

CS-524 Intro to Cloud Computing

Creating a BitCoin Price Prediction Model

Using AWS SageMaker

Final Report

Members:

**David DeLaus, Abhishek Desai,
Hardi Patel, Jayant Kumar, Jiahn Liu, Vishal
Kuchadi, Malika Thakur
(TEAM 5)**

CONTENTS

SECTION	TITLE	PAGE NUMBER
Introduction	Introduction	3
	Explain Technology used	4
Deployment Guide	Setting up AWS Services	6
	Data Analysis and Cleaning	14
	Creating the Models	20
	Deletion of the Environment	22
Conclusion	Results	31
References	Useful Links	32

INTRODUCTION

Abstract

Bitcoin is the most well-known and longest-running cryptocurrency, having been published as an open source project in 2009 by the anonymous Satoshi Nakamoto. Bitcoin is a decentralized digital currency that allows transactions to be checked and documented in a public distributed ledger (the blockchain) without the need for a trustworthy record-keeping authority or a central intermediary. A transaction block is made up of a collection of transactions.

Content

bitstampUSD 1-min data 2012-01-01 to 2021-03-31.csv is the dataset used for select bitcoin exchanges, with minute-by-minute updates of OHLC (Open, High, Low, Close), for the period January 2012 to December March 2021. Unix time is used for timestamps. The data fields of timestamps with no trades or operation are loaded with NaNs.

How Bitcoin Works:

Bitcoin is the most popular and valuable cryptocurrency that is currently being traded and mined. It was launched in 2009 and ever since then the price has continued to rise with the ever increasing demand. The demand for Bitcoin is due to its nature of being a digital currency and the scarcity that is caused by its ever increasing difficulty in mining. Mining is the process of creating new Bitcoin. It works by having computers solve very difficult math problems that help update the blockchain ledger for the cryptocurrency and create new coins.

Technology Used

AWS S3 Bucket

Amazon Simple Storage Service (Amazon S3) is an object storage service that offers elite levels of performance, versatility, and accessibility. It has very easy to use and understand management tools allowing the user to customize their buckets to fit their needs.

You can simply upload the resources you need to the S3 Bucket and then access it anytime from a website or another AWS service. There is no charge for an S3 Bucket as it is in the free tier.

Benefits:

1. Fast and simple to access
2. Available anytime
3. High read speeds

AWS SageMaker

AWS SageMaker is Amazon's most comprehensive machine learning service. It is the first IDE for machine learning and it was designed from the ground up to be as functional as possible. SageMaker comes with the SageMaker studio which simplifies the process for the development of machine learning models.

All you need to do is upload a notebook file to the service and it will run it for you at a fraction of the time it would take a personal machine to run the algorithms.

Benefits:

1. Much faster than using a personal computer to train models
2. Very easy to work with
3. Can be accessed by multiple people

AWS Sagemaker JupyterLab

JupyterLab is a web-based development platform that allows you to work with Jupyter notebooks, code, and data. You can write and test code in Jupyter notebooks without having to recompile entire files because you can compile line by line. Notebooks often make it simple to share your code by allowing you to share it as a notebook, pdf, or html file.

Benefits:

1. Easily share code to speed development process
2. Lightweight to run on any machine
3. Accepted by AWS SageMaker

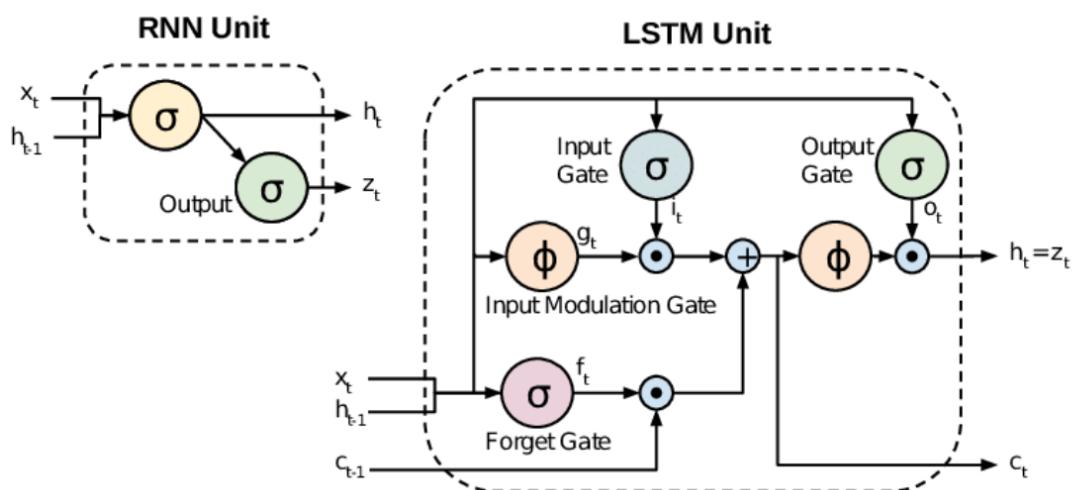
Prediction Models

Simple RNN

RNN or recurrent neural network is a class of artificial neural networks. It works by using connections between nodes to form directed graphs that can be used to then predict outcomes of events.

LSTM

LSTM which stands for Long short-term memory. It is an artificial RNN that is used in the deep learning field of study. It is commonly used for speech recognition, machine translation, and financial modeling. It works by using feedback connections as opposed feedforward.



Deployment Guide

The project deployment works in 4 major stages:

1. Setting up the **AWS service i.e S3 bucket, and Creating AWS Sagemaker notebook instance with ml.t2.medium and AWS IAM instance.**
2. Writing code for Data analysis and cleaning, and
3. Writing the machine learning code in **Python** and uploading it to the Sage Maker instance and deploying it on AWS.
4. Deleting the environment

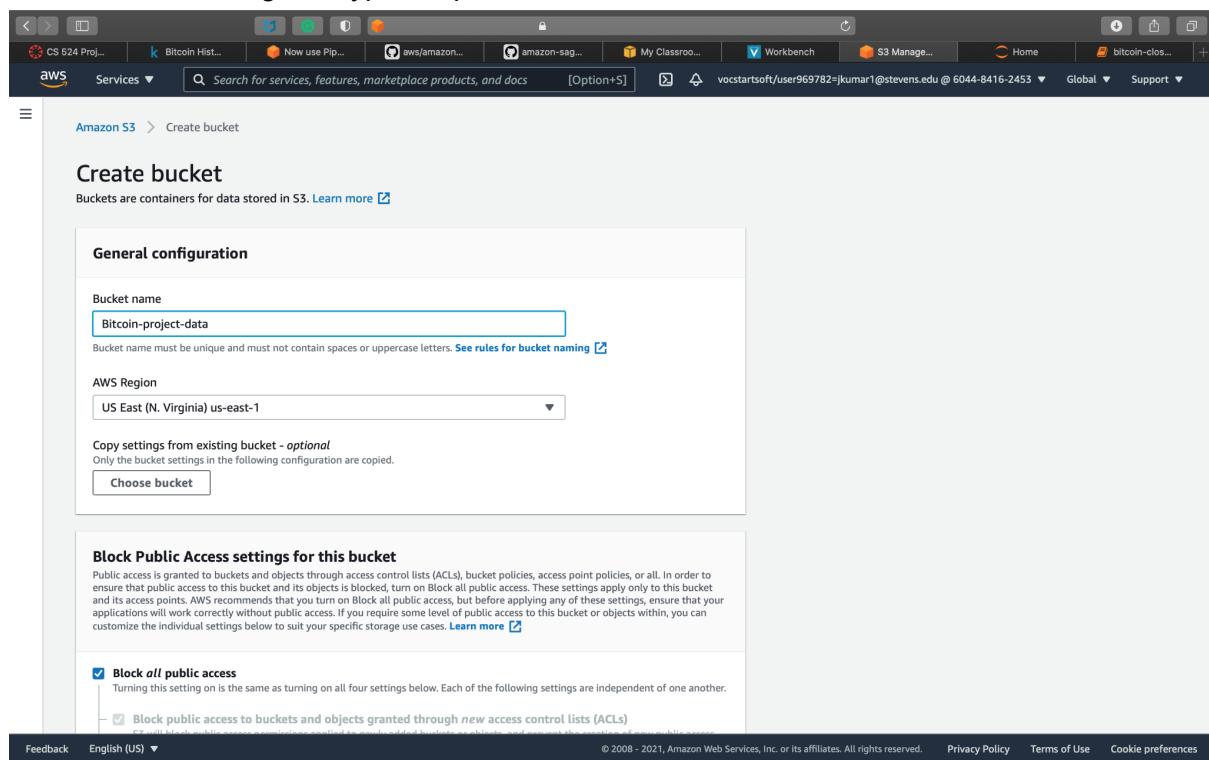
Setting Up AWS Services

We decided to use SageMaker as it was a strong machine learning service that AWS has and seemed to fit our needs the best. We also had to utilize an S3 bucket in order to have SageMaker access the bitcoin sale data.

Step 1: Creating a S3 bucket for our data.

Add bucket name, Select AWS region, Block all Public access for not giving public access of the URL.

Select the versioning, encryption options and click on CREATE BUCKET.



The screenshot shows the AWS S3 Bucket creation wizard. The configuration steps include:

- Bucket Versioning:** Set to **Disable**.
- Tags (0) - optional:** No tags associated with this bucket. A button to **Add tag** is present.
- Default encryption:** Automatically encrypt new objects stored in this bucket. Set to **Disable**.
- Server-side encryption:** Set to **Disable**.
- Advanced settings:** A section with a note: "After creating the bucket you can upload files and folders to the bucket, and configure additional bucket settings."

At the bottom right are **Create bucket** and **Cancel** buttons.

S3 bucket has been successfully created.

The screenshot shows the AWS S3 Buckets page. The main message is: "Successfully created bucket 'bitcoin-project-data'. To upload files and folders, or to configure additional bucket settings choose View details." Below this, the **Account snapshot** provides metrics for total storage (647.1 MB), object count (4), and average object size (161.8 MB). The **Buckets (1)** section lists the newly created bucket:

Name	AWS Region	Access	Creation date
bitcoin-project-data	US East (N. Virginia) us-east-1	Bucket and objects not public	May 3, 2021, 17:12:21 (UTC-04:00)

At the bottom right are **Create bucket** and other management buttons.

Step 2: Uploading our data to our bucket.

The screenshot shows the AWS S3 'Upload' interface. In the 'Files and folders' section, a single file 'bitstampUSD_1-min_data_2012-01-01_to_2021-03-31.csv' is listed with a size of 302.8 MB. In the 'Destination' section, the path 's3://bitcoin-project-data' is specified. The 'Permissions' section is collapsed. At the bottom, standard AWS navigation links are visible.

The screenshot shows the 'Upload: status' page. It displays a summary table with one succeeded file (0 files, 640.0 KB) and zero failed files. Below this, the 'Files and folders' table shows the same single file entry with a status of 'In Progress (0%)'. The 'Configuration' tab is also visible at the bottom of the main content area.

Upload: status

The information below will no longer be available after you navigate away from this page.

Destination	Succeeded	Failed
s3://bitcoin-project-data	1 file, 302.8 MB (100.00%)	0 files, 0 B (0%)

Files and folders Configuration

Files and folders (1 Total, 302.8 MB)

Name	Type	Size	Status	Error
bitstampUSD_1-min_data_2012-01-01_to_2021-03-31.csv	text/csv	302.8 MB	Succeeded	-

Step 3: Configuring our access list to make sure the SageMaker instance could have access to it.

Edit access control list

Access control list (ACL)
Grant basic read/write permissions to AWS accounts. [Learn more](#)

Grantee	Objects	Object ACL
Object owner (your AWS account)	<input checked="" type="checkbox"/> Read	<input checked="" type="checkbox"/> Read <input checked="" type="checkbox"/> Write
Canonical ID: 95c6dea36f6b244ebf22dc5756cb619ceda5fa0298c669b3660c0bc328df665		
Everyone (public access)	<input checked="" type="checkbox"/> Read	<input checked="" type="checkbox"/> Read <input type="checkbox"/> Write
Authenticated users group (anyone with an AWS account)	<input type="checkbox"/> Read	<input type="checkbox"/> Read <input type="checkbox"/> Write

⚠️ When you grant access to the Everyone or Authenticated users group grantees, anyone in the world can access this object.
[Learn more](#)

I understand the effects of these changes on this object.
⚠️ You must select the check box to continue.

S

Authenticated users group Read Read Write

Group: http://acs.amazonaws.com/groups/global/AuthenticatedUsers

⚠ When you grant access to the Everyone or Authenticated users group grantees, anyone in the world can access this object.

[Learn more](#)

I understand the effects of these changes on this object.

Access for other AWS accounts

No other AWS accounts associated with the resource.

[Add grantee](#)

Specified objects

Name	Version ID	Type	Last modified	Size
bitstampUSD_1-min_data_2012-01-01_to_2021-03-31.csv	-	csv	May 3, 2021, 17:13:26 (UTC-04:00)	302.8 MB

[Cancel](#) [Save changes](#)

Feedback English (US) ▾ © 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use Cookie preferences

URL of our dataset extracted from Kaggle and uploaded in our S3 bucket:

https://bitcoin-project-data.s3.amazonaws.com/bitstampUSD_1-min_data_2012-01-01_to_2021-03-31.csv

Bitcoin Historical Data | Kaggle S3 Management Console Home https://bitcoin-project-data.s3.amazonaws.com/bitstampUSD_1-min_data_2012-01-01_to_2021-03-31.csv

```
Timestamp,Open,High,Low,Close,Volume (BTC),Volume (Currency),Weighted_Price
1325317920,4.39,4.39,4.39,4.39,0.45558087,2.000000193,4.39
1325317980,Nan,Nan,Nan,Nan,Nan,Nan
1325318040,Nan,Nan,Nan,Nan,Nan,Nan
1325318100,Nan,Nan,Nan,Nan,Nan,Nan
1325318160,Nan,Nan,Nan,Nan,Nan,Nan
1325318220,Nan,Nan,Nan,Nan,Nan,Nan
1325318280,Nan,Nan,Nan,Nan,Nan,Nan
1325318340,Nan,Nan,Nan,Nan,Nan,Nan
1325318400,Nan,Nan,Nan,Nan,Nan,Nan
1325318460,Nan,Nan,Nan,Nan,Nan,Nan
1325318520,Nan,Nan,Nan,Nan,Nan,Nan
1325318580,Nan,Nan,Nan,Nan,Nan,Nan
1325318640,Nan,Nan,Nan,Nan,Nan,Nan
1325318700,Nan,Nan,Nan,Nan,Nan,Nan
1325318760,Nan,Nan,Nan,Nan,Nan,Nan
1325318820,Nan,Nan,Nan,Nan,Nan,Nan
1325318880,Nan,Nan,Nan,Nan,Nan,Nan
1325318940,Nan,Nan,Nan,Nan,Nan,Nan
1325318960,Nan,Nan,Nan,Nan,Nan,Nan
1325319060,Nan,Nan,Nan,Nan,Nan,Nan
1325319120,Nan,Nan,Nan,Nan,Nan,Nan
1325319180,Nan,Nan,Nan,Nan,Nan,Nan
1325319240,Nan,Nan,Nan,Nan,Nan,Nan
1325319300,Nan,Nan,Nan,Nan,Nan,Nan
1325319360,Nan,Nan,Nan,Nan,Nan,Nan
1325319420,Nan,Nan,Nan,Nan,Nan,Nan
1325319480,Nan,Nan,Nan,Nan,Nan,Nan
1325319540,Nan,Nan,Nan,Nan,Nan,Nan
1325319600,Nan,Nan,Nan,Nan,Nan,Nan
1325319660,Nan,Nan,Nan,Nan,Nan,Nan
1325319720,Nan,Nan,Nan,Nan,Nan,Nan
1325319780,Nan,Nan,Nan,Nan,Nan,Nan
1325319840,Nan,Nan,Nan,Nan,Nan,Nan
1325319900,Nan,Nan,Nan,Nan,Nan,Nan
1325319960,Nan,Nan,Nan,Nan,Nan,Nan
1325320020,Nan,Nan,Nan,Nan,Nan,Nan
1325320080,Nan,Nan,Nan,Nan,Nan,Nan
1325320140,Nan,Nan,Nan,Nan,Nan,Nan
1325320200,Nan,Nan,Nan,Nan,Nan,Nan
1325320260,Nan,Nan,Nan,Nan,Nan,Nan
1325320320,Nan,Nan,Nan,Nan,Nan,Nan
1325320380,Nan,Nan,Nan,Nan,Nan,Nan
1325320440,Nan,Nan,Nan,Nan,Nan,Nan
1325320500,Nan,Nan,Nan,Nan,Nan,Nan
1325320560,Nan,Nan,Nan,Nan,Nan,Nan
1325320620,Nan,Nan,Nan,Nan,Nan,Nan
1325320680,Nan,Nan,Nan,Nan,Nan,Nan
1325320740,Nan,Nan,Nan,Nan,Nan,Nan
1325320800,Nan,Nan,Nan,Nan,Nan,Nan
1325320860,Nan,Nan,Nan,Nan,Nan,Nan
1325320920,Nan,Nan,Nan,Nan,Nan,Nan
1325320980,Nan,Nan,Nan,Nan,Nan,Nan
```

Step 4: Creating our SageMaker instance

Give the “notebook instance name”, “notebook instance type” , enable “IAM role” and keep the root access “enable”.

- Click on “Create Notebook Instance” and check the status - “InService”

Permissions and encryption

IAM role
Notebook instances require permissions to call other services including SageMaker and S3. Choose a role or let us create a role with the **AmazonSageMakerFullAccess** IAM policy attached.

AmazonSageMaker-ExecutionRole-20210429T114782

Root access - optional
 Enable - Give users root access to the notebook
 Disable - Don't give users root access to the notebook
Lifecycle configurations always have root access

Encryption key - optional
Encrypt your notebook data. Choose an existing KMS key or enter a key's ARN.
No Custom Encryption

Network - optional

Git repositories - optional

Tags - optional

Create notebook instance

Name	Instance	Creation time	Status	Actions
Introtocloudproject	ml.t2.medium	May 04, 2021 16:18 UTC	InService	Open Jupyter Open JupyterLab
Introtocloud	ml.t2.medium	May 03, 2021 19:59 UTC	Stopped	Start

The screenshot shows the Amazon SageMaker Studio interface. On the left, there is a sidebar with various navigation options like Dashboard, Search, Images, Ground Truth, Notebook, Processing, Training, Inference, Edge Manager, Augmented AI, and AWS Marketplace. The main area is titled 'Introtocloudproject' and contains two sections: 'Notebook instance settings' and 'Git repositories'. Under 'Notebook instance settings', there is a table with the following data:

Name	Status	Notebook instance type
Introtocloudproject	InService	m1t2.medium

Under 'Git repositories', there is a table with the following columns: Name, Repository URL, and Type. A message indicates 'There are currently no resources.'

- Open the previously created **Sagemaker Jupyter Notebook**.

Url for jupyter notebook:

<https://introtocloudproject.notebook.us-east-1.sagemaker.aws/tree>

The screenshot shows the Jupyter Notebook interface. At the top, there are tabs for Files, Running, Clusters, SageMaker Examples, and Conda. Below that, there is a list of files with the following details:

Name	Last Modified	File size
bitcoin-close-value-price-prediction.ipynb	Running 15 minutes ago	956 kB

Data Analysis and Cleaning:

Step 1: Exploratory Data Analysis

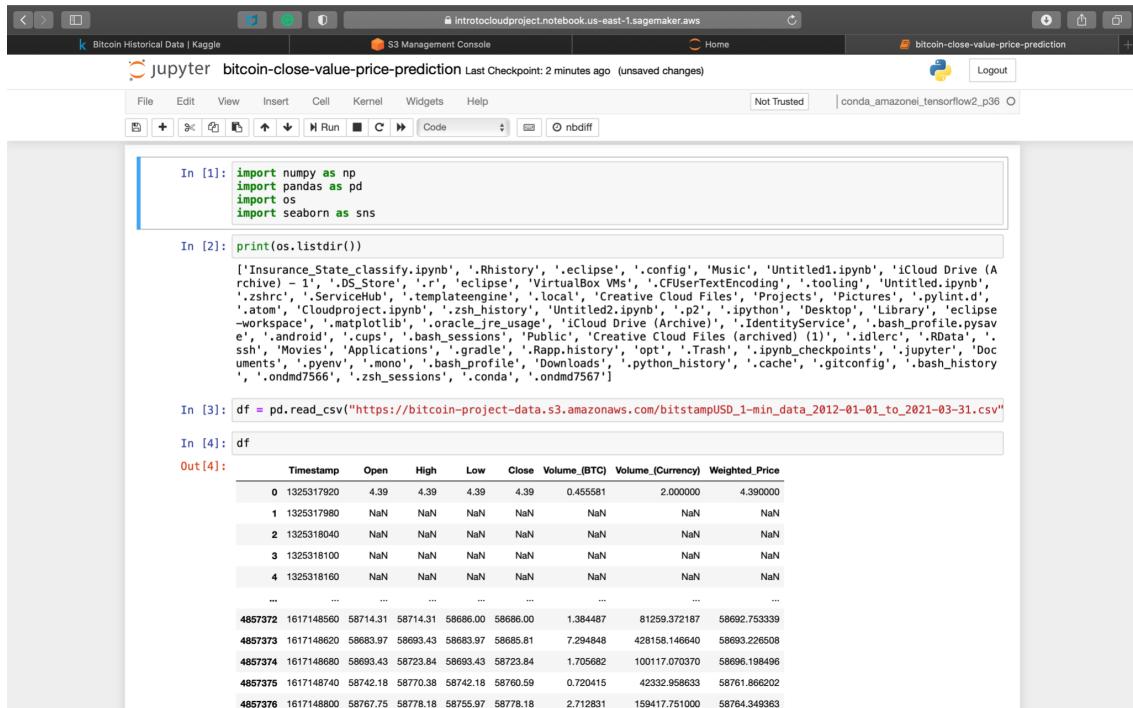
This is the standard first step for any machine learning project. There were large amounts of junk data found in the csv file given by kaggle. The first step is to examine all the data through a pandas dataframe. While using Trend Analysis we figure it out our data analysis by importing matplotlib, scipy, pandas, numpy, seaborn, ARIMA model frameworks.

Data features to be considered for prediction:

The opening and closing price on a particular day is given by the **Open** and **Close** columns.

The highest and the lowest price on a particular day is given by **High** and **Low** columns.

The total volume of traded on a particular day is defined by **Volume** column and The **Weighted price** is a trading benchmark which is mostly used by traders. Both the trend and value of a security insight info is given by the weighted price to traders. We are predicting here the close value of Bitcoin price using deep learning models.



The screenshot shows a Jupyter Notebook interface running on AWS Sagemaker. The browser tabs include 'Bitcoin Historical Data | Kaggle', 'S3 Management Console', 'Home', and the current notebook 'bitcoin-close-value-price-prediction'. The notebook has a single cell with the following code and output:

```
In [1]: import numpy as np
import pandas as pd
import os
import seaborn as sns

In [2]: print(os.listdir())
['Insurance_State_classify.ipynb', 'Rhhistory', 'eclipse', '.config', 'Music', 'Untitled1.ipynb', 'iCloud Drive (Archive) - 1', '.DS_Store', 'r', 'eclipse', 'VirtualBox VMs', '.CFUserTextEncoding', '.tooling', 'Untitled.ipynb', '.zshrc', '.ServiceHub', '.templateengine', '.local', 'Creative Cloud Files', 'Projects', 'Pictures', '.pylint.d', '.atom', 'Cloudproject.ipynb', '.zsh_history', 'Untitled2.ipynb', '.p2', '.ipython', 'Desktop', 'Library', 'eclipse-workspace', '.matplotlib', '.oracle_jre_usage', 'iCloud Drive (Archive)', '.IdentityService', '.bash_profile.pyav', 'e', '.android', '.cups', '.bash_sessions', 'Public', 'Creative Cloud File (Archived) (1)', '.idler', '.RData', '.ssh', '.Movie', 'AppIcons', '.gradle', '.app.history', 'opt', '.Trash', '.ipython_checkpoints', 'jupyter', 'Documents', '.pyenv', '.mono', '.bash_profile', 'Downloads', '.python_history', '.cache', '.gitconfig', '.bash_history', '.ondemand7566', '.zsh_sessions', '.conda', '.ondemand7567']

In [3]: df = pd.read_csv("https://bitcoin-project-data.s3.amazonaws.com/bitstampUSD_1-min_data_2012-01-01_to_2021-03-31.csv")
In [4]: df
```

The output of cell [4] is a Pandas DataFrame with the following schema and data:

	Timestamp	Open	High	Low	Close	Volume_(BTC)	Volume_(Currency)	Weighted_Price
0	1325317920	4.39	4.39	4.39	4.39	0.455581	2.000000	4.390000
1	1325317980	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	1325318040	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	1325318100	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	1325318160	NaN	NaN	NaN	NaN	NaN	NaN	NaN
...
4857372	1617148560	58714.31	58714.31	58686.00	58686.00	1.384487	81259.372187	58692.753339
4857373	1617148620	58683.97	58693.43	58683.97	58685.81	7.294848	428158.146640	58693.226508
4857374	1617148680	58693.4	58723.84	58693.4	58723.84	1.705682	100117.070370	58696.198496
4857375	1617148740	58742.18	58770.38	58742.18	58760.59	0.720415	42332.958633	58761.866202
4857376	1617148800	58767.7	58778.18	58755.97	58778.18	2.712831	159417.751000	58764.349363

The screenshot shows a Jupyter Notebook interface with several cells of code and their corresponding outputs.

```

In [3]: df = pd.read_csv("https://bitcoin-project-data.s3.amazonaws.com/bitstampUSD_1-min_data_2012-01-01_to_2021-03-31.csv")
In [4]: df
Out[4]:
   Timestamp  Open  High  Low  Close  Volume_(BTC)  Volume_(Currency)  Weighted_Price
0  1325317920  4.39  4.39  4.39  0.45581  2.000000          4.390000
1  1325317980  NaN  NaN  NaN  NaN  NaN          NaN          NaN
2  1325318040  NaN  NaN  NaN  NaN  NaN          NaN          NaN
3  1325318100  NaN  NaN  NaN  NaN  NaN          NaN          NaN
4  1325318160  NaN  NaN  NaN  NaN  NaN          NaN          NaN
...
4857372  1617148560  58714.31  58714.31  58686.00  58686.00  1.384487  81259.372187  58692.753339
4857373  1617148620  58683.97  58683.97  58685.81  7.294848  428158.146640  58693.226508
4857374  1617148680  58693.43  58693.43  58723.84  58693.43  1.705682  100117.070370  58696.198496
4857375  1617148740  58742.18  58770.38  58742.18  58760.59  0.720415  42332.558633  58761.866202
4857376  1617148800  58767.75  58778.18  58755.97  58778.18  2.712831  159417.751000  58764.349363
4857377 rows x 8 columns

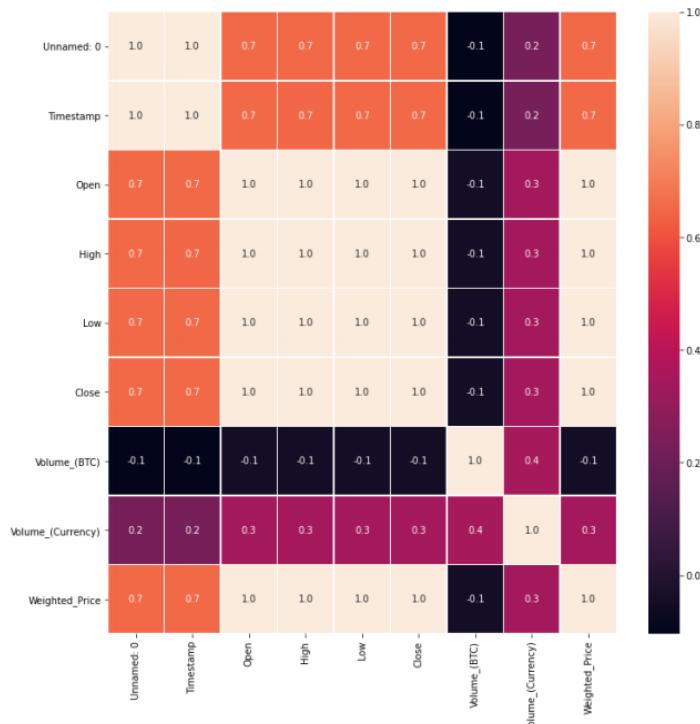
In [5]: df.isnull().sum()
Out[5]:
Timestamp      0
Open        1243608
High        1243608
Low         1243608
Close       1243608
Volume_(BTC) 1243608
Volume_(Currency) 1243608
Weighted_Price 1243608
dtype: int64

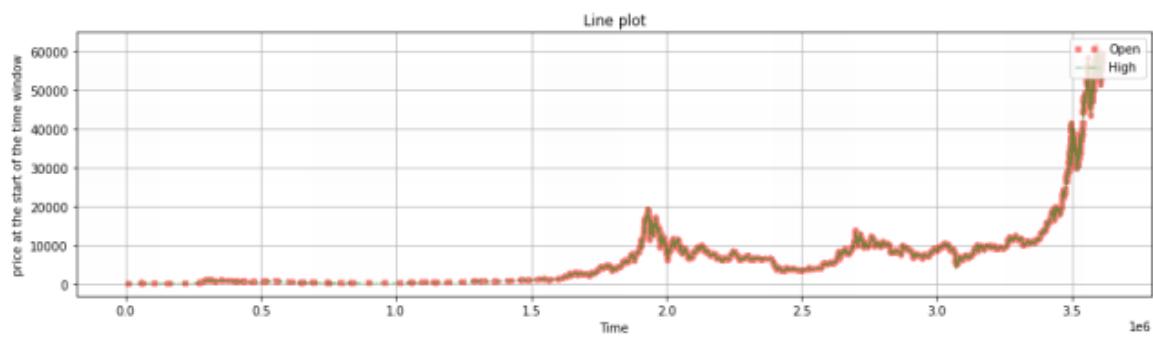
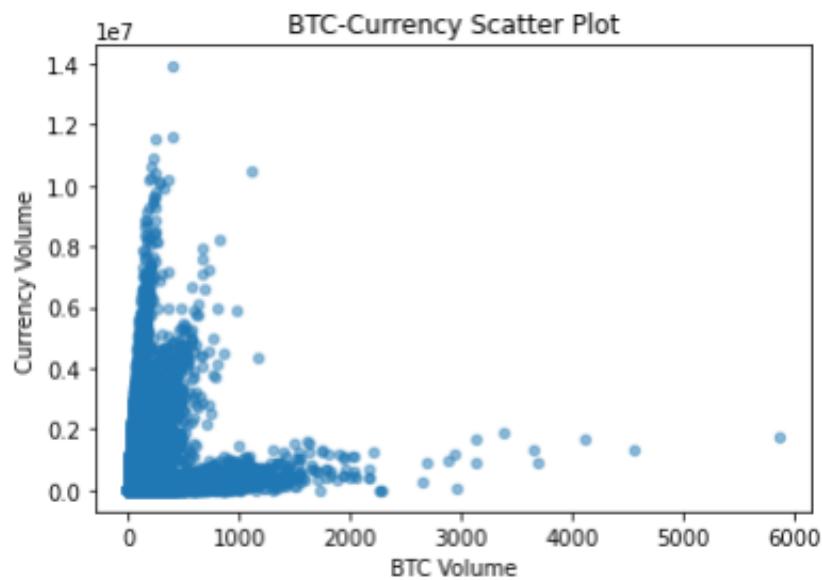
In [6]: df.dropna(inplace= True)

```

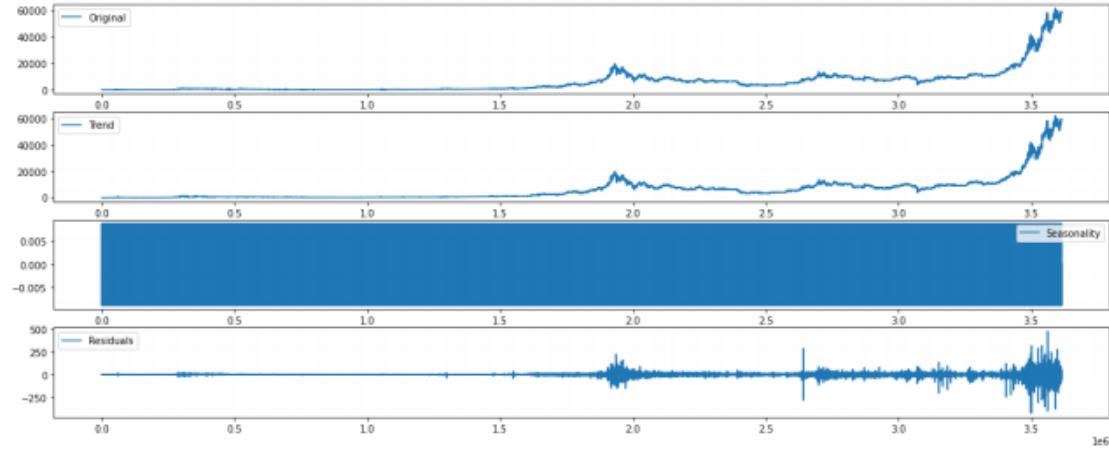
Data Visualization

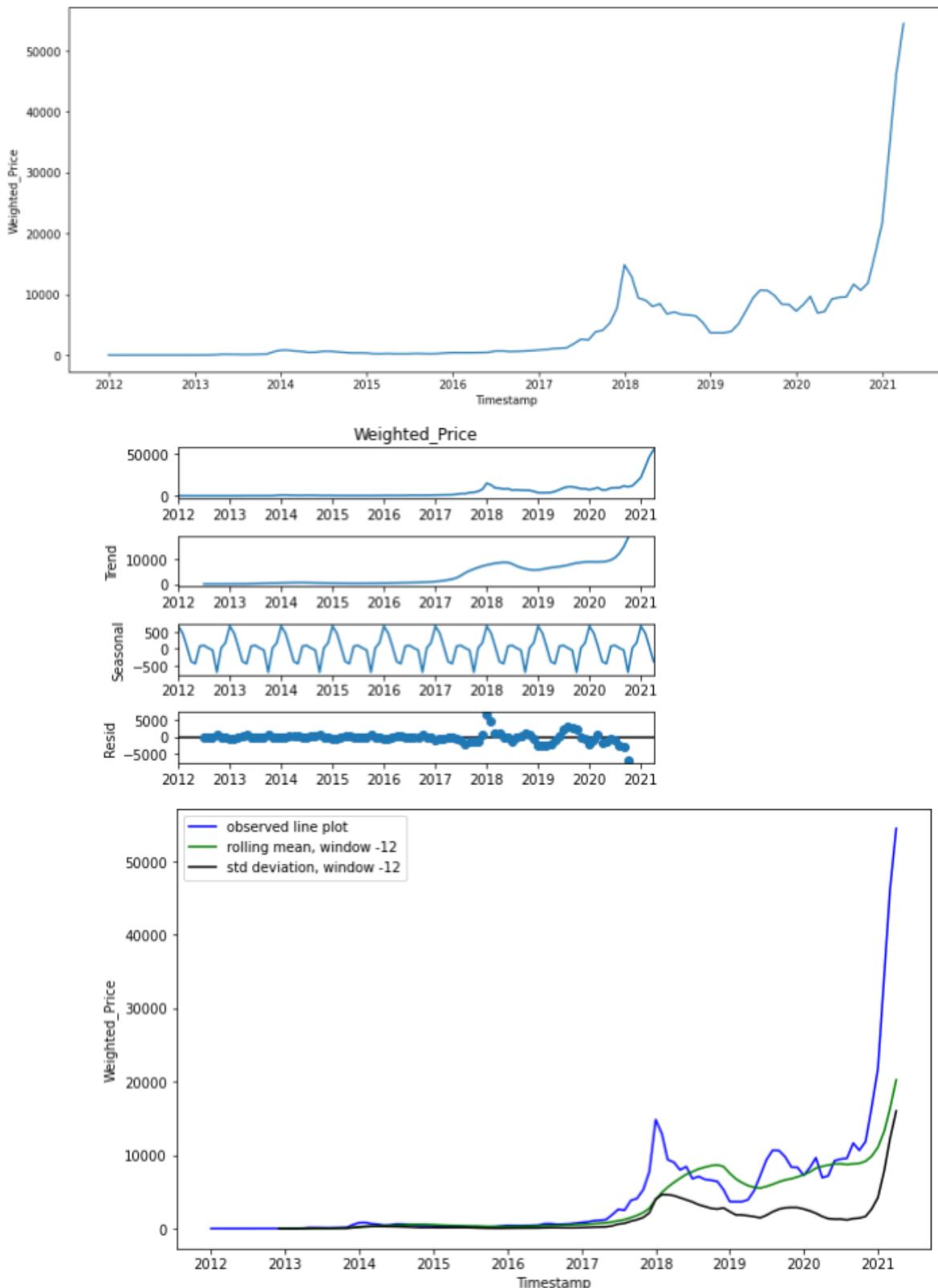
Following these first steps of examining the data we then began to **visualize** different aspects of the data in order to have a better understanding of the **various correlations that existed within the data that can impact the predicted price**.

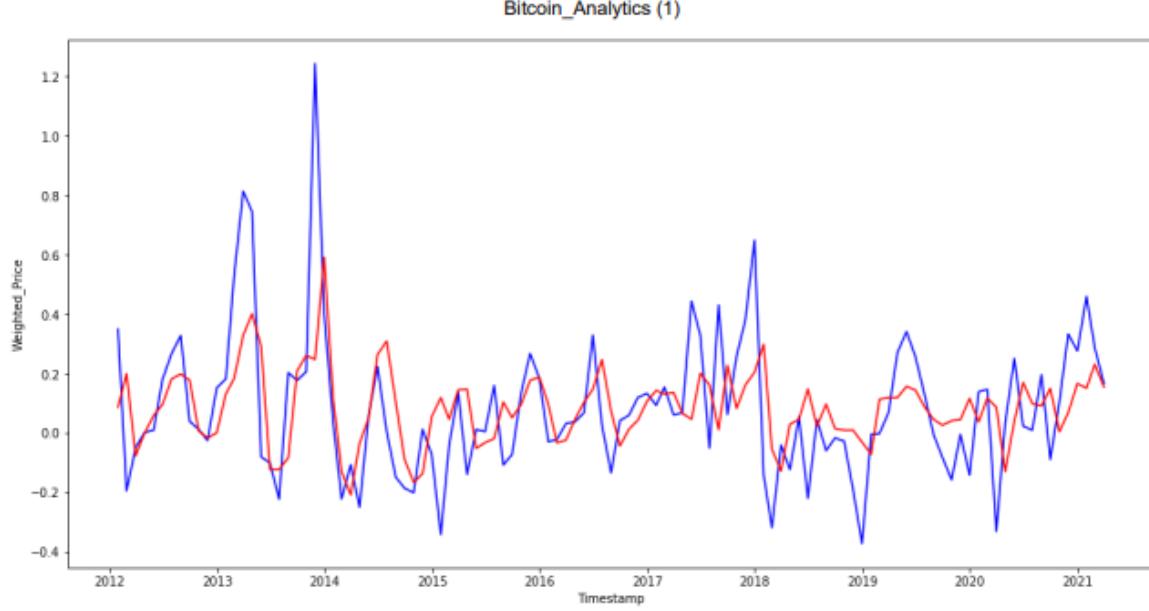
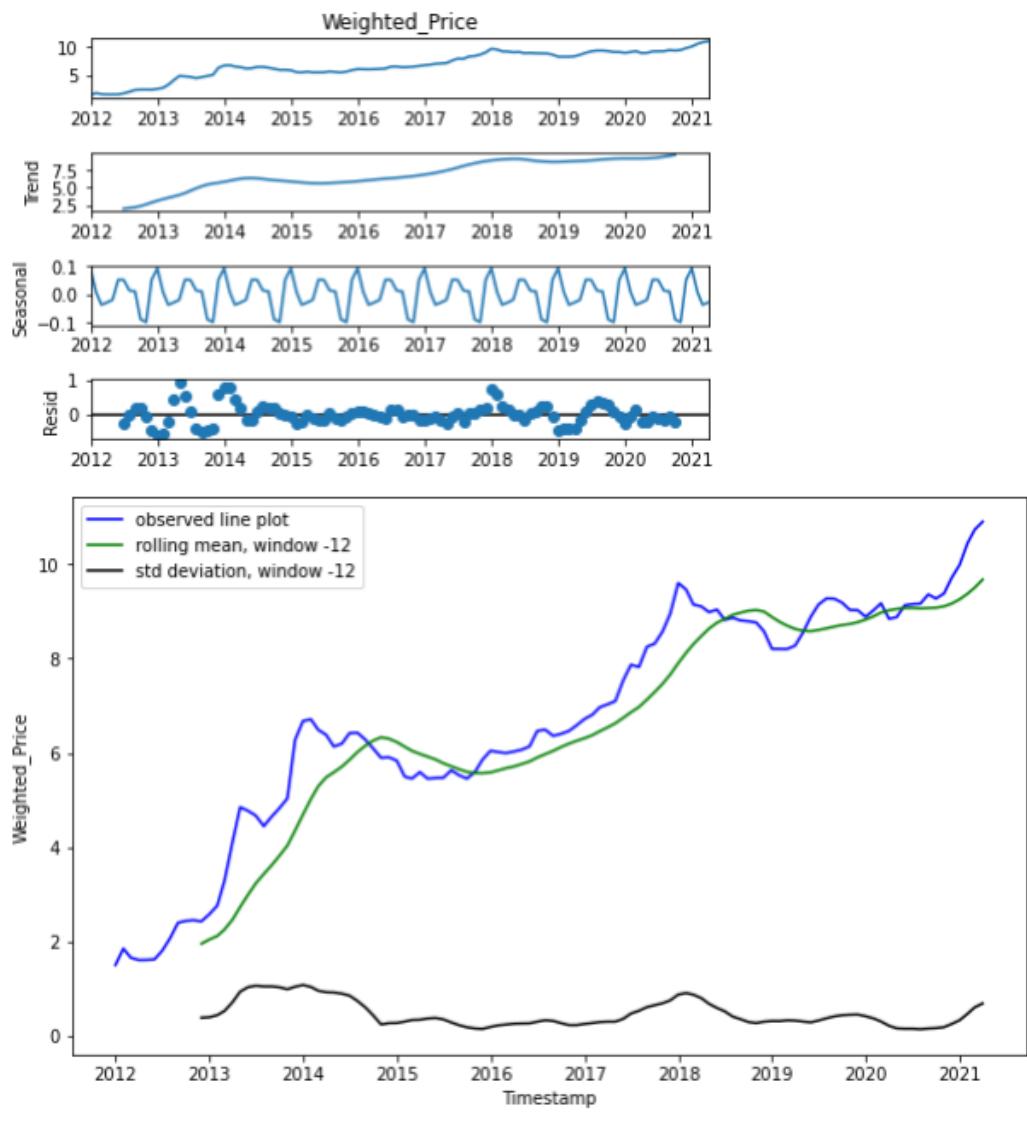


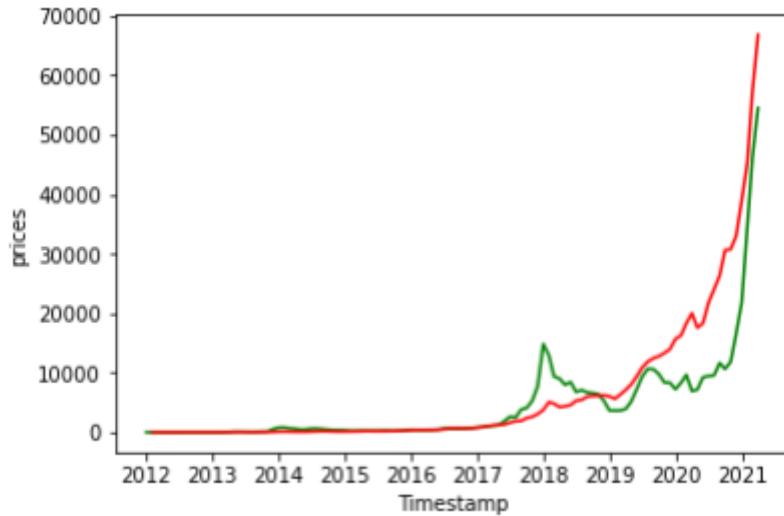


Decomposition of Prices Data









After doing our data analysis it was determined that we should use all the data types for our model as they all provided valuable information for training our model. We broke it down into `x_train`, `x_test`, `y_train`, `y_test`. The `x` values were all the data points besides the closing price and the `y` values were the closing price.

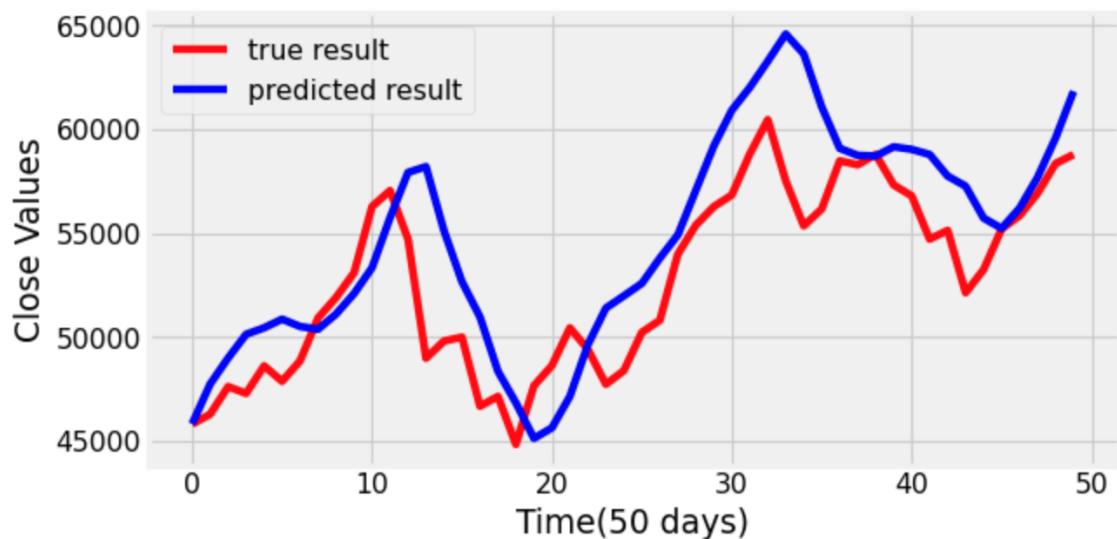
Creating the Models

LSTM Model

The first model we used was LSTM, this model proved very effective at predicting the price of bitcoin.

Using LSTM

```
1 from sklearn.metrics import mean_absolute_error
2 from keras.models import Sequential
3 from keras.layers import Dense, LSTM, Dropout, Flatten
4
5 Lstm_m=Sequential()
6
7 Lstm_m.add(LSTM(10,input_shape=(None,1)))
8
9 Lstm_m.add(Activation('relu'))
10
11 Lstm_m.add(Dense(1))
12
13 Lstm_m.compile(loss="mean_squared_error",optimizer="adam")
14
15 Lstm_m.fit(x_train,y_train,epochs=50,batch_size=32)
```



As it can be seen by the above graph our model was very close to the correct value over the course of the testing data.

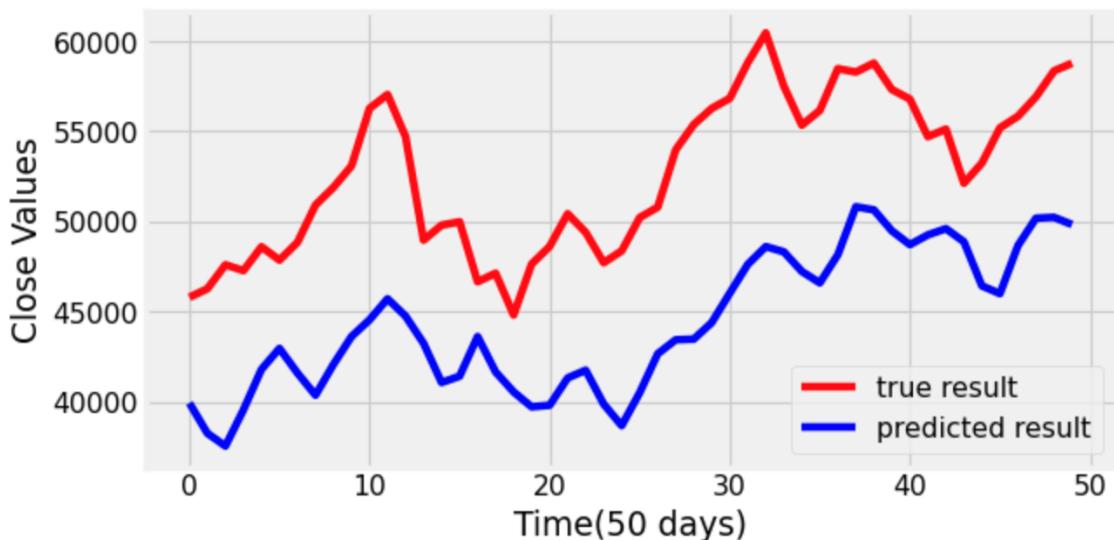
Simple RNN model

RNN model

```

1 model=Sequential()
2
3 model.add(SimpleRNN(128,return_sequences=True,input_shape=(x_train.shape[1],1)))
4 model.add(Activation('relu'))
5 model.add(Dropout(0.25))
6
7 model.add(SimpleRNN(256,return_sequences=True,activation='relu'))
8 model.add(Dropout(0.20))
9
10 model.add(SimpleRNN(256,return_sequences=True,activation='relu'))
11 model.add(Dropout(0.30))
12
13 model.add(SimpleRNN(512,return_sequences=True,activation='relu'))
14 model.add(Dropout(0.45))
15
16 model.add(SimpleRNN(256,return_sequences=True,activation='relu'))
17 model.add(Dropout(0.20))
18
19 model.add(SimpleRNN(128,return_sequences=True,activation='relu'))
20 model.add(Dropout(0.20))
21
22 model.add(Flatten())
23
24 model.add(Dense(1))
25
26
27 model.compile(optimizer='adam',loss='mean_squared_error')
28 model.fit(x_train,y_train,epochs=50,batch_size=64)
29

```



As it can be seen above the RNN model was not nearly as effective as the LSTM model. This was for a simple reason. As RNN trains further and further it loses memory of early training rounds. This means that it loses context for data and causes inaccurate predictions

Deletion of Environment:

Deleting S3 bucket and AWS Sagemaker:

Deleting the dataset and emptying bucket for deletion:

The screenshot shows the AWS S3 Management Console interface. On the left, the sidebar has sections for Buckets, Storage Lens, Feature spotlight, and AWS Marketplace for S3. The main area shows the 'bitcoin-project-data' bucket under 'Amazon S3'. The bucket is publicly accessible. The 'Objects' tab is selected, showing one object: 'bitstampUSD_1-min_data_2012-01-01_to_2021-03-31.csv'. The object details show it's a CSV file from May 3, 2021, at 21:32:44 UTC, with a size of 302.8 MB and a storage class of Standard.

The screenshot shows the 'Delete objects' dialog within the AWS S3 Management Console. It lists the same object as the previous screenshot. Below the list, there's a section titled 'Delete objects?' with a text input field containing the word 'delete'. At the bottom are 'Cancel' and 'Delete objects' buttons.

The screenshot shows the AWS S3 Management Console interface. At the top, there's a green banner indicating "Successfully deleted objects". Below it, a modal window titled "Delete objects: status" displays summary information: "Source s3://bitcoin-project-data", "Successfully deleted 1 object, 302.8 MB", and "Failed to delete 0 objects". Under the "Failed to delete" tab, it says "(0)". A table below shows "No objects failed to delete." At the bottom of the page, there are standard AWS navigation links like Feedback, English (US), Privacy Policy, Terms of Use, and Cookie preferences.

Emptying bucket:

The screenshot shows the AWS S3 Management Console interface. The URL in the address bar is "Amazon S3 > bitcoin-project-data > Empty bucket". The main content area is titled "Empty bucket" and contains a warning message: "Emptying the bucket deletes all objects in the bucket and cannot be undone. Objects added to the bucket while the empty bucket action is in progress might be deleted." It also suggests creating a lifecycle rule for efficiency. Below this, a section asks "Permanently delete all objects in bucket 'bitcoin-project-data'?". A text input field contains the text "permanently delete". At the bottom right are "Cancel" and "Empty" buttons. At the very bottom of the page, there are standard AWS navigation links.

Successfully emptied bucket "bitcoin-project-data"
View details below. If you want to delete this bucket, use the [delete bucket configuration](#).

Empty bucket: status

The details below are no longer available after you navigate away from this page.

Summary		
Source s3://bitcoin-project-data	Successfully deleted 4 objects, 605.6 MB	Failed to delete 0 objects

Failed to delete (0)

Name	Prefix	Version ID	Type	Last modified	Size	Error
No failed object deletions						

Feedback English (US) ▾ © 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use Cookie preferences

Services ▾ Search for services, features, marketplace products, and docs [Option+S] Workbench S3 Management Console +

Amazon S3 > bitcoin-project-data > Delete bucket

Delete bucket

⚠ • Deleting a bucket cannot be undone.
• Bucket names are unique. If you delete a bucket, another AWS user can use the name.
[Learn more](#)

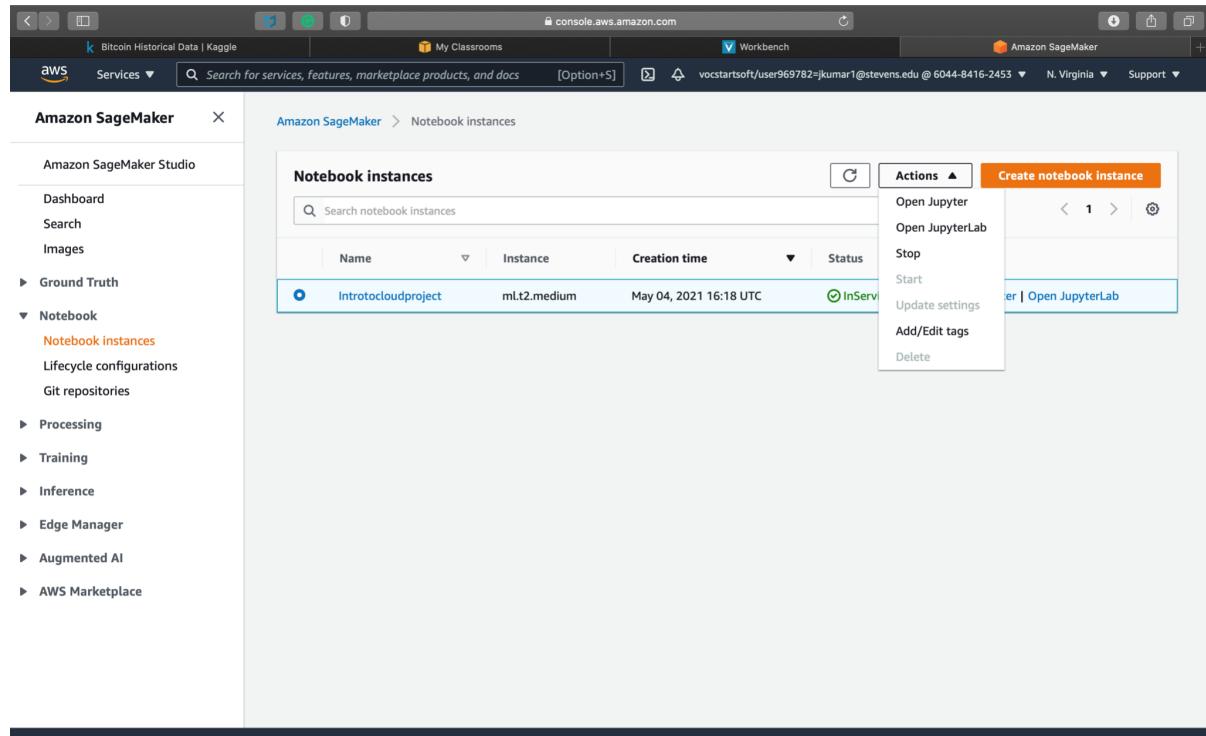
Delete bucket "bitcoin-project-data"?

To confirm deletion, enter the name of the bucket in the text input field.

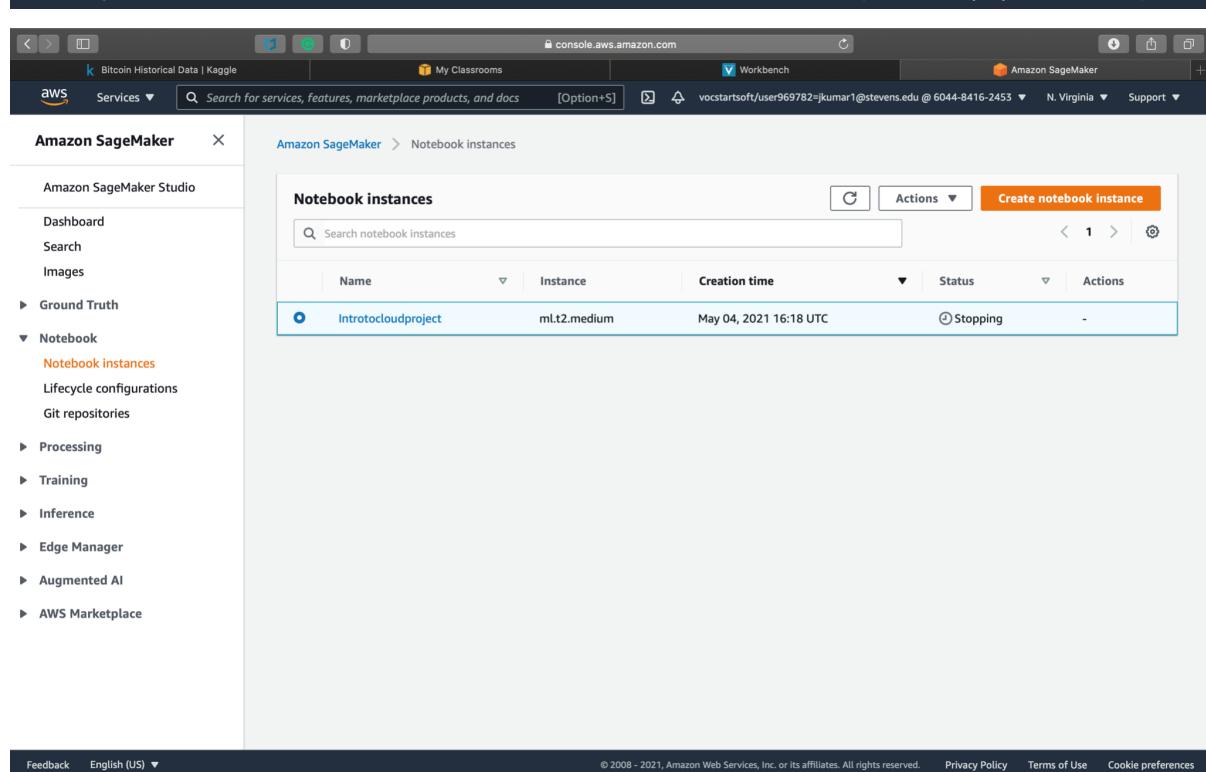
[Cancel](#) [Delete bucket](#)

The screenshot shows the AWS S3 Management Console interface. At the top, there are tabs for Services, Workbench, and S3 Management Console. The main navigation bar shows 'Amazon S3 > bitcoin-project-data > Delete bucket'. Below this, a modal window titled 'Delete bucket' displays a warning message: 'Deleting a bucket cannot be undone.' and 'Bucket names are unique. If you delete a bucket, another AWS user can use the name.' A 'Learn more' link is also present. The next section is titled 'Delete bucket "bitcoin-project-data"?'. It contains a text input field with the value 'bitcoin-project-data'. At the bottom of the modal are 'Cancel' and 'Delete bucket' buttons. After performing the deletion, a success message 'Successfully deleted bucket "bitcoin-project-data"' is displayed in a green banner. The main S3 dashboard shows an account snapshot with 605.6 MB total storage, 3 objects, and an average object size of 201.9 MB. The 'Buckets' section shows a table with one row: 'Name' (No buckets), 'AWS Region' (No buckets), 'Access' (No buckets), and 'Creation date' (No buckets). A 'Create bucket' button is available. The left sidebar includes sections for Buckets, Access Points, Object Lambda Access Points, Batch Operations, Access analyzer for S3, Block Public Access settings, Storage Lens, Dashboards, AWS Organizations settings, Feature spotlight, and AWS Marketplace for S3.

Deleting the AWS sagemaker notebook instance:



The screenshot shows the AWS SageMaker Studio interface. On the left, a sidebar menu includes options like Dashboard, Search, Images, Ground Truth, Notebook (selected), Processing, Training, Inference, Edge Manager, Augmented AI, and AWS Marketplace. The main content area displays a table titled "Notebook instances" with one entry: "Introtocloudproject" (ml.t2.medium, May 04, 2021 16:18 UTC, InService). A context menu is open over this entry, showing options: Open Jupyter, Open JupyterLab, Stop, Start, Update settings, Add/Edit tags, and Delete.



The second screenshot shows the same interface after the "Stop" action has been initiated. The "Actions" dropdown now includes "Starting" and "Stopped". The "Introtocloudproject" row in the table shows its status as "Stopping".

- Waiting for instance to stop and then we can delete the instance:

The screenshot shows two screenshots of the AWS SageMaker console side-by-side.

Left Screenshot (Notebook instances page):

- The URL is `console.aws.amazon.com`.
- The sidebar shows the navigation menu: Services > Amazon SageMaker > Notebook instances.
- The main content shows a table titled "Notebook instances" with one row:

Name	Instance	Creation time	Status	Actions
Introtocloudproject	mlt2.medium	May 04, 2021 16:18 UTC	Stopped	Start

Right Screenshot (Introtocloudproject instance details):

- The URL is `console.aws.amazon.com`.
- The sidebar shows the navigation menu: Services > Amazon SageMaker > Notebook instances > Introtocloudproject.
- The main content shows the "Introtocloudproject" instance settings:

Name	Status	Notebook instance type
Introtocloudproject	Stopped	mlt2.medium

 Other details shown include ARN, Creation time, Last updated, and Volume Size.
- Below the settings, there are sections for "Git repositories" and "Permissions and encryption".

The screenshot shows the AWS SageMaker console with the URL [console.aws.amazon.com](https://console.aws.amazon.com/sagemaker/). The left sidebar shows navigation options like Dashboard, Search, Images, Ground Truth, Notebook, Processing, Training, Inference, Edge Manager, Augmented AI, and AWS Marketplace. The main content area shows a notebook instance named "Introtocloudproject". A modal dialog titled "Delete Introtocloudproject?" is open, asking if the user wants to permanently delete the notebook instance. The dialog includes fields for Name (Introtocloudproject), Status (Stopped), ARN (arn:aws:sagemaker:us-east-1:604484162453:notebook-instance/introtocloudproject), Lifecycle (-), and a note stating "This will permanently delete your notebook instance and cannot be undone. This may affect other resources." Buttons for "Delete" and "Cancel" are at the bottom. To the right of the modal, the notebook instance settings show the name "Introtocloudproject", status "Stopped", ARN, and a "Notebook Instance type" of "mlt2.medium". Below the settings is a section for "Git repositories" with a message "There are currently no resources." At the bottom of the page is a "Permissions and encryption" section.

This screenshot shows the same AWS SageMaker console interface after the deletion. The notebook instance "Introtocloudproject" is now listed with a status of "Deleting". The ARN field shows the ARN of the deleted instance. The "Notebook instance type" is still listed as "mlt2.medium". The "Git repositories" section remains empty. The "Permissions and encryption" section is also present at the bottom.

The screenshot shows the Amazon SageMaker console interface. On the left, there is a navigation sidebar with the following menu items:

- Amazon SageMaker Studio
- Dashboard
- Search
- Images
- ▶ Ground Truth
- ▼ Notebook
 - Notebook instances**
 - Lifecycle configurations
 - Git repositories
- ▶ Processing
- ▶ Training
- ▶ Inference
- ▶ Edge Manager
- ▶ Augmented AI
- ▶ AWS Marketplace

The main content area is titled "Notebook instances" and displays a table with one row of data:

Name	Instance	Creation time	Status	Actions
Introtocloudproject	mL.t2.medium	May 04, 2021 16:18 UTC	Deleting	-

At the bottom of the page, there are links for Feedback, English (US), Privacy Policy, Terms of Use, and Cookie preferences.

This screenshot is identical to the one above, but it shows a different state. The table in the main content area now displays the message "There are currently no resources." indicating that no notebook instances have been created.

Logging out:

The screenshot shows the AWS SageMaker Studio interface. On the left is a navigation sidebar titled "Amazon SageMaker" with sections for Dashboard, Search, Images, Ground Truth, Notebook (selected), Processing, Training, Inference, Edge Manager, Augmented AI, and AWS Marketplace. The main content area is titled "Notebook instances" and shows a search bar and a table with columns for Name, Instance, and Creation time. A message at the bottom of the table says "There are currently no resources". On the right side, there is a dark sidebar with account information: "Federated Login: vocstartsoft/user969782=jkumar1@stevens.edu", "My Account: 604484162453", "My Organization", "My Service Quotas", "My Billing Dashboard", and "Switch Roles". At the bottom of this sidebar is a "Sign Out" button. The top of the screen shows the AWS logo, services like My Classrooms, Workbench, and Amazon SageMaker, and a user profile with the email "vocstartsoft/user969782=jkumar1@stevens.edu". The URL in the address bar is "console.aws.amazon.com".

Conclusion:

We have used AWS S3 bucket for storage of data, AWS IAM for encryption, AWS sagemaker for creating the project and using the Deep learning libraries, AWS sagemaker Jupyter notebook instance for creating Python files and training our model with various Deep learning algorithms.

As we see LSTM and RNN diagrams on Page 20 and 21, the LSTM is all the more firmly related with the real value expectation than the RNN model. At the point when the preparation information spreads in reverse then it gets more vulnerable and more fragile, when it gets to those neurons that address more seasoned information focused presently arrangement it has no juice to change them appropriately. This issue is called Vanishing Gradient.

LSTM is also one of the kind or say extensions of RNN. The difference is that it keeps the vital data around the past and overlooks the insignificant data. It won't be devoured by pointless data when gradient back-propagates. We are able to increment the number of epochs to refine our show execution and it also improves the prediction of close value of Bitcoin price results overall.

We can say that, when we move from RNN to LSTM (Long Short-Term Memory), we are presenting more and additional controlling handles, which control the stream and blending of Inputs according to prepared Weights. What's more, hence, acquiring greater adaptability in controlling the yields. Along these lines, **LSTM gives us the most Control-capacity and hence, Better Results and when used with the AWS S3 bucket provides faster data fetching and AWS sagemaker libraries gives faster computation.**

AWS Sagemaker has been an incredible arrangement for our predictive analysis algorithm to achieve a really start to finish ML arrangement. It dealt with abstracting a huge load of programming improvement abilities important to achieve the errand while being exceptionally powerful and adaptable and practical. Above all, it helped us focus on the core Deep Learning experiments and supplements the remaining necessary skills with easy abstracted tools similar to our existing workflow.

References:

1. <https://www.kaggle.com/mczielinski/bitcoin-historical-data><https://aws.amazon.com/blogs/machine-learning/training-debugging-and-running-time-series-forecasting-models-with-the-gluonts-toolkit-on-amazon-sagemaker/>
2. https://github.com/aws/amazon-sagemaker-examples/tree/master/introduction_to_amazon_algorithms
3. https://www.linkedin.com/learning/learning-amazon-sagemaker?trk=share_ios_course_learning&shareId=A769egtZRB6UhHuZGBIF6A
4. <https://www.linkedin.com/learning/learning-bitcoin-and-other-cryptocurrencies/how-bitcoin-works?u=56742337>