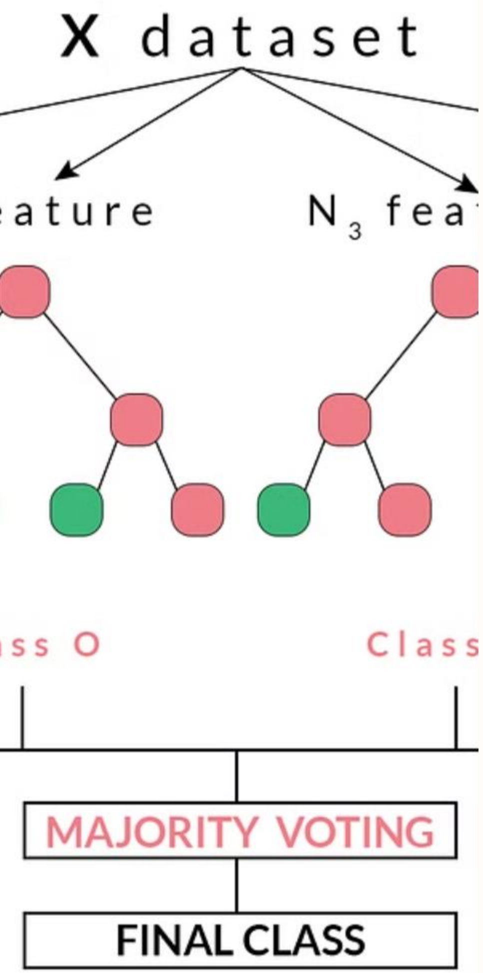
Random Forest

Random Forest is a popular supervised learning algorithm used for both classification and regression tasks. It employs ensemble learning, combining multiple decision trees trained on different subsets of the dataset to improve predictive accuracy. By aggregating predictions from individual trees, Random Forest produces a final output based on majority voting.

Ensemble learning

1. **Bagging (Bootstrap Aggregating)**
2. **Boosting**
3. **Voting**
4. **Stacking**

Random Forest employs ensemble learning, specifically bagging
Bagging is an ensemble technique in which different samples are collected for making a decision. Here in Random Forest, we choose such samples from the population and form decision trees and not one decision tree. Here are there many trees formed from different samples(bootstrap samples)which are all combined to form Random Forest.



Why Random Forest?

1 **Robust to Overfitting**

The ensemble nature of Random Forest reduces the risk of overfitting.

2 **Handles Nonlinearity**

It can capture complex non-linear relationships in the data.

3 **Handling of High-Dimensional Data:**
Stock market data often contains numerous features, such as price, volume, technical indicators, and economic variables. Random Forest can handle high-dimensional datasets effectively, making it suitable for incorporating diverse information sources.

Features of Random Forest

Decision Trees

Random Forest is an ensemble of multiple decision trees.

Bagging

It uses bootstrap aggregating to reduce variance and improve stability.

Random Subspaces

Each tree is trained on a random subset of features for diversity.

Flexibility and Tunability:

Flexible and can be customized through hyperparameter tuning for specific prediction tasks. Parameters such as the no. of trees, maximum depth, and minimum samples per leaf can be adjusted to fine-tune the model.

Model Training and Hyperparameter Tuning

1

Data Split

Divide the dataset into training, validation, and testing sets. Usually 80:20 split is done .

2

Data Cleaning:

It is always important to assess and clean the data as most real life datasets are untidy and messy.

3

Data Filtering

This is done because historical data might not reflect current market trends.

4

Model Training

- Random Forest belongs to a category of algorithms called ensemble methods.
- The core idea is to combine multiple models (often decision trees) to get a more robust and accurate prediction than any single model could achieve on its own.

5

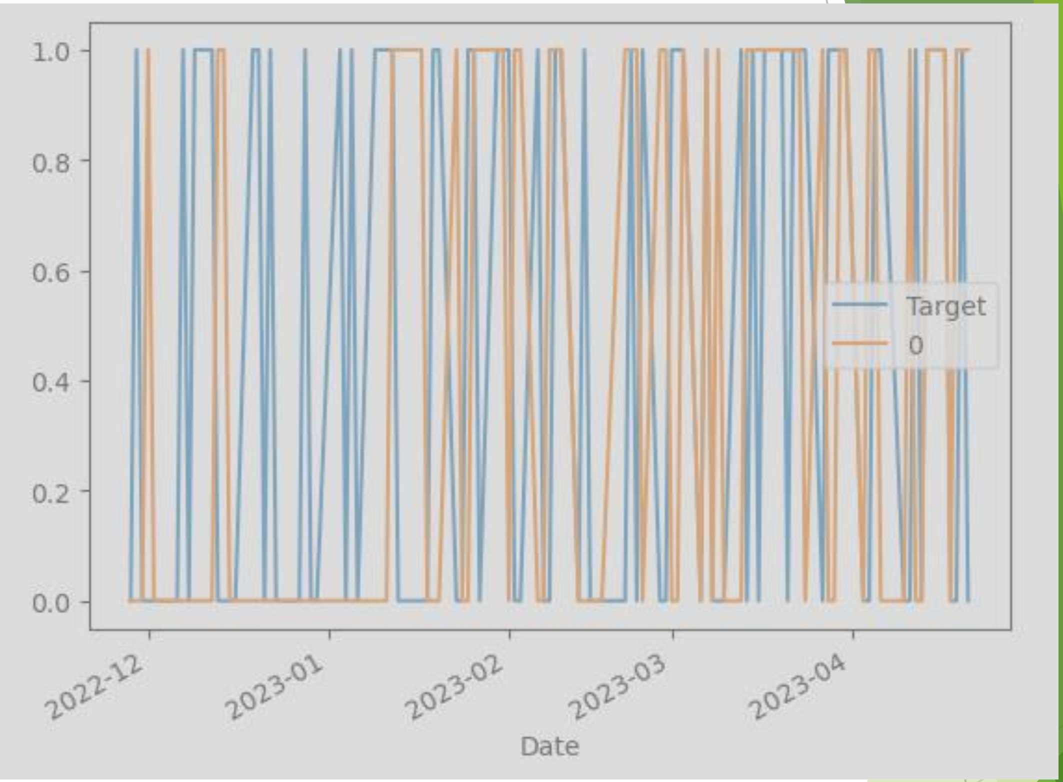
Hyperparameter tuning

These are external configuration settings that define the model's architecture and learning behavior

Why is Hyperparameter Tuning Important?

- **Underfitting:** The model fails to capture the patterns in the data, resulting in poor accuracy.

- **Overfitting:** The model memorizes the training data too well and performs poorly on unseen data.



Model training

```
from sklearn.ensemble import RandomForestClassifier

model = RandomForestClassifier(n_estimators=500, min_samples_split=50, random_state=1)
```

- `n_estimators`: This specifies the number of decision trees to be used in

  Random Forest (set to 500 here).
- `random_state`: This ensures reproducibility by setting a
- seed for the random number generator (set to 1 here).

Disadvantages of Random Forest:

- **Interpretability:** Unlike simpler models like decision trees, Random Forest models can be less interpretable, making it challenging to understand the exact reasoning behind its predictions.

- **Tuning Hyperparameters:** Random Forest has several hyperparameters that control its behavior. Tuning these parameters can be time-consuming and requires experimentation to find the optimal settings.

Rolling Averages

List of Horizons:

- The `horizons` list defines a set of time periods (2, 5, 60, 250, 1000 days) used for calculating rolling averages.

Empty List for New Predictors:

- The `new_` list is initially empty. It will be populated with the names of the newly created columns.

Looping through Horizons:

- The code iterates through each horizon in the `horizons` list.

Creating Rolling Averages:

- Inside the loop, `sp500.rolling(window=horizon).mean()` calculates the rolling average for the 'Close' column of the `sp500` DataFrame with a window size to the current `horizon`.
- This new column is added to the `sp500` DataFrame. It stores the ratio between the daily closing price and the corresponding rolling average closing price.
- A similar approach is used to create a new column name with the format "Trend_{horizon}".

Creating Trend Columns:

- A new column name is constructed using string formatting (f-string) with the format "Closing_ratio_{horizon}".

- `rolling(horizon).sum()['Target']` calculates the sum of the 'Target' column within a rolling window of size `horizon` on the shifted DataFrame. This provides a count of how many days within the horizon window (excluding the current day) had an upward trend (Target = 1).

Conclusion and Future Considerations

1

2

3

Accurate Predictions

The Random Forest model demonstrated strong predictive capabilities on historical data.

Scalability

The model can be further improved by incorporating more data sources and advanced techniques.

Real-Time Monitoring

Developing a system to continuously monitor and update the model for real-time predictions.