

## Authors

Palak Sharma, [sharma.pala@husky.neu.edu](mailto:sharma.pala@husky.neu.edu)

Dhanisha Phadte, [phadate.d@husky.neu.edu](mailto:phadate.d@husky.neu.edu)

Dharani Thirumalaisamy, [thirumalaisamy.d@husky.neu.edu](mailto:thirumalaisamy.d@husky.neu.edu)

### HEADING : Data Science in FinTech Industries(Cryptocurrency)

The goal of this blog post is to provide a brief introduction of FinTech Industries and an easy way to analyse the data using Python and R.

#### What is FinTech?

FinTech stands for Financial Technologies, and in its broadest definition, that's exactly what it is: technologies used and applied in the financial services sector, chiefly used by financial institutions themselves on the back end of their businesses.

#### Cryptocurrency – the next new buzz word

The word crypto comes from the fact that most of the cryptocurrencies (Yes there's more than one!), use cryptography principles for generation, transfers etc. Cryptocurrency is a form of digital money that is designed to be secure and, in many cases, anonymous. It is a currency associated with the internet that uses cryptography, the process of converting legible information into an almost uncrackable code, to track purchases and transfers.

#### Why is it required on first place?

Here are 7 ways how data science is at the core of the current transformation of the financial sector.

1. Payment and Transactions: Analysis and prediction of transaction volumes is key to enhance product value for customers. Data science enables better classification of payment records and thus allows banks to tailor additional services to their client's needs. This may vary from simple analytical features to more advanced features such as the integration of payment records and personal data to allow for recommendation, loyalty rewards and other forms of proactive engagement.

2. Credit Risk Evaluation : With the vision to make "credit accessible for more people", various FinTech startups are on the rush for clients and VC money. Their value proposition boils down

to a “faster and more accurate credit risk evaluation” process than at traditional banks, which enables them to reach a broader client base and minimize credit default rates.

3. Revenue and Debt Collection : Data science enables the utilization of powerful predictive models in order to optimize revenue and debt collection. Already at the moment of purchase it is possible to predict a probability of timely payment, thus making revenue collection more transparent.

4. Customer Journey Attribution : Customer Acquisition Costs and Customer Lifetime-Value are – as in most business models – key metrics for banks and financial service providers. Thus, minimizing churn rates and optimizing conversion rates are crucial activities within most financial organizations.

5. Fraud Detection and Prevention : While they have been on the agenda even before data science became a recognized term, fraud detection and prevention are still among the top priorities of FinTech executives

6. Portfolio Optimization and Asset Management : While portfolio optimization and asset management are generally among the more mature fields of application of data science, there are some exciting new avenues. Impact analyses and correlation with asset price developments are now executed in a seamless manner and allow for more creativity and insight. Testing and optimization of capital allocation strategies can be fully automated, which makes cuts on overhead costs feasible.

7. Corporate Compliance & Service Quality : Certainly not on the innovation end, but for larger institutions still of vital importance: implementation and tracking mechanisms for compliant behavior across the entire organization.

### **What are we dealing with?**

Different types of datasets available for cryptocurrency :

a. Cross-sectional dataset

It groups the available data into groups based on the similarities in the data and builds the model. In statistical terms , it is a dataset that is taken at a particular time. The error between 2 observations will be zero. The predicted output will be numeric. Example : the consumption expenditure of individuals for a period of one month.

b. Time-series dataset

It refers to the data that is taken over a period of time at specific and equal-intervals of time. The observations made are independent of one another. The predicted output is numeric. Usually regression method is used to create a model. This mainly has 4 subtypes : Date , Date Time ,Time, Timestamp . Example : Stock market closing rate for a time span of one year.

c. Panel dataset

In statistics and econometrics, panel data or longitudinal data are multi-dimensional data involving measurements over time. Panel data contain observations of multiple phenomena obtained over multiple time periods for the same firms or individuals. Time series and cross-sectional data can be thought of as special cases of panel data that are in one dimension only (one panel member or individual for the former, one time point for the latter).

### Where to find them???

A number of data sources are available on internet, where data will be available freely to perform analysis and predict.

- [www.kaggle.com/datasets](http://www.kaggle.com/datasets) ::: Formats Available : csv , JSON,SQLite
- <http://archive.ics.uci.edu/ml/datasets> ::: Formats Available : csv
- <https://elitedatascience.com/datasets> ::: Formats Available : csv
- <https://www.quandl.com/> ::: Formats Available : csv, JSON,xml
- <https://www.census.gov/data.html> ::: Formats Available : csv, sas
- <https://relational.fit.cvut.cz/> ::: Formats Available : MySQL database

### CSV format :

```
import urllib2
import csv
url = ("website from where the data is taken")
data = urllib2.urlopen(url)
```

We can use ScraperWiki library instead of urllib .

```
import csv
import scraperwiki
url = ("website from where the data is taken")
data = scraperwiki.scrape(url)
```

### JSON format :

1. Create a base class to handle the website :

```
import re, json
import mechanize
import urlparse
import urllib
from bs4 import BeautifulSoup
```

2. Add a scrape method :

```
def scrape(self):
    self.open_search_openings_page()
    self.submit_search_form()
    self.scrape_jobs()
```

3. Submit the search form :

```
def submit_search_form(self):
    self.soupify_form(soup=None, form_name='aspnetForm')
    self.br.select_form('aspnetForm')
    self.br.submit()
    self.soup = BeautifulSoup(self.br.response().read())
```

4. It runs the form through BeautifulSoup and sets the current page in HTML form . No using the attributes we can load the JSON data in to it :

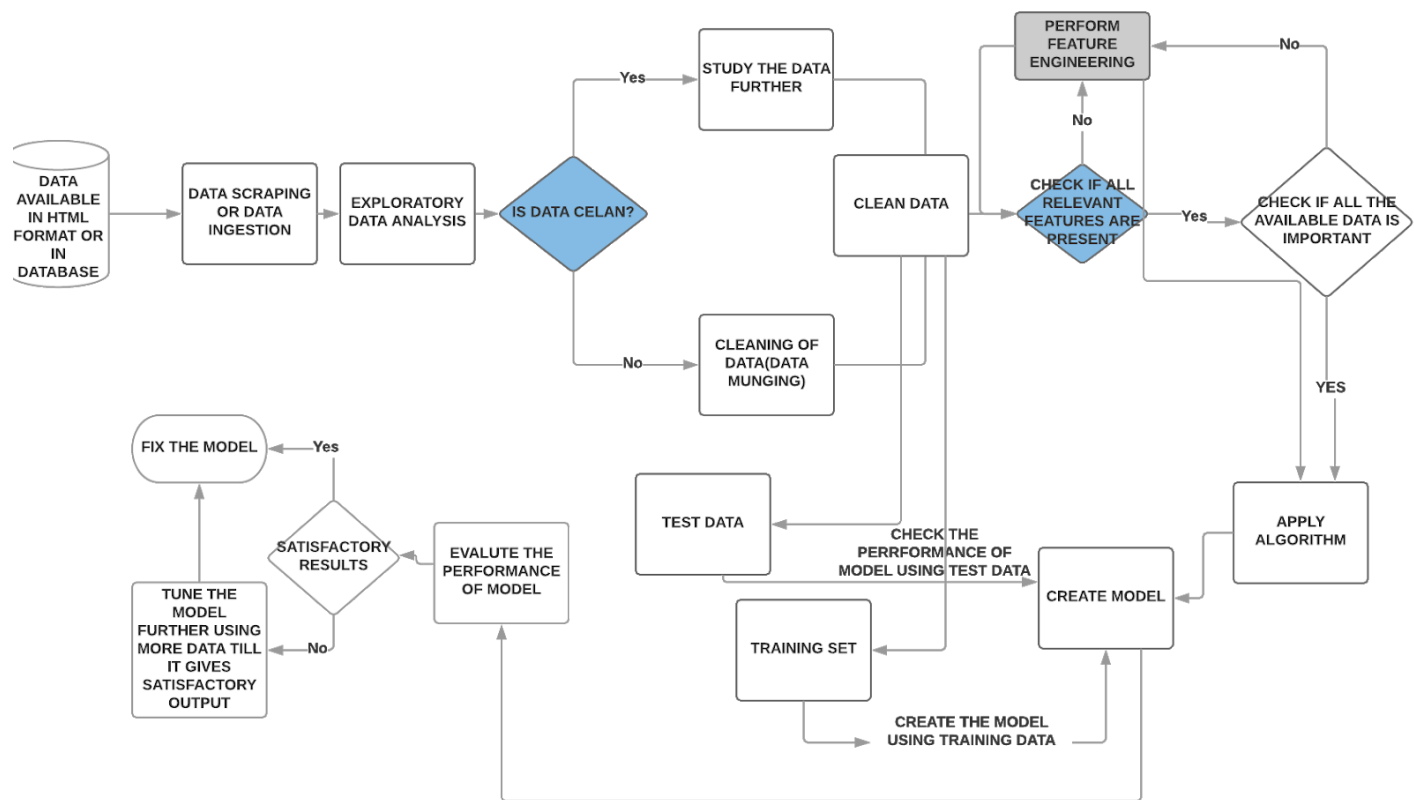
```
def scrape_data(self):
    while not self.seen_all_data():
        t = 'ctl00_MainContent_GridFormatter_json_tabledata'
        i = self.soup.find('input', id=t)
        j = json.loads(i['value'])
```

**XML format :**

**PARSING :**

- Parsing the XML package has 2 basic models - DOM & SAX.
- Document Object Model (DOM) in which tree is stored internally as C, or as regular R objects
- Uses XPath to query nodes of interest, to extract info.
- It writes recursive functions to "visit" nodes which extracts information as it descends tree.
- It extracts information to R data structures via handler functions that are called for particular XML elements by matching XML name.
- For processing very large XML files with low-level state machine via R handler functions closures are used.

## FLOW DIAGRAM OF DATA ANALYSIS



## How to access?

### 1. Retrieve Bitcoin data using Quandl API

Quandl offers free Bitcoin exchange rates for 30+ currencies from a variety of exchanges. Quandl's simple API gives access to Bitcoin exchanges and daily Bitcoin values. Accessing Bitcoin data via the API is no different than the mechanism for all data on Quandl.

## Import the Dependencies at the Top of the Notebook

```
In [13]: #to extract the data using QuadAPI
import os # OS module provides function that allows you to interface with the operating system that Pyt
import numpy as np #provides a high-performance multidimensional array object,
import pandas as pd
import pickle # Pickling is a way to convert a python object (list, dict, etc.) into a character stream
#serializing and de-serializing a Python object structure
import quandl #Quandl requires NumPy (v1.8 or above) and pandas (v0.14 or above) to work.
#Package for quandl API access
#Basic wrapper to return datasets from the Quandl website as Pandas dataframe objects with a timeseries
```

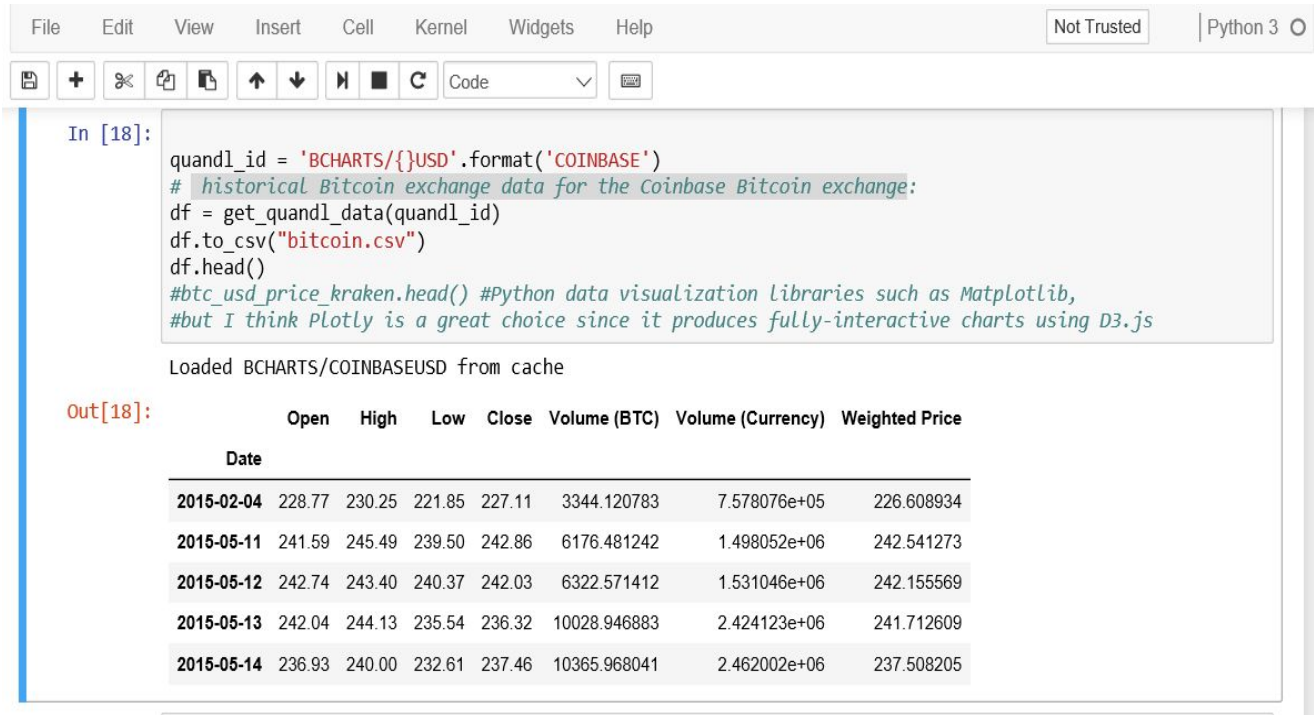
To get Bitcoin pricing data using Quandl's free Bitcoin API.

Define a function to download and cache datasets from Quandl.

```
In [15]: def get_quandl_data(quandl_id):
    '''Download and cache Quandl dataseries'''
    #taking data series from cached path
    cache_path = '{}.pkl'.format(quandl_id).replace('/', '-') # creat pkl file as quandl_id.pkl and repl
    #cache_path = BCHARTS-KRAKENUSD.pkl
    try:
        f = open(cache_path, 'rb') #open the pkl file to read
        df = pickle.load(f) # load the data from file into df object
        print('Loaded {} from cache'.format(quandl_id)) # loaded from the pkl file which was cached
    except (OSError, IOError) as e:
        print('Downloading {} from Quandl'.format(quandl_id))
        # getting data from quandl website using quandl module and API as pandas dataframe
        df = quandl.get(quandl_id, returns="pandas")
        df.to_pickle(cache_path)# storing the data from df to pkl file
        print('Cached {} at {}'.format(quandl_id, cache_path))
    return df
```

Using pickle to serialize and save the downloaded data as a file, which will prevent our script from re-downloading the same data each time we run the script. The function will return the data as a Pandas dataframe.

Pull Historical Bitcoin exchange data for the Coinbase Bitcoin exchange



```
In [18]:
quandl_id = 'BCHARTS/{}USD'.format('COINBASE')
# historical Bitcoin exchange data for the Coinbase Bitcoin exchange:
df = get_quandl_data(quandl_id)
df.to_csv("bitcoin.csv")
df.head()
#btc_usd_price_kraken.head() #Python data visualization libraries such as Matplotlib,
#but I think Plotly is a great choice since it produces fully-interactive charts using D3.js

Loaded BCHARTS/COINBASEUSD from cache

Out[18]:
```

	Open	High	Low	Close	Volume (BTC)	Volume (Currency)	Weighted Price
Date							
2015-02-04	228.77	230.25	221.85	227.11	3344.120783	7.578076e+05	226.608934
2015-05-11	241.59	245.49	239.50	242.86	6176.481242	1.498052e+06	242.541273
2015-05-12	242.74	243.40	240.37	242.03	6322.571412	1.531046e+06	242.155569
2015-05-13	242.04	244.13	235.54	236.32	10028.946883	2.424123e+06	241.712609
2015-05-14	236.93	240.00	232.61	237.46	10365.968041	2.462002e+06	237.508205

## 2. Fetching data using Web Scrapping

- When performing data science tasks, it's common to want to use data found on the internet. You'll usually be able to access this data in csv format, or via an Application Programming Interface (API). However, there are times when the data you want can only be accessed as part of a web page. In cases like this, you'll want to use a technique called web scrapping to get the data from the web page into a format you can work with in your analysis.
- It is very necessary to be familiar with the HTML structure before you start to scrap the data. Scrapping can be done with Python using BeautifulSoup and requests libraries.pandas, and matplotlib library is used to perform some simple analysis using.
- Use urllib library to request the web base from server by the passing the url page.

## EXPLORATORY DATA ANALYSIS on the data we scarped via API (Python)

Exploratory Data Analysis (EDA) is an approach/philosophy for data analysis that employs a variety of techniques (mostly graphical) to :

1. maximize insight into a data set;
2. uncover underlying structure;

3. extract important variables;
4. detect outliers and anomalies;
5. test underlying assumptions;
6. develop parsimonious models; and
7. determine optimal factor settings.

Lets get started with some python codes leading to EDA on the data we scraped using APIs in the previous section :

A. Know the datatypes you are dealing with and club them into one category

```
In [1]: import pandas as pd
```

```
In [2]: data= pd.read_csv(r"C:\Users\palak\Documents\Python\bitcoin.csv")
```

```
In [3]: data.dtypes
```

```
Out[3]: Date                object
Open                float64
High                float64
Low                 float64
Close               float64
Volume (BTC)        float64
Volume (Currency)   float64
Weighted Price       float64
dtype: object
```

```
In [7]: def get_var_category(series):
        unique_count = series.nunique(dropna = False)
        total_count = len(series)
        if pd.api.types.is_numeric_dtype(series):
            return 'Numerical'
        elif pd.api.types.is_datetime64_dtype(series):
            return 'Date'
        elif unique_count == total_count:
            return 'Text (Unique)'
        else:
            return 'Categorical'
    def print_categories(df):
        for column_name in df.columns:
            print(column_name, " :", get_var_category(df[column_name]))
    print_categories(data)

Date : Text (Unique)
Open : Numerical
High : Numerical
Low : Numerical
Close : Numerical
Volume (BTC) : Numerical
Volume (Currency) : Numerical
```



B. Calculate if any missing values

```
In [11]: #data['Volume'].astype(str).astype(int)
lengtho = len(data['Open'])
print(lengtho)
```

659360

```
In [13]: count = data['Open'].count()
print(count)
```

658996

```
In [14]: missing_rows = length-count
perc_miss_data = float(missing_rows / length)
perc_miss_data = "{0:.1f}%".format(perc_miss_data*100)
print(perc_miss_data)
```

0.1%

C. Computing Summary Metrics for the overall data

```
In [5]: print("Minimim Value : ", data["Open"].min())
```

Minimim Value : 1e-10

```
In [6]: print("Maximum Value : ", data["Open"].max())
```

Maximum Value : 99917.8

```
In [7]: print(data["Open"].mode())
```

0 0.000002  
dtype: float64

```
In [9]: mean = data["Open"].mean()
print(mean)
```

35.58682048554945

```
In [11]: median = data["Open"].median()
print(median)
```

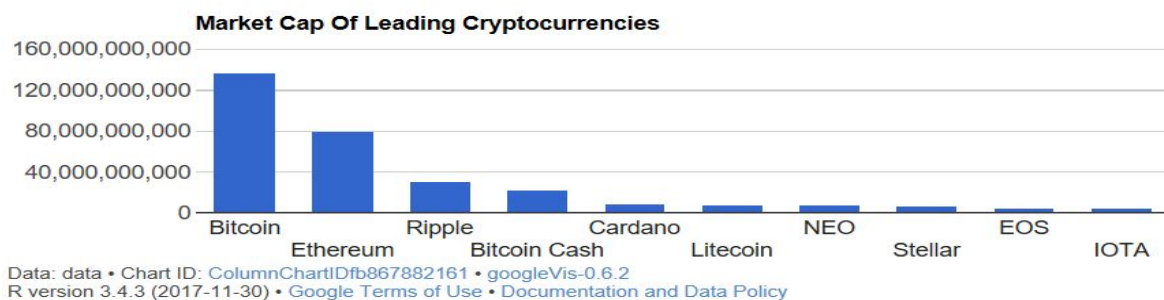
0.006664

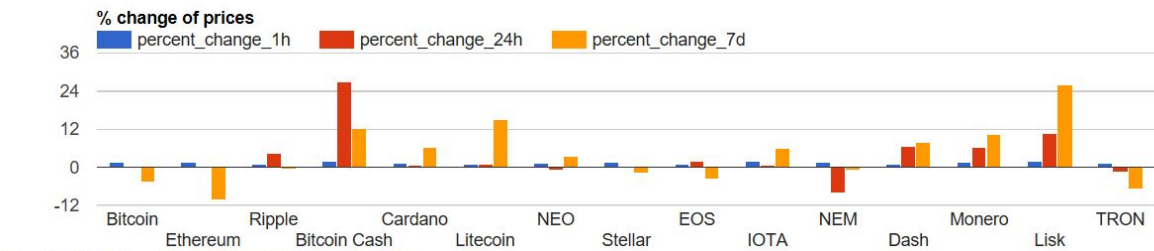
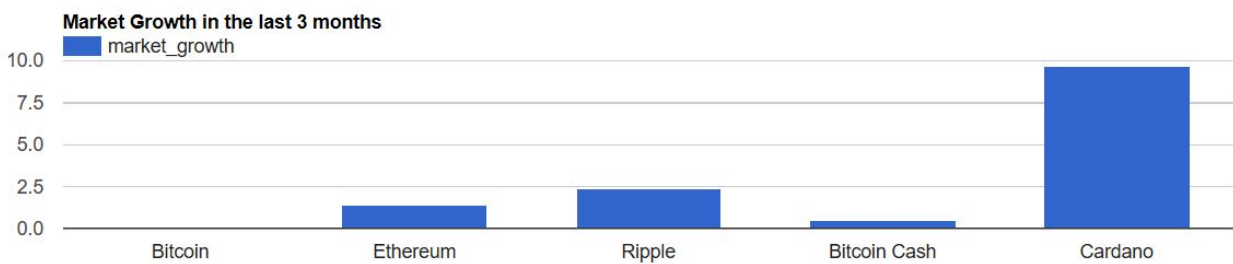
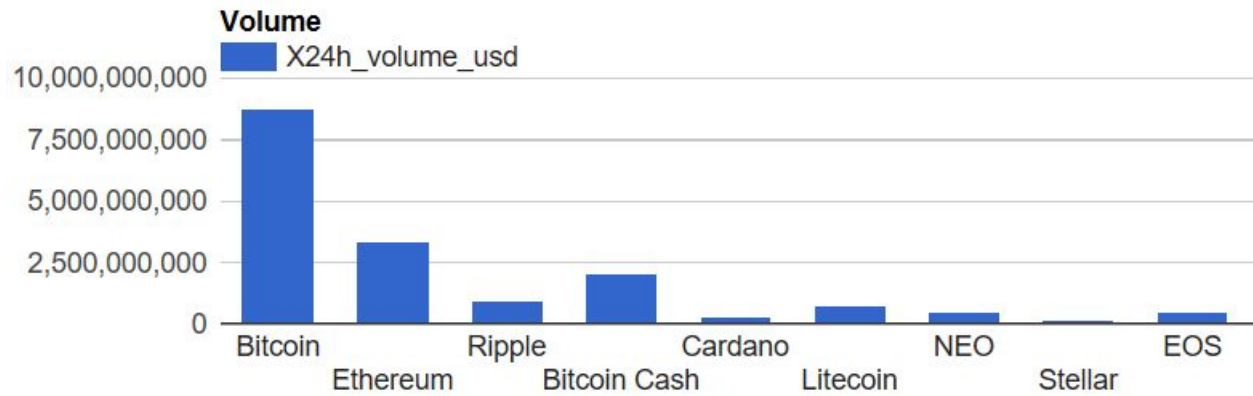
## EXPLORATORY DATA ANALYSIS on real time data (R Studio)

```

1
2 library(coinmarketcapr) #to get the cryptocurrencies market cap prices from "coin market cap"
3 library(formatrR) # to reformat R code to increase readability
4 library(yaml) #convert R data int yaml(to create ordered maps)
5 library(googlevis) #R interface to google chart
6 library(knitr) #packet for dynamic report generation
7 data_sample <- get_marketcap_ticker_all()
8 mapply(class,data_sample)
9
10 for(i in c(4:ncol(data_sample))) {
11   data_sample[,i] <- as.double(data_sample[,i])
12 }
13
14 mark <- gviscolumnchart(data_sample[1:15, ], "name", "market_cap_usd", options = list(title = "Market Cap of Leading Cryptocurrencies",
15   legend = "left"))
16 plot(mark)

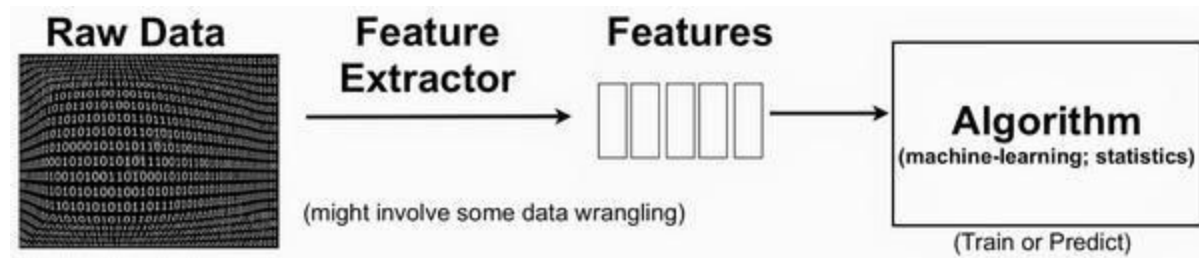
```

[illegible]



## FEATURE ENGINEERING :

Feature engineering is defined as a process of transforming raw data into features that represent the problems to models that one creates , which results in a better model with better accuracy for future predictions with new set of data.



Steps involved :

Feature extraction - reducing the dimensionality of the available data set so that it gives better result .

Out of this not all features will be of importance to us. So we have to select features that are more relevant to the analysis that one is performing. So , the next step is Feature Selection.

Sometimes , the feature which is at most important to the analysis will not be available. In that case one has to develop a feature. It is called Feature Construction. This feature should be related to the other existing features.

Once the data is with us it is equally important to know about the features - Feature Learning.

Process involved :

Brainstorming - coming up with a list of features which might fit best into the model.

Selection of feature - picking up the most relevant one

Evaluation of models - checking if the new data fits in the model that is already created .

Example :

This is a simple example that explains feature engineering.

Let's consider we have item\_colour as one feature which has red , yellow, green and no colour as attributes.

In this case we can create another feature called value\_of\_colour and set it to 1 if the item has colour and to 0 if it has no colour. This will give us a better idea about the data that we have.