SAPIENZA
UNIVERSITÀ DI ROMA

**Alessandro Di Patria, Matricola: 1844538**

---

# Twitter Sentiment Analysis

## About the dataset

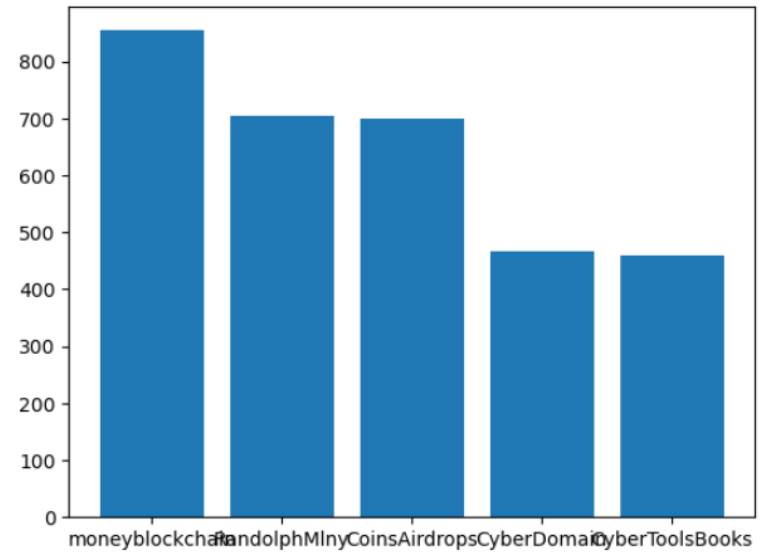
The project is about predicting the sentiment of tweets about Bitcoin.
The dataset is provided at this link : [https://www.kaggle.com/datasets/skularat/bitcoin-tweets](https://www.kaggle.com/datasets/skularat/bitcoin-tweets) .
The dataset choiced contains only few but important features like text,username,Date,Dateonly, Url of the tweet, Retweet by Users, number of likes and finally the label column composed by the following classes: “Positive”,”Negatives” and “Neutral”

| | Date | Text | Users | d | e | Retweet | Url | Sentiment | Mycol | Dataonly |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 2018-03-23 00:40:32+00:00 | RT @ALXTOKEN: Paul Krugman, Nobel Luddite. I h... | myresumerocket | 16522 | 0 | [] | <a href="http://twitter.com" rel="nofollow">Tw... | ['neutral'] | 2018-03-23 00:40:32+00:00 | 2018-03-23 |
| 1 | 2018-03-23 00:40:34+00:00 | @lopp @_Kevin_Pham @psycho_sage @naval But @Pr... | BitMocro | 1295 | 0 | [u'Bitcoin'] | <a href="http://twitter.com/download/android" ... | ['neutral'] | 2018-03-23 00:40:34+00:00 | 2018-03-23 |
| 2 | 2018-03-23 00:40:35+00:00 | RT @tippereconomy: Another use case for #block... | hojachotopur | 6090 | 0 | [u'blockchain', u'Tipper', u'TipperEconomy'] | <a href="http://twitter.com" rel="nofollow">Tw... | ['positive'] | 2018-03-23 00:40:35+00:00 | 2018-03-23 |
| 3 | 2018-03-23 00:40:36+00:00 | free coins https://t.co/DiuoePJdap | denies_distro | 2626 | 0 | [] | <a href="http://twitter.com" rel="nofollow">Tw... | ['positive'] | 2018-03-23 00:40:36+00:00 | 2018-03-23 |
| 4 | 2018-03-23 00:40:36+00:00 | RT @payvxofficial: WE are happy to announce th... | aditzgraha | 184 | 0 | [] | <a href="http://twitter.com/download/android" ... | ['positive'] | 2018-03-23 00:40:36+00:00 | 2018-03-23 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 54 | 2018-03-23 08:55:16+00:00 | RT @fixy_app: Fixy Network brings popular cryp... | quoting_lives | 5673 | 0 | [] | <a href="http://twitter.com" rel="nofollow">Tw... | ['positive'] | 2018-03-23 08:55:16+00:00 | 2018-03-23 |

At First I decided to check if there are missing values and delete them with entire columns because It's quite difficult to replace a text.

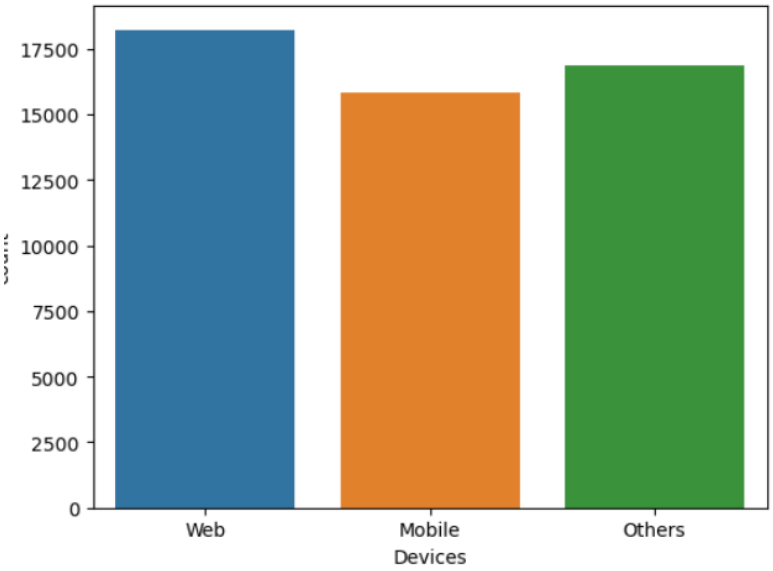
## Data Analysis

After Doing that I began to analyze the dataset and make some statistics about it. Initially I decided to analyze what are the most active users (users that tweet most). The graph in the section below shows the result that I found.

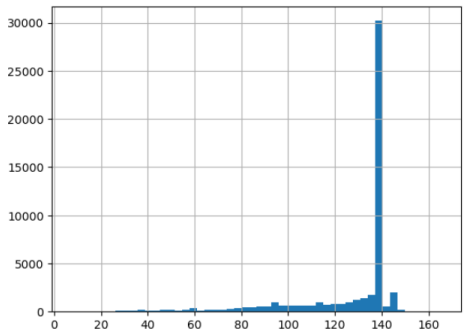


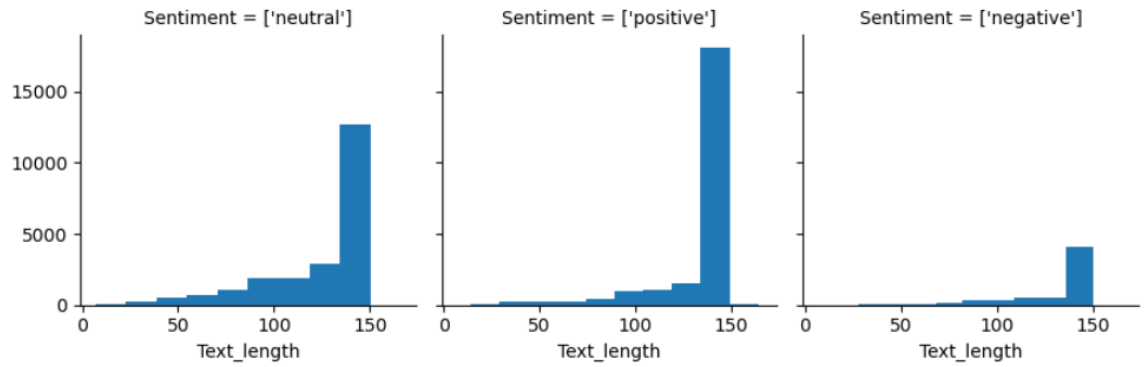

1# 'Moneyblockchain' with 854,
2# 'RandolphMlny' with 705
3#' ‘CoinsAirdrops', with 700
4# 'CyberDomain' with 467
5# 'CyberToolsBooks' 459

After I Decide to create a function that extrapolate in the URL the device used by the users. I found that people use Web Browsers rather than mobile apps to make tweets. The graph in the section above shows a more accurate description.

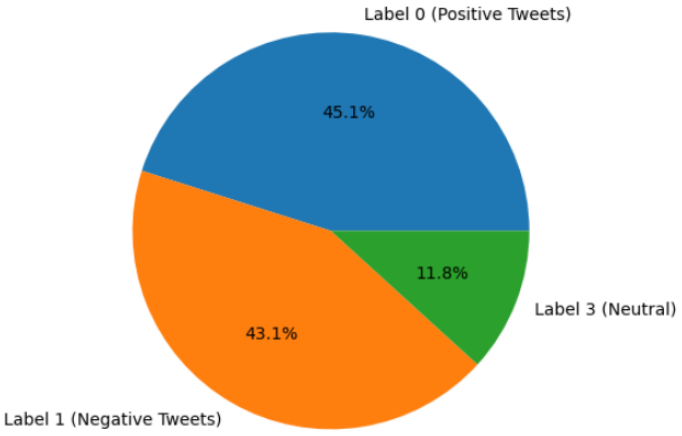


Now I have decided to analyze the text length of tweets and I noticed that tweets with 140 characters (more or Less) are predominant. After I decided to count words for each class of sentiment, I noticed that generally the predominant value is around 140 characters.




Then I decide to count the number of positive negative and neutral tweets and I noticed that the predominant class is Positive I noticed that tree class are almost unbalanced (neutral comments are very few respect the others)

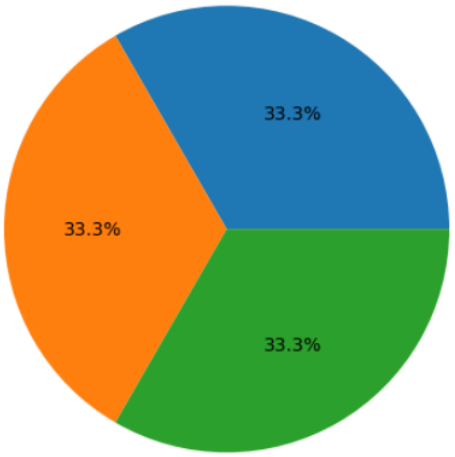


## Data Preprocessing

In the second part I start to make some preprocessing of the dataset. At the beginning I began to clean the feature "tweets". It was full of emoji, tags and other different symbols/words that would compromise the prediction phase. So I decided to use regular expressions using the modul **re** in order to clean all the not relevant symbols.A regular expression (or RE) specifies a set of strings that matches it; the functions in this module let you check if a particular string matches a given regular expression.
Finally I decided to encode the label class using a label encoding in order to increase speed in the training phase.

### SMOTE

The Synthetic Minority Oversampling Technique works by selecting examples that are close in the feature space, drawing a line between the examples in the feature space and drawing a new sample at a point along that line. Creating new synthetic samples helps to balance the 3 classes in order to increase accuracy.The above figures show the data after the smote procedure.




### TOKENIZATION

In the last section I split data in 80 % Train and 20% test set, which is a good split for having a robust model. To work with phrases and words we need to tokenize it. Tokenization is a common task in Natural Language Processing (NLP). Tokenization is a way of separating a piece of text into smaller units called tokens. Here, tokens can be either words, characters, or subwords. I decided to use the TFIDF Vectorizer by sklearn module. TF-IDF is an abbreviation for Term Frequency Inverse Document Frequency. This is a very common algorithm to transform text into a meaningful representation of numbers which is used to fit an algorithm for prediction. TF-IDF

measures the originality of a word by comparing the numbers of time a word appears in the doc with the numbers of doc the words appears in.

| | aa | aabb | aahl | aaptiv | aaron | aavitsland | ab | ababa | abaca | abad | ... |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... |
| 1 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... |
| 2 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... |
| 3 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... |
| 4 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 564 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... |
| 565 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... |
| 566 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... |
| 567 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... |
| 568 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... |

Tfidf matrix example

## Prediction

Now our data are ready for the model selection phase in which we have created different models and compared their performance. The models that we have chosen are :
**Logistic Regression, Random Forest, Support Vector Machine ,Multinomial Naive Bayes**
I decided to use these models because they are the most used estimators.
I excluded Decision Tree because it is the oldest estimator and the performance is slow.
I moreover excluded the NN because I would like to build RNN .
I tried with RNN in particular with **LSTM** for text elaboration but I have had some problems with Anaconda dependencies and I decided to leave it as a future development.

In order to increase the accuracy we have tuned hyperparameters for each model.
The accuracy of these models are around 50% but Logistic Regression seems to be the best estimator with an accuracy of 55%.

### LOGISTIC REGRESSION

This type of statistical model is often used for classification and predictive analytics. Logistic regression estimates the probability of an event occurring, such as voted or didn't vote, based on a given dataset of independent variables.
I build the model at first and then I decide to tune hyperparameters in order to increase the accuracy of predictions.

We have tuned the following hyperparameters :

```
space = dict()
space['solver'] = ['newton-cg', 'lbfgs','saga']
space['penalty'] = ['none','l2']
space['C'] = [1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1, 10, 100]
```

**Penalty :** Specify the norm of the penalty in our case none and L2.

**C :** smaller values specify stronger regularization.

**solver :** Algorithm to use in the optimization problem

The best Hyperparameters found are :

```
Best Hyperparameters: {'C': 1e-05, 'penalty': 'none', 'solver': 'newton-cg'}
```

The accuracy of logistic regression with turned hyperparameters is around 55%.

**RANDOM FOREST**

The random forest algorithm is an extension of the bagging method as it utilizes both bagging and feature randomness to create an uncorrelated forest of decision trees.
I build the model at first and then I decide to tune hyperparameters in order to increase the accuracy of predictions.

We have tuned the following hyperparameters :

```
parameters = {
    'n_estimators': [5,50],
    'max_depth': [2,10,20]
}
```

The best Hyperparameters found are :

```
BEST PARAMS: {'max_depth': 2, 'n_estimators': 5}
```

The accuracy of logistic regression with turned hyperparameters is around 45%.

**N_estimators are** : The number of trees in the forest.

**max_depth:** The maximum depth of the tree.

**SVM**

Support Vector Machine or SVM is a Supervised Learning algorithm, which is used for Classification as well as Regression problems.The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.
I build the model at first and then I decide to tune hyperparameters in order to increase the accuracy of predictions.
I have tuned the following hyperparameters :

```
param_grid = {'C': [0.1, 1],
              'gamma': [0.1, 0.01],
              'kernel': ['rbf']}
```

The best Hyperparameters found are :

```
BEST PARAMS: {'C': 0.1, 'gamma': 0.1, 'kernel': 'rbf'}
```

**C:**Regularization parameter. The strength of the regularization is inversely proportional to C. Must be strictly positive. The penalty is a squared l2 penalty.

**kernel** : Specifies the kernel type to be used in the algorithm. If none is given, 'rbf' will be used. If a callable is given it is used to pre-compute the kernel matrix from data matrices; that matrix should be an array of shape

**Gamma** : kernel coefficient for 'rbf', 'poly' and 'sigmoid'.

**MULTINOMIAL NAIVE BAYES**

The Multinomial Naive Bayes algorithm is a Bayesian learning approach popular in Natural Language Processing (NLP). The program guesses the tag of a text, such as an email or a newspaper story, using the Bayes theorem. It calculates each tag's likelihood for a given sample and outputs the tag with the greatest chance.
I build the model at first and then I decide to tune hyperparameters in order to increase the accuracy of predictions.
I have tuned the following hyperparameters :

```
parameters = {
'alpha': (1, 0.1, 0.01, 0.001, 0.0001, 0.00001),
'fit_prior':('True','False')
}
```

The best Hyperparameters found are :

```
MultinomialNB(alpha=1, fit_prior='True')
```

The accuracy of multinomial naive bayes with turned hyperparameters is around 30%.

**Alpha** : Additive (Laplace/Lidstone) smoothing parameter (0 for no smoothing).
**Fit_prior** : Whether to learn class prior probabilities or not. If false, a uniform prior will be used.

## RESULTS

The Best estimator is Logistic Regression with an accuracy of 55%.
The worst estimator is Naive Bayes classifier with an accuracy of 30%.

## FUTURE DEVELOPMENT

In the future I would like to use the RNN, Long Short Term Frequencies (LSTM) to predict the task.
I think that this model performs better because it is optimized for sequential data like text.