

***Artificial Intelligence Fundamentals***

**Project of**

**Cryptocurrency Market price Prediction**

***Krzysztof Milde, Daniel Pławiak,***

***Marcin Świerkot, Łukasz Zalewski***

Computer Science and IS

**17 March 2021**

|  |  |  |  |
| --- | --- | --- | --- |
| **History of changes** | | | |
| **Version** | **Date** | **Who** | **Description** |
| **01** | 2021-03-17 | Krzysztof Milde, Daniel Pławiak | Initial version, definition |
| **02** | 2021-03-23 | Łukasz Zalewski, Marcin Świerkot | Analysis of a problem |
| **03** | 2021-04-06 | Łukasz Zalewski, Marcin Świerkot | Some existing solutions  Description of a preferred solution |
| **04** | 2021-04-13 | Krzysztof Milde | User Interface (Description of a preferred solution) |
| **05** | 2021-05-25 | Łukasz Zalewski,  Marcin Świerkot | Some implementation problem |
| **06** | 2021-06-08 | Łukasz Zalewski | Simulation results from the application, Conclusion,  References, Final version |

|  |  |
| --- | --- |
| **Responsibilities** | |
| **Name** | **Tasks** |
| **Krzysztof Milde** | 1. Description 2. Topic research 3. Frontend and backend implementation 4. Presentation |
| **Daniel Pławiak** | 1. Description 2. Topic research 3. Final model implementation 4. Presentation |
| **Marcin Świerkot** | 1. Description 2. Topic research 3. First models attempts 4. Presentation |
| **Łukasz Zalewski** | 1. Description 2. Topic research 3. First model attempts 4. Presentation |

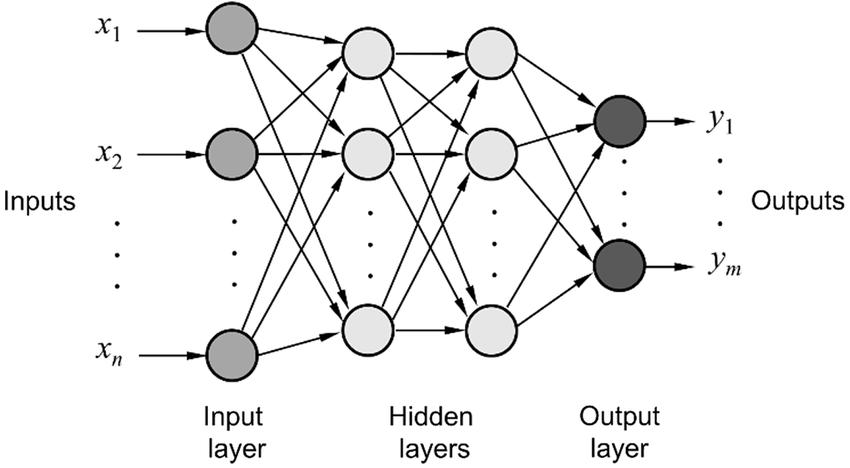
**Table of Contents**

1. Description of a problem and technology [4](#_heading=h.gjdgxs)
   1. Technology [4](#_heading=h.30j0zll)
   2. Description [4](#_heading=h.1fob9te)
2. Analysis of a problem [5](#_heading=h.3znysh7)
   1. Model [5](#_heading=h.2et92p0)
   2. Application 6
3. Some existing solutions 7
   1. Commercial applications 7
   2. Non-commercial applications 8
4. Description of a preferred solution 9
   1. Backend application 9
   2. Frontend application 11
   3. User Interface 11
5. Some implementation problems 14
6. Simulation results form the application 14
7. Conclusions 16
8. Literature 17

# Description of a problem and technology

## Technology

In order to implement a prediction model for cryptocurrency prices Python will be used to develop a *Deep Learning Neural Network.* Such technology is capable of learning patterns which are often hard to spot by the human eye.



***Figure 1 Neural Network Schema***

*Source: https://www.researchgate.net/figure/Typical-structure-of-a-feed-forward-multilayer-neural-network\_fig1\_291339457*

## Description

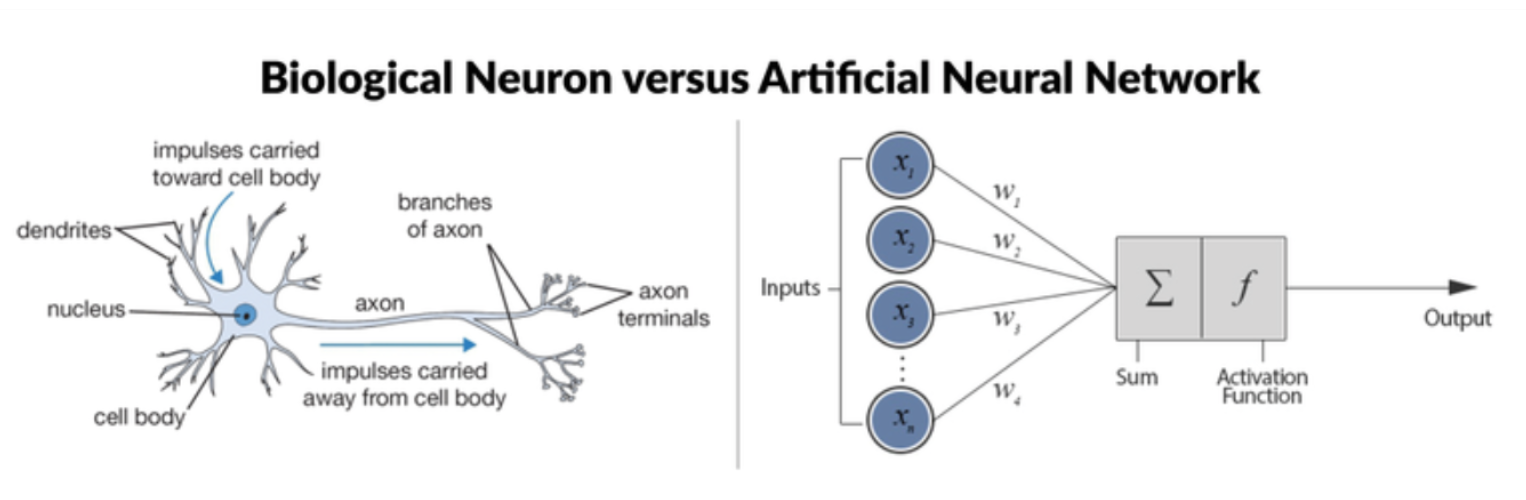
Cryptocurrency Market is one of the most unpredictable fields of investment. It can be affected by many external factors and indecisive human nature. Hence, the area of price prediction is extremely complicated for analysis and causes a great challenge for predictors. However, due to the rapid and constant Artificial Intelligence development, new techniques of approach have been introduced. One of the new ways to aid in predicting is usage of Deep Neural Networks. This paper will focus on implementing, training and testing prognostic models. Based on historical data, the system will predict the short time trend of a cryptocurrency (decrease/increase) and estimation of the price value itself. In our work, we will be mostly focusing on Bitcoin, although once the working model for that cryptocurrency is prepared, we may also add options for other assets, such as Ethereum, Binance Coin, or other so known “altcoins”.

# Analysis of a problem

## Model

As we pointed out at the beginning we will use artificial neural networks to build the desired model. The goal of the model will be not only to predict future trends on the cryptocurrency market, but also to predict the future price range of the highest capitalised and most promising cryptocurrencies.

The idea behind neural networks is to mimic the way the human brain works and processes information. The fundamental type of cells, the human brain consists of are neurons, which are the basic working unit of the brain that transmits information to other cells. Each neuron consists of the cell body, dendrites and axioms. The information, in the form of an electric pulse, is received by the dendrites, then processed within the cell body and further passed to the axon, which delivers it to the next neuron. Artificial neural networks are an attempt to recreate a similar structure that can be used in solving many artificial intelligence problems.

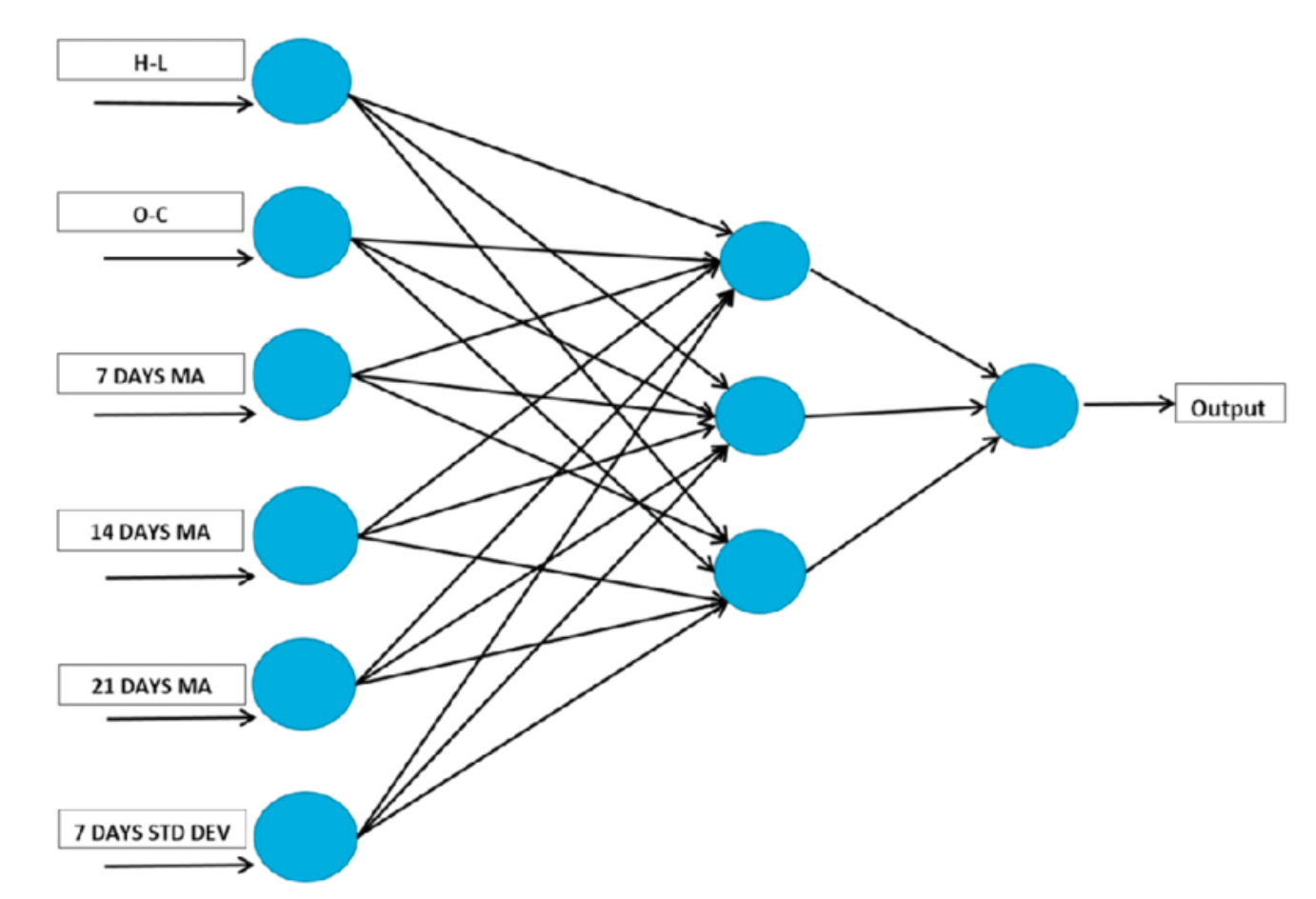
***Figure 2 Biological neurons and their artificial counterpart***

*Source: https://www.researchgate.net/figure/Biological-Neuron-versus-Artificial-Neural-Network\_fig4\_325870973*

The similarities between neurons and their artificial counterparts, called perceptrons, are captured in the Figure 2 image above.

One of the applications of artificial neural networks is predicting time series, thus they should fit the characteristics of our task. The second advantage of neural networks is their ability to model nonlinear relationships in the data.

The architecture of the artificial neural network we want to build and train, shall resemble the one pictured below. The input data will come from the most popular technical indicators such as moving averages, moving standard deviation, days highs and lows, which will be then processed in hidden layers (**Fig 3)**, and then the final output in the form of future price will be obtained.

***Figure 3 Neural Network Schema for price prediction***

*Source: https://www.researchgate.net/figure/Detailed-architecture-of-Artificial-Neural-Network-ANN-for-stock-price-prediction\_fig1\_340698512*

## Application

Building the model is the biggest challenge, as it not only has to work correctly, but we also need a way to present it in a clear, yet pleasant way for the user's eyes. The appropriate presentation of the model’s output is also crucial.

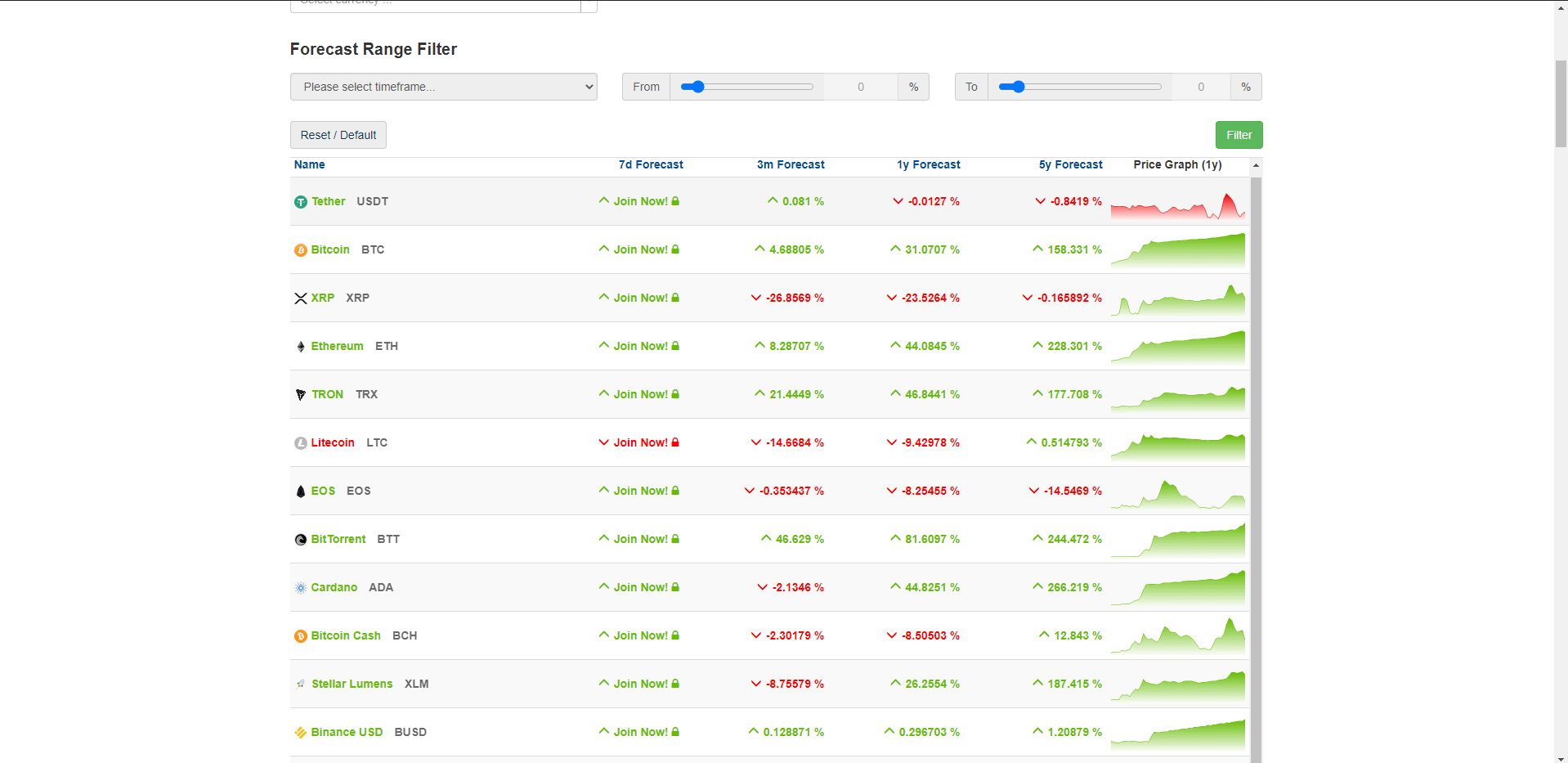
The user will be able to see the predictions via a single web app. It will fetch the data being the result of the model’s prediction via its endpoint. The application will then show the final prediction for the given cryptocurrency shares, as well as some basic information about the object. A given simulation presenting the graph of the price prediction for a certain period of time will be drawn, and a final decision regarding the state of buying the given asset will be delivered to the user as well.

1. **Some existing solutions**

As predicting cryptocurrency prices has been a hot topic in the last years, many different applications and solutions for that problem have been created. For that case we can differentiate between two different sorts of applications.

## Commercial applications

Commercial applications are the programs, which usually need some sort of paid membership to take full advantage of them. A user can create a free account, or use the application with no account at all, but it usually has fewer functionalities, or they are available for only a certain amount of time. However, in many cases they come with not only tools for the price prediction, but also with some other automated features that can fully utilize it, such as trading bots. While it’s hard to make any predictions on what the general price-prediction techniques are used in these types of solutions, or what programming language and libraries are used, we can see how a well-constructed, professional application making those sorts of predictions should look like. Below in Figure 4 we can see an example of such a web application, which contains many of the qualities described above.



***Figure 4. Example of commercial web application***

All commercial applications can be split into two types. The first type is a simple mobile application, which gives fast information about the current trends. The UI in those sort of programs is usually minimal and clear, as its goal is to quickly provide the information to the user when he’s outside, and just wants to quickly check the probable price of the asset. Based on that, a person can quickly perform actions to try to make any potential profits.

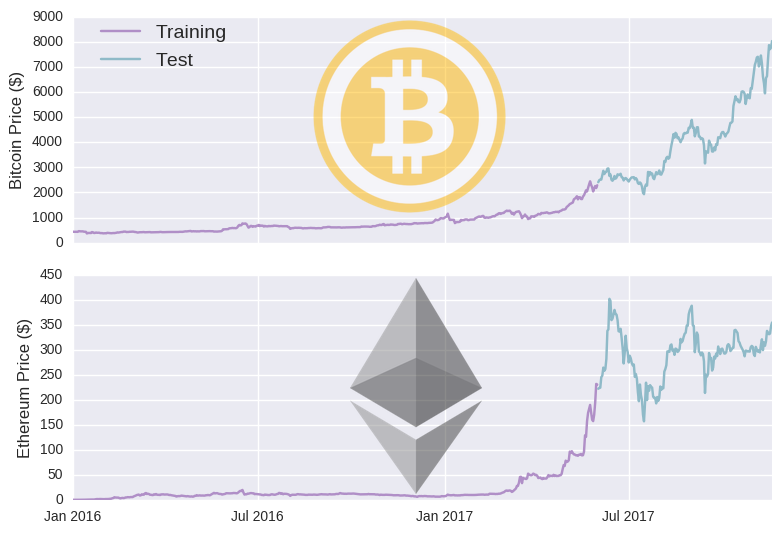
The other type of application is web application. In that case, the UI is usually filled with a lot more tools and features. In many cases, a user can choose many different cryptocurrency tokens or coins, and compare between these assets. Also, there are usually sorts of price predictions for given years along with some long-term plots. As an additional functionality for premium users, there are sometimes other features provided as well, such as trading bots.

## Non-commercial applications

For the non-commercial solutions, the most popular choice of programming language is Python. It’s not a surprise, as it comes with a huge variety of different tools and modules for writing quick and easy machine learning programs.

A usual solution consists of first reading the data, which is prepared beforehand. It usually consists of the information about the name of the different asset, date, closing and opening price as well as the volume. However, there are sometimes even more different variables, which come in handy for the analysis. Then, some sort of feature engineering is usually done on the given set of data. It can be done manually, although many solutions use different sorts of automated tools for feature engineering. At last, a chosen set of models can process through the data, and print the results in form of a single information, or a plot. An example of a result plot from that program can be seen in Figure 5, which presents

These simple applications are usually very easy and quick to write, however they usually lack any further form of customization, or even user interface. They are mostly centered around usage of a given set of data, and they lack any sort of automatization. They are still, however, a good starting point.



***Figure 5. Example of a plot constructed in Python for price prediction of BTC and ETC***

*Source: https://dashee87.github.io/deep%20learning/python/predicting-cryptocurrency-prices-with-deep-learning/*

1. **Description of a preferred solution**

For our application, we decided to use the Python programming language to do most of the work regarding the AI part. It provides many libraries and modules, which really come in handy when designing the solution for price-prediction problems. They will be the core of our algorithm. It also comes with a few reliable ways to then provide the data to web applications, which will work as our tool for communication with the user. For that case, we decided to split the application into two distinct parts – backend and frontend application.

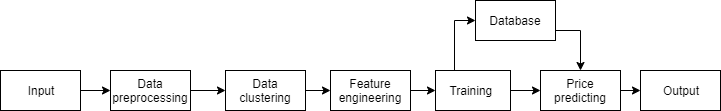
## Backend application

The backend application will fetch the data regarding the chosen asset. It will then process it using the deep learning neural network algorithms, and hand them into the frontend application for the user. For the sake of that part, Python provides many modules, which come in handy when creating suck applications.

**Pandas** module is an easy solution for reading the data through the csv file, and it will be a good solution for at least testing purposes at the beginning. In the later stages, however, the fetch will be done using an API provided by a third-party site, such as for example **Binance**, **Yahoo Finance** or **CoinMarketCap**. This way, we will be able to provide an automatic way to analyse any cryptocurrency, and we will not be limited to only a certain amount of assets prepared beforehand.

For any mathematical operations on the data, we are going to use the **numpy** module. It provides easy to use tools for any operations, and it may be useful for any situations where we need to modify our data manually.

For deep learning neural networks, Python provides many different modules. **Tensorflow** and **Keras** are the ones which we are most interested in, and we will probably stick to one. They both provide simple functions for creating and using our deep learning neural networks. With that, we can then show our results, and analyze it even further beyond. The example block diagram of the possible process of the AI part can be seen below, in Figure 6.



***Figure 6. Example block diagram of example system for price prediction***

For providing the results coming from our neural network, we may be using simple plots and graphs. They could be constructed with the help of **Matplotlib,** which is a Python module for creating plots and graphs. However, it’s also highly possible that we will try to accomplish these graphs by using modules or frameworks in the frontend application. Nevertheless, that module could still be useful if we would like to see the graphs outside of the frontend application, for our analysis or for comparison.

As for providing the client services for the frontend application, **Flask REST Api** is going to be used. It provides an easy way to start any web-based service, and with that we are going to run our frontend application.

After the data is processed, and the final results are ready, the data will be formatted in a certain way, which will then be processed by the frontend application.

## Frontend application

Frontend application is made in the form of a web application. It will get the data prepared by the backend regarding the chosen asset, and will show the results to the user.

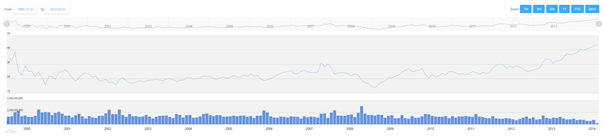
The preferred language of choice here is **Javascript with HTML.** We could also have been using the **React** framework for our application,but the JS alone was enough to create an application that suits our needs. That combination makes it possible to create a good-looking web application with many different capabilities with not much of a hustle.

For the price trend prediction, the user gets information about the possible trends of given cryptocurrency prices trend predictions in a clear form, using **HTML + CSS** elements.

For asset management and choice, a user-friendly interface is provided. In the preferred solution, the user would be first able to log in to the application to his personal account. In that account, he would be able to choose the cryptocurrency, for which he wishes to get the prediction. Additionally he would also have the possibility to choose cryptocurrencies he is particularly interested in, so every time he logs in he will see the predictions of the cryptocurrencies he chose. However, in the end, we decided that logging into account is not necessary, as we want to just focus on the prediction aspect, and to do that just using the page alone without a need of an account is enough.

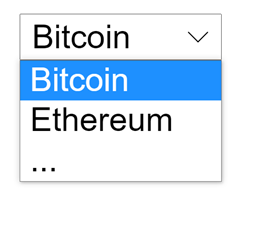
**4.3.**  **User Interface**

In order to implement User Interface, a web application is implemented. There are multiple Rest APIs available to connect and access financial data. The most popular is **Yahoo Finance API.** We tried it, but it wasn’t optimal for fetching live data. Thus, for our case we used **CoinGecko**. It is because the API from that site is free, easy to use, and we can fetch the current data without any problems. Having knowledge about livestock data, the plot of it’s price will be shown. There are various JavaScript graph plotting frameworks, which are free to use. In our case we chose to use the **amCharts** extension **(Fig. 7)**.



***Figure 7. Example of a stock price chart constructed in amCharts***

In this work, we focus on Cryptocurrencies. That is why a user has a choice of their favorite currency. Even though we are, most and foremost, focusing on Bitcoin and its price trend, we still want to create an application, which will give the possibility to go through Dogecoin and Ethereum as well, with potential to add even other currencies in the future. The list of available options is available in the drop down list **(Fig. 8)**.



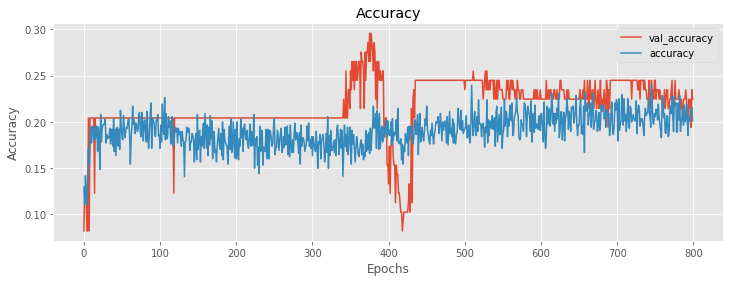
***Figure 8. Drop down with available currencies***

After each change of Cryptocurrency diagrams will refresh to show appropriate data. At the right top of the page a price trend prediction **(Fig. 9)** will be placed to give the user a quick and easy way to access the prediction for given crypto’s price trends for 1, 3, 5 and 7 days,

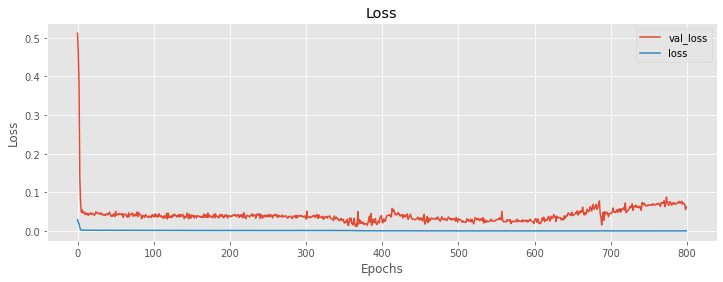


***Figure 9. Example of the price prediction***

After that, a user can also look at the accuracy and loss charts, which are below the current price chart. These charts are simple images, which were created during one of many cryptocurrency model’s training and fitting, as we didn’t want to recreate the chart over and over again. For those reasons, the accuracy and loss charts are always the same just to give the user some idea about the model itself. The example accuracy chart (**Figure 10**) and example loss chart (**Figure 11**) can be seen below.



***Figure 10. Example of the accuracy chart for the model***

**

***Figure 11. Example of the loss chart for the model***

For the preferred solution, we were at first thinking about implementing the multiselection option for all the currencies (**Fig. 12**). However, as it turned out, the multiselect would unnecessarily add too many charts for accuracy and loss for models predicting given crypto, and we decided to stick to one currency only. Thanks to that, we keep our solution clear and easy to read.



***Figure 12. Idea of selection of multiple available currencies***

# Some implementation problems

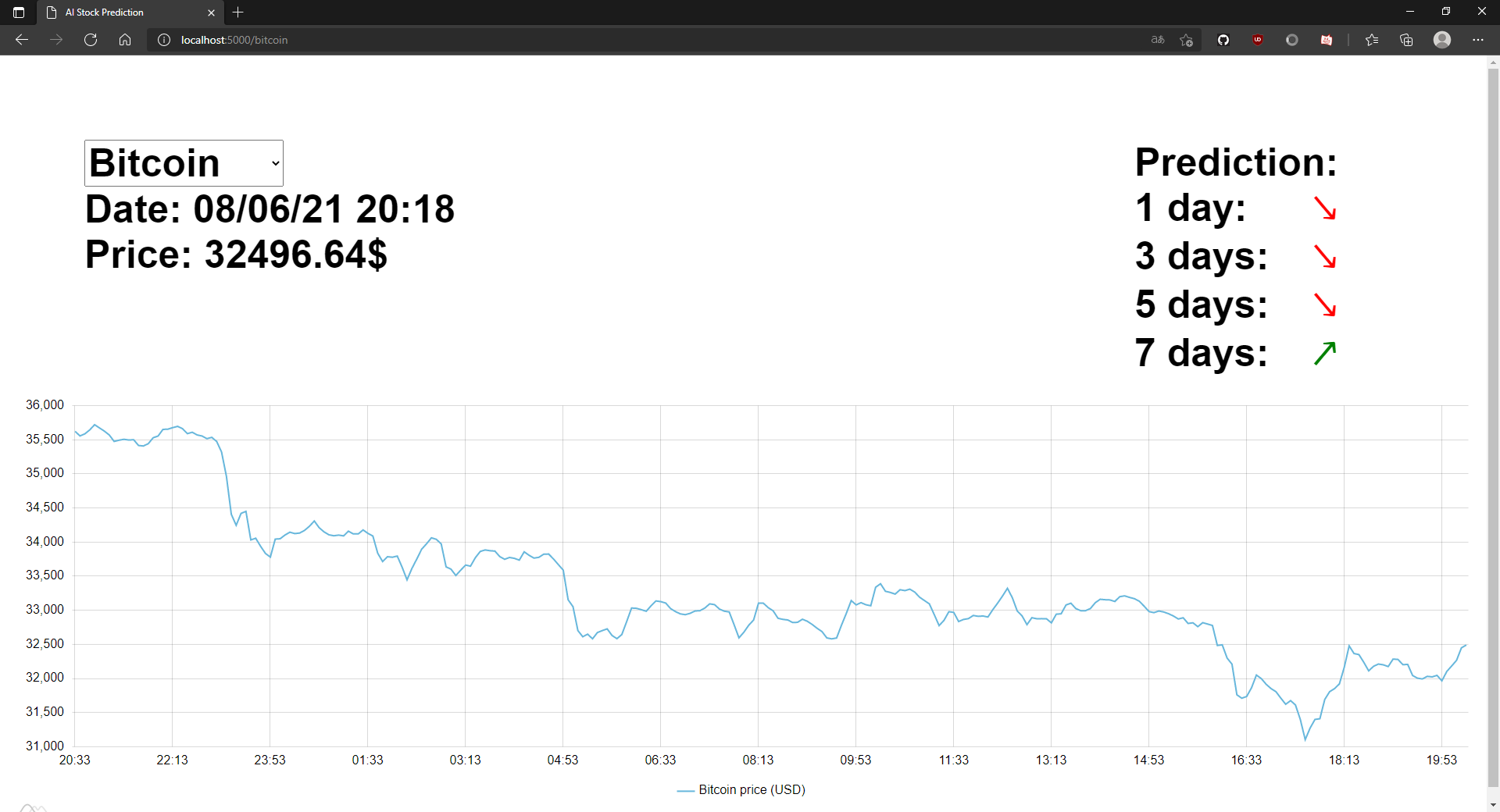
The biggest problem with implementing the solution was related to finding the best set of data for our case. At first, we wanted to use the data set taken from the one minute price chart of Bitcoin, starting from around 2013. However, as it turned out, the data set was too large, and training the neural networks on that set was taking too much time for our case. In some cases, the estimated time was around 36 hours. The results from the neural networks trained on that set weren’t satisfying as well, and it seemed that they weren’t close to the real prices of Bitcoin. Because of that, implementation of a correctly working model took a lot more time than expected.

Because of the problems with the dataset, it was also really hard to establish the best layers and options for our neural network. Dataset really had an impact on choosing the best option for us, as the results were usually unacceptable. Only after changing the dataset, we were able to establish the model, which was the best in our situation.

Because of that, we also had to change our strategy in our prediction. At first, we were thinking about making the price prediction, but the prices were varying too much, as in some of our tests they were sometimes even more than 2 times bigger or smaller than the original price. Because of that, we decided to focus on the aspect of the price trend, as for these there are no problems with such situations, and from that a user can get a somewhat reliable result.

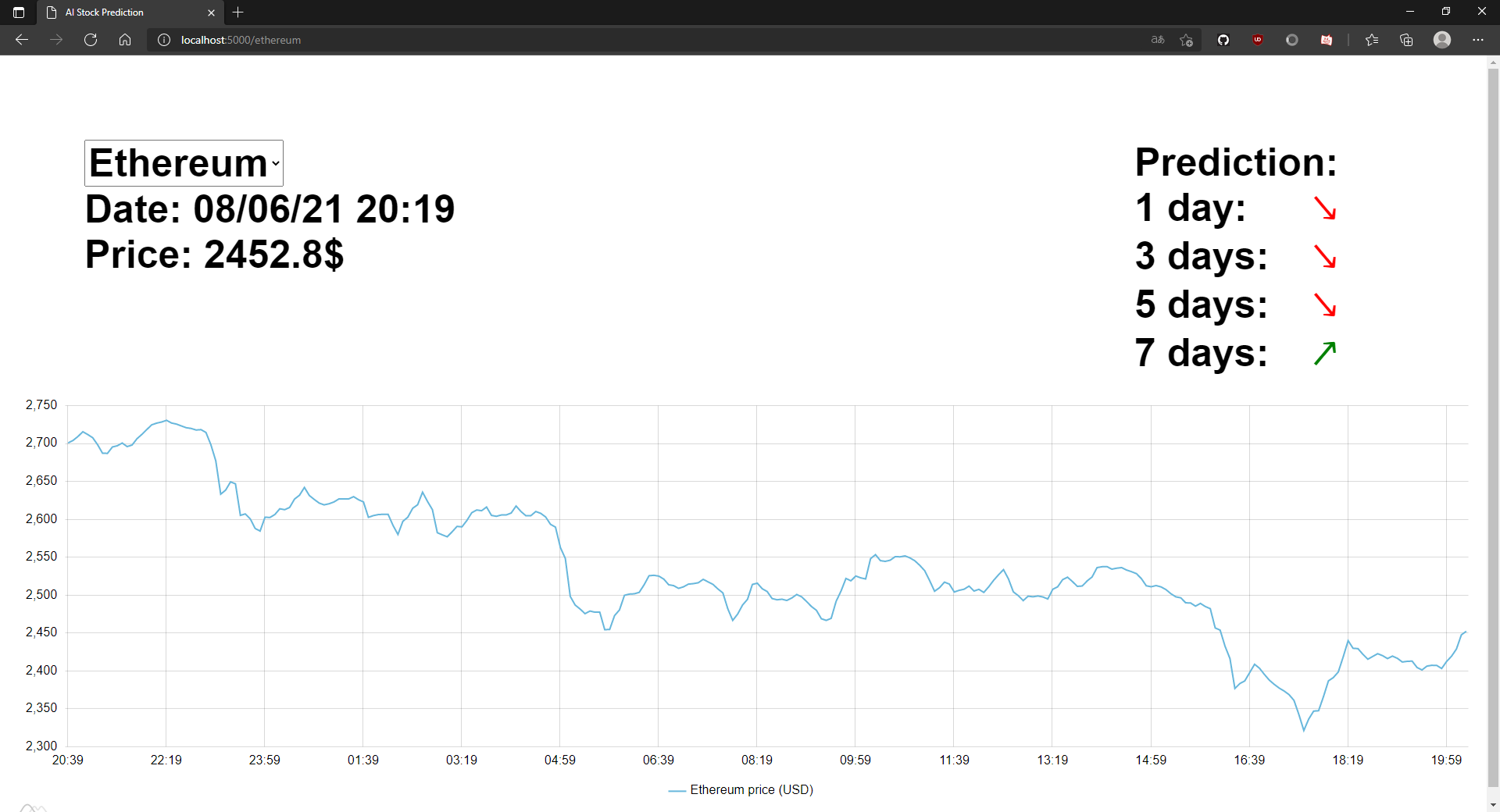
# Simulation results from application

Below, our final web application can be seen. It shows the chart with real prices for Bitcoin (**Fig. 13**) as well as for Ethereum (**Fig. 14**) and Dogecoin, along with the prediction for a timespan of 1 day, 3 days, 5 days and 7 days. Each period is symbolised by either red arrow pointing down, which is a signal to sell, or by green arrow pointing up, which is a sign that the user should buy Bitcoin, as it is going up. The prediction is based on the work performed by our neural networks. They are trained on the dataset containing prices for Bitcoin only, assuming that Ethereum and Dogecoin follow the price trends created by Bitcoin. With that in mind, we can use the prediction for these currencies as well.



***Figure 13. Fetched price of Bitcoin along with the prediction***

As we can see above, the price of Bitcoin and the prediction as well can be checked really easily. The same goes for Ethereum and Dogecoin.



***Figure 14. Fetched price of Ethereum along with the prediction***

After seeing the above charts and information, the user can scroll down to see the accuracy and loss charts for each cryptocurrency. Example of this behaviour can be observed below for Dogecoin (**Fig. 15**).



***Figure 15. View for Dogecoin with live price chart and accuracy and loss charts***

# Conclusion

Neural networks are a great application of artificial intelligence in predicting many different things in an environment, where there are not that many changes, or they are constant. However, the cryptocurrency market is really hard to predict, as there are many small factors, on which not only the user doesn’t have an impact on, but he also has no real possibilities to foresee them. Situations such as some well-known person tweeting about the cryptocurrency market, or the government suddenly banning the whole market are not things that a human nor neural network system can predict, and these situations create dramatic price swings, which have a huge impact on the price.

The neural network, if trained correctly, can definitely make some predictions in regards to the cryptocurrency market, although one has to be careful and observe the market so that he can act whenever something unexpected happens. While neural networks may be a better solution for other types of markets, where these big swings don’t happen, a person can still gain a lot of knowledge from such predictions.

# Literature

* [François Chollet](https://www.amazon.pl/s/ref=dp_byline_sr_book_1?ie=UTF8&field-author=Fran%C3%A7ois+Chollet&search-alias=books) “Deep Learning with Python”
* [Aurélien Géron](https://www.amazon.pl/s/ref=dp_byline_sr_book_1?ie=UTF8&field-author=Aur%C3%A9lien+G%C3%A9ron&search-alias=books) “Hands-On Machine Learning with Scikit-Learn, Keras, and Tensorflow: Concepts, Tools, and Techniques to Build Intelligent Systems”
* Aggarwal Charu C. “Neural Networks and Deep Learning”
* Suresh Samudral “Machine Intelligence: Demystifying Machine Learning, Neural Networks and Deep Learning”
* Wiley John&Sons Inc. “Bitcoin For Dummies”