Title: Bayesian Regression

Introduction

We aim to predict Bitcoin prices using Bayesian Regression. We use historical data from the CoinMarketCap API, which includes features such as opening price, closing price, high, low, and volume.

Data Collection

We fetch the data from the CoinMarketCap API. The data includes the following features:

- Date
- Low
- High
- Open
- Close
- Volume

We also create a new feature, 'Mean', which is the average of 'Low' and 'High' prices.

Data Preprocessing

We clean the data by dropping any rows with NaN or Null fields. We also drop redundant columns that are not needed for our prediction model.

Model Training

We use the Bayesian Regression model from the sklearn library. We train the model on the 'Open' price and predict the 'Mean' price. We use the first 2441 data points for training and the last 272 data points for testing.

Model Evaluation

We evaluate the model by calculating the Root Mean Square Error (RMSE) between the predicted and actual 'Mean' prices.


actual
predicted
12000
11000
10000
9000
8000
7000
6000
5000
0
50
100
150
200
250

Title: Polynomial Regression

Introduction

In this section, we aim to predict Bitcoin prices using Polynomial Regression. We use historical data from the CoinMarketCap API, which includes features such as opening price, closing price, high, low, and volume.

Data Collection

We fetch the data from the CoinMarketCap API. The data includes the following features:

- Date
- Low
- High
- Open
- Close
- Volume

We also create a new feature, 'Mean', which is the average of 'Low' and 'High' prices.

Data Preprocessing

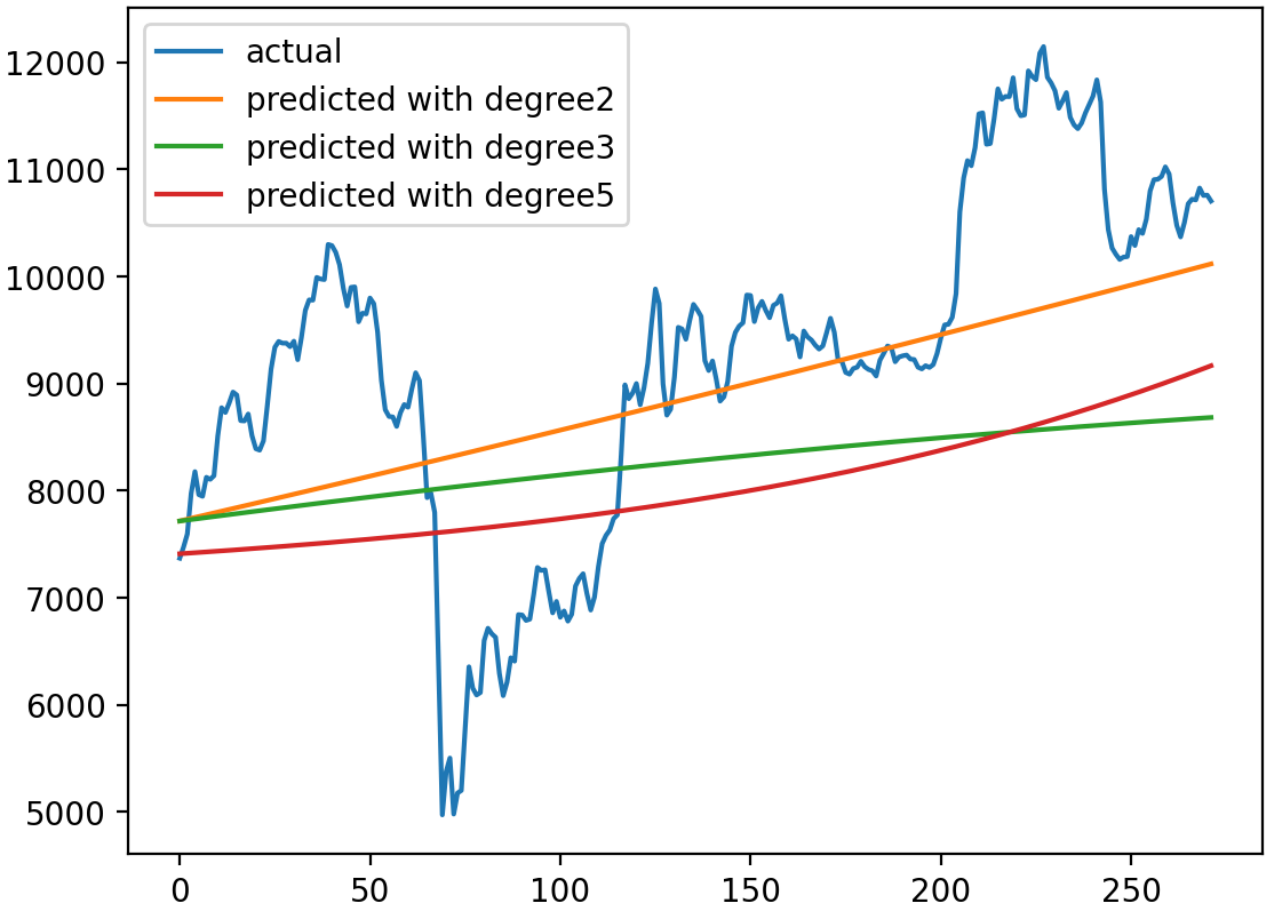
We clean the data by dropping any rows with NaN or Null fields. We also drop redundant columns that are not needed for our prediction model.

Model Training

We use the Polynomial Regression model from the sklearn library. We perform a grid search to find the optimal degree for the polynomial. We train the model on the 'Open' price and predict the 'Mean' price. We use the first 2411 data points for training and the last 272 data points for testing.

Model Evaluation

We evaluate the model by calculating the Root Mean Square Error (RMSE) between the predicted and actual 'Mean' prices. We perform this evaluation for different degrees of the polynomial to find the optimal degree.


actual
predicted with degree2
predicted with degree3
predicted with degree5
12000
11000
10000
9000
8000
7000
6000
5000
0
50
100
150
200
250

Title: Vector Autoregression (VAR)

Introduction

In this section, we aim to predict Bitcoin prices using Vector Autoregression (VAR). We use historical data from the CoinMarketCap API, which includes features such as opening price, closing price, high, low, and volume.

Data Collection

We fetch the data from the CoinMarketCap API. The data includes the following features:

- Date
- Low
- High
- Open
- Close
- Volume

We also create a new feature, 'Mean', which is the average of 'Low' and 'High' prices.

Data Preprocessing

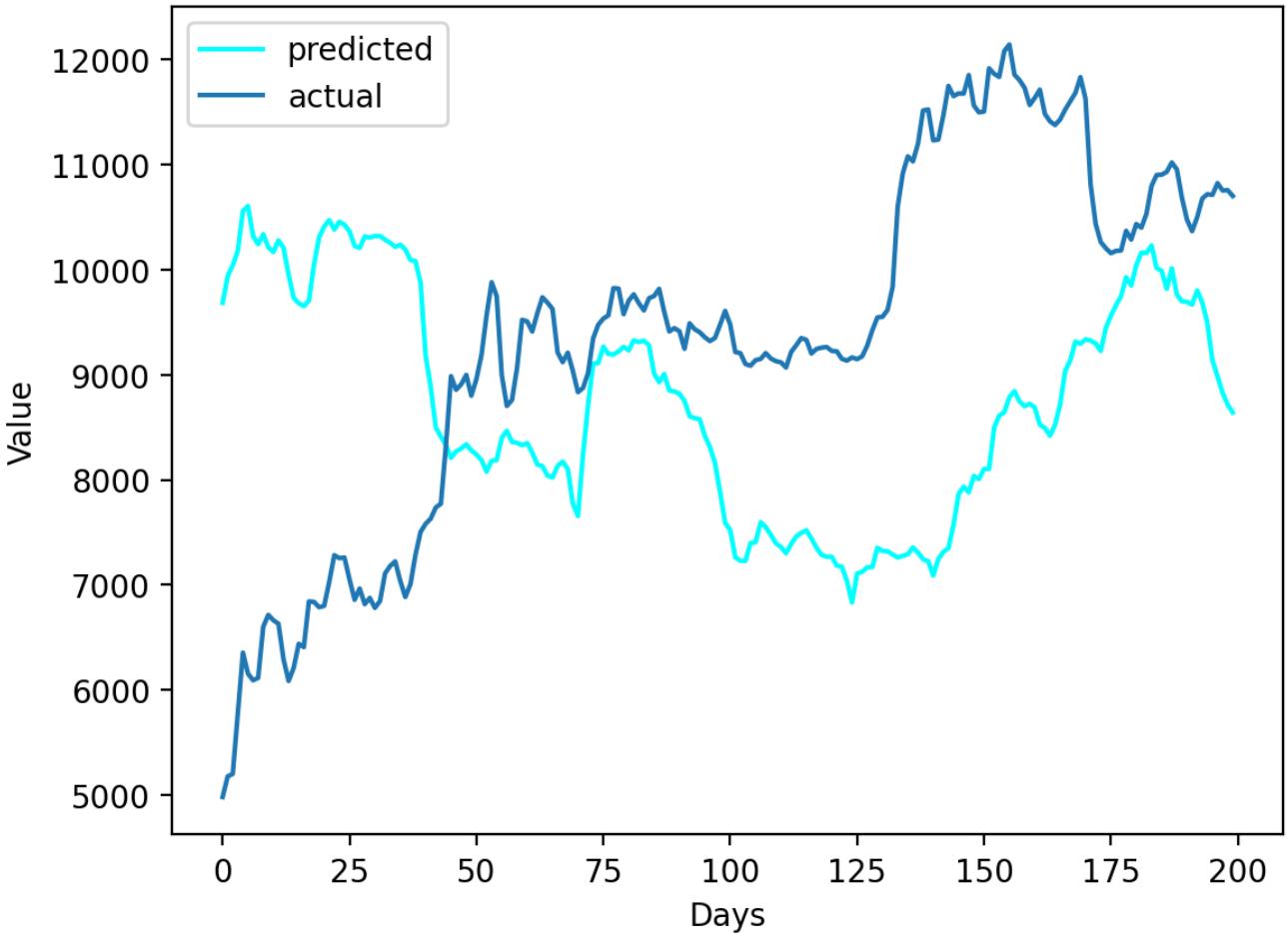
We clean the data by dropping any rows with NaN or Null fields. We also drop redundant columns that are not needed for our prediction model.

Model Training

We use the Vector Autoregression (VAR) model from the statsmodels library. We train the model on the 'Mean' and 'Close' prices, with 'Open' price as the exogenous variable. We use the first 2500 data points for training.

Model Evaluation

We evaluate the model by calculating the Root Mean Square Error (RMSE) between the predicted and actual 'Mean' prices for the last 200 data points.


predicted
actual
12000
11000
10000
9000
8000
7000
6000
5000
Value
0
25
50
75
100
125
150
175
200
Days

Title: Predicting Bitcoin Prices Using Auto ARIMA

Introduction

We tried to predict Bitcoin prices using the Auto ARIMA model. We use historical data from the CoinMarketCap API, which includes features such as opening price, closing price, high, low, and volume.

Data Collection

We fetch the data from the CoinMarketCap API. The data includes the following features:

- Date
- Low
- High
- Open
- Close
- Volume

We also create a new feature, 'Mean', which is the average of 'Low' and 'High' prices.

Data Preprocessing

We clean the data by dropping any rows with NaN or Null fields. We also drop redundant columns that are not needed for our prediction model.

Model Training

We use the Auto ARIMA model from the pmdarima library. We perform a grid search to find the optimal parameters for the ARIMA model. We train the model on the 'Mean' price, with 'Low', 'High', 'Open', 'Close', and 'Volume' as exogenous variables. We use the first 90% of data points for training.

Model Evaluation

We evaluate the model by calculating the Root Mean Square Error (RMSE) between the predicted and actual 'Mean' prices for the last 10% of data points.

Data Looks like

	Date	Low	High	Open	Close	Volume	Mean
0	2013-04-29 23:59:59	134	147.488	134.444	144.54	0	140.744
1	2013-04-30 23:59:59	134.05	146.93	144	139	0	140.49
2	2013-05-01 23:59:59	107.72	139.89	139	116.99	0	123.805
3	2013-05-02 23:59:59	92.2819	125.6	116.38	105.21	0	108.9409
4	2013-05-03 23:59:59	79.1	108.128	106.25	97.75	0	93.614

Normalized X

Date	Low	High	Open	Close	Volume	Mean
2013-04-30 23:59:59	0.0036	0.0036	0.0039	0.0036	0	0.0036
2013-05-01 23:59:59	0.0022	0.0033	0.0036	0.0025	0	0.0027
2013-05-02 23:59:59	0.0014	0.0026	0.0025	0.0019	0	0.002
2013-05-03 23:59:59	0.0007	0.0017	0.0019	0.0015	0	0.0012
2013-05-04 23:59:59	0.0014	0.002	0.0015	0.0023	0	0.0017

Normalized y

Date	BTC Price next day
2013-04-30 23:59:59	0.0036
2013-05-01 23:59:59	0.0036
2013-05-02 23:59:59	0.0027
2013-05-03 23:59:59	0.002
2013-05-04 23:59:59	0.0012

Title: Predicting Bitcoin Prices Using SARIMAX

Introduction

In this section, we aim to predict Bitcoin prices using the SARIMAX model. We use historical data from the CoinMarketCap API, which includes features such as opening price, closing price, high, low, and volume.

Data Collection

We fetch the data from the CoinMarketCap API. The data includes the following features:

- Date
- Low
- High
- Open
- Close
- Volume

We also create a new feature, 'Mean', which is the average of 'Low' and 'High' prices.

Data Preprocessing

We clean the data by dropping any rows with NaN or Null fields. We also drop redundant columns that are not needed for our prediction model. We then normalize the data using the MinMaxScaler from the sklearn library.

Model Training

We use the SARIMAX model from the statsmodels library. We train the model on the 'Mean' price, with 'Low', 'High', 'Open', 'Close', and 'Volume' as exogenous variables. We use the first 90% of data points for training.

Model Evaluation

We evaluate the model by calculating the Root Mean Square Error (RMSE) between the predicted and actual 'Mean' prices for the last 10% of data points.


Pred
Actual
12000
11000
10000
9000
8000
7000
6000
5000
2020-01
2020-02
2020-03
2020-04
2020-05
2020-06
2020-07
2020-08
2020-09
2020-10

Title: Predicting Using Random Forest Regression

Introduction

In this project, we aim to predict Bitcoin prices using the Random Forest Regression model. We use historical data from the CoinMarketCap API, which includes features such as opening price, closing price, high, low, and volume.

Data Collection

We fetch the data from a CSV file named 'BTC-Hourly.csv'. The data includes the following features:

- Date
- Open
- High
- Low
- Close
- Volume_BTC
- Volume_USD

Data Preprocessing

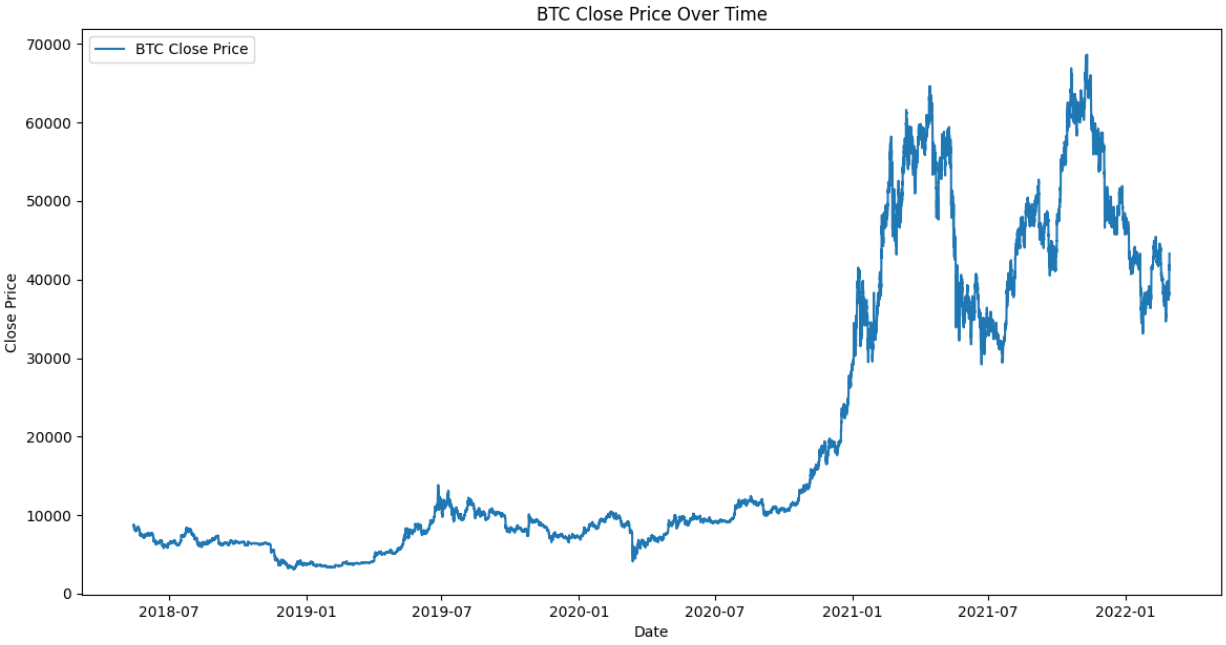
We clean the data by converting the 'date' column to datetime and sorting the data by date. We also create new features such as 'return', 'return_lag1', 'ma7', 'ma30', and 'std30'. We then drop any rows with NaN values.

Model Training

We use the Random Forest Regressor model from the sklearn library. We scale the features using the StandardScaler from the sklearn library. We split the data into training and testing sets, with 80% of the data for training and 20% for testing. We train the model on the features and predict the 'Close' price.

Model Evaluation

We evaluate the model by calculating the Mean Squared Error (MSE) between the predicted and actual 'Close' prices.


BTC Close Price Over Time
BTC Close Price
Close Price
Date
0
10000
20000
30000
40000
50000
60000
70000
2018-07
2019-01
2019-07
2020-01
2020-07
2021-01
2021-07
2022-01