Milestone 3 Mick van Hulst (s1013954)

My project is Product-oriented as I'm going to develop a predictive model that will predict whether or not the Bitcoin price will go up or down in the coming 24h. Before even using these methods, I'll have to preprocess the data which will consist of:

1. Focusing solely on users who have at least been active over a set amount of time (have to decide on the boundaries).
2. Focusing solely on English tweets and tweets who only have a single type of ticker symbol in their tweet (either '$btc' or '$BTC'). Approach based on [1].
3. Replacing certain words like negation with the keyword 'NEGATION' as prior research found that to be useful. Approach based on [0].
4. Removal of retweets. Although retweets have an impact on the reach of a certain tweet, they do not provide additional sentiment/textual information.
5. Other based on further research and/or advise tutors.

After preprocessing, I'd like to propose the following models, which can be divided in two categories and where each model predicts whether the price of Bitcoin will go up or down after 24h from making a prediction:

1. **Classification using keywords of Tweets:** This model will consist of a ConvNet which requires a matrix input. The input for this model will be a 2-dimensional matrix consisting of the usage of keywords over t timesteps for a certain user.
    a. To generate data, I'll first get a certain price of Bitcoin at time t and t-1 and take the difference of those two (this will tell me whether the price went up/down). This difference will be weighted as traders pay a fee to make trades.
    b. Second, I'll generate 2d matrices based on keyword usage of users. These keywords will be generated first based on a model described below. For each user, I can then generate a: k x t matrix, where k is a set of keywords that's present or not in the set of tweets[1] for a certain user at timestep t (t is the amount of timesteps we look in the past, e.g. 30 days). This matrix will be assigned a binary variable (price going up/down) based on the fluctuation of Bitcoin price at t+1 using the proposed method (see 1a).
    Extra explanation for classification: given an arbitrary matrix m, which consists of keywords from timestep t-30 to t, we predict whether or not the price will go up after 24h. We then evaluate this prediction by using the price of t+1 (i.e. price after 24h) to see whether or not the model was correct or not.
    c. As the set of keywords will have to be fixed, I'll need to do some preprocessing to find which keywords in a tweet are most important. This means that I'll first have to do a correlation analysis to find words which relate to a fluctuation in price of Bitcoin. These words will be assigned a token such that I can use them in the model.
    d. Classification will consist of gathering keywords for certain users from timestep t-n to t (where t-n is the amount of time we look in the past and t is the current date), then classifying these matrices on a per user basis, to predict whether or not the price will go up or down after 24h. This will result in a list of predictions for each user, where we'll

[1] If a certain user tweets several messages per day, these tweets will be aggregated into one tweet.

perform a majority vote/weighted average to come to our final prediction for the price of Bitcoin going up/down after 24h.

2. **Classification using the stance of Twitter users on Bitcoin:** For this model I'd like to use the sentiment of tweets. Prior research [1] used a dictionary[2] that classifies words in a tweet as either NEG or POS and then uses certain formulas to determine negativity/positivity of an entire tweet. As they found that the used formulas give different results, my research will use all three formulas to generate three sentiment indexes. This sentiment will be used to propose three ConvNets, where I'll use the same form of assigning a class to a data point as model 1:
    a. *Single user sentiment:* Single user 1d ConvNet that uses a 1 x t x s matrix[3], where we classify the price as going up/down based on the development of the sentiment of a user over timesteps t by using three channels. I'd then train the classifier on all users at timestep t. Prediction will be based on classifying all users for a certain timestep and then taking a majority vote/weighted average which is comparable to method 1.
    b. *Ranking based model*: I'll use a ranking model to rank users based on their Twitter meta-data (e.g. user activity) and perhaps content of their tweets. Based on that ranking model I'll select the top n users and use them as my user input. Then each datapoint will consist of: n x t x s matrix, where t is the amount of timesteps (e.g. 30 days), n is the sentiment of the users that we're following and s consists of three sentiment channels. For this model it's highly important that we follow users who are active as inactive users means that there's no sentiment to track for a given day.
    Problem[4]: Assuming I have 365 days of data, then I'd be able to only generate 335 data points as I'm following a fixed amount of users, meaning that n is set and the only thing that varies is t. ***To solve this I propose model c.***
    c. *User cluster model*: I'll use a clustering model to cluster users based on their tweet content. Based on this model, each datapoint will consist of: n x t x s matrix, where t is the amount of timesteps (e.g. 30 days), n is the set of clusters which were generated using a clustering algorithm and s consists of three sentiment channels. As we're working with clusters, the cluster will consist of multiple users and I'll be able to sample users from these clusters to generate multiple rows, thus creating multiple data points with these clusters per set of timesteps.
3. **Ensemble:** If all goes well, I'd like to make an ensemble of the models by either performing some form of an average or cutting the models off at a certain point and add another neural net to process the signals of both models.

[2] This dictionary was used for the same use-case, but instead of focusing on Bitcoin they focused on the stock market. The authors who wrote this paper mention that: *"We used the Harvard-IV dictionary (Jorgenson & Vu, 2005), which is a commonly used source for word classification in the financial content analysis of popular press articles and Web news sites, used, for example, by Tetlock(2007), Tetlock etal. (2008), and Da, Engelberg, and Gao(2011).".* There's a chance that some words that are tweeted will not be known by the dictionary, these words will be hand-labeled (by either myself or some people I know from the domain) as positive or negative and mentioned in the research as a possible limitation.

[3] This model assumes that there's no benefit in moving a filter over the several different sentiment indexes at the same time. If there is a benefit there, then I'd have to change it to a s x t matrix, such that I can apply a filter that moves over the three different sentiment indexes at the same time.

[4] I've added this model as the tutors might know some form of a solution to this problem and/or it might spark new ideas which can further aid me during my research. If not, then I'll abandon this idea.

In terms of techniques, I'll use Python/R for data-preprocessing. Furthermore, I'll use Python and the packages PyTorch/Keras (most likely PyTorch as I'd like to develop my skills using this package, but Keras if I don't have sufficient time) for the Deep Learning part.

Resources wise, I'll gather data using the Twitter API. At the moment I've been scraping for about a week and my strategy consists of gathering t amount of days of unique tweets which contain the keyword '$BTC'. These unique tweets will then give me a list of users who tweet about Bitcoin. As the Twitter API doesn't allow me to go back in time more than 14 days, I had to find a workaround, which consists of scraping Tweets of individual users. This method does allow me to go back further in time than 14 days.

In terms of priorities, I'd like to focus on model 1 and model 2a first. Model 1c is a bit of a wild-card as I'm not sure how effective clustering will be. After speaking to Florian Kunneman, he mentioned that there hasn't been that much research regarding computational methods and predicting the Bitcoin price, so in that regard, I'd like to see this as a form of a basis which I or others can use for further research.

Sources (sorry for the ugly references, I promise I'll do better when I write my report :P):

[0] Smailovic, J., Grcar, M., Lavrac, N., & Znidarsic, M. (2014). Stream-based active learning for sentiment analysis in the financial domain. *Information Sciences, 285(32),* 181–203

[1] Sul, H.K., Dennis, A.R. & Yuan, L. I. (2017). Trading on Twitter: Using Social Media Sentiment to Predict Stock Returns. *Decision Sciences, Vol. 48, No3.*