**Stock Analysis and Forecast –**

**ML project in Python.**

cid:image002.png@01D248F5.A145CFA0

Nataliya Lisjö

Mjukvaruutvecklare inom Industriell IoT 2020

**Utförd: 2022**

Innehållsförteckning

Abstract 1

Preface 2

Introduction 3

Bakground 4

Purpose and question 4

Metod 4

Main part 5

How does predictive analytics work? 5

Dataset 6

Reading and Analyzing the Data 7

Normalizing the data 8

How to build a LSTM model in Keras with Tensorflow 10

Importing the Right Modules 10

What is LSTM? 11

Understanding Input and Output shapes in LSTM | Keras 12

Creating a LSTM Neural Network with Keras 13

Parameters that we can change if we want to do model better 16

Decision Tree Regression and Linear Regression 18

Dashboard 20

Problems 21

Analys av resultat 22

Final discussion 23

Bibliography 24

# Abstract

*Machine Learning is known for Predictive Modeling. In other words, we can predict the value of a dependent variable . Generally, we use predictive modeling or predictive analytics in order to forecast future outcomes.*

*Machine learning have several methods to predict result . I want to compare them which one give the best result. For that I will use neural network LSTM , Decision tree and Linear regression methods*



# Preface

I became interesting in Machine Learning and Data Analyses after our curs " Integrering och overvakning" . Sometimes I buy and sell Bitcoin or other cryptocurrency and decided to try test if Machine Learnings Predictive Modeling work with it .

Diagram

Description automatically generated

# Introduction .

*In my thesis work I got idea to start experimenting hands-on Machine Learning projects on Stock Prices Predictor. Today are companies on the lookout for software that can monitor and analyze the company performance and predict future prices of various stocks.*

*Machine learning has significant applications in the stock price prediction. In this machine learning project, we will be talking about predicting the prices on stocks. This is a very complex task and has uncertainties.*

*We will develop this project into two parts:*

1. *First, we will learn how to predict stock price using the LSTM neural network.*
2. *We will use Decision Tree Regression and Linear Regression methods to predict and compare result with LSTM .*
3. *Then we will build a dashboard using Plotly dash for analysis of Tesla stocks.*

*This thesis work will show you a market analysis and forecasting tool implemented as a Python code in VisualStudio Code with help of the keras , pandas , numpy and another library*

## Background

I became interesting in Machine Learning and Data Analyses after our curs " Integrering och overvakning" . Sometimes I buy and sell Bitcoin or other cryptocurrency and decided to try test if Machine Learnings Predictive Modeling work with it .

## Purpose and question

I want to compare several methods to predict result .Which one give the best result. For that I will use LSTM , Decision tree and Linear regression methods

## Metod

To get result I will work with the most widely used predictive models In Machine Learning like Decision tree Regression and Linear regression and neural network LSTM.

* I will take Data from Yahoo Finance, Normalize and prepare it for usage. Clean the data outputs.
* Then I will split prepared data on training and test sets .
* Create a model from training set
* Test the created model on test set
* Show result
* I will try different approaches.
* When I find a model that accurately forecasts the load, I move it into Dash system and test it in Tesla stocks. I will try the model against new data to see how well it performs.

That I will do with different methods and show result in Graph for we can see det.

During my thesis work I will explain process and theoretical questions regarding this process.

# Main Part

**How does predictive analytics work?**

The algorithms perform the data mining and statistical analysis, determining trends and patterns in data.

The most widely used predictive models are:

* **Decision trees:** Decision trees are a simple, but powerful form of multiple variable analysis. They are produced by algorithms that identify various ways of splitting data into branch-like segments. Decision trees partition data into subsets based on categories of input variables, helping you to understand someone’s path of decisions.
* **Regression (linear and logistic)** Regression is one of the most popular methods in statistics. Regression analysis estimates relationships among variables, finding key patterns in large and diverse data sets, and how they relate to each other.
* **Neural networks.** Patterned after the operation of neurons in the human brain, neural networks (also called artificial neural networks) are a variety of deep learning technologies. They’re typically are incredibly useful for analyzing large data sets. They are great at handling nonlinear relationships in data – and work well when certain variables are unknown

### Predictive Analytics Workflow:

* Import data from varied sources, such as web archives, databases, and spreadsheets.
* Clean the data by removing outliers and combining data sources. Identify data spikes, missing data, or anomalous points to remove from the data
* Develop an accurate predictive model based on the aggregated data using statistics, curve fitting tools, or machine learning.
* Develop an accurate predictive model based on the aggregated data using statistics, curve fitting tools, or machine learning. You might choose to use neural networks to build and train a predictive model.
* Iterate through your training data set to try different approaches. When the training is complete, you can try the model against new data to see how well it performs.

**Start**

The first step when we start a project is to ask where we are getting the dataset we will be analyzing from. How good is the dataset and can we trust it?

**Dataset**

 At **Yahoo Finance**, [Bitcoin USD (BTC-USD) Price History & Historical Data - Yahoo Finance](https://finance.yahoo.com/quote/BTC-USD/history?p=BTC-USD) I get free stock quotes, portfolio management resources, international market data. To build the stock price prediction model, we will use one of the dataset that we download from hear.

The dataset also contains a date-wise price of stock with **open, close, high,** and **low prices** along with **volume** traded as well as turnover on that day.

It is an excellent database for people who want to try learning techniques of data visualization, data analytics, and many different forms of data processing.

#### Data Format: Date, Open, High, Low, Last, Close , Volume.

Table

Description automatically generated with medium confidence

Text

Description automatically generated

We download file BTC-USD.csv for our project. we can choose Time interval. My is Daily. I have 2 dataset BTC-USD.csv and BTC-USD2.csv Different are

BTC-USD.csv have data for 6-year period daily interval from 2015 to 2022

BTC-USD2.csv have data for 1 year period daily interval from 2021to Maj 2022

## Reading and Analyzing the Data

We start work with file **main.ipybn**

We can read the data into frame as shown below :

Text

Description automatically generated

We have total 2391 items , each representing a day’s stock market attributes for the Bitcoin.

We can check for NULL value:

Text

Description automatically generated

## Normalizing the data

The data is not normalized and the range for each column varies, especially Volume. Normalizing data helps the algorithm in converging i.e. to find local/ global minimum efficiently. Normalisation is a feature scaling technique that puts our variable values inside a defined range (like 0-1) so that they all have the same range.

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

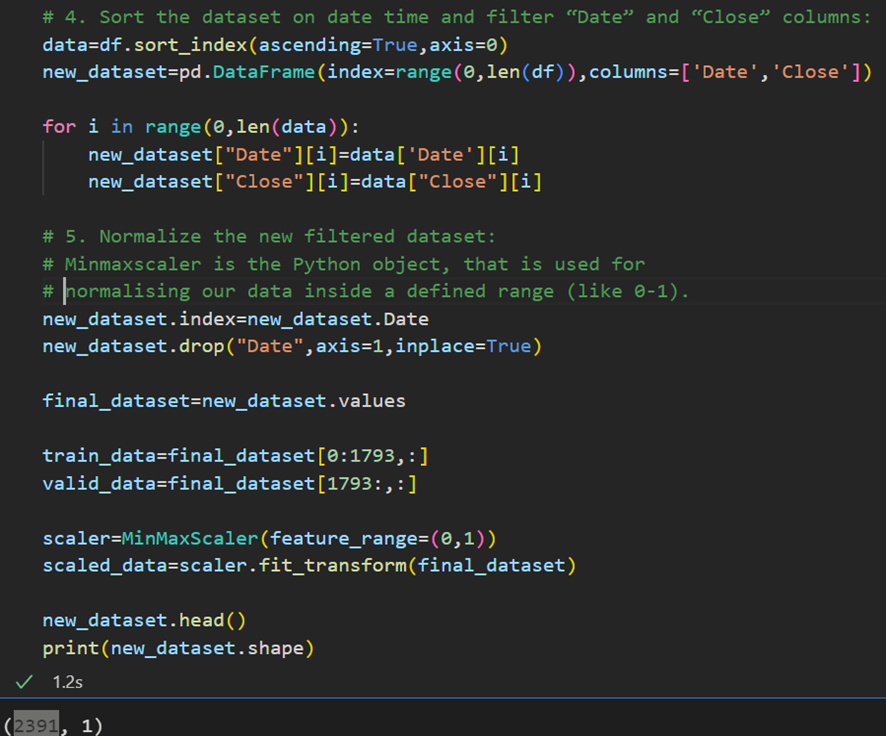
%matplotlib inline

from matplotlib.pylab import rcParams

rcParams['figure.figsize']=20,10

from sklearn.preprocessing import MinMaxScaler

Also, I will convert the **DataFrame** to nd-array in the process and take data info from column DATE and CLOSE that we are interested in .

****

I will use **MinMaxScaler** from **Sci-kit Learn** to normalizing data.

But before that we have to split the dataset into training and testing datasets. For training set **train-data** we take ¾ of the total 2392 datasets items and it is around 1793 units , If we change dataset we should change volume of the units that we train, usual it is 70- 75 % of the all data.

Graphical user interface, chart

Description automatically generated

Text

Description automatically generated

My input is a matrix of 2392 rows and 2 columns.

Graphical user interface, application

Description automatically generated with medium confidence

We should input our data in the form of a 3D array to the LSTM model , but we have 2D array , for this we have function **np.reshape()** that gives a new shape to an array without changing its data.

First, we create data in 60 timesteps before using numpy to convert it into an array. Finally, we convert the data into a 3D array with X\_train samples, 60 timestamps, and one feature at each step.

Graphical user interface, application

Description automatically generated

Now we have prepared dataset **X\_TRAIN\_DATA** and we are ready to create model.

**How to build a LSTM model in Keras with Tensorflow.**

In this thesis work I’ll use Keras and Tensorflow to create a LSTM model, and train and test it on the dataset. Here are the steps we’ll go through:

* Importing the Right Modules
* What is an LSTM?
* Creating a LSTM Neural Network with Keras
* Adding Layers to Your Model
* Training and Testing our LSTM on the Dataset
* Load the dataset
* Compile the LSTM model
* Train and Fit the Model
* Test your LSTM Model

## Importing the Right Modules

## We need to install from <https://pypi.org/>

## Keras is an implementation of the Keras API that uses TensorFlow as a backend.

pip install tensorflow

pip install sklearn

pip install keras

Using Keras and Tensorflow makes building neural networks much easier to build. It’s much easier to build neural networks with these libraries than from scratch.

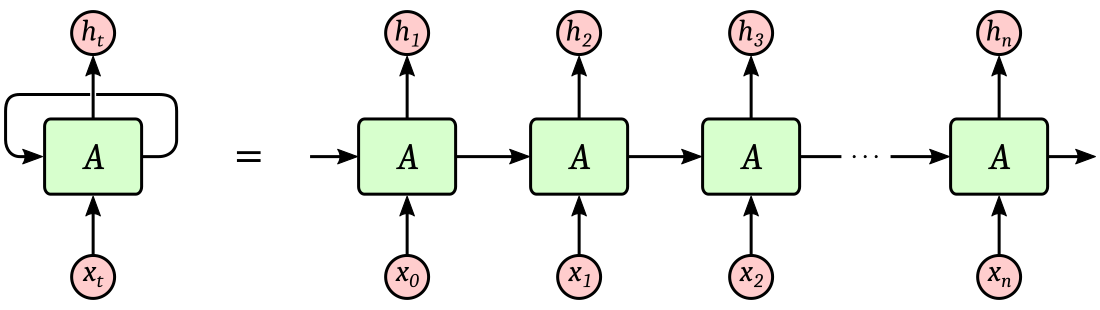
from keras.models import Sequential

from keras.layers import LSTM,Dropout,Dense

**What is Long Short-Term Memory (LSTM)?**

**Long Short-Term Memory (LSTM)** networks are a modified version of recurrent neural networks-**RNN** neural network that has an internal memory, which makes it easier to remember past data in memory.

**RNN-Recurrent Neural Network,** it is a network that works on the present input by taking into consideration the previous output (feedback) and storing in its memory for a short period of time (short-term memory).



**LSTM** is well-suited to classify, process and predict time series given time lags of unknown duration. It trains the model by using back-propagation. In an LSTM network, three gates are present:

Diagram

Description automatically generated

Bild. Gate LSTM

**Input gate** — discover which value from i**nput** should be used to modify the memory.

**Forget gate**— discover what details **to be discarded** from the block.

**Output gate** — the input and the memory of the block is used to decide **the output**.

The advantage of the Long Short-Term Memory (LSTM) network over other recurrent networks back. The LSTM can deal with more complex problems than the RNN by keeping a constant flow of error throughout the backpropagation from cell to cell.

It is important to mention that as more experiments are performed with LSTM network stops being able to learn. Generally, it goes to the thousand-time steps before it happens .

# Understanding Input and Output shapes in LSTM | Keras

Let’s first understand the Input and its shape in LSTM Keras. The input data to **LSTM** looks like the following diagram.

A picture containing diagram

Description automatically generated

We always have to give a three-dimensional **3D array** as an input to your LSTM network. Where :

* the first dimension represents the **batch size**,
* the second dimension represents the **time-steps**
* and the third dimension represents the number of **units** in one input sequence.

**batch size** . As it turns out using very **small** **batch size** reduces the speed of training and on the other hand using too **big batch size** reduces the models ability to generalize to different data and it also consumes more memory. So we have to try out various values on our data and find the sweet spot.

Table

Description automatically generated

**time\_steps** is the unit of time used for the network. This value is determined by how many units back in time is viewed in the LSTM.

If you look at the green part of the picture above CLOSE column, it will be :

**time\_steps**: 3 (0.6201, 0.6201, 06122)

because we are using three pieces of data to output the next one.

**Features** is the number of attributes used to represent each time step. In LSTM, the input is represented by multiple time\_steps inputs, which is the number of its attributes. Here are 2 of them: Date, Closed .

**input\_dim** is the number of dimensions of the features, in my case that is just 2. The equivalent notation for **input\_shape**, which is an actual dimensional shape, is (2,)

**Creating a LSTM Neural Network with Keras**

Keras model represents the actual neural network model. Keras provides a two mode to create the model, simple and easy to use ***Sequential API*** as well as more flexible and advanced ***Functional API*.**

**Sequential**

The core idea of Sequential API is simply arranging the Keras layers in a sequential order and so, it is called Sequential API. Most of the ANN also has layers in sequential order and the data flows from one layer to another layer in the given order until the data finally reaches the output layer.

ANN model can be created by simply calling Sequential() as specified below –

lstm\_model=Sequential()

**Add layers**

Afterwards, we’ll add an LSTM layer. This is what makes this an LSTM neural network. Then we’ll add a batch normalization layer and a **dense** output layer.

Text

Description automatically generated

In this code Units-16 basically means that the dimension of the output vector, h, is 16.

LSTM with 16 neurons.

Why do we need to care of calculating number of parameters in LSTM layer ?

In ANN models, number of parameters is a really important metric for understanding the model capacity and complexity. We need adjust the number of parameters of each layer in the model to handle **overfitting** or **underfitting** situations.

Later can we change parameters if we are not satisfied with result .

**A common debugging workflow: add() + summary()**

When building a new **Sequential** architecture, it's useful to incrementally stack layers with **add()** and frequently print model summaries.

We get LSTM model layers:

Text

Description automatically generated

Note that the Input object is not displayed as part of model.layers, since it isn't a layer.

Models built with a predefined input shape like this always have weights and always have a defined output shape.

Model weight =8:

Text

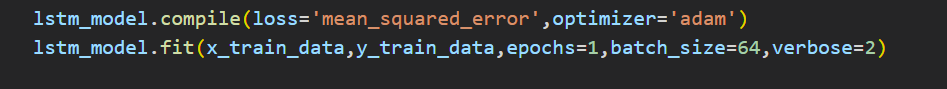
Description automatically generated

Outputs layer =1:

Text

Description automatically generated

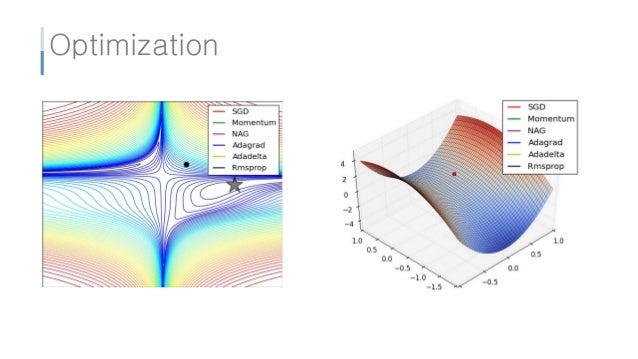
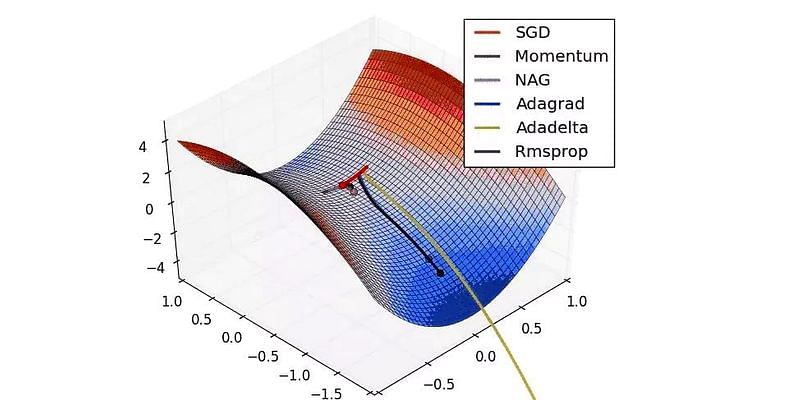
**Compile the model and train :**



**Loss= "mean squared error".** Mean squared error (MSE) measures the amount of error in statistical models. It assesses the average squared difference between the observed and predicted values. When a model has no error, the MSE equals zero. As model error increases, its value increases.

# Optimizer Adam . Optimizer that implements the Adam algorithm.

Adam optimization is a stochastic gradient descent method that is based on adaptive estimation of first order and second-order moments.

**Parameters that we can change if we want to do model better**.

A screenshot of a computer

Description automatically generated with medium confidence

* **batch size** . As it turns out using very **small** **batch size** reduces the speed of training and on the other hand using too **big batch size** reduces the models ability to generalize to different data and it also consumes more memory. So we have to try out various values on our data and find the sweet spot.
* **Units in Layer** . We need adjust the number of parameters of each layer in the model to handle **overfitting** or **underfitting** situations.
* We can even **Add()** one more or several **Layer to model.**
* **Epoch** can be from 1 to 1000. Epoch: generally defined as "one pass over the entire dataset", used to separate training into distinct phases

Text

Description automatically generated

We see that in each epoch **Loss= "mean squared error"** became smallest. Model is training each time, each epoch.

That we can see I graph here: the value x- epoch, Y - loss

A picture containing shape

Description automatically generated

Now we have trained LSTM model and we can save it :

Text

Description automatically generated

It will look like binary file :

Text

Description automatically generated

# Decision Tree Regression and Linear Regression

We will work with file **Variant.ipynb**

**What is a regression tree in machine learning?**

Diagram, engineering drawing

Description automatically generated

A regression tree is basically a decision tree that is used for the task of regression which can be used to predict continuous valued outputs. In Decision Trees for Classification, we saw how the tree asks **right questions** at the **right node** to give accurate and efficient classifications.

Graph is printed using **tree** from **sklearn** function

A screenshot of a computer

Description automatically generated with medium confidence

The branches/edges represent the **True/False** of the statement and take makes a decision based on that in the example below which shows our decision tree that evaluates:

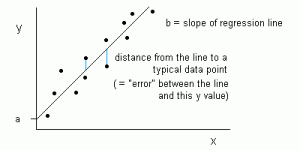
A computer screen capture

Description automatically generated with low confidence

Decision tree regression observes features of an object and trains a model in the structure of a tree to predict data in the future.

**Linear Regression**

In linear regression, we have a set of input variables (x) that are used to determine an output variable (y). A relationship exists between the input variables and the output variable. The goal of ML is to quantify this relationship.



In Linear Regression, the relationship between the input variables (x) and output variable (y) is expressed as an equation of the **form y = a + bx**. Thus, the goal of linear regression is to find out the values of coefficients **a** and **b**. Here, a is the intercept and b is the slope of the line.

From our dataset I got this visualized plot:

Chart, scatter chart

Description automatically generated

#### Dashboard

#### Build the dashboard using Plotly dash

In this section, we will build a dashboard to analyze stocks in a file **stock\_app.py**

[**Dash**](https://dash.plotly.com/introduction) is a python framework that provides an abstraction over flask and react.js to build analytical web applications.

pip install dash

Before moving ahead, you need to install dash. Run the below command in the terminal.

A screenshot of a computer

Description automatically generated with medium confidence

Then we **Run Debugg** a file **stock\_app.py**

In **Localhost: 127.0.0.1:8050** it will open app **Dash,** where we can see visual our dataset and predict graph.

To develop the dashboard for stock analysis we will use another stock dataset with multiple stocks like Apple, Microsoft, Facebook, Tesla.

Graphical user interface, chart, application

Description automatically generated

**Problems**

* I hade problem with reading DateTime format :

*TypeError: float() argument must be a string or a number, not 'Timestamp'*

I converted it to float with the**strftime** () method takes one or more format codes as an argument and returns a formatted string based on it.

df["Date"]=pd.to\_datetime(df.Date,format="%Y-%m-%d").dt.strftime('%Y%m%d').astype(float)

* During my work I changed dataset often to get best training model and didn’t remember to change the size of dataset

Text

Description automatically generated

To Text

Description automatically generated with medium confidence

# Analysis of results

# I got several results from different predictive models . If we compare them, we see the best result is from neural network LSTM

# Linear Regression model:Graphical user interface, chart Description automatically generated Decision Tree Regression model:Graphical user interface, chart Description automatically generated neural network LSTM model:

# Graphical user interface, chart Description automatically generated

**LSTM recurrent networks prove to be a useful model to help analyze sequential data.**

I'm not brave enough to use it to invest in real stocks, in the process I learned a lot about open-source code, different library such a keras, seaborn, Dash. I want to share my results and code with you.

Only when I plunged into a real project, in which a problem arose that I had not solved before, was once again convinced that practice is better theory!

**Final discussion**

Cryptocurrency prices are highly driven by speculation, and I would add analysis from social media like **twitter** or **Google Search Trends** to help predict future prices by using machine learning algorithms to analyze the emotion behind data from people.

**Google Search Trends** -What are people searching for, and how is it changing over time? Google search trends help you gain these insights.

**Tweepy, Twitter for Python**

**Tweepy** is open-sourced, hosted on GitHub and enables Python to communicate with Twitter platform and use its API.

One of the main usage cases of tweepy is monitoring for tweets and doing actions when some event happens. Key component of that is the object, which monitors tweets in real time and catches them. **StreamListener.**

Probably well-trained LSTM model , plus analysis of peoples emotion could give good result, I guess it is possible to make it useful.

# Bibliography:

1. [E-Mini S&P 500 Jun 22 (ES=F) Stock Historical Prices & Data - Yahoo Finance](https://finance.yahoo.com/quote/ES%3DF/history?p=ES%3DF) At **Yahoo Finance**, you get free stock quotes
2. [What Is Predictive Analytics? - 3 Things You Need to Know - MATLAB & Simulink (mathworks.com)](https://www.mathworks.com/discovery/predictive-analytics.html#:~:text=Typically%2C%20the%20workflow%20for%20a%20predictive%20analytics%20application,load%20forecasting%20system%20in%20a%20production%20environment.%20)
3. [Machine Learning models for prediction - Data Science (simulatoran.com)](https://simulatoran.com/machine-learning-models-for-prediction/)
4. [How the LSTM improves the RNN. Understand the differences between… | by Tiago Miguel | Towards Data Science](https://towardsdatascience.com/how-the-lstm-improves-the-rnn-1ef156b75121)
5. [Long Short-Term Memory (LSTM) in Keras - PythonAlgos](https://pythonalgos.com/long-short-term-memory-lstm-in-keras/#:~:text=keras%20lstm%20code%20model%20%3D%20keras.Sequential%28%29%20model.add%28layers.LSTM%2864%2C%20input_shape%3D%28None%2C,than%20the%20Simple%20RNN%20we%20built%20with%20Keras.)
6. [python - Understanding Keras LSTMs - Stack Overflow](https://stackoverflow.com/questions/38714959/understanding-keras-lstms/50235563#50235563)
7. [Predicting stock prices using Deep Learning LSTM model in Python - Thinking Neuron](https://thinkingneuron.com/predicting-stock-prices-using-deep-learning-lstm-model-in-python/)
8. [Machine Learning to Predict Stock Prices | by Roshan Adusumilli | Towards Data Science](https://towardsdatascience.com/predicting-stock-prices-using-a-keras-lstm-model-4225457f0233)
9. [forecastbtc/testing the blog code.ipynb at master · JafferWilson/forecastbtc · GitHub](https://github.com/JafferWilson/forecastbtc/blob/master/testing%20the%20blog%20code.ipynb)
10. [TestRepo/TestBookLSTM.ipynb at master · danmoller/TestRepo · GitHub](https://github.com/danmoller/TestRepo/blob/master/TestBookLSTM.ipynb)
11. [Indexing and selecting data — pandas 1.4.2 documentation (pydata.org)](https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html)
12. [Understanding LSTM Networks -- colah's blog](http://colah.github.io/posts/2015-08-Understanding-LSTMs/)
13. [Visualize a Decision Tree in 4 Ways with Scikit-Learn and Python | MLJAR](https://mljar.com/blog/visualize-decision-tree/)
14. [Python strftime() - datetime to string (programiz.com)](https://www.programiz.com/python-programming/datetime/strftime)
15. [Visualising linear regression| by Rishabh Roy | The Startup (medium.com)](https://medium.com/swlh/visualising-linear-regression-dfac98624d27)
16. [Example gallery — seaborn 0.11.2 documentation (pydata.org)](https://seaborn.pydata.org/examples/index.html)
17. [What Is Google Search Trends? How to Start Using It Now (2022) (tutsplus.com)](https://business.tutsplus.com/tutorials/what-is-google-search-trends--cms-39538)
18. [Introduction to tweepy, Twitter for Python - Python Central](https://www.pythoncentral.io/introduction-to-tweepy-twitter-for-python/)
19. [Tweepy](https://www.tweepy.org/)