Course project - Big Data Concepts

Yash Pratap Solanky

Github link - https://github.com/ysolanky/big-data-project

Introduction

Cryptocurrencies have recently become a widely recognised way for many individuals to invest their money. Cryptocurrency values have increased dramatically during the last several years. As a result, I tried numerous ways to find an efficient and accurate model to predict the price using machine learning algorithms. For the course project, I analysed four cryptocurrencies and attempted to predict the price of bitcoin. The CSV time series data is uploaded to GitHub. The time-series data in CSV format is posted to GitHub. The GitHub repository is then cloned into a Jupyter notebook under the Google cloud platform's AI platform. The time-series graph and the prediction graph are then saved as png files and uploaded to Github and to a bucket on the Google cloud platform that has been created separately.

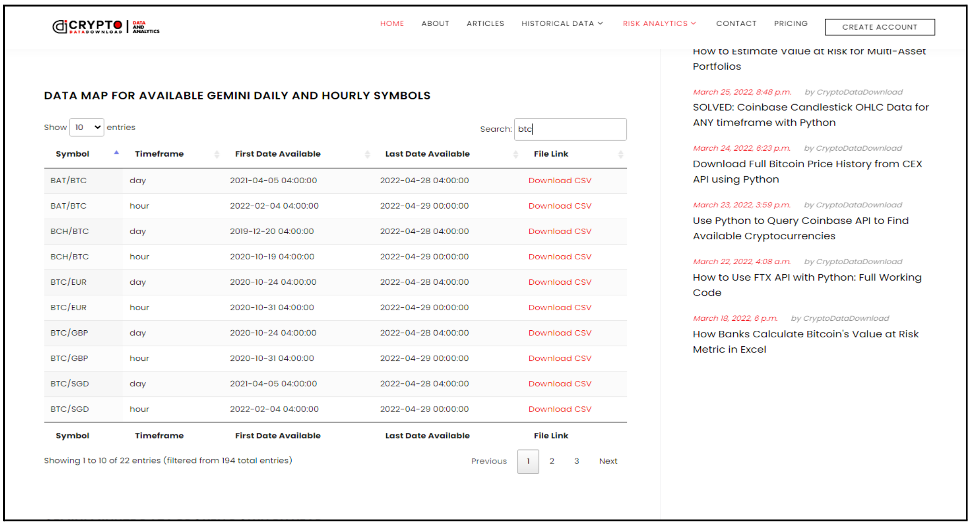
Background

I attempted to predict cryptocurrencies due to the fluctuating nature and fluctuation in the values of cryptocurrencies since the pandemic has piqued the interest of people all around the world. We can completely compute, store, analyse, and predict this project on the Google cloud platform thanks to the usage of virtual machines. Although this project could have been completed on a local system, the usage of a virtual machine makes it extremely simple for anybody to replicate it for any time series data. It might be for stocks, cryptocurrency, or something else.

Methodology

Step 1: Downloading the time series data of various cryptocurrencies from https://www.cryptodatadownload.com/.

I downloaded the daily prices of Bitcoin, Litecoin, Ethereum and Dogecoin from this website. The range of time series of all the four coins was different.



DATA MAP FOR AVAILABLE GEMINI DAILY AND HOURLY SYMBOLS

Show 10 entries

Search: btc

Symbol	Timeframe	First Date Available	Last Date Available	File Link
BAT/BTC	day	2021-04-05 04:00:00	2022-04-28 04:00:00	Download CSV
BAT/BTC	hour	2022-02-04 04:00:00	2022-04-29 00:00:00	Download CSV
BCH/BTC	day	2019-12-20 04:00:00	2022-04-28 04:00:00	Download CSV
BCH/BTC	hour	2020-10-19 04:00:00	2022-04-29 00:00:00	Download CSV
BTC/EUR	day	2020-10-24 04:00:00	2022-04-28 04:00:00	Download CSV
BTC/EUR	hour	2020-10-31 04:00:00	2022-04-29 00:00:00	Download CSV
BTC/GBP	day	2020-10-24 04:00:00	2022-04-28 04:00:00	Download CSV
BTC/GBP	hour	2020-10-31 04:00:00	2022-04-29 00:00:00	Download CSV
BTC/SGD	day	2021-04-05 04:00:00	2022-04-28 04:00:00	Download CSV
BTC/SGD	hour	2022-02-04 04:00:00	2022-04-29 00:00:00	Download CSV
Symbol	Timeframe	First Date Available	Last Date Available	File Link

Showing 1 to 10 of 22 entries (filtered from 194 total entries)

Previous 1 2 3 Next

How to Estimate Value at Risk for Multi-Asset Portfolios

March 25, 2022, 8:48 p.m. by CryptoDataDownload

SOLVED: Coinbase Candlestick OHLC Data for ANY timeframe with Python

March 24, 2022, 6:23 p.m. by CryptoDataDownload

Download Full Bitcoin Price History from CEX API using Python

March 23, 2022, 3:59 p.m. by CryptoDataDownload

Use Python to Query Coinbase API to Find Available Cryptocurrencies

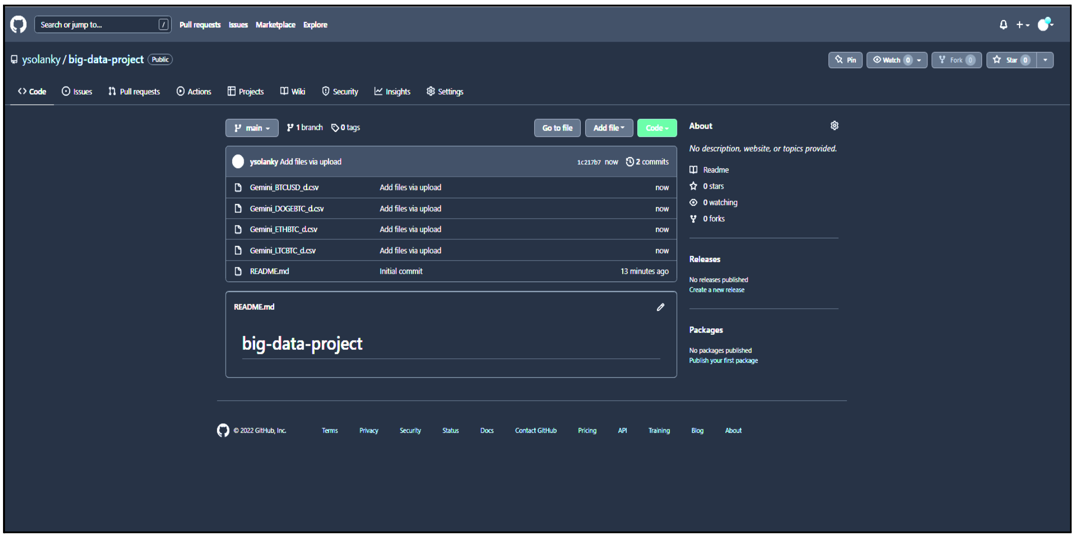
March 22, 2022, 4:08 a.m. by CryptoDataDownload

How to Use FTX API with Python: Full Working Code

March 18, 2022, 6 p.m. by CryptoDataDownload

How Banks Calculate Bitcoin's Value at Risk Metric in Excel

Step 2: Creating a repository and uploading the downloaded data to the repository



ysolanky Add files via upload 1c217b7 now 2 commits

Gemini_BTCUSD_d.csv	Add files via upload	now
Gemini_DOGEBTC_d.csv	Add files via upload	now
Gemini_ETHBTC_d.csv	Add files via upload	now
Gemini_LTCBTC_d.csv	Add files via upload	now
README.md	Initial commit	13 minutes ago

README.md

big-data-project

About

No description, website, or topics provided.

Readme
0 stars
0 watching
0 forks

Releases

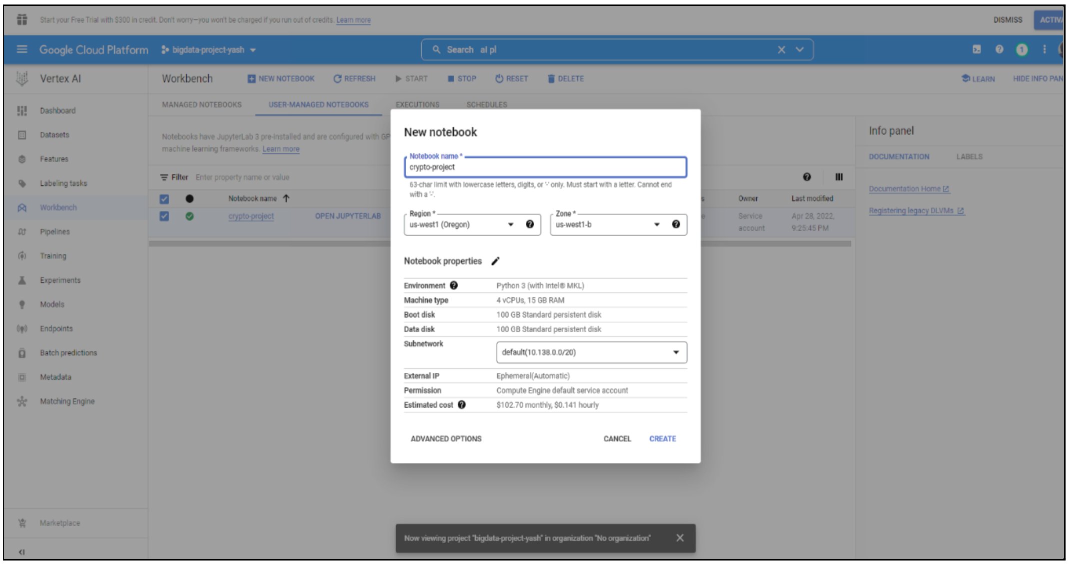
No releases published
Create a new release

Packages

No packages published
Publish your first package

Step 3: Creating a Jupyter notebook on the AI platform on the Google cloud platform and then cloning the repository to this Jupyter environment.

The AI platform supports Jupyter notebooks and has inbuilt support for git repositories. Making it easy to link the notebook with repository,



New notebook

Notebook name *
crypto-project

63-char limit with lowercase letters, digits, or '-' only. Must start with a letter. Cannot end with a '-'.

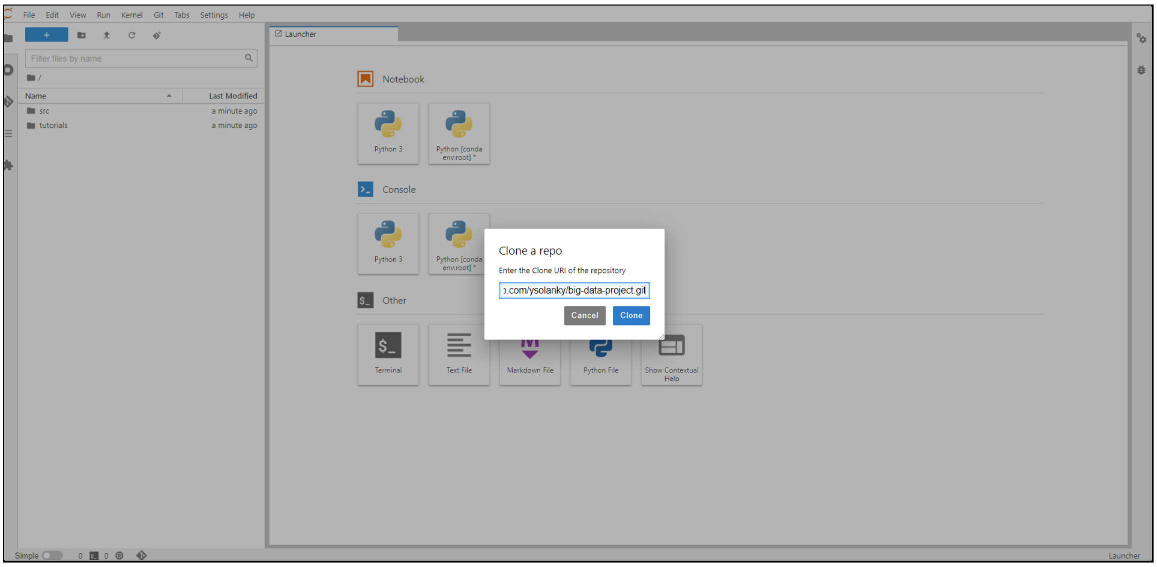
Region *
us-west1 (Oregon)

Zone *
us-west1-b

Notebook properties

Environment	Python 3 (with Intel® MKL)
Machine type	4 vCPUs, 15 GB RAM
Boot disk	100 GB Standard persistent disk
Data disk	100 GB Standard persistent disk
Subnetwork	default(10.138.0.0/20)
External IP	Ephemeral(Automatic)
Permission	Compute Engine default service account
Estimated cost	$102.70 monthly, $0.141 hourly

Now viewing project "bigdata-project-yash" in organization "No organization"



Clone a repo

Enter the Clone URI of the repository

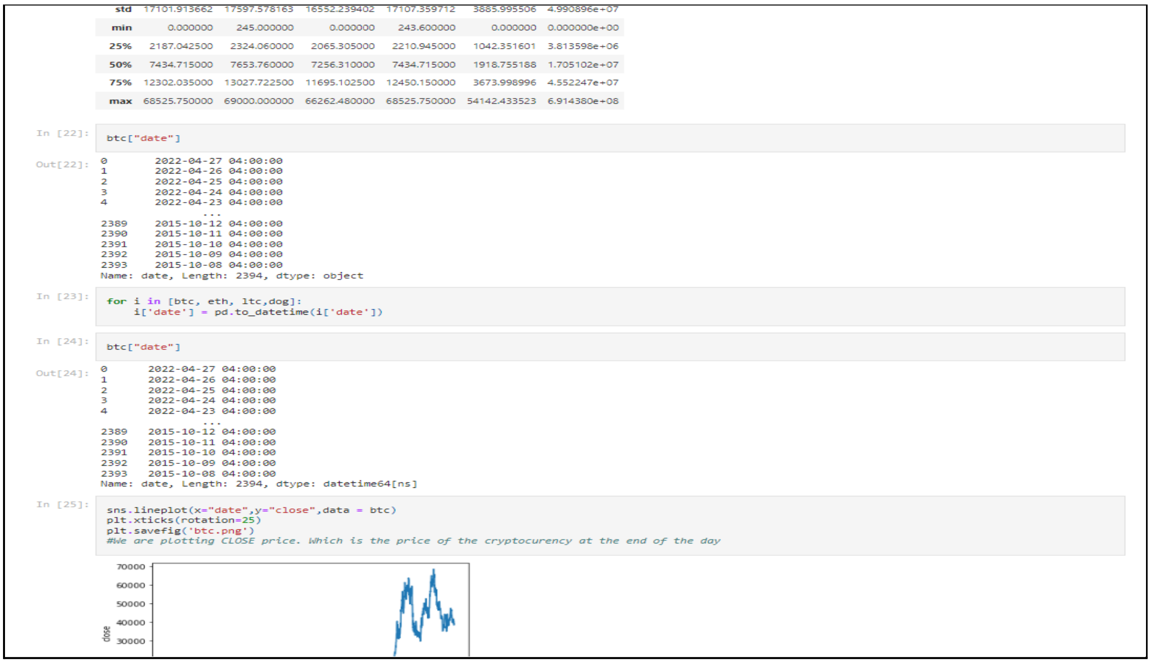
).com/ysolanky/big-data-project.git

Cancel Clone

Step 4: Performing exploratory data analysis on the data.

We will performing data engineering (cleaning data, transforming data), and then using the final dataset to make predictions using the autoregressive integrated moving average (ARIMA) algorithm.We then will also be using facebook's prophet library to make predictions.

For ARIMA we will be using 2364 rows for training data and will use the last 30 days for testing. For the hyperparameters we will be using (10,1,5). For prophet we will be using 2300 days for training and testing on the final 94 days.

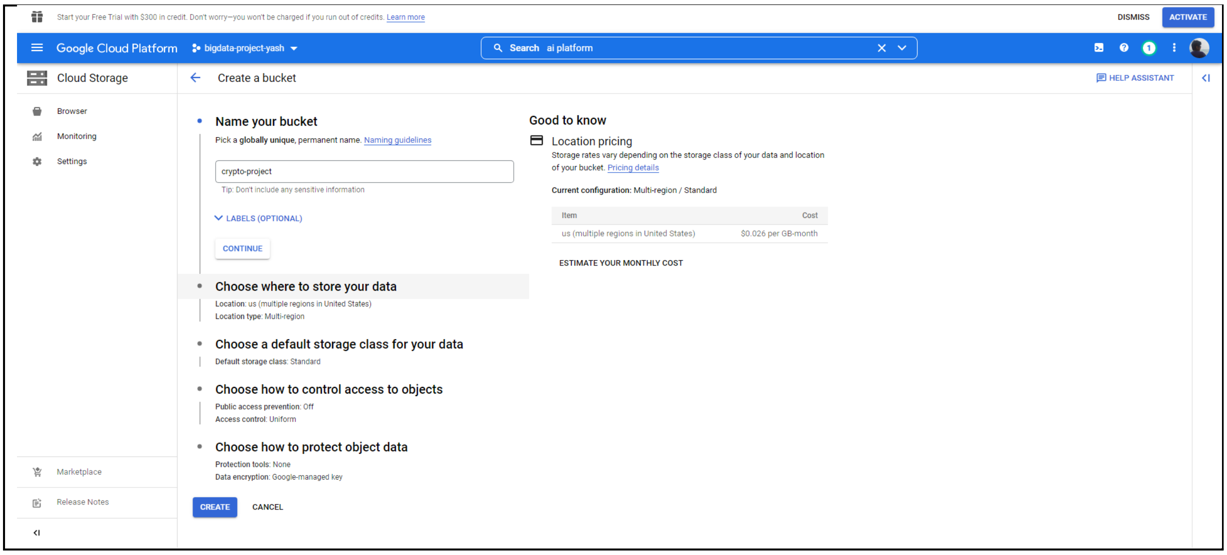


std	17101.913662	17597.578163	16552.239402	17107.359712	3885.995506	4.990896e+07
min	0.000000	245.000000	0.000000	243.600000	0.000000	0.000000e+00
25%	2187.042500	2324.060000	2065.305000	2210.945000	1042.351601	3.813598e+06
50%	7434.715000	7653.760000	7256.310000	7434.715000	1918.755188	1.705102e+07
75%	12302.035000	13027.722500	11695.102500	12450.150000	3673.998996	4.552247e+07
max	68525.750000	69000.000000	66262.480000	68525.750000	54142.433523	6.914380e+08

```
In [22]: btc["date"]

Out[22]: 0       2022-04-27 04:00:00
         1       2022-04-26 04:00:00
         2       2022-04-25 04:00:00
         3       2022-04-24 04:00:00
         4       2022-04-23 04:00:00
                         ...
         2389    2015-10-12 04:00:00
         2390    2015-10-11 04:00:00
         2391    2015-10-10 04:00:00
         2392    2015-10-09 04:00:00
         2393    2015-10-08 04:00:00
         Name: date, Length: 2394, dtype: object

In [23]: for i in [btc, eth, ltc,dog]:
             i['date'] = pd.to_datetime(i['date'])

In [24]: btc["date"]

Out[24]: 0      2022-04-27 04:00:00
         1      2022-04-26 04:00:00
         2      2022-04-25 04:00:00
         3      2022-04-24 04:00:00
         4      2022-04-23 04:00:00
                        ...
         2389   2015-10-12 04:00:00
         2390   2015-10-11 04:00:00
         2391   2015-10-10 04:00:00
         2392   2015-10-09 04:00:00
         2393   2015-10-08 04:00:00
         Name: date, Length: 2394, dtype: datetime64[ns]

In [25]: sns.lineplot(x="date",y="close",data = btc)
         plt.xticks(rotation=25)
         plt.savefig('btc.png')
         #We are plotting CLOSE price. Which is the price of the cryptocurency at the end of the day
```

Step 5: Creating a google cloud platform bucket

In addition to saving the outputs from our exploratory data analysis and the ARIMA prediction to Github, we wish to publish the results on a google cloud platform bucket.



Start your Free Trial with $300 in credit. Don't worry—you won't be charged if you run out of credits. Learn more

Create a bucket

Name your bucket

Pick a **globally unique**, permanent name. Naming guidelines

crypto-project

Tip: Don't include any sensitive information

Choose where to store your data

Location: us (multiple regions in United States)
Location type: Multi-region

Choose a default storage class for your data

Default storage class: Standard

Choose how to control access to objects

Public access prevention: Off
Access control: Uniform

Choose how to protect object data

Protection tools: None
Data encryption: Google-managed key

Good to know

Location pricing

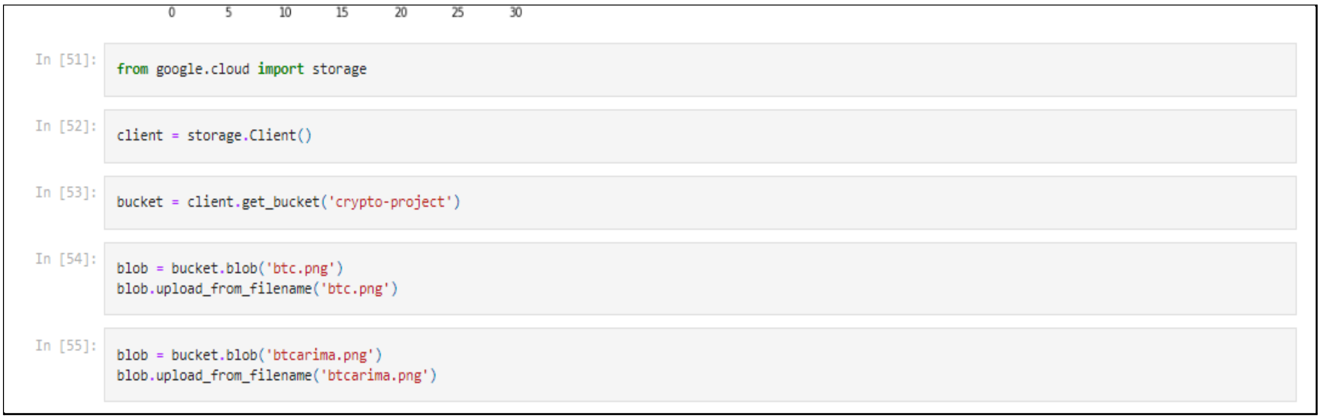
Storage rates vary depending on the storage class of your data and location of your bucket. Pricing details

Current configuration: Multi-region / Standard

Item	Cost
us (multiple regions in United States)	$0.026 per GB-month

Step 6: Saving the graphs from the project to the bucket.

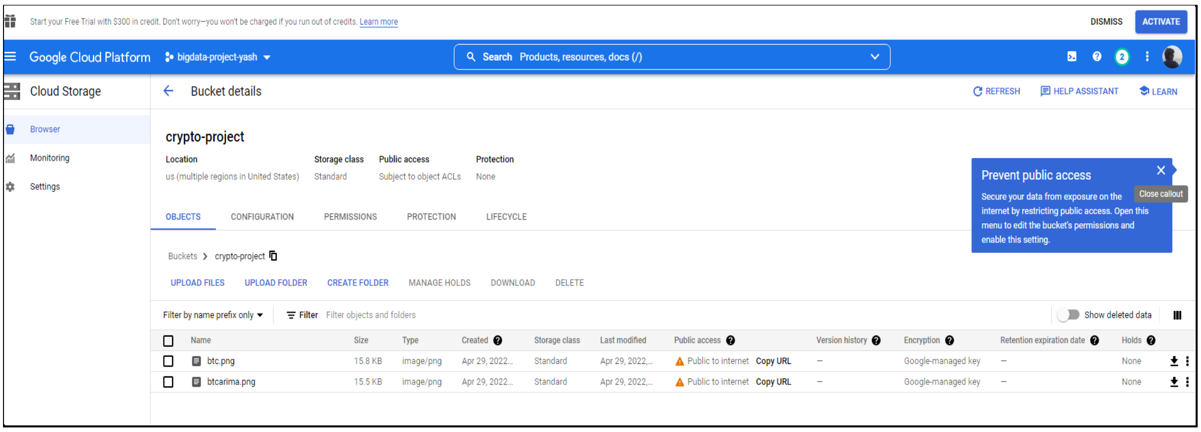
Using the google.cloud library in Python, we can link our pre-existing GCP bucket with the Jupyter notebook. And save the outputs to the bucket.



```
In [51]: from google.cloud import storage

In [52]: client = storage.Client()

In [53]: bucket = client.get_bucket('crypto-project')

In [54]: blob = bucket.blob('btc.png')
         blob.upload_from_filename('btc.png')

In [55]: blob = bucket.blob('btcarima.png')
         blob.upload_from_filename('btcarima.png')
```

Step 7: Making the graphs public from the bucket

We can now make the contents of the bucket publicly available through a URL. This can be done by editing the access of the files individually.



crypto-project

Location	Storage class	Public access	Protection
us (multiple regions in United States)	Standard	Subject to object ACLs	None

Prevent public access

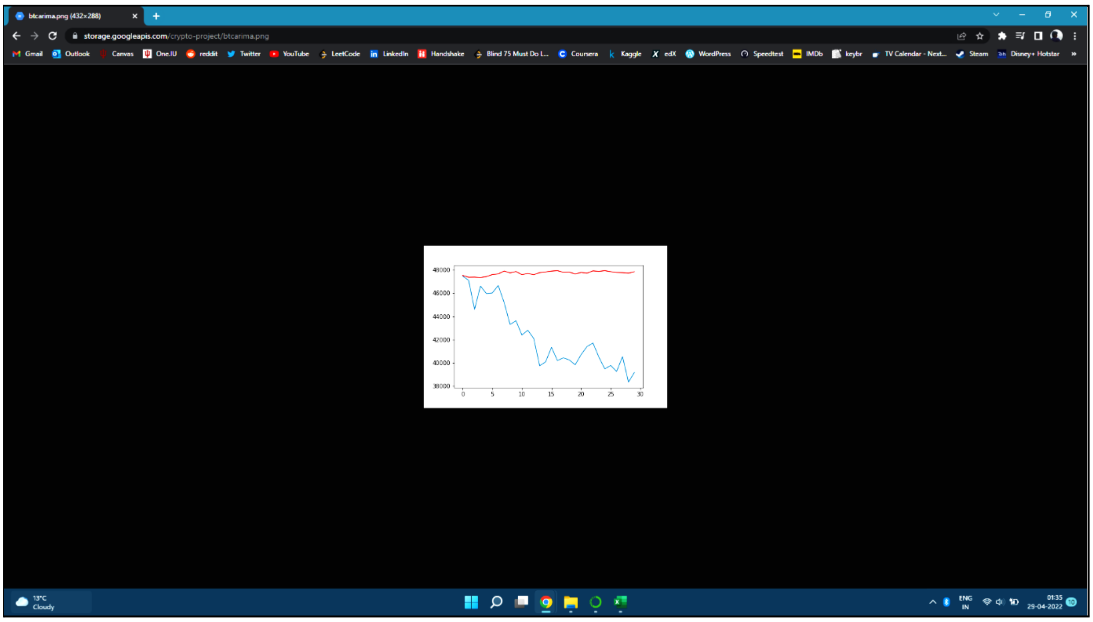
Secure your data from exposure on the internet by restricting public access. Open this menu to edit the bucket's permissions and enable this setting.

□	Name	Size	Type	Created	Storage class	Last modified	Public access	Version history	Encryption	Retention expiration date	Holds
□	btc.png	15.8 KB	image/png	Apr 29, 2022...	Standard	Apr 29, 2022,...	Public to internet Copy URL	–	Google-managed key	–	None
□	btcarima.png	15.5 KB	image/png	Apr 29, 2022...	Standard	Apr 29, 2022,...	Public to internet Copy URL	–	Google-managed key	–	None

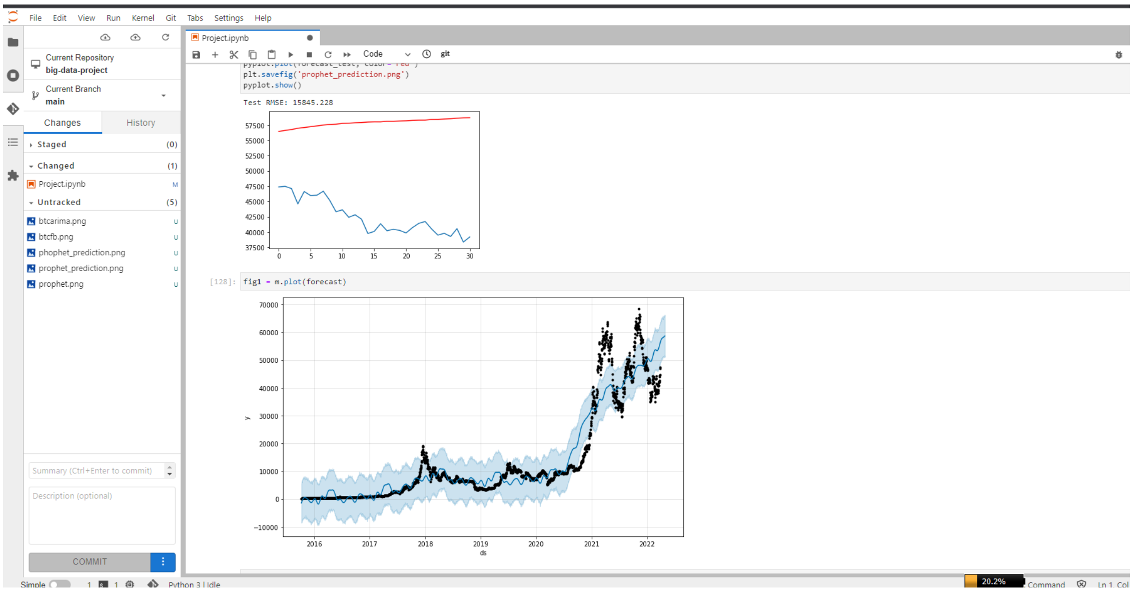
Results

Through our exploratory analysis we observed that towards the end of 2020, all the cryptocurrencies experienced a rapid increase in prices. There is still some debate as to what caused it, as the rise in crypto prices is in contrast to the world economy which due to the COVID-19 pandemic has been in a decline.

Using Arima, we were not able to accurately anticipate the price of Bitcoin. The algorithm predicted the price of bitcoin to remain stable, yet it plummeted significantly. This is rather expected as Bitcoin and all other cryptocurrencies follow an erratic pattern.



Above, you can see the prediction (in red line) significantly different from the actual data (blue line). Hence, for any real world application, this prediction is not acceptable.



```
pyplot.plot(forecast_test, color='red')
plt.savefig('prophet_prediction.png')
pyplot.show()
```

```
Test RMSE: 15845.228
```

```
[128]: fig1 = m.plot(forecast)
```

Surprisingly, even worse results were obtained using the prophet library. Above you can see the first graph with prediction (in red line) significantly different from the actual data (blue line). The second graph is a forecast of the time series.

Discussion

An interpretation of the results

Even though ARIMA is a very sophisticated algorithm and is considered one of the best ways to make predictions on time series data, it proved to be relatively poor for predicting cryptocurrencies. The same result was obtained when prediction was done using facebook's prophet library. This leads to the conclusion that cryptocurrencies are hard to predict using the measures that have been proven to work in great effect for stocks.

A discussion of how you employed the technologies/skills from this course.

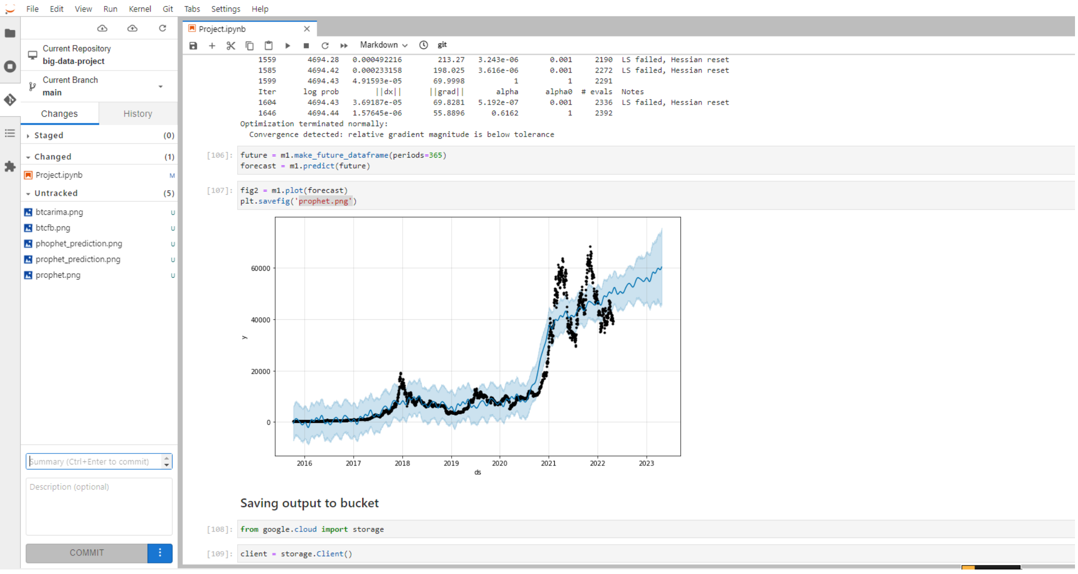
For this project, I did the following:

- Implemented a pipeline (e.g., download - transform - summarize - visualize) - The pipeline was implemented on a jupyter notebook in google cloud platform. The data was downloaded in the form of csv files, it was transformed by changing variables to datetime objects and removing the unnecessary columns. It was summarised and then visualised.
- Investigated a big data cloud platform environment - Google cloud platforms AI platform and storage bucket were used in the scope of this project.
- Developed a data management plan that describes how this dataset is being collected, stored, preserved, and shared - The dataset is being collected through a cryptocurrency historical archive and stored,preserved and shared on github
- (1) Pipelines, (2) Established techniques for data cleaning/quality assurance (3) Publishing data in a repository (automatically or manually). All three of the above stated points were implemented in the scope of this project. The pipeline was created

in a Jupyter notebook on the AI platform on GCP, the technique for data cleaning and quality assurance has been demonstrated in the notebook. The results in the form of graphs have been published to GitHub.

Any barriers or failures you encountered?

The traditional and the modern techniques used to predict stock prices have resulted in fruitless results for prediction of cryptocurrencies. Very high value for the Root mean square error (RMSE) was obtained in these experiments, hence these results are far from acceptable in real world application. I think that the sudden rise and drop in price make it very difficult to predict using parametric linear model constructions, yet if with more information of weekly, seasonal and yearly trends, the prediction accuracy may be improved.



```
       1559      4694.28   0.000492216       213.27   3.243e-06       0.001      2190  LS failed, Hessian reset
       1585      4694.42   0.000233158      198.025   3.616e-06       0.001      2272  LS failed, Hessian reset
       1599      4694.43   4.91593e-05      69.9998           1           1      2291
       Iter      log prob        ||dx||      ||grad||       alpha      alpha0  # evals  Notes
       1604      4694.43   3.69187e-05      69.8281   5.192e-07       0.001      2336  LS failed, Hessian reset
       1646      4694.44   1.57645e-06      55.8896      0.6162           1      2392
Optimization terminated normally:
  Convergence detected: relative gradient magnitude is below tolerance
```

```
[106]: future = m1.make_future_dataframe(periods=365)
       forecast = m1.predict(future)
```

```
[107]: fig2 = m1.plot(forecast)
       plt.savefig('prophet.png')
```

Saving output to bucket

```
[108]: from google.cloud import storage
```

```
[109]: client = storage.Client()
```

Above is a 365 day forecast of BTC. It predicts a somewhat steady growth.

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

Conclusion

Cryptocurrency prices are hard to predict using traditional methods for stock prediction because of the lack of trends that stocks follow, as cryptocurrencies are more volatile than stocks. Recently it has been observed that a single event could trigger an unprecedented rise or fall in the price of crypto or stock. For example, Elon Musk tweeted that everyone should be buying GameStop stocks and Dogecoin cryptocurrency which led to a huge increase in their respective price. Events like this are hard to be integrated as a part of input information for forecasting.

But using the GCP and github, this project can be modeleted to implement prediction on any time series data. I expect it to perform much better for traditional time series like stocks.

References

1. https://www.qwiklabs.com/focuses/1846?catalog_rank=%7B%22rank%22%3A1%2C%22num_filters%22%3A0%2C%22has_search%22%3Atrue%7D&parent=catalog&search_id=7008005

2. https://cloud.google.com/ai-platform/training/docs/working-with-cloud-storage

3. https://www.cryptodatadownload.com

4. https://github.com/googleapis/python-storage

5. https://stackoverflow.com/questions/37003862/how-to-upload-a-file-to-google-cloud-storage-on-python-3