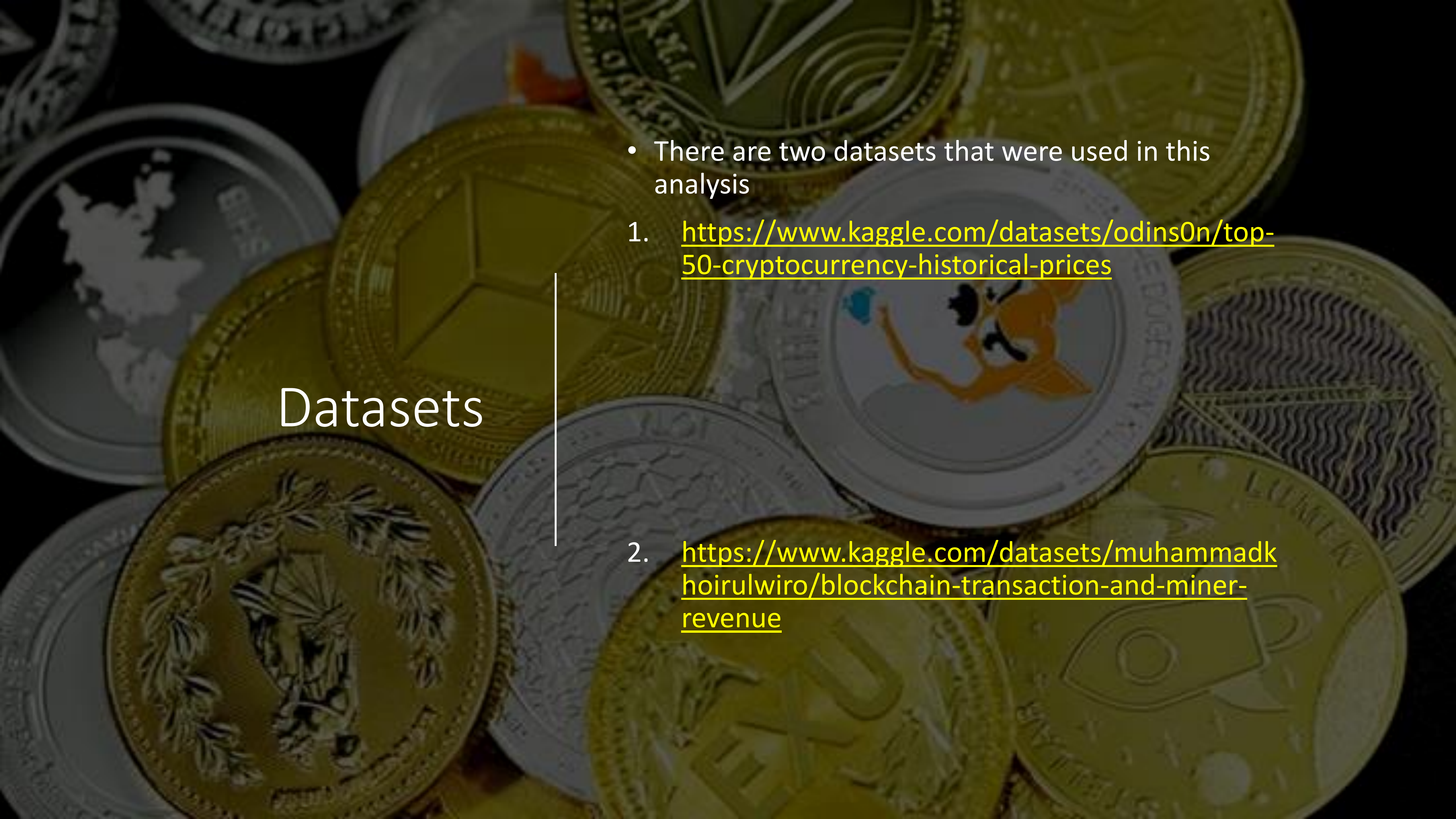
A Crypto Analysis



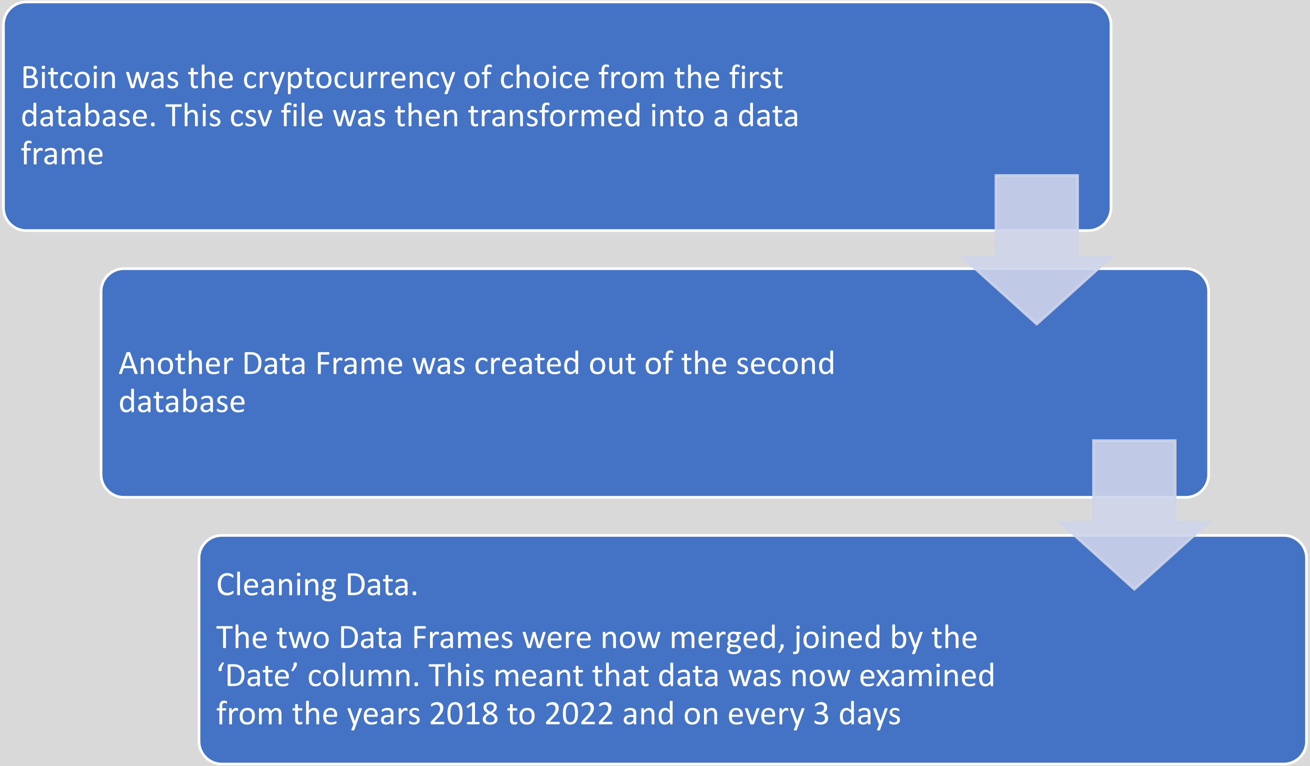
Datasets

- There are two datasets that were used in this analysis

1. https://www.kaggle.com/datasets/odins0n/top-50-cryptocurrency-historical-prices

2. https://www.kaggle.com/datasets/muhammadkhoirulwiro/blockchain-transaction-and-miner-revenue

Data Frames

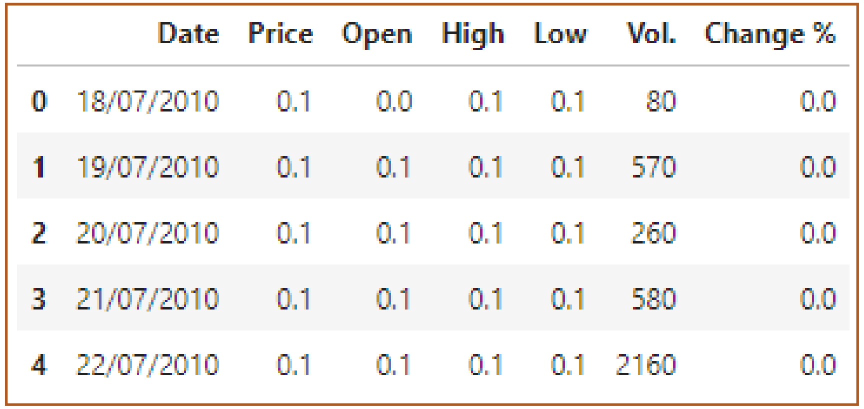


Bitcoin was the cryptocurrency of choice from the first database. This csv file was then transformed into a data frame

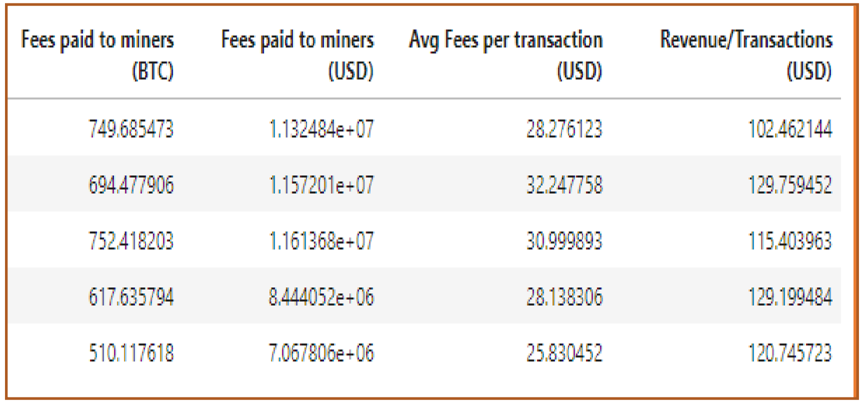
Another Data Frame was created out of the second database

Cleaning Data.

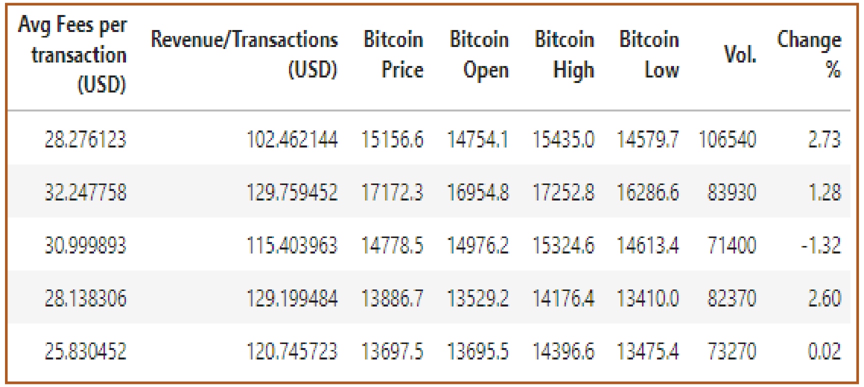
The two Data Frames were now merged, joined by the 'Date' column. This meant that data was now examined from the years 2018 to 2022 and on every 3 days



	Date	Price	Open	High	Low	Vol.	Change %
0	18/07/2010	0.1	0.0	0.1	0.1	80	0.0
1	19/07/2010	0.1	0.1	0.1	0.1	570	0.0
2	20/07/2010	0.1	0.1	0.1	0.1	260	0.0
3	21/07/2010	0.1	0.1	0.1	0.1	580	0.0
4	22/07/2010	0.1	0.1	0.1	0.1	2160	0.0



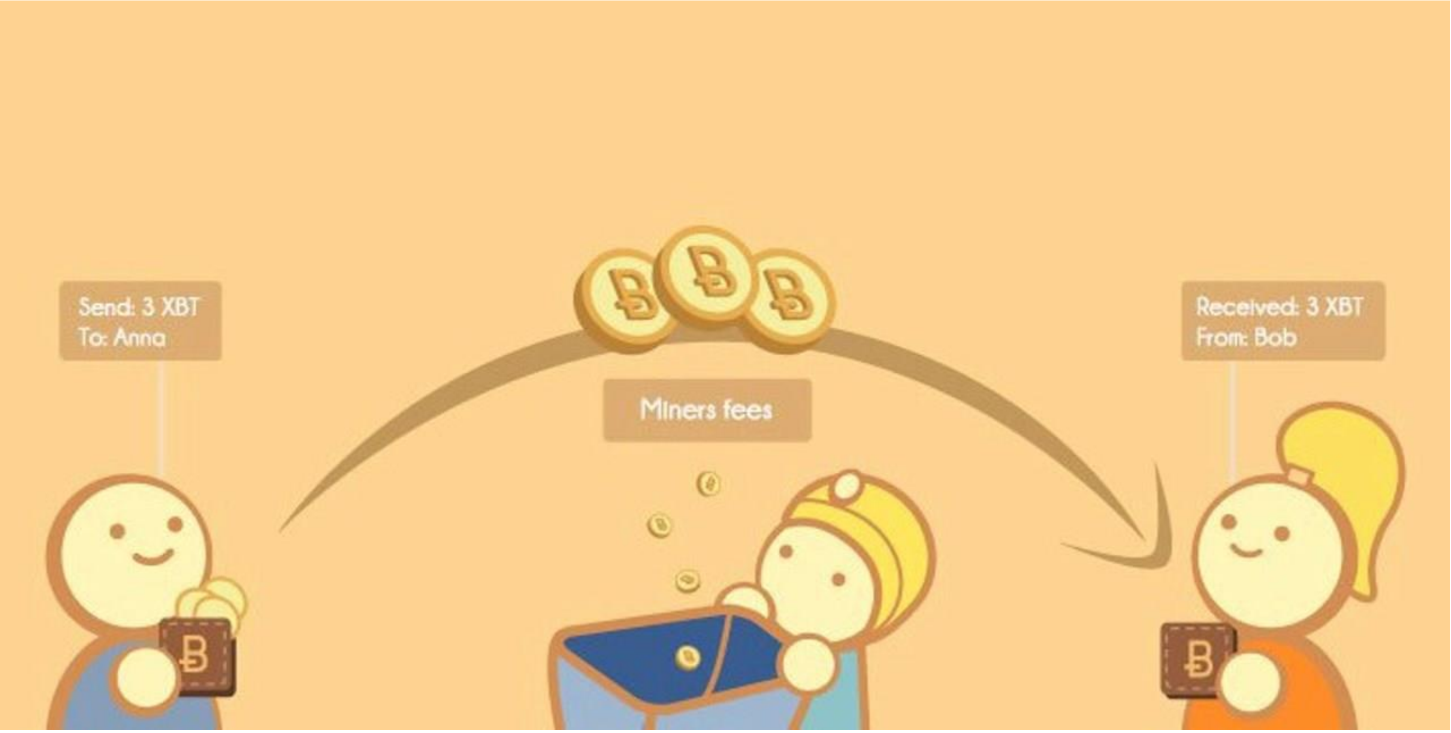
Fees paid to miners (BTC)	Fees paid to miners (USD)	Avg Fees per transaction (USD)	Revenue/Transactions (USD)
749.685473	1.132484e+07	28.276123	102.462144
694.477906	1.157201e+07	32.247758	129.759452
752.418203	1.161368e+07	30.999893	115.403963
617.635794	8.444052e+06	28.138306	129.199484
510.117618	7.067806e+06	25.830452	120.745723



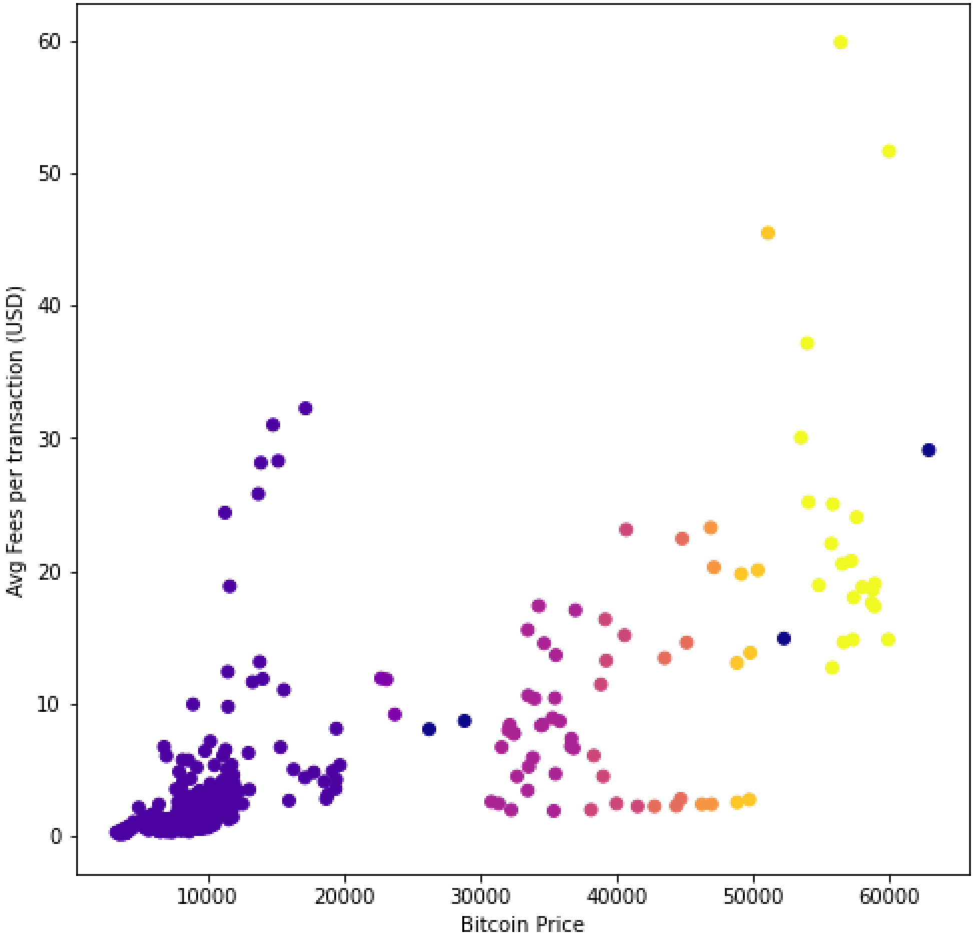
Avg Fees per transaction (USD)	Revenue/Transactions (USD)	Bitcoin Price	Bitcoin Open	Bitcoin High	Bitcoin Low	Vol.	Change %
28.276123	102.462144	15156.6	14754.1	15435.0	14579.7	106540	2.73
32.247758	129.759452	17172.3	16954.8	17252.8	16286.6	83930	1.28
30.999893	115.403963	14778.5	14976.2	15324.6	14613.4	71400	-1.32
28.138306	129.199484	13886.7	13529.2	14176.4	13410.0	82370	2.60
25.830452	120.745723	13697.5	13695.5	14396.6	13475.4	73270	0.02

Target Variable

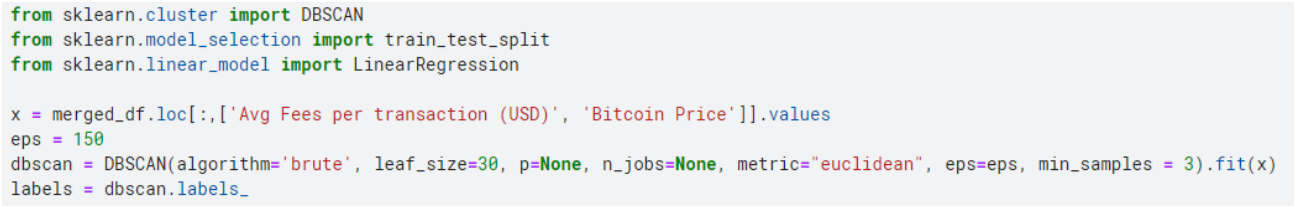
- The target variable of this project was initially the price of Bitcoin. Many of the different variables were used to investigate its effect on the Bitcoin Price
- Later, the target variable changed to measure the effect of the Bitcoin price on the average transaction fees



DBSCAN



- A Density-Based clustering algorithm (DBSCAN) is an unsupervised learning algorithm that’s used to handle noise, detect outliers and find patterns
- A large number of epsilons (eps) was needed due to the large number of data points provided , as well as this , the x value “Bitcoin Price” ranges from less than 5000 to greater than 60000, which means that the distance between multiple data points can be extremely large



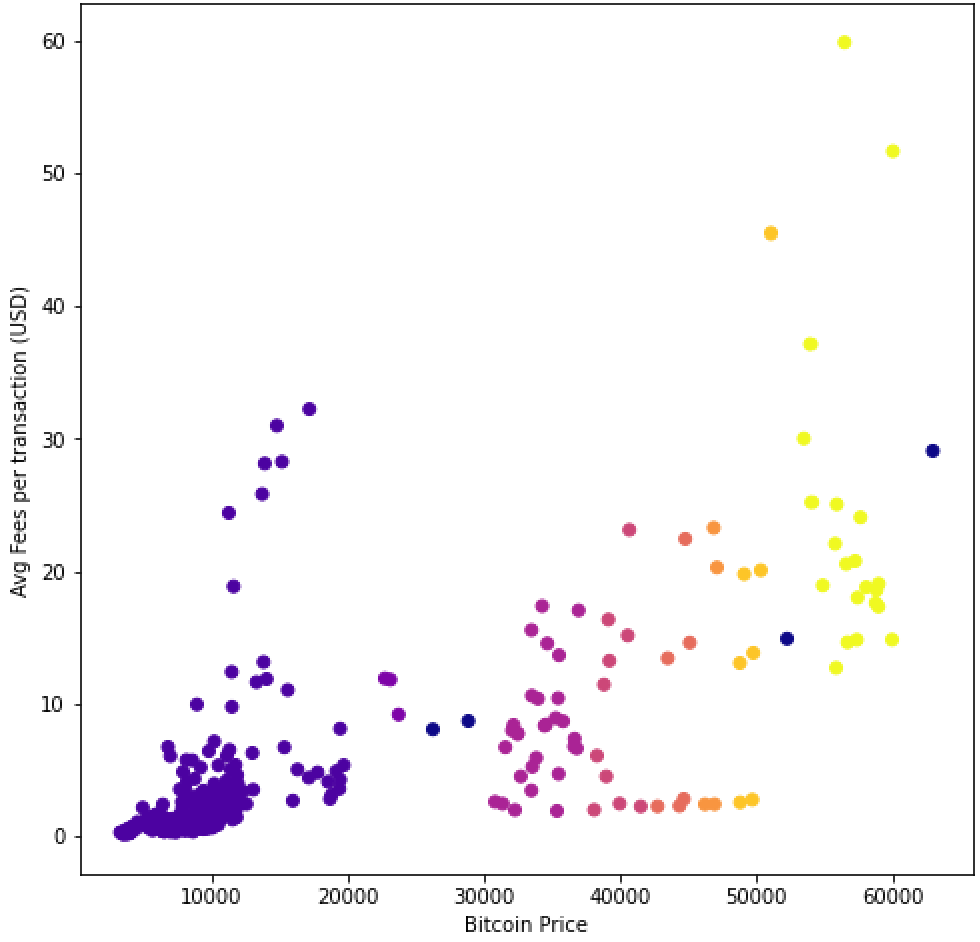
```
from sklearn.cluster import DBSCAN
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression

x = merged_df.loc[:,['Avg Fees per transaction (USD)', 'Bitcoin Price']].values
eps = 150
dbscan = DBSCAN(algorithm='brute', leaf_size=30, p=None, n_jobs=None, metric="euclidean", eps=eps, min_samples = 3).fit(x)
labels = dbscan.labels_
```

DBSCAN Results

- The silhouette is a good metric to use when evaluating the performance of clustering algorithms.
- The Silhouette Coefficient is bounded between 1 and -1. The best value is 1, the worst is -1. A higher score indicates that the model has better defined, more dense clusters. Values close to 0 indicate overlapping clusters, while negative values usually indicate that data points have been assigned to the wrong clusters.
- Two scores are used to calculate the silhouette coefficient:
  - a: The average distance between one data point and all other points in the same cluster
  - b: The average distance between one data point and all other points in the next nearest cluster.

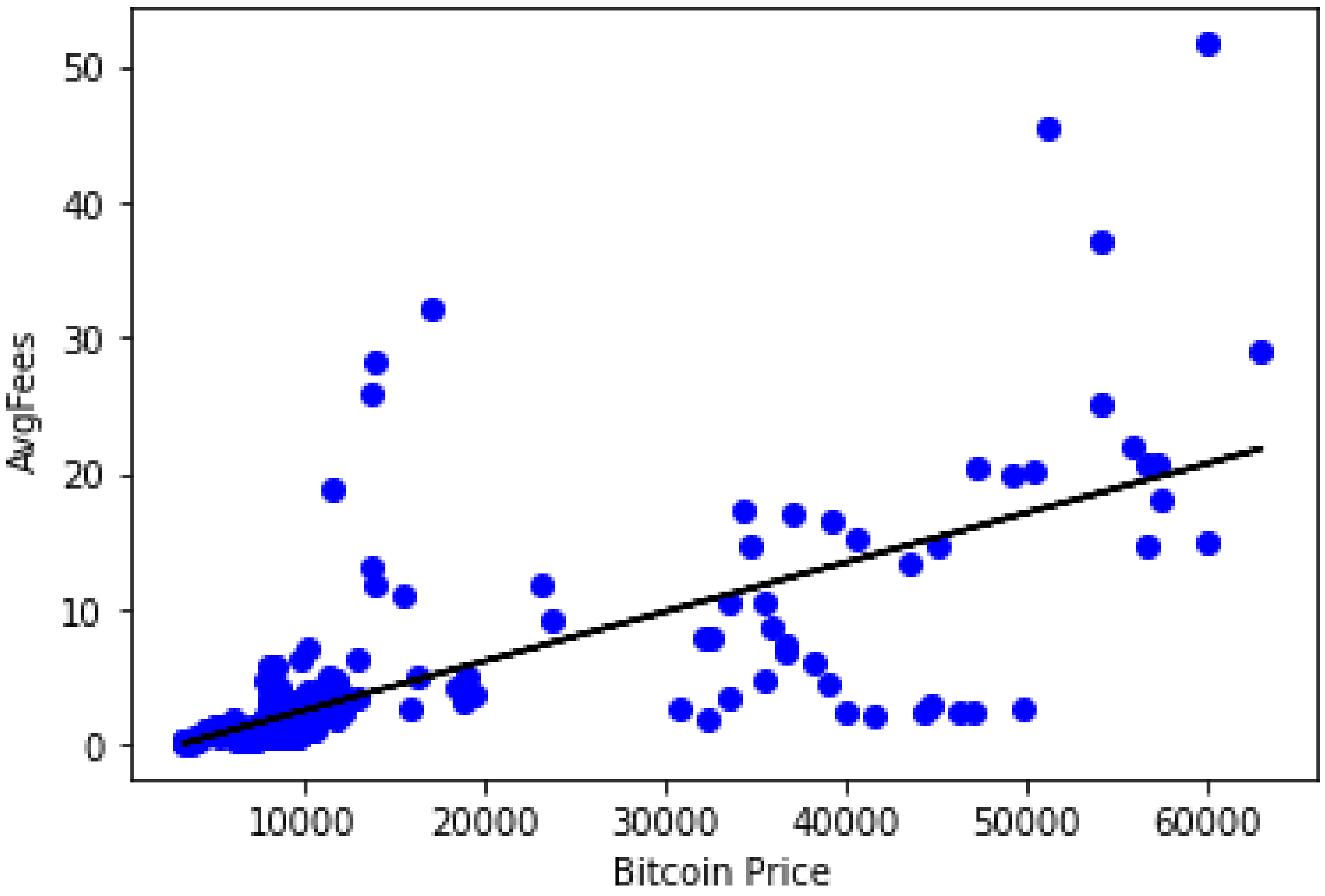
- A Score of 0.71 shows the model doesn’t have any mislabelled data points or overlapping clusters

```
Silhouette score:  0.7050517249524746
```



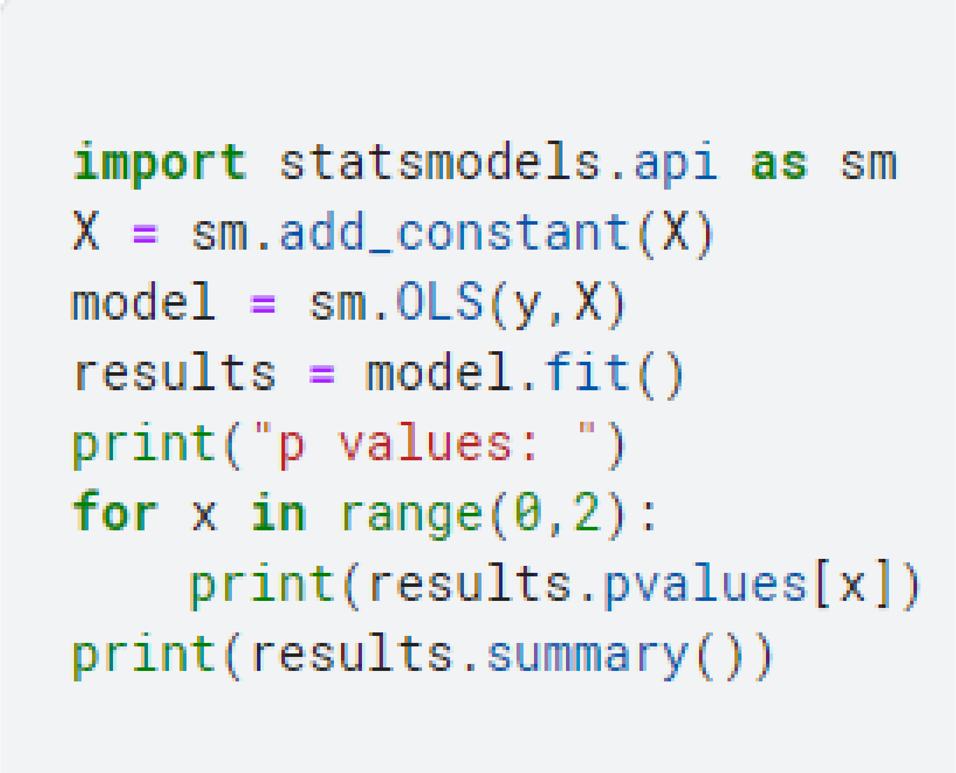
Linear Regression

- Linear Regression is another model used to look at the relationship between the average transaction fees and Bitcoin Price is Linear regression.
- The model creates a line that best fits the variables provided.

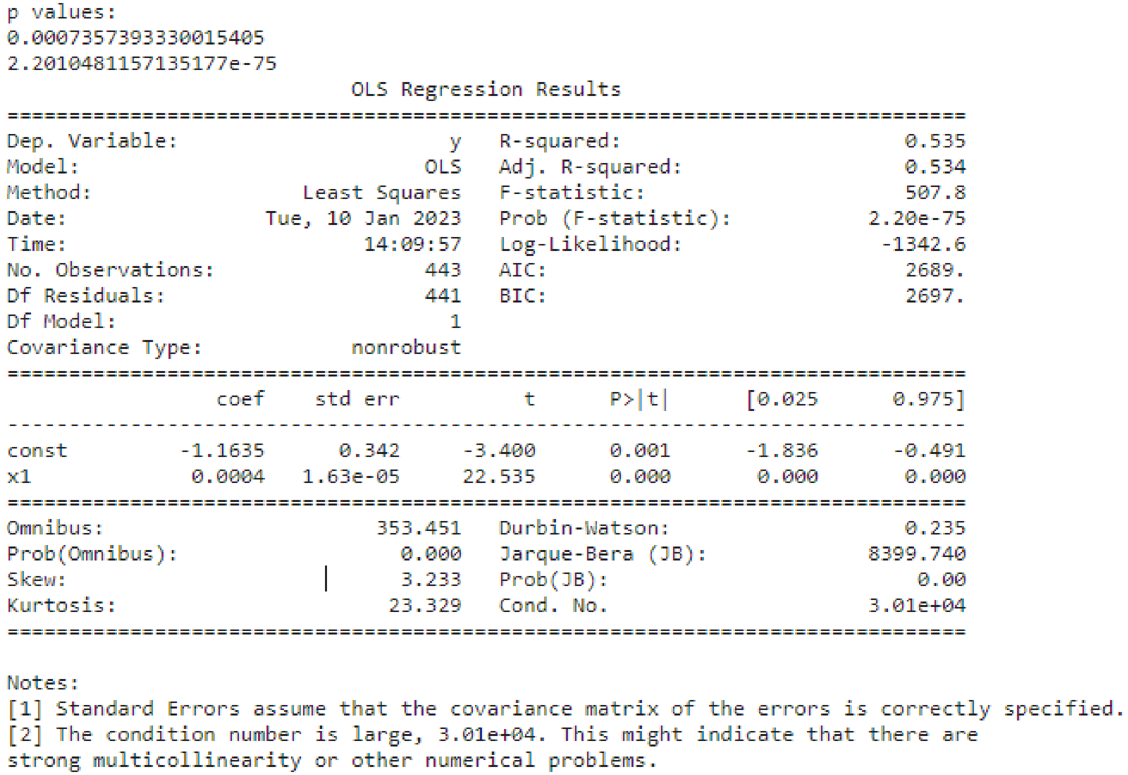


Linear Regression – P values

- The OLS (Ordinary Least Squares Function) tool is used to analyze the linear regression. Comparing the difference between points in the data and the predicted best fit line



```python
import statsmodels.api as sm
X = sm.add_constant(X)
model = sm.OLS(y,X)
results = model.fit()
print("p values: ")
for x in range(0,2):
    print(results.pvalues[x])
print(results.summary())
```

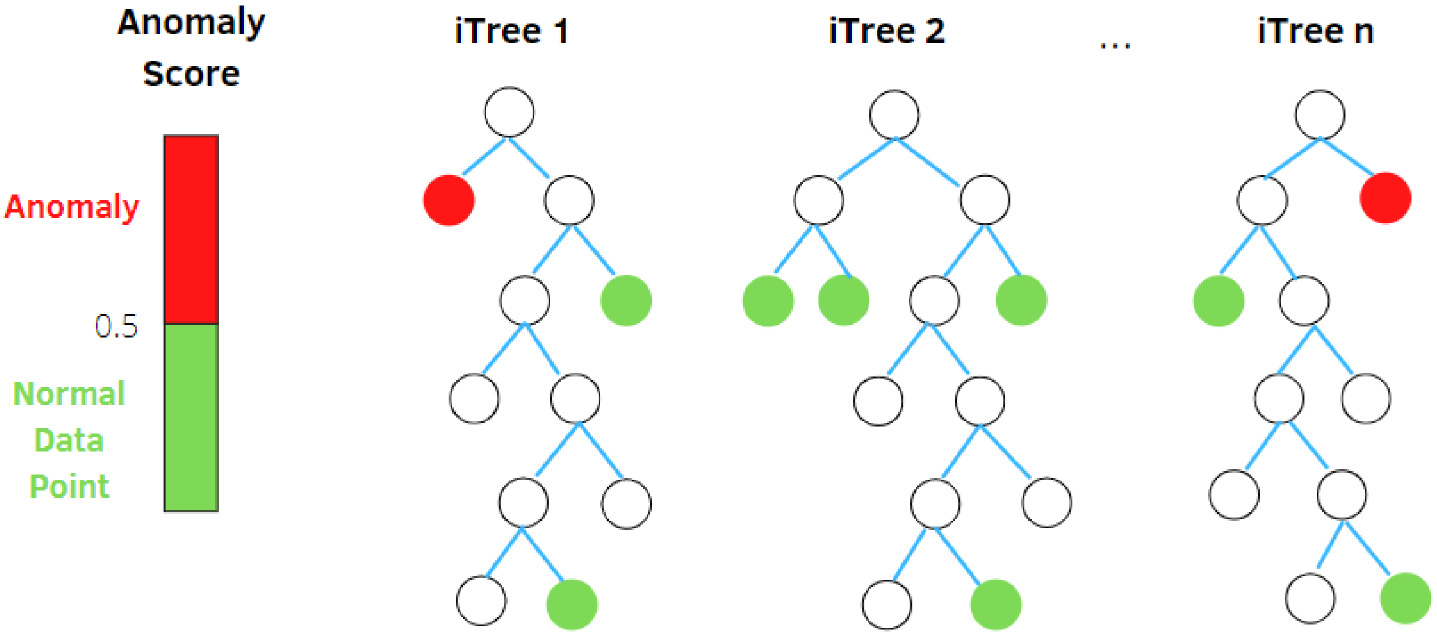


```
p values:
0.0007357393330015405
2.2010481157135177e-75
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.535
Model:                            OLS   Adj. R-squared:                  0.534
Method:                 Least Squares   F-statistic:                     507.8
Date:                Tue, 10 Jan 2023   Prob (F-statistic):           2.20e-75
Time:                        14:09:57   Log-Likelihood:                -1342.6
No. Observations:                 443   AIC:                             2689.
Df Residuals:                     441   BIC:                             2697.
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const         -1.1635      0.342     -3.400      0.001      -1.836      -0.491
x1             0.0004   1.63e-05     22.535      0.000       0.000       0.000
==============================================================================
Omnibus:                      353.451   Durbin-Watson:                   0.235
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             8399.740
Skew:                           3.233   Prob(JB):                         0.00
Kurtosis:                      23.329   Cond. No.                     3.01e+04
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 3.01e+04. This might indicate that there are
strong multicollinearity or other numerical problems.
```

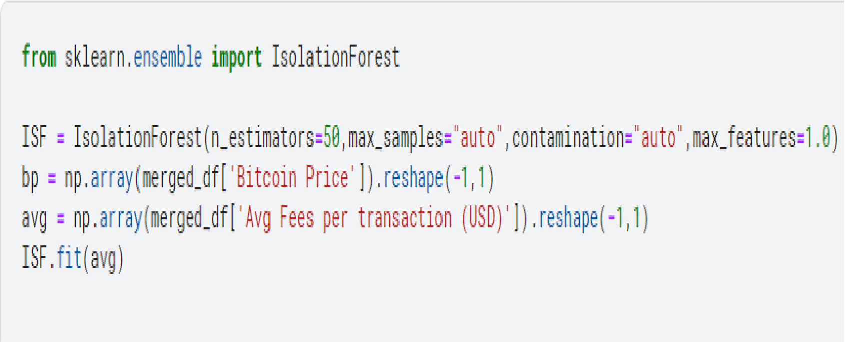
Results Explained

- R Squared - This represents the quality of the model fit. (How much the independent variable is explained by the changes in the dependent variable) A score of 0.535 means that the model can explain 53.5 % of the change in the dependant variable.
- Coef - shows the coefficient for each independent variable and intercept variable
- Std-error/t - An estimate of the standard deviation of the corresponding variable's coefficient across all data points.
- F-statistic - Examines whether the group of variables is ***statistically significant*** by comparing this model with another model where the effect of the variables are reduced to 0.
- P>t - How likely is it that the coefficient is measured through the model by chance?

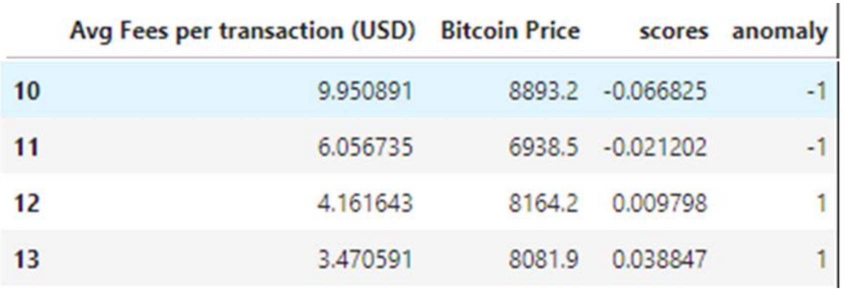
Detecting and Removing Anomalies



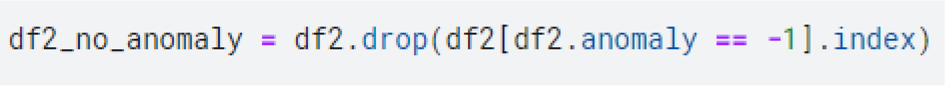
Isolation Forest Implementation



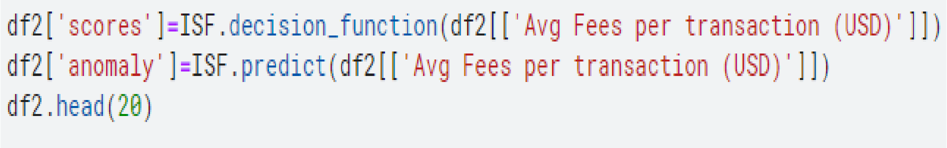
```
from sklearn.ensemble import IsolationForest

ISF = IsolationForest(n_estimators=50,max_samples="auto",contamination="auto",max_features=1.0)
bp = np.array(merged_df['Bitcoin Price']).reshape(-1,1)
avg = np.array(merged_df['Avg Fees per transaction (USD)']).reshape(-1,1)
ISF.fit(avg)
```



	Avg Fees per transaction (USD)	Bitcoin Price	scores	anomaly
10	9.950891	8893.2	-0.066825	-1
11	6.056735	6938.5	-0.021202	-1
12	4.161643	8164.2	0.009798	1
13	3.470591	8081.9	0.038847	1



```
df2_no_anomaly = df2.drop(df2[df2.anomaly == -1].index)
```



```
df2['scores']=ISF.decision_function(df2[['Avg Fees per transaction (USD)']])
df2['anomaly']=ISF.predict(df2[['Avg Fees per transaction (USD)']])
df2.head(20)
```

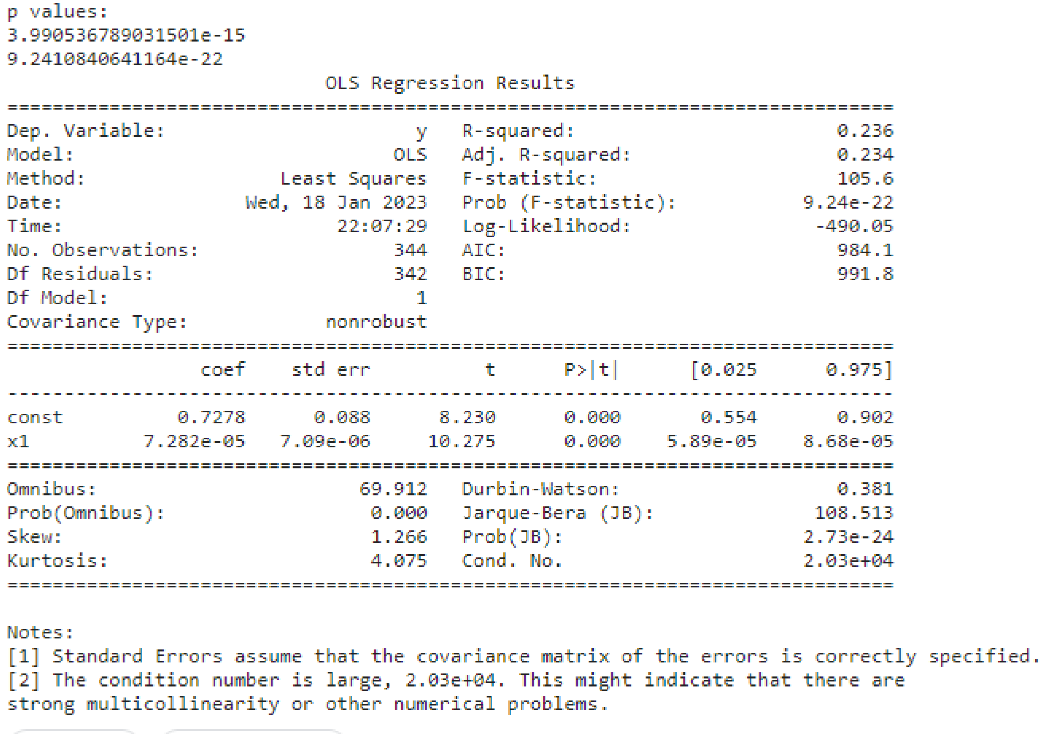
This Data Frame shows each data point within both variables, an anomaly score and an anomaly prediction.

-1 indicates an anomaly in the dataset.

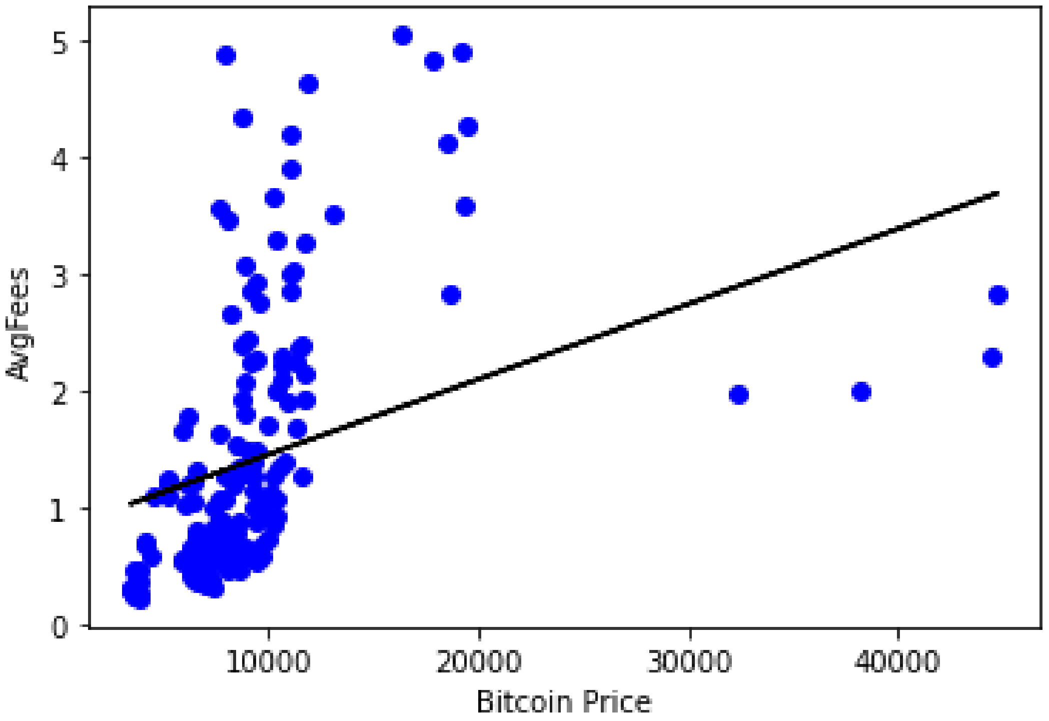
A new dataframe is then created with the removal of all data points that are considered an ‘anomaly’

Linear Regression without anomalies

- The addition of the isolation forest to detect outliers had produced no improvement on the p-values or r-squared scores. In fact, the r-squared score was lower than before. The isolation forest algorithm detected well over 100 anomalies. Looking at the graph, it seems as if the forest removed any point that were outside the big cluster in the bottom-left corner, instead of taking in account every single data point on the graph.



```
p values:
3.990536789031501e-15
9.2410840641164e-22
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.236
Model:                            OLS   Adj. R-squared:                  0.234
Method:                 Least Squares   F-statistic:                     105.6
Date:                Wed, 18 Jan 2023   Prob (F-statistic):           9.24e-22
Time:                        22:07:29   Log-Likelihood:                -490.05
No. Observations:                 344   AIC:                             984.1
Df Residuals:                     342   BIC:                             991.8
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          0.7278      0.088      8.230      0.000       0.554       0.902
x1          7.282e-05   7.09e-06     10.275      0.000    5.89e-05    8.68e-05
==============================================================================
Omnibus:                       69.912   Durbin-Watson:                   0.381
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              108.513
Skew:                           1.266   Prob(JB):                     2.73e-24
Kurtosis:                       4.075   Cond. No.                     2.03e+04
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 2.03e+04. This might indicate that there are
strong multicollinearity or other numerical problems.
```



K-Means Clustering

- What is it?
- A type of unsupervised algorithm that groups data based on each point’s Euclidean distance to a central point(centroid)
- It also partitions all points in the sample based on similarity (calculated by the Euclidean distance)

- Methods:

- Elbow Method
  - Indicates how many clusters there are within the dataset by bending at a certain number

- Silhouette Coefficient
  - Used when the number of clusters in a dataset is unknown , it then calculates the density of possible clusters using a model

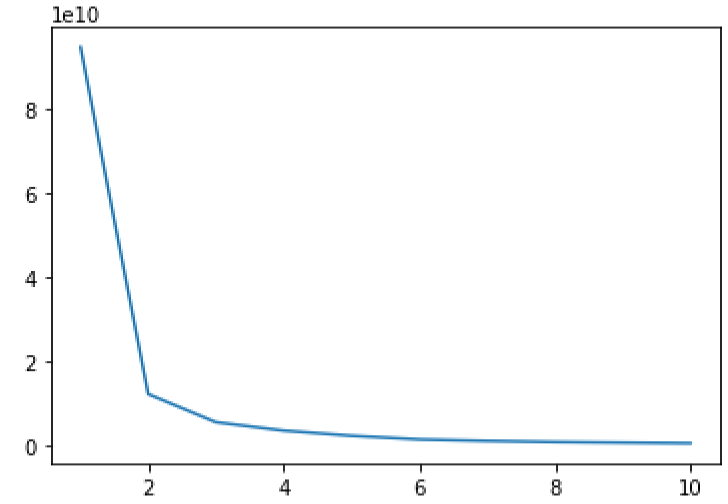
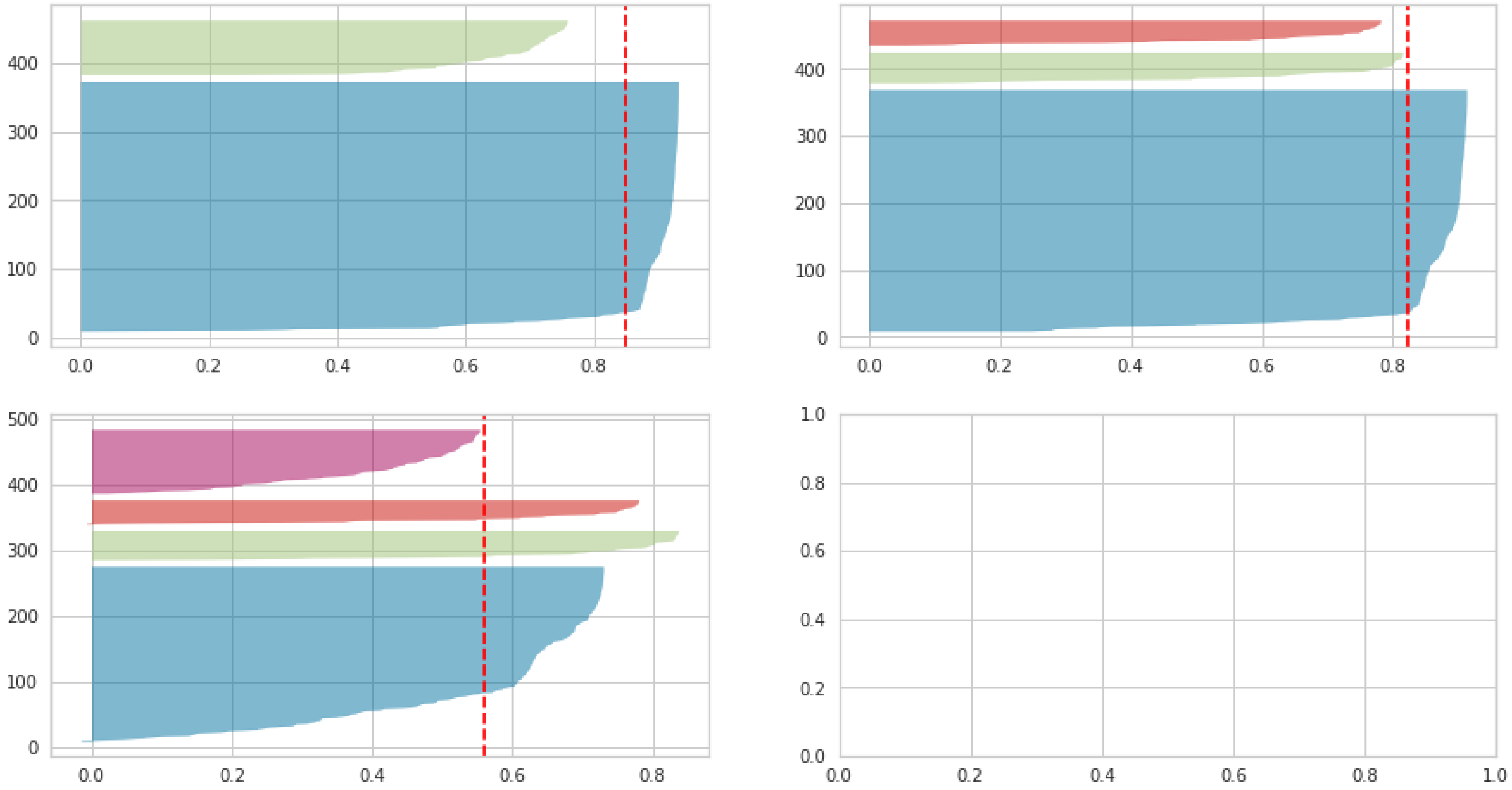


Figure showing the elbow method for determining the optimal value of k.

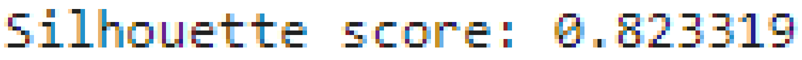
For the optimal number of clusters we must select the value of k for the point after the distortions starts to decrease in a liner fashion. That occurs at k = 3



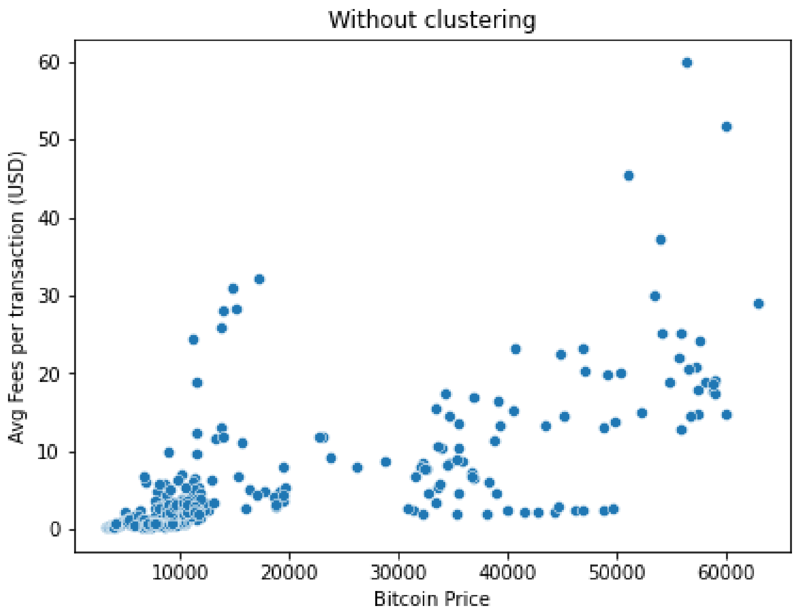
The silhouette analysis graph shows the different average silhouette scores (dotted red line) for each value of k

K – Means Clustering Results

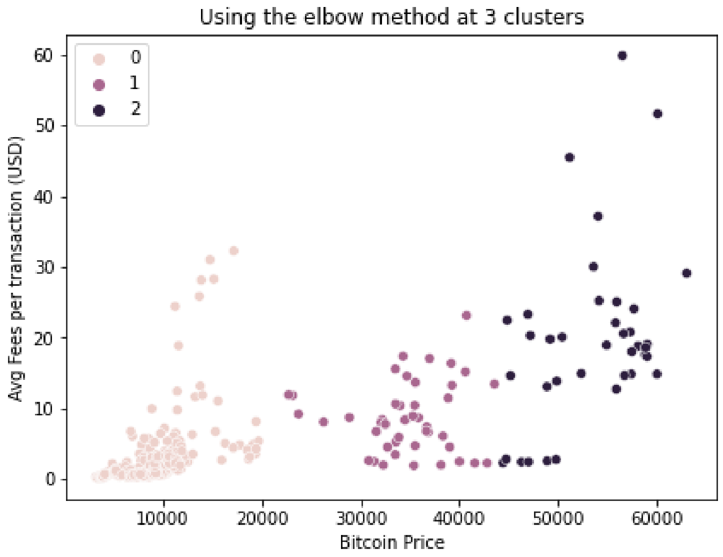
- Libraries:
  - Seaborn
  - Scikit Learn – ‘K-Means’
- 3 Clusters are identified.
  - A Silhouette score of 0.82 shows that the 3 clusters are of a significant distance away from each other and the samples within each cluster are very close to their centroids



```
Silhouette score: 0.823319
```



Without clustering



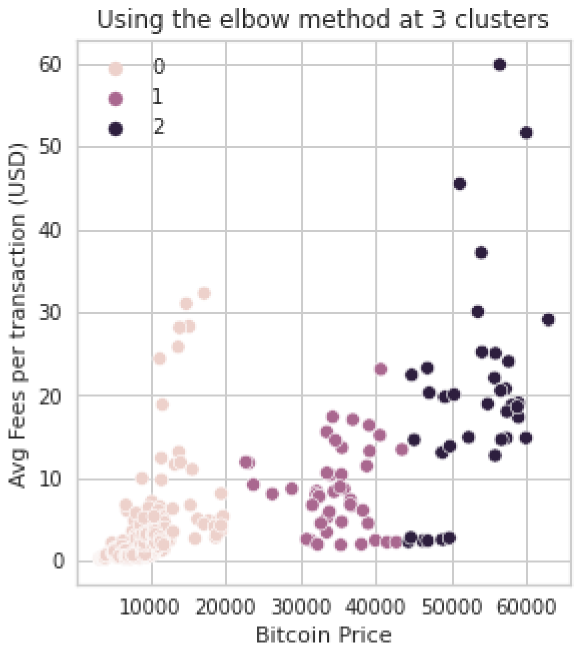
Using the elbow method at 3 clusters

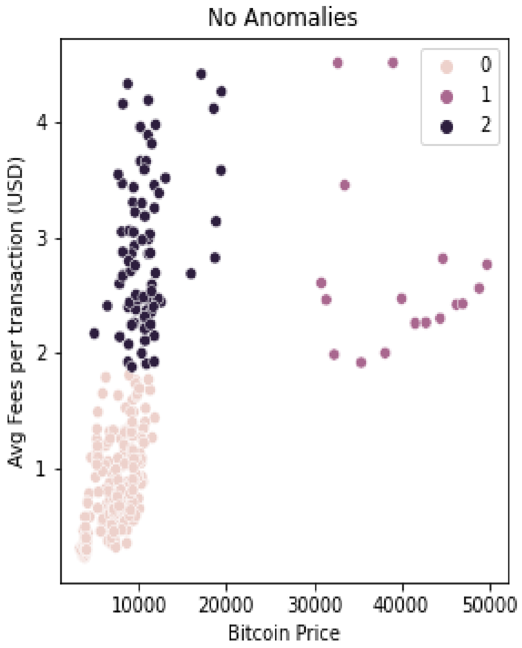
K means clustering without anomalies

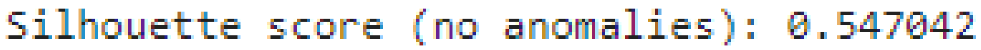
The K-means clustering algorithm is of course sensitive to outliers, due to the algorithm taking an average of all data points related to a certain cluster to create a centroid.

Removal of the outliers will give a more accurate reading of the silhouette score.

The silhouette score dropped from 0.82 to 0.55.







```
Silhouette score (no anomalies): 0.547042
```

Findings

- As shown by the Linear regressions , there is a strong positive relationship between the price of bitcoin and the average transaction fees. However, there is no indication that the bitcoin price is the most significant factor in affecting the transaction fees.
- This is shown by the anomalies in the first linear regression. At points, the transaction fees are low whilst the Bitcoin price is at the top 10% all-time, whilst the clusters show that transaction fees should be low when the prices are low.