[DATA ANALYSIS AND VISUALIZATION ON REAL WORLD DATASETS]

February 6

2022

A DETAILED REPORT ON ANALYSIS AND VISUALIZATION ON A CRYPTO DATASET.

THIS PROJECT IS PREAPARED BY THE STUDENTS OF THE UNIVERSITY OF KARACHI (DEPARTMENT OF MATHEMATICS).

DRGREE PROGRAMME NAME:

FINANCIAL MATHEMATICS

COURSE NAME:

PROGRAMING LANGUAGE II (FM- 410)

COURSE INSTRUCTOR:

SYED UMAID AHMED

GROUP MEMBERS:

OMEME SHAKIR PATEL

TOOBA KHAN

S. M. MEESUM ABBAS

ABSTRACT

Data analysis is becoming a very influential tool for decision-making today both in industry and academia. The incorporation of data-driven concepts in the core curriculum would be very beneficial to graduates, making them competitive in today's market. The main objective of this work is to develop tools and case studies to increase student's understanding and appreciation of data for decision making, introduce students to basic data analytics and machine learning methods, introduce students to basic data visualization tools, and exhibit that the interactions between rigorous simulations and data could lead to improved solutions. This report about data analysis using Python will be very useful to the ones looking for inner workings of basic data processing.

The dataset that we used for our analysis is on cryptocurrency. The report contains two main parts: the first section of the report gives brief overview of data science, data analysis, and python; and the second part of the report seeks to provide a concise yet comprehensive analysis of the cryptocurrency market. The main purpose of the report is to investigate data by utilizing logical techniques, procedures, calculations and frameworks to understand the relevant information.

Cryptocurrencies are built on the principles of blockchain technology or what is more accurately known as distributed ledger technologies (DLTs). There are theoretically two types of DLTs, open and closed, more formally, 'permission-less (open)' and 'permissioned (closed)' blockchain.

We picked different features from the dataset to understand different trends and patterns between 14 cryptocurrencies. With help of different libraries in python we analyzed and produced different outputs each time also visualized our results. We not only understood the data subjectively but also objectively, for example we not only analyzed data as a whole but also looked individually that how the data features works for only one asset.

Table of Contents

SECTION 1: INTRODUCTION

1.1 WHAT IS DATA?

Data are individual facts, statistics, or items of information, often numeric. In a more technical sense, data are a set of values of qualitative or quantitative variables about one or more persons or objects.

Although the terms "data" and "information" are often used interchangeably, this term has distinct meanings. In some popular publications, data are sometimes said to be transformed into information when they are viewed in context or in post-analysis. However, in academic treatments of the subject data are simply units of information. Data are used in scientific research, businesses management (e.g., sales data, revenue, profits, stock price), finance, governance (e.g., crime rates, unemployment rates, literacy rates), and in virtually every other form of human organizational activity (e.g., censuses of the number of homeless people by non-profit organizations).

1.2 DATA PHRASES IN TECHNOLOGY

Data has become the forefront of many mainstream conversations about technology. New innovations constantly draw commentary on data, how we use and analyze it, and broader implications for those effects. As a result, the popular IT vernacular has come to include a number of phrases new and old:

- **Big data:** A massive volume of structured and unstructured data that is too large to process using traditional database and software technologies.
- **Big data analytics:** The process of collecting, organizing, and synthesizing large sets of data to discover patterns or other useful information.
- **Data center:** Physical or virtual infrastructure used by enterprises to house computer, storage, and networking systems and components for the company's IT needs.
- **Data integrity:** The validity of data, which can be compromised in a number of ways including human error or transfer errors.
- **Data miner:** A software application that monitors and/or analyzes the activities of a computer, and subsequently its user, to collect information.
- **Data mining:** A class of database applications that look for hidden patterns in a group of data that can be used to predict/anticipate future behavior.
- **Data warehouse:** A data management system that uses data from multiple sources to promote business intelligence.
- **Database:** A collection of data points organized in a way that is easily maneuvered by a computer system.
- **Metadata:** Summary information about a data set.
- **Raw data:** Information that has been collected but not formatted or analyzed.
- **Structured data:** Any data that resides in a fixed field within a record or file, including data contained in relational databases and spreadsheets.
- **Unstructured data:** Information that does not reside in a traditional column-row database like structured data.

1.3 What is Data Science?

Data science combines multiple fields, including statistics, scientific methods, and artificial intelligence (AI), and data analysis, to extract value from data. Those who practice data science are called data

scientists, and they combine a range of skills to analyze data collected from the web, smartphones, customers, sensors, and other sources to derive actionable insights.

1.4 What's the difference between Data Science, Artificial Intelligence, and Machine Learning?

To better understand data science and how you can harness, it's equally important to know other terms related to the field, such as artificial intelligence (AI) and machine learning. Often, you'll find that these terms are used interchangeably, but there are nuances.

Here's a simple breakdown:

- **AI** means getting a computer to mimic human behavior in some way.
- **Data science** is a subset of AI, and it refers more to the overlapping areas of statistics, scientific methods, and data analysis—all of which are used to extract meaning and insights from data.
- **Machine learning** is another subset of AI, and it consists of the techniques that enable computers to figure things out from the data and deliver AI applications. And for good measure, we'll throw in another definition.
- **Deep learning** is a subset of machine learning that enables computers to solve more complex problems.

1.5 Tools for Data Science

Building, evaluating, deploying, and monitoring machine learning models can be a complex process. That's why there's been an increase in the number of data science tools. Data scientists use many types of tools, but one of the most common is open source notebooks, which are web applications for writing and running code, visualizing data, and seeing the results—all in the same environment.

Some of the most popular notebooks are Jupiter, RStudio, and Zeppelin. Notebooks are very useful for conducting analysis, but have their limitations when data scientists need to work as a team. Data science platforms were built to solve this problem.

To determine which data science tool is right for you, it's important to ask the following questions: What kind of languages do your data scientists use? What kind of working methods do they prefer? What kind of data sources are they using?

For example, some users prefer to have a data source-agnostic service that uses open source libraries. Others prefer the speed of in-database, machine learning algorithms.

1.6 What is a Data Scientist?

As a specialty, data science is young. It grew out of the fields of statistical analysis and data mining. *The Data Science Journal* debuted in 2002, published by the International Council for Science: Committee on Data for Science and Technology. By 2008 the title of data scientist had emerged, and the field quickly took off. There has been a shortage of data scientists ever since, even though more and more colleges and universities have started offering data science degrees.
A data scientist's duties can include developing strategies for analyzing data, preparing data for analysis, exploring, analyzing, and visualizing data, building models with data using programming languages, such as Python and R, and deploying models into applications.

The data scientist doesn't work solo. In fact, the most effective data science is done in teams. In addition to a data scientist, this team might include a business analyst who defines the problem, a data engineer

who prepares the data and how it is accessed, an IT architect who oversees the underlying processes and infrastructure, and an application developer who deploys the models or outputs of the analysis into applications and products.

1.7 How Data Science Is Applied?

Data science incorporates tools from multiple disciplines to gather a data set, process, and derive insights from the data set, extract meaningful data from the set, and interpret it for decision-making purposes. The disciplinary areas that make up the data science field include mining, statistics, machine learning, analytics, and programming.

Data mining applies algorithms to the complex data set to reveal patterns that are then used to extract useful and relevant data from the set. Statistical measures or predictive analytics use this extracted data to gauge events that are likely to happen in the future based on what the data shows happened in the past.

Machine learning is an artificial intelligence tool that processes mass quantities of data that a human would be unable to process in a lifetime. Machine learning perfects the decision model presented under predictive analytics by matching the likelihood of an event happening to what actually happened at a predicted time.

Using analytics, the data analyst collects and processes the structured data from the machine learning stage using algorithms. The analyst interprets, converts, and summarizes the data into a cohesive language that the decision-making team can understand. Data science is applied to practically all contexts and, as the data scientist's role evolves, the field will expand to encompass data architecture, data engineering, and data administration.

1.8 What Is Data Analysis?

Data analysis is defined as a process of cleaning, transforming, and modeling data to discover useful information for business decision-making. The purpose of Data Analysis is to extract useful information from data and taking the decision based upon the data analysis.
A simple example of Data analysis is whenever we take any decision in our day-to-day life is by thinking about what happened last time or what will happen by choosing that particular decision. This is nothing but analyzing our past or future and making decisions based on it. For that, we gather memories of our past or dreams of our future. So that is nothing but data analysis. Now same thing analyst does for business purposes, is called Data Analysis.

1.9 Data Analysis Tools

Data analysis tools make it easier for users to process and manipulate data, analyze the relationships and correlations between data sets, and it also helps to identify patterns and trends for interpretation.

There are several data analysis tools available in the market, each with its own set of functions. The selection of tools should always be based on the type of analysis performed and the type of data worked. Here is a list of a few compelling tools for Data Analysis.

- **Excel:** It has various compelling features, and with additional plugins installed, it can handle a massive amount of data. So, if you have data that does not come near the significant data margin, Excel can be a versatile tool for data analysis.
- **Tableau**: It falls under the BI Tool category, made for the sole purpose of data analysis. The essence of Tableau is the Pivot Table and Pivot Chart and works towards representing data in

the most user-friendly way. It additionally has a data cleaning feature along with brilliant analytical functions.

- **Power BI:** It initially started as a plugin for Excel, but later on, detached from it to develop in one of the most data analytics tools. It comes in three versions: Free, Pro, and Premium. Its PowerPivot and DAX language can implement sophisticated advanced analytics similar to writing Excel formulas.
- **Fine Report:** Fine Report comes with a straightforward drag and drops operation, which helps design various reports and build a data decision analysis system. It can directly connect to all kinds of databases, and its format is similar to that of Excel. Additionally, it also provides a variety of dashboard templates and several self-developed visual plug-in libraries.
- **R & Python:** These are programming languages that are very powerful and flexible. R is best at statistical analysis, such as normal distribution, cluster classification algorithms, and regression analysis. It also performs individual predictive analyses like customer behavior, spending, items preferred by him based on his browsing history, and more. It also involves concepts of machine learning and artificial intelligence.
- **SAS:** It is a programming language for data analytics and data manipulation, which can easily access data from any source. SAS has introduced a broad set of customer profiling products for web, social media, and marketing analytics. It can predict their behaviors, manage, and optimize communications.

1.10 Types of Data Analysis

There are a half-dozen popular types of data analysis available today, commonly employed in the worlds of technology and business. They are:

- **Diagnostic Analysis:** Diagnostic analysis answers the question, "Why did this happen?" Using insights gained from statistical analysis (more on that later!), analysts use diagnostic analysis to identify patterns in data. Ideally, the analysts find similar patterns that existed in the past, and consequently, use those solutions to resolve the present challenges hopefully.

- **Predictive Analysis:** Predictive analysis answers the question, "What is most likely to happen?" By using patterns found in older data as well as current events, analysts predict future events. While there's no such thing as 100 percent accurate forecasting, the odds improve if the analysts have plenty of detailed information and the discipline to research it thoroughly.

- **Prescriptive Analysis:** Mix all the insights gained from the other data analysis types, and you have prescriptive analysis. Sometimes, an issue can't be solved solely with one analysis type, and instead requires multiple insights.

- **Statistical Analysis:** Statistical analysis answers the question, "What happened?" This analysis covers data collection, analysis, modeling, interpretation, and presentation using dashboards. The statistical analysis breaks down into two sub-categories:

    1. **Descriptive:** Descriptive analysis works with either complete or selections of summarized numerical data. It illustrates means and deviations in continuous data and percentages and frequencies in categorical data.

    2. **Inferential:** Inferential analysis works with samples derived from complete data. An analyst can arrive at different conclusions from the same comprehensive data set just by choosing different samplings.

- **Text Analysis:** Also called "data mining," text analysis uses databases and data mining tools to discover patterns residing in large datasets. It transforms raw data into useful business information. Text analysis is arguably the most straightforward and the most direct method of data analysis.

1.11 What Is the Data Analysis Process?

Answering the question "what is data analysis" is only the first step. Now we will look at how it's performed. The data analysis process, or alternately, data analysis steps, involves gathering all the information, processing it, exploring the data, and using it to find patterns and other insights. The process consists of:

- **Data Requirement Gathering:** Ask yourself why you're doing this analysis, what type of data analysis you want to use, and what data you are planning on analyzing.
- **Data Collection:** Guided by the requirements you've identified, it's time to collect the data from your sources. Sources include case studies, surveys, interviews, questionnaires, direct observation, and focus groups. Make sure to organize the collected data for analysis.
- **Data Cleaning:** Not all of the data you collect will be useful, so it's time to clean it up. This process is where you remove white spaces, duplicate records, and basic errors. Data cleaning is mandatory before sending the information on for analysis.
- **Data Analysis:** Here is where you use data analysis software and other tools to help you interpret and understand the data and arrive at conclusions. Data analysis tools include Excel, Python, R, Looker, Rapid Miner, Chartio, Metabase, Redash, and Microsoft Power BI.
- **Data Interpretation:** Now that you have your results, you need to interpret them and come up with the best courses of action, based on your findings.
- **Data Visualization:** Data visualization is a fancy way of saying, "graphically show your information in a way that people can read and understand it." You can use charts, graphs, maps, bullet points, or a host of other methods. Visualization helps you derive valuable insights by helping you compare datasets and observe relationships.

1.12 Why is Data Analysis Important?

Here is a list of reasons why data analysis is such a crucial part of doing business today.

- **Better Customer Targeting:** You don't want to waste your business's precious time, resources, and money putting together advertising campaigns targeted at demographic groups that have little to no interest in the goods and services you offer. Data analysis helps you see where you should be focusing your advertising efforts.
- **You Will Know Your Target Customers Better:** Data analysis tracks how well your products and campaigns are performing within your target demographic. Through data analysis, your business can get a better idea of your target audience's spending habits, disposable income, and most likely areas of interest. This data helps businesses set prices, determine the length of ad campaigns, and even help project the quantity of goods needed.
- **Reduce Operational Costs:** Data analysis shows you which areas in your business need more resources and money, and which areas are not producing and thus should be scaled back or eliminated outright.

- **Better Problem-Solving Methods:** Informed decisions are more likely to be successful decisions. Data provides businesses with information. You can see where this progression is leading. Data analysis helps businesses make the right choices and avoid costly pitfalls.

- **You Get More Accurate Data:** If you want to make informed decisions, you need data, but there's more to it. The data in question must be accurate. Data analysis helps businesses acquire relevant, accurate information, suitable for developing future marketing strategies, business plans, and realigning the company's vision or mission.

1.13 Python for Data Analysis

Python is the internationally acclaimed programming language to help in handling your data in a better manner for a variety of causes.

We live in the digital era of high technologies, smart devices, and mobile solutions. Data is an essential aspect of any enterprise and business. It's crucial to gather, process, and analyze the data flow and to do that as quickly and accurately as possible. Nowadays, the data volume can be large, which makes information handling time-consuming and expensive. Due to this precise reason, the data science industry is growing at a rapid pace, creating new vacancies and possibilities.

Now lots of new approaches to recording, storing, and analyzing data have emerged to extract cognitive info effectively, gain insights and knowledge. Not only can you choose from a list of options, features, and tools, but you can also utilize them to process operations as well as leverage techniques to convert information into the knowledge and insights by means of reports or visualization.

There is a host of prominent programming languages to utilize for data reduction. C, C++, R, Java, JavaScript, and Python are a few among them. Each one offers unique features, options, and tools that suit the different demands depending on your needs. Some are better than others for specific industry needs. For example, one industry survey states Python has established itself as a leading choice for developing fintech software and other application areas.

There are two main factors that make Python a widely-used programming language in scientific computing, in particular:

- The stunning ecosystem.
- A great number of data-oriented feature packages that can speed up and simplify data processing, making it time-saving.

In addition to that, Python is initially utilized for actualizing data analysis. It is among those languages that are being developed on an ongoing basis. Thereby, Python is called the topmost language with a high potential in the data science field more than other programming languages.

1.14 What Makes Python a Fantastic Option for Data Analysis?

Python is a cross-functional, maximally interpreted language that has lots of advantages to offer. The object-oriented programming language is commonly used to streamline large complex data sets. Over and above, having a dynamic semantics plus unmeasured capacities of *RAD (rapid application development)*, Python is heavily utilized to script as well. There is one more way to apply Python – as a coupling language.
Another Python's advantage is high readability that helps engineers to save time by typing fewer lines of code for accomplishing the tasks. Being fast, Python jibes well with data analysis. And that's due to heavy support; availability of a whole slew of open-source libraries for different purposes, including but not limited to scientific computing. Therefore, it's not surprising at all that it's claimed to be the preferred programming language for data science. There is a scope of unique features provided that

makes Python *a-number-one* option for data analysis. Seeing is believing. So, just let's overlook each option one by one.

- **Easy to Learn**: Being involved in development for web services, mobile apps, or coding, you have a notion that Python is widely recognized thanks to its clear syntax and readability. Yes, these are the most famous language characteristics. More than that, a low and, thus, fast learning curve is the next pre-eminence of Python when comparing it with older languages on offer. C#, Ruby, Java, others in the roll are much harder to master, especially for entry-level programmers. Python is focused on simplicity as well as readability, providing a host of helpful options for data analysts/scientists simultaneously. Thus, newbies can easily utilize its pretty simple syntax to build effective solutions even for complex scenarios. Most notably, that's all with fewer lines of code used. That's why it's an ideal tool for beginners.
- **Well-Supported**: Having the experience of using some tools for free, you probably know that it is a challenge to get decent support. That's not the case with Python, though. Despite the high simplicity, there can be situations when you still need help with Python. Being in widespread use in industrial alongside academic areas, Python has a broad array of helpful libraries with tons of helpful and support materials. The great benefit is that all the libraries are available at no cost. The higher the popularity of the language is, the more cognitive info about real user experience is contributed. Hence, you've got access to the user-contributed codes, Stack Overflow, documentation, mailing lists, and so forth. Users around the world can ask more experienced programmers for advice and help when it's needed.
- **Flexibility**: The cool options don't end there. So, let's observe another reason why Python is really a fantastic option for data processing. Another strong feature of the language is the hyper flexibility that makes Python highly requested among data scientists and analysts. Due to that, it's possible to build data models, systematize data sets, create ML-powered algorithms, web services, and apply data mining to accomplish different tasks in a brief period of time. Yes, such an advantage makes Python an ideal solution that the data science industry needs.
- **Scalability**: This Python's feature is described right after the flexibility, not by accident, but because it is closely connected with the previous option. Comparing with other languages like R, Go, and Rust, Python is much faster and more scalable. Therefore, Python is good for different usages in various fields that can solve a wide range of problems. That's why many companies have migrated to Python. Additionally, this language is perfect for the RAD of all kinds (as we've already stated above). What's more, the data analysis is in the list of the industries where the language can be applied successfully.
- **Huge Libraries Collection**: As we have already mentioned, Python is one of the most supported languages nowadays. It has a long list of totally free libraries available for all the users. That's a key factor that gives a strong push for Python at all, and in the data science, too. If you're involved in the field, more than likely, you are acquainted with such names as *Pandas, SciPy, Stats Models*, and other libraries that are intensively utilized in the data science community. Noteworthy is that the libraries constantly grow, providing robust solutions. Herewith, you can easily find a solution needed hassle-free without additional expenses.
- **Exceeding Python Community**: It's a kind of open-source language. That means you get at least two strong advantages. Python is free, plus it employs a community-based model for development. Yes, this issue and the previous paragraph are inextricably linked too. Besides of open-source libraries such as *Statistics, Data Visualization, and Manipulation, ML*, and more, Python has a massive community base with pieces of training and forums available. That's the way people all over the globe can exchange experiences, thoughts, and knowledge, as well as provide solutions, codes, and ask questions. We recommend you to go to the Python Package Index in case you are eager to learn more about the multifarious Python's aspects.
- **Graphics and Visualization Tools**: It's a well-known fact that visual information is much easier to understand, operate, and remember. Here is another portion of a piece of good news for you. There is a pack of diverse visualization options available. That makes Python a must-

have tool not only for data analysis but for all data science. You can make the data more accessible and easier-to-use by means of creating various charts and graphics, as well as web-ready interactive plots. Yes, Python provides you with the capability to get a good sense of data.

- **Extended Pack of Analytics Tools Available**: Straight after you gather data, you're to handle it. Python suits this purpose supremely well. So, seeking for the perfect tool for complex data processing or self-service analytics, we can't but mention Python's built-in data analytics tools. Dozens of data mining companies over the globe utilize Python to reduce data. Python also has the ability to penetrate patterns easily as well as correlate information in large sets and give better insights alongside other critical matrices in evaluating performance.
- **Bottom Line**: The success of your business directly depends on the ability to extract knowledge and insights from data to make effective strategic decisions, stay competitive, and make progress. Python is the internationally acclaimed programming language to help in handling your data in a better manner for a variety of causes. First and foremost, it is one of the most easy-to-learn languages, pretty simple in use, with the best price ever (actually, it's free!), with an excellent pack of features provided. Though Python is an open-source language, it remains well-supported by a huge community. All that makes Python perfect for newbies in the programming. In addition to that, Python is scalable and flexible enough to be applied in different fields and for various purposes. Thanks to the pack of graphical options along with visualization tools that make data more accessible, Python is named as the most preferred language among the data analysts and data scientists. What's more, it evolves constantly and becomes more effective, multi-feature, and tight.

1.15 Python Libraries for Data Analysis

NUMPY

One of the most fundamental packages in Python, NumPy is a general-purpose array processing package. It provides high-performance multidimensional array objects and tools to work with the arrays. NumPy is an efficient container of generic multi-dimensional data.

NumPy's main object is the homogeneous multidimensional array. It is a table of elements or numbers of the same datatype, indexed by a tuple of positive integers. In NumPy, dimensions are called axes and the number of axes is called rank. NumPy's array class is called ndarray aka array.

NumPy is used to process arrays that store values of the same datatype. NumPy facilitates math operations on arrays and their vectorization. This significantly enhances performance and speeds up the execution time correspondingly. Some of the functions of NumPy are:

- Basic array operations: add, multiply, slice, flatten, and reshape, index arrays.
- Advanced array operations: stack arrays, split into sections, broadcast arrays.
- Work with Date Time or Linear Algebra.
- Basic Slicing and Advanced Indexing in NumPy Python.

PANDAS

Pandas is an open-source Python package that provides high-performance, easy-to-use data structures and data analysis tools for the labeled data in Python programming language. Pandas stand for Python Data Analysis Library. Pandas is a perfect tool for data wrangling or munging. It is designed for quick and easy data manipulation, reading, aggregation, and visualization.

Pandas take data in a CSV or TSV file or a SQL database and create a Python object with rows and columns called a data frame. The data frame is very similar to a table in statistical software, say Excel or SPSS.

Functions of Pandas are as follows:

- Indexing, manipulating, renaming, sorting, merging data frame.
- Update, add, and delete columns from a data frame.
- Impute missing files, handle missing data or NANs.
- Plot data with histogram or box plot.

This makes Pandas a foundation library in learning Python for Data Science.

MATPLOTLIB

This is undoubtedly a quintessential Python library. You can create stories with the data visualized with Matplotlib. Another library from the SciPy Stack, Matplotlib plots 2D figures. Matplotlib is the plotting library for Python that provides an object-oriented API for embedding plots into applications. It is a close resemblance to MATLAB embedded in Python programming language.

Histogram, bar plots, scatter plots, area plot to pie plot, Matplotlib can depict a wide range of visualizations. With a bit of effort and tint of visualization capabilities, with Matplotlib, you can create just any visualizations like line plots, scatter plots, area plots, bar charts, histograms, pie charts, stem plots, contour plots, quiver plots, spectrograms.

Matplotlib also facilitates labels, grids, legends, and some more formatting entities.

SCIPY

The SciPy library is one of the core packages that make up the SciPy stack. Now, there is a difference between SciPy Stack and SciPy, the library. SciPy builds on the NumPy array object and is part of the stack which includes tools like Matplotlib, Pandas, and SymPy with additional tools.

SciPy library contains modules for efficient mathematical routines as linear algebra, interpolation, optimization, integration, and statistics. The main functionality of the SciPy library is built upon NumPy and its arrays. SciPy makes significant use of NumPy.

SciPy uses arrays as its basic data structure. It has various modules to perform common scientific programming tasks as linear algebra, integration, calculus, ordinary differential equations, and signal processing.

SEABORN

It is defined as the data visualization library based on Matplotlib that provides a high-level interface for drawing attractive and informative statistical graphics. Putting it simply, seaborn is an extension of Matplotlib with advanced features.

So, what is the difference between Matplotlib and Seaborn? Matplotlib is used for basic plotting; bars, pies, lines, scatter plots and stuff whereas, seaborn provides a variety of visualization patterns with less complex and fewer syntax.

Functions of Seaborn are as follows:

- Determine relationships between multiple variables (correlation)
- Observe categorical variables for aggregate statistics.
- Analyze uni-variate or bi-variate distributions and compare them between different data subsets

- Plot linear regression models for dependent variables
- Provide high-level abstractions, multi-plot grids
- Seaborn is a great second-hand for R visualization libraries like corrplot and ggplot.

SCIKIT LEARN

Introduced to the world as a Google Summer of Code project, Scikit Learn is a robust machine learning library for Python. It features ML algorithms like SVMs, random forests, k-means clustering, spectral clustering, mean shift, cross-validation and more. Even NumPy, SciPy and related scientific operations are supported by Scikit Learn with Scikit Learn being a part of the SciPy Stack.

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. Supervised learning models like Naive Bayes to grouping unlabeled data such as KMeans, Scikit learn would be your go-to.

Functions of Scikit Learn are as follows:

- Classification: Spam detection, image recognition
- Clustering: Drug response, Stock price
- Regression: Customer segmentation, Grouping experiment outcomes
- Dimensionality reduction: Visualization, Increased efficiency
- Model selection: Improved accuracy via parameter tuning
- Pre-processing: Preparing input data as a text for processing with machine learning algorithms.

Scikit Learn focuses on modeling data; not manipulating data. We have NumPy and Pandas for summarizing and manipulation.

SECTION 2: ANALYSIS ON A CRYPTO DATASET

2.1 About Data

The dataset is about the cryptocurrency market collected by a G-Research company. G-Research is Europe's leading quantitative finance research firm. They hire the brightest minds in the world to tackle some of the biggest questions in finance. They pair this expertise with machine learning, big data, and some of the most advanced technology available to predict movements in financial markets.

In the first file we have access to millions of rows of minute-by-minute cryptocurrency trading data of 14 different assets simultaneously. In the second file we have the names of 14 different assets, their weight (popularity) and the asset id already assigned to them.

2.2 The Cryptocurrency Market

First, a quick introduction to the crypto world. A cryptocurrency is a digital currency that is created and managed through the use of advanced encryption techniques known as cryptography. Cryptocurrencies have become an extremely popular and volatile market, delivering massive returns (as well as losses) to investors. Thousands of cryptocurrencies have been created with a few major ones that many of you will have heard of including Bitcoin (BTC), Ether (ETH) or Dogecoin (DOGE).

Cryptocurrencies are traded extensively across crypto-exchanges, with an average volume of $41 billion traded daily over the last year, according to CryptoCompare (as of 25th July 2021).Changes in prices between different cryptocurrencies are highly interconnected. For example, Bitcoin has historically been a major driver of price changes across cryptocurrencies but other coins also impact the market.

Some of the limitations that cryptocurrencies presently face – such as the fact that one's digital fortune can be erased by a computer crash, or that a virtual vault may be ransacked by a hacker – may be overcome in time through technological advances. What will be harder to surmount is the basic paradox that bedevils cryptocurrencies – the more popular they become, the more regulation and government scrutiny they are likely to attract, which erodes the fundamental premise for their existence.

While the number of merchants who accept cryptocurrencies has steadily increased, they are still very much in the minority. For cryptocurrencies to become more widely used, they have to first gain widespread acceptance among consumers. However, their relative complexity compared to conventional currencies will likely deter most people, except for the technologically adept.

A cryptocurrency that aspires to become part of the mainstream financial system may have to satisfy widely divergent criteria. It would need to be mathematically complex (to avoid fraud and hacker attacks) but easy for consumers to understand; decentralized but with adequate consumer safeguards and protection; and preserve user anonymity without being a conduit for tax evasion, money laundering and other nefarious activities. Since these are formidable criteria to satisfy, is it possible that the most popular cryptocurrency in a few years' time could have attributes that fall in between heavily-regulated fiat currencies and today's cryptocurrencies? While that possibility looks remote, there is little doubt that as the leading cryptocurrency at present, Bitcoin's success (or lack thereof) in dealing with the challenges it faces may determine the fortunes of other cryptocurrencies in the years ahead.

2.3 Assets Description

The cryptocurrency market has evolved erratically and at unprecedented speed over the course of its short lifespan. There short details of the 14 different cryptocurrencies are given below:

- Bitcoin is a decentralized digital currency, without a central bank or single administrator, which can be sent from user to user on the peer-to-peer bitcoin network without the need for intermediaries.
- Ethereum is a blockchain-based platform that is best known for its cryptocurrency, ETH. The blockchain technology that powers Ethereum enables secure digital ledgers to be publicly created and maintained.
- Litecoin (LTC) is a cryptocurrency that was founded in 2011, two years after Bitcoin, by a former Google engineer named Charlie Lee. Like Bitcoin, Litecoin is based on an open-source global payment network that is not controlled by any central authority. Litecoin differs from Bitcoin in aspects like faster block generation rate and use of Scrypt as a proof-of-work (POW) scheme.
- The binance coin is used as a utility token for the Binance exchange and allows users to pay for transactions and trading fees at a lower rate than they would be with other tokens. Binance uses the process of token "burns," meaning they use the profit from token sales to repurchase more BNB and then burn (destroy) them.
- Dogecoin is a cryptocurrency created by software engineers Billy Markus and Jackson Palmer, who decided to create a payment system as a "joke", making fun of the wild speculation in cryptocurrencies at the time. It is considered both the first "meme coin", and, more specifically, the first "dog coin".
- Cardano is a public blockchain platform. It is open-source and decentralized, with consensus achieved using proof of stake. It can facilitate peer-to-peer transactions with its internal cryptocurrency, ADA. Cardano was founded in 2015 by Ethereum co-founder Charles Hoskinson.
- IOTA is an open-source distributed ledger and cryptocurrency designed for the Internet of things. It uses a directed acyclic graph to store transactions on its ledger, motivated by a potentially higher scalability over blockchain based distributed ledgers.
- Bitcoin Cash is a cryptocurrency that is a fork of Bitcoin. Bitcoin Cash is a spin-off or altcoin that was created in 2017. In November 2018, Bitcoin Cash split further into two cryptocurrencies: Bitcoin Cash and Bitcoin SV.
- Dai/maker is a stable coin cryptocurrency which aims to keep its value as close to one United States dollar as possible through an automated system of smart contracts on the Ethereum blockchain.
- TRON is a decentralized, open-source blockchain-based operating system with smart contract functionality, proof-of-stake principles as its consensus algorithm and a cryptocurrency native to the system, known as Tronix.
- Stellar, or Stellar Lumens, is an open source, decentralized protocol for digital currency to fiat money low-cost transfers which allows cross-border transactions between any pair of currencies was released on July 31,2014
- EOS launched In June 2018 after an initial coin offering that raised $4.1 billion in crypto for Block.one, the company that developed the open-source software called EOS.IO that is used on the platform.
- Monero uses a public distributed ledger with privacy-enhancing technologies that obfuscate transactions its supply is unlimited and it block reward is 1.16 XMR right now.

2.4 Dataset Features

We can see the different features included in the dataset. Specifically, the features included per asset are the following:

- **Timestamp**: All timestamps are returned as second UNIX timestamps (the number of seconds elapsed since 1970-01-01 00:00:00.000 UTC). Timestamps in this dataset are multiple of 60, indicating minute-by-minute data.
- **Asset ID**: The asset ID corresponding to one of the cryptocurrencies (e.g. Asset ID = 1 for Bitcoin). The mapping from Asset ID to crypto asset is contained in asset_details.csv.
- **Count**: Total number of trades in the time interval (last minute).
- **Open**: Opening price of the time interval (in USD).
- **High**: Highest price reached during time interval (in USD).
- **Low**: Lowest price reached during time interval (in USD).
- **Close**: Closing price of the time interval (in USD).
- **Volume**: Quantity of asset bought or sold, displayed in base currency USD.
- **VWAP**: The average price of the asset over the time interval, weighted by volume. VWAP is an aggregated form of trade data.
- **Target**: Residual log-returns for the asset over a 15 minute horizon.

2.5 Analysis and Results

Step 1

DATA ANALYSIS ON A CRYPRO DATASET

Importing Libraries

```
In [1]: # First we are gonna import all the necessary.

        import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        %matplotlib inline
        import seaborn as sns
```

Step 2

Importing Our Data Files

```
In [2]: # By the help of pandas we are now reading our data files.

        df = pd.read_csv('supplemental_train.csv')
        fd=pd.read_csv('asset_details.csv ')
```

Step 3

```
In [3]: # Having a glance at the first fifty records of the first dataset.

        df.head(50)
```

Out[3]:

	timestamp	Asset_ID	Count	Open	High	Low	Close	Volume	VWAP	Target
0	1623542400	3	1201.0	1.478556	1.486030	1.478000	1.483681	6.547996e+05	1.481439	-0.002594
1	1623542400	2	1020.0	580.306667	583.890000	579.910000	582.276667	1.227988e+03	581.697038	-0.009143
2	1623542400	0	626.0	343.789500	345.108000	343.640000	344.598000	1.718833e+03	344.441729	-0.004525
3	1623542400	1	2888.0	35554.289632	35652.464650	35502.670000	35602.004286	1.638115e+02	35583.469303	0.003096
4	1623542400	4	433.0	0.312167	0.312600	0.311920	0.312208	5.855774e+05	0.312154	0.001426
5	1623542400	5	359.0	4.832550	4.845900	4.822900	4.837583	4.714355e+04	4.836607	-0.000579
6	1623542400	7	541.0	55.223080	55.494000	55.182000	55.344680	6.625202e+03	55.298816	-0.003998
7	1623542400	6	2186.0	2371.194286	2379.200000	2369.670000	2374.380714	1.214129e+03	2374.335307	0.002565
8	1623542400	8	35.0	1.003150	1.019800	0.987300	1.003300	7.061928e+03	1.002936	-0.005097
9	1623542400	9	560.0	161.933429	162.480000	161.730000	162.214714	1.485009e+03	162.231310	0.000686
10	1623542400	10	61.0	2939.862750	2952.160000	2936.230000	2947.078025	9.584785e+00	2945.110614	-0.004899
11	1623542400	13	229.0	0.068132	0.068240	0.068038	0.068158	3.046438e+06	0.068158	-0.003036
12	1623542400	12	383.0	0.327973	0.329272	0.327650	0.328829	5.364911e+05	0.328582	-0.003314
13	1623542400	11	123.0	243.137500	243.810000	242.960000	243.532500	3.079589e+02	243.452697	-0.001360
14	1623542460	3	672.0	1.482410	1.483759	1.479200	1.482043	2.858286e+05	1.481495	-0.003286
15	1623542460	2	1251.0	581.800000	585.590000	580.380000	582.358333	1.405285e+03	583.451389	-0.006528
16	1623542460	0	468.0	344.253500	344.700000	343.620000	344.089500	1.217362e+03	344.188716	0.001328

Step 4

```
In [4]: #Let's see what we have in the second dataset file.

        fd
```

Out[4]:

	Asset_ID	Weight	Asset_Name
0	2	2.397895	Bitcoin Cash
1	0	4.304065	Binance Coin
2	1	6.779922	Bitcoin
3	5	1.386294	EOS.IO
4	7	2.079442	Ethereum Classic
5	6	5.894403	Ethereum
6	9	2.397895	Litecoin
7	11	1.609438	Monero
8	13	1.791759	TRON
9	12	2.079442	Stellar
10	3	4.406719	Cardano
11	8	1.098612	IOTA
12	10	1.098612	Maker
13	4	3.555348	Dogecoin

Step 5

Preprocessing Data

```
In [5]: #Lookin at the number of rows and columns in the dataset

        df.shape
```

```
Out[5]: (2015112, 10)
```

Step 6

```
In [6]: #Getting the frequency of most aseets in this data.

        df['Asset_ID'].value_counts()
```

```
Out[6]: 1     143998
        2     143998
        3     143998
        4     143998
        5     143998
        6     143998
        7     143998
        9     143998
        12    143998
        13    143993
        11    143956
        0     143921
        8     143717
        10    143543
        Name: Asset_ID, dtype: int64
```

Step 7

```
In [7]: #Reading each column of the dataset.

        df.columns
```

```
Out[7]: Index(['timestamp', 'Asset_ID', 'Count', 'Open', 'High', 'Low', 'Close',
               'Volume', 'VWAP', 'Target'],
              dtype='object')
```

Step 8

```
In [8]: # Let's see if there are any missing values in our dataset.

        df.isna().values.any()
```

```
Out[8]: True
```

Step 9

```
In [10]: #Checking missing values in each column of our data.

         df.isna().any()
```

```
Out[10]: timestamp    False
         Asset_ID     False
         Count        False
         Open         False
         High         False
         Low          False
         Close        False
         Volume       False
         VWAP         False
         Target        True
         dtype: bool
```

Step 10

```
In [10]: #Checking missing values in each column of our data.

         df.isna().any()
```

```
Out[10]: timestamp    False
         Asset_ID     False
         Count        False
         Open         False
         High         False
         Low          False
         Close        False
         Volume       False
         VWAP         False
         Target        True
         dtype: bool
```

Step 11

```
In [11]: #Droping the unnesecary columns.

         df = df.drop(['Target','timestamp'],axis=1)
```

Step 12

```
In [12]: #Checking again for the missing values in each column of our data.

         df.isna().any()
```

```
Out[12]: Asset ID    False
         Count       False
         Open        False
         High        False
         Low         False
         Close       False
         Volume      False
         VWAP        False
         dtype: bool
```

Step 13

```
In [13]: # Confirming that the columns have been dropped or not.

         df.head()
```

Out[13]:

	Asset_ID	Count	Open	High	Low	Close	Volume	VWAP
0	3	1201.0	1.478556	1.48603	1.47800	1.483681	654799.561103	1.481439
1	2	1020.0	580.306667	583.89000	579.91000	582.276667	1227.988328	581.697038
2	0	626.0	343.789500	345.10800	343.64000	344.598000	1718.832569	344.441729
3	1	2888.0	35554.289632	35652.46465	35502.67000	35602.004286	163.811537	35583.469303
4	4	433.0	0.312167	0.31260	0.31192	0.312208	585577.410442	0.312154

Step 14

In [14]:
```
# A descriptive statiscal analysis of the dataset.

df.describe()
```

Out[14]:

	Asset_ID	Count	Open	High	Low	Close	Volume	VWAP
count	2.015112e+06	2.015112e+06	2.015112e+06	2.015112e+06	2.015112e+06	2.015112e+06	2.015112e+06	2.015112e+06
mean	6.499139e+00	5.163656e+02	3.408915e+03	3.412519e+03	3.405373e+03	3.408919e+03	3.386617e+05	3.408879e+03
std	4.031342e+00	1.060273e+03	1.052321e+04	1.053392e+04	1.051262e+04	1.052323e+04	1.387742e+06	1.052310e+04
min	0.000000e+00	1.000000e+00	4.652540e-02	4.695000e-02	4.637000e-02	4.652140e-02	9.610000e-06	4.654178e-02
25%	3.000000e+00	8.200000e+01	9.160500e-01	9.381000e-01	8.970000e-01	9.160000e-01	1.457512e+02	9.159005e-01
50%	6.000000e+00	2.010000e+02	7.554037e+01	7.578950e+01	7.535660e+01	7.553027e+01	1.352555e+03	7.553674e+01
75%	1.000000e+01	5.390000e+02	5.637334e+02	5.642800e+02	5.631500e+02	5.637578e+02	1.021959e+05	5.637045e+02
max	1.300000e+01	8.747800e+04	5.288442e+04	5.295647e+04	5.285604e+04	5.288209e+04	1.261933e+08	5.289832e+04

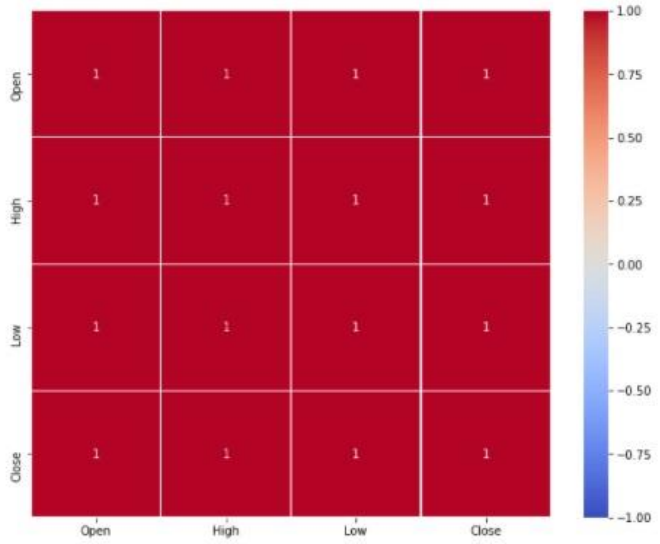
Step 15

Data Exploration

1. Show that how the different prices for all the 14 assets correlate.

In [15]:
```
#Let us create a correlation plot on the actual different category of prices for all the 14 assets.

plt.figure(figsize=(10,8))
sns.heatmap(df[['Open','High','Low','Close']].corr(), vmin=-1.0, vmax=1.0, annot=True, cmap='coolwarm', linewidths=0.1)
plt.show()
```



Step 16

In [17]:
```
# Adding the asset names column according to the asset id.

Assets= ['Binance Coin', 'Bitcoin', 'Bitcoin Cash', 'Cardano', 'Dogecoin', 'EOS.IO', 'Ethereum', 'Ethereum Classic', 'IOTA',
        'Litecoin','Maker','Monero','Stellar','TRON']
df_new['Asset_Names'] = Assets
df_new
```

Out[17]:

Asset_ID	Count	Open	High	Low	Close	Volume	VWAP	Asset_Names
0	453.032018	366.452795	366.784035	366.107393	366.450339	1.114897e+03	366.445454	Binance Coin
1	1905.615703	40643.194661	40685.846152	40601.500966	40643.264847	8.073969e+01	40642.768582	Bitcoin
2	171.830262	567.988415	568.530901	567.437762	567.990972	1.581713e+02	567.985001	Bitcoin Cash
3	721.523056	1.808935	1.811549	1.806243	1.808950	3.358984e+05	1.808903	Cardano
4	775.669294	0.249300	0.249689	0.248947	0.249299	2.079518e+06	0.249295	Dogecoin
5	316.173030	4.470072	4.475532	4.464574	4.470081	3.326959e+04	4.470026	EOS.IO
6	1571.874797	2707.255325	2710.406453	2704.112950	2707.262326	1.006284e+03	2707.237822	Ethereum
7	274.079668	55.040199	55.128570	54.957270	55.040195	2.656985e+03	55.040021	Ethereum Classic
8	73.938943	1.009116	1.028026	0.990583	1.009150	2.108542e+04	1.009097	IOTA
9	326.979812	156.308956	156.503782	156.117641	156.309523	1.137219e+03	156.307372	Litecoin
10	42.199627	2962.282641	2965.458924	2959.029529	2962.260816	4.882707e+00	2962.247638	Maker
11	67.012427	247.637204	247.921439	247.346961	247.640111	1.126452e+02	247.635525	Monero
12	209.294455	0.300870	0.301264	0.300475	0.300872	2.525648e+05	0.300866	Stellar
13	317.361288	0.076188	0.076264	0.076109	0.076188	2.010745e+06	0.076187	TRON

Step 17

In [17]:
```
# Adding the asset names column according to the asset id.

Assets= ['Binance Coin', 'Bitcoin', 'Bitcoin Cash', 'Cardano', 'Dogecoin', 'EOS.IO', 'Ethereum', 'Ethereum Classic', 'IOTA',
        'Litecoin','Maker','Monero','Stellar','TRON']
df_new['Asset_Names'] = Assets
df_new
```

Out[17]:

Asset_ID	Count	Open	High	Low	Close	Volume	VWAP	Asset_Names
0	453.032018	366.452795	366.784035	366.107393	366.450339	1.114897e+03	366.445454	Binance Coin
1	1905.615703	40643.194661	40685.846152	40601.500966	40643.264847	8.073969e+01	40642.768582	Bitcoin
2	171.830262	567.988415	568.530901	567.437762	567.990972	1.581713e+02	567.985001	Bitcoin Cash
3	721.523056	1.808935	1.811549	1.806243	1.808950	3.358984e+05	1.808903	Cardano
4	775.669294	0.249300	0.249689	0.248947	0.249299	2.079518e+06	0.249295	Dogecoin
5	316.173030	4.470072	4.475532	4.464574	4.470081	3.326959e+04	4.470026	EOS.IO
6	1571.874797	2707.255325	2710.406453	2704.112950	2707.262326	1.006284e+03	2707.237822	Ethereum
7	274.079668	55.040199	55.128570	54.957270	55.040195	2.656985e+03	55.040021	Ethereum Classic
8	73.938943	1.009116	1.028026	0.990583	1.009150	2.108542e+04	1.009097	IOTA
9	326.979812	156.308956	156.503782	156.117641	156.309523	1.137219e+03	156.307372	Litecoin
10	42.199627	2962.282641	2965.458924	2959.029529	2962.260816	4.882707e+00	2962.247638	Maker
11	67.012427	247.637204	247.921439	247.346961	247.640111	1.126452e+02	247.635525	Monero
12	209.294455	0.300870	0.301264	0.300475	0.300872	2.525648e+05	0.300866	Stellar
13	317.361288	0.076188	0.076264	0.076109	0.076188	2.010745e+06	0.076187	TRON

Step 18

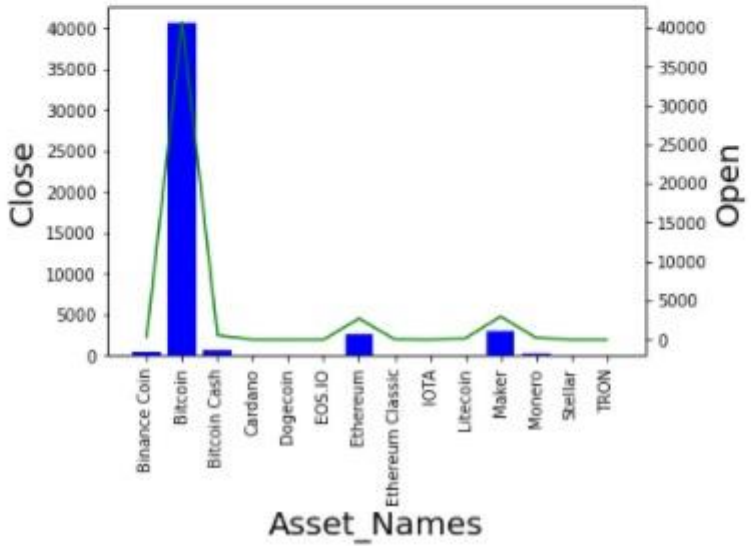
```
In [18]: # Now we are gonna visualize our results.

         fig, ax1 = plt.subplots()

         ax2 = ax1.twinx()
         ax1.bar(df_new.Asset_Names, df_new.Open, color='b')
         ax2.plot(df_new.Asset_Names,df_new.Close, color='g')

         ax1.set_xlabel('Asset_Names', size= 20)
         ax1.set_ylabel('Close', size= 20)
         ax2.set_ylabel('Open', size= 20)
         ax1.set_xticklabels(df_new.Asset_Names, rotation= 'vertical', size= 10)

         fig.show()
```



Step 19

3. Determine the two most popoular assets of all the 14 assets through weight percentage wise.

In [19]:
```
# Arranging the weights of the asset from highest to lowest.

fd.sort_values("Weight", ascending=False)
```

Out[19]:

	Asset_ID	Weight	Asset_Name
2	1	6.779922	Bitcoin
5	6	5.894403	Ethereum
10	3	4.406719	Cardano
1	0	4.304065	Binance Coin
13	4	3.555348	Dogecoin
0	2	2.397895	Bitcoin Cash
6	9	2.397895	Litecoin
4	7	2.079442	Ethereum Classic
9	12	2.079442	Stellar
8	13	1.791759	TRON
7	11	1.609438	Monero
3	5	1.386294	EOS.IO
11	8	1.098612	IOTA
12	10	1.098612	Maker

Step 20

In [20]:
```
# Let us see the percentage of weights given to each of the assets.


fd["Weight_percentage"] = (fd["Weight"] / fd["Weight"].sum()) * 100
fd.sort_values("Weight", ascending=False)
```

Out[20]:

	Asset_ID	Weight	Asset_Name	Weight_percentage
2	1	6.779922	Bitcoin	16.584998
5	6	5.894403	Ethereum	14.418848
10	3	4.406719	Cardano	10.779686
1	0	4.304065	Binance Coin	10.528574
13	4	3.555348	Dogecoin	8.697068
0	2	2.397895	Bitcoin Cash	5.865715
6	9	2.397895	Litecoin	5.865715
4	7	2.079442	Ethereum Classic	5.086716
9	12	2.079442	Stellar	5.086716
8	13	1.791759	TRON	4.382990
7	11	1.609438	Monero	3.936996
3	5	1.386294	EOS.IO	3.391144
11	8	1.098612	IOTA	2.687418
12	10	1.098612	Maker	2.687418

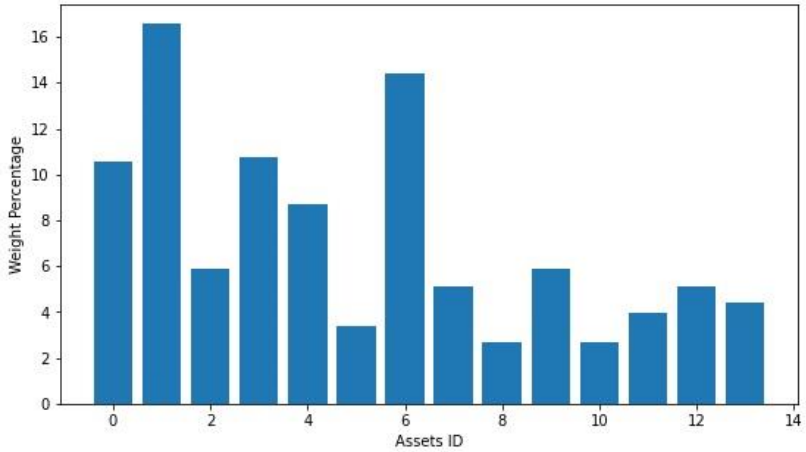
Step 21

```
In [21]: #Visulalizing our results.

         plt.figure(figsize=(9,5))

         plt.xlabel("Assets ID")
         plt.ylabel("Weight Percentage")

         plt.bar(fd.Asset_ID,fd.Weight_percentage)
         plt.show()
```



Step 22

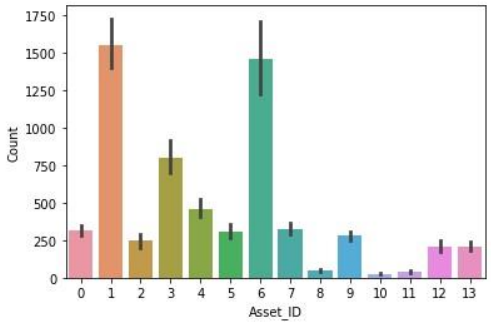
4. Prove the popularity of those two assets through count (total number of trades in the time interval).

```
In [22]: #Comparing the counts with asset id.

         sns.barplot("Asset_ID","Count",data=df.head(1000),orient="v")
```

```
C:\Users\dell\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variables as
s: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without
keyword will result in an error or misinterpretation.
  warnings.warn(
```

```
Out[22]: <AxesSubplot:xlabel='Asset_ID', ylabel='Count'>
```



Step 23

5. Show the regression trend between the VWAP and Volume for Bitcoin and Ethereum only.

In [23]:
```
#Extracting the bitcoin data.

btc =df[df.Asset_ID==1].reset_index(drop=True)
btc
```

Out[23]:

	Asset_ID	Count	Open	High	Low	Close	Volume	VWAP
0	1	2888.0	35554.289632	35652.464650	35502.67	35602.004286	163.811537	35583.469303
1	1	2006.0	35596.771429	35621.000000	35533.38	35555.397143	93.363659	35584.861196
2	1	3531.0	35550.271250	35576.590000	35402.87	35488.287500	220.535164	35480.068897
3	1	2901.0	35478.867162	35503.460134	35381.01	35423.490000	118.802511	35438.243466
4	1	1968.0	35419.640459	35476.000000	35384.64	35453.218571	115.564721	35434.811663
...	...	...	...	...	...	...	...	...
143993	1	1940.0	42983.780000	43001.850849	42878.26	42899.012857	56.850913	42935.489499
143994	1	2026.0	42904.197143	42932.000000	42840.16	42860.005714	80.993326	42879.576084
143995	1	1986.0	42859.385714	42887.500000	42797.20	42827.020000	65.677734	42844.090693
143996	1	4047.0	42839.012802	43042.160000	42818.10	43017.277143	138.335477	42935.761938
143997	1	2698.0	43009.961250	43048.510000	42961.64	43002.505000	128.206820	43011.414052

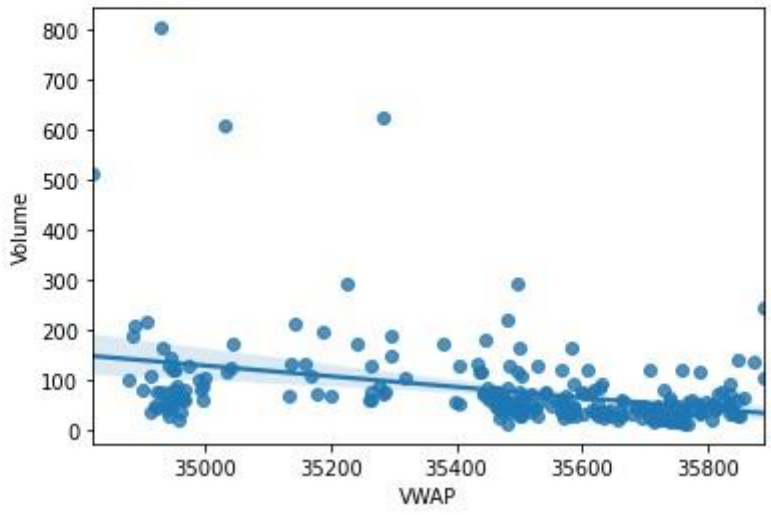
143998 rows × 8 columns

Step 24

In [24]:
```
# Visualizing the results for bitcoin.

sns.regplot(x='VWAP', y='Volume', data=btc.head(250))
```

Out[24]: <AxesSubplot:xlabel='VWAP', ylabel='Volume'>



Step 25

In [25]:
```
#Let us inspect the data for second most popular asset, Ethereum.


eth =df[df.Asset_ID==6].reset_index(drop=True)
eth
```

Out[25]:

	Asset_ID	Count	Open	High	Low	Close	Volume	VWAP
0	6	2186.0	2371.194286	2379.200000	2369.67	2374.380714	1214.128692	2374.335307
1	6	1261.0	2373.970101	2375.350000	2369.37	2371.790000	786.738453	2372.809830
2	6	1856.0	2370.880011	2371.950076	2363.00	2365.590000	764.080469	2367.128372
3	6	2624.0	2365.769427	2367.500000	2359.01	2360.505714	2253.662759	2362.394059
4	6	3084.0	2360.050000	2364.670000	2356.29	2362.864536	3501.689885	2361.092854
...	...	...	...	...	...	...	...	...
143993	6	2162.0	2973.728686	2976.100000	2962.09	2964.711429	751.256906	2968.339295
143994	6	1976.0	2965.461446	2967.950000	2958.05	2960.845714	729.113672	2963.499199
143995	6	2262.0	2960.321429	2962.600000	2954.14	2957.398571	807.935362	2958.464868
143996	6	3594.0	2958.771429	2978.710000	2956.75	2977.190000	1723.473979	2970.867698
143997	6	2205.0	2976.858333	2978.820000	2969.35	2972.603333	1204.825710	2975.213919

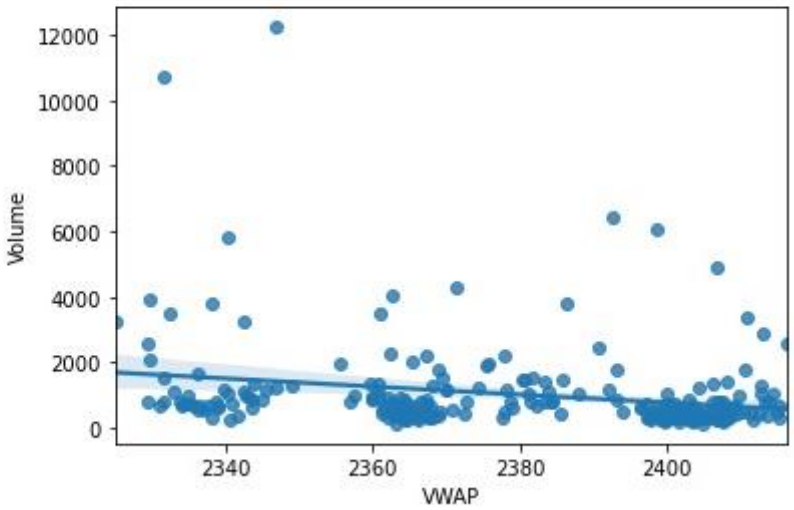
143998 rows × 8 columns

Step 26

In [26]:
```
# Visualizing the results for ethereum.

sns.regplot(x='VWAP', y='Volume', data=eth.head(250))
```

Out[26]: <AxesSubplot:xlabel='VWAP', ylabel='Volume'>



2.6 Analysis Conclusion

Through our analysis we came to some very interesting conclusions which are given below:

- Despite the fact that you may see a considerable difference in the values of different categories of prices i.e. opening price, closing price, highest price, and the lowest price for all the 14 assets, but still they are highly correlated with each other.
- We understood that on average, the closing price was always the highest than the opening price for all the 14 assets.
- We did an analysis by finding out weight percentages of each assets and got to know that Bitcoin and Ethereum are two most popular assets in the crypto market.
- Through visualizing the total number if trades in minute for every asset we again confirmed that Bitcoin and Ethereum are two most popular currencies as they both have the highest count.
- By plotting a regression trend of Bitcoin and Ethereum only, between the average price and quantity of asset bought and sold we came to know that when the prices were lower people tend to buy that asset in larger quantity and vice versa.

Without a doubt we can say that data powered decisions will always give you an edge in this competitive world and understand any work in much better way.

References

- https://www.kaggle.com/c/g-research-crypto-forecasting
- https://docs.ie.edu/cgc/research/cryptocurrencies/CGC-Cryptocurrencies-and-the-Future-of-Money-Full-Report.pdf
- https://repository.upenn.edu/cgi/viewcontent.cgi?article=1133&context=wharton_research_scholars
- https://medium.com/@springboard_ind/data-science-vs-data-analytics-how-to-decide-which-one-is-right-for-you-41e7bdec080e#:~:text=Data%20analysis%20involves%20answering%20questions,information%20to%20uncover%20actionable%20data.&text=Data%20science%20is%20a%20multi,solve%20analytically%20complex%20business%20problems.
- https://www.upgrad.com/blog/data-science-vs-data-analytics/