ECS766P: Data Mining

Assignment 1

In [1]:
```
import pandas as pd
from sklearn.svm import OneClassSVM
from mpl_toolkits.mplot3d import Axes3D
import numpy as np
import matplotlib.pyplot as plt

from scipy.stats import pearsonr
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
import seaborn as sns
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import LabelEncoder
```

1(a) What is the advantage of using the Apriori algorithm in comparison with computing the support of every subset of an itemset in order to find the frequent itemsets in a transaction dataset? [5 marks]

The advantage of the Apriori algorithm in a transaction dataset is that it reduces the search space. It relies on all subsets of a frequent itemset also being frequent. By applying this principle, the algorithm can remove itemsets with low weight without examining its supersets. Instead of looking at every possible group, which can take a very long time like computing the support of every subset, which is often not possible due to the large amounts of possible subsets within itemsets. The computational cost would be high due to the exponential increase in possible combinations. Apriori algorithm efficiently reduces the search space by eliminating itemsets whose subsets are infrequent.

(b) Let $L1$ denote the set of frequent 1-itemsets. For $k \geq 2$ why must every frequent k-itemset be a superset of an itemset in $L1$? [6 marks]

Every frequent k-itemset must be a superset of an itemset in $L1$ because if any item in the k-itemset was not frequent on its own (not in $L1$), the entire k-itemset cannot be frequent. This follows the Apriori principle that all subsets of a frequent itemset must also be frequent.

(c) Let $L2$ = {{1,2},{1,4},{2,3},{2,4},{3,5}}. Compute the set of candidates $C3$ that is obtained by joining every pair of joinable itemsets from $L2$. [5 marks]

To get C3 from L2, we join itemsets in L2 that share the first k−1 items.

$L2$={{1,2},{1,4},{2,3},{2,4},{3,5}}

The joinable sets are {1,2} and {1,4} because they share the first item (1), aligning with the rule that joinable sets must share their first k-1 items. Giving {1,2,4}

{2,3} and {2,4} giving {2,3,4}.

So, C3 from $L2$ by joining pairs C3={{1,2,4},{2,3,4}}.

(d) Let $S1$ denote the support of the association rule: $\{boarding\ pass, passport\} \Rightarrow \{flight\}$ Let $S2$ denote the support of the association rule: $\{boarding\ pass\} \Rightarrow \{flight\}$ What is the relationship between $S1$ and $S2$? [5 marks]

The support $S1$ is in proportion to transactions with both 'boarding pass' and 'passport', while $S1$ considers all transactions with 'boarding pass'. Then $S1 \leq S2$ a subset of S2. This is because the presence of 'boarding pass' and 'passport' together in S1 is a subset covered by S2, which only requires 'boarding pass', making S1 less frequent or equal to S2.

(e) What is the support of the rule: $\{\} \Rightarrow \{Eggs\}$ in the transaction dataset below shown in Figure 1? [5 marks]

The rule:$\{\} \Rightarrow \{Eggs\}$ in the dataset is the proportion of transactions have eggs. It is in 3/5 so support is 60% to get this i counted the number of transactions containing eggs and divided by the total number of transactions to get the support value.

(f) In the transaction dataset shown in Figure 1, what is the maximum length of a frequent itemset for a support threshold of 0.2?[5 marks]

The support threshold of 0.2 means an itemset is frequent if it appears in at least 1 transaction. The maximum length of a frequent itemset is 6 as thats the longest transaction. The longest transaction determines the maximum length because any itemset shorter than this length will appear at least once (meeting the 0.2 threshold).

(2a) For a system designed to prevent identity theft in online transactions, we are focusing on identifying unusual transaction patterns. Propose 2 possible contextual attributes and 2 possible behavioural attributes that could be integrated into this system's algorithm. Provide a rationale for classifying each attribute as either contextual or behavioural. [6 marks]

In a system to prevent identity theft in online transactions, integrating attributes that identify unusual transaction patternsis important. Two contextual attributes could be transaction time and location of transaction. Behavioural attributes could be frequency of transactions and transaction amount. Transaction location and time is a contextual attribute as its when purchases are made at unusual locations/hours atypical of the user it could show warning signs for fraud.

For behavioural attributes, an increase in frequency of transactions might indicate that an account has been compromised. Any unusual patterns in transation amount could also be a sign of fraudulent activity. These attributes are related to user behaviour and have potential use in identifying fraudulent activities. Contextual attributes reflect the environment where the transaction is made and behavioural attributes reflect the patterns of the user's transactional behaviour.

(b) Assume that you are provided with the University of Wisconsin breast cancer dataset from the Week 3 lab, and that you are asked to detect outliers from this dataset. Additional information on the dataset attributes can be found online. Explain one possible outlier detection method that you could apply for detecting outliers for this particular dataset, explain what is defined as an outlier for your suggested approach given this particular

dataset, and justify why would you choose this particular method for outlier detection. [7 marks]

To detect outliers in a dataset like the [University of Wisconsin breast cancer dataset] you could use the Interquartile Range. This method identifies outliers by looking at the spread of the data points. In this dataset, an outlier can be defined as a data point that lies 1.5 times above the third quartile and below the first quartile for any given feature.

The IQR method is a suitable approach as opposed to standard deviation, because it is less affected by extreme values. It handles the variability in medical data and can handle deviations from a normal distribution. In medical data outliers may represent actual cases of interest e.g.tumor detection. The IQR approach can effectively identify outliers in skewed distributions, this can help in flagging data points that need further study or represent data entry errors in the dataset.

(c) The monthly rainfall in the London borough of Tower Hamlets in 2019 had the following amount of precipitation (measured in mm, values from January-December 2018): {22.93, 20.69, 25.75, 23.84, 25.34, 3.25, 23.55, 28.28, 23.72, 22.42, 26.83, 23.82}. Assuming that the data is based on a normal distribution, identify outlier values in the above dataset using the maximum likelihood method. [7 marks]

Using the maximum likelihood method, the mean and standard deviation of the dataset was calculated and then determine which values fall outside the typical range more than 3 standard deviations from the mean. The choice of 3 standard deviations is based on the empirical rule, where approximately 99.7% of data in a normal distribution lies within this range. Given that the data follows a normal distribution 3.25 mm of rainfall was an outlier.

In [2]:
```
rainfall_data = np.array([22.93, 20.69, 25.75, 23.84, 25.34, 3.25, 23.55, 28.28, 23
```

In [3]:
```
mean_rainfall = np.mean(rainfall_data)
std_dev_rainfall = np.std(rainfall_data)
upper = mean_rainfall + 3 * std_dev_rainfall
lower = mean_rainfall - 3 * std_dev_rainfall
print('mean:',mean_rainfall)
print('std:',std_dev_rainfall)
print('lower_threshold:', lower)
print('upper_threshold:', upper)
```

```
mean: 22.534999999999997
std: 6.130045540885756
lower_threshold: 4.14486337734273
upper_threshold: 40.92513662265726
```

In [4]:
```
outliers = rainfall_data[(rainfall_data > upper) | (rainfall_data < lower)]
outliers
```

Out[4]:
```
array([3.25])
```

(d) Using the stock prices (stocks.csv included in the supplementary material) dataset used in sections 1 and 2 of Week 9 lab, estimate the outliers in the dataset using the one-class SVM classifier approach. As input to the classifier, use the percentage of changes in the daily closing price of each stock, as was done in section 1 of the notebook. Use the same SVM settings as in the lab notebook. Plot a 3D scatterplot of the dataset, where each object is

color coded according to whether it is an outlier or an inlier. Also compute a histogram and the frequencies of the estimated outlier and inlier labels. In terms of the plotted results, how does the one-class SVM approach for outlier detection differ from the parametric and proximity-based methods used in the lab notebook? What percentage of the dataset objects are classified as outliers? [8 marks]

In [5]:
```
df = pd.read_csv('stocks.csv')
df
```

Out[5]:

	Date	MSFT	F	BAC
0	1/3/2007	29.860001	7.51	53.330002
1	1/4/2007	29.809999	7.70	53.669998
2	1/5/2007	29.639999	7.62	53.240002
3	1/8/2007	29.930000	7.73	53.450001
4	1/9/2007	29.959999	7.79	53.500000
...	...	...	...	...
2513	12/23/2016	63.240002	12.46	22.600000
2514	12/27/2016	63.279999	12.39	22.610001
2515	12/28/2016	62.990002	12.25	22.330000
2516	12/29/2016	62.900002	12.23	22.000000
2517	12/30/2016	62.139999	12.13	22.100000

2518 rows × 4 columns

In [6]:
```
print(df.head())
```

```
       Date       MSFT     F        BAC
0  1/3/2007  29.860001  7.51  53.330002
1  1/4/2007  29.809999  7.70  53.669998
2  1/5/2007  29.639999  7.62  53.240002
3  1/8/2007  29.930000  7.73  53.450001
4  1/9/2007  29.959999  7.79  53.500000
```

In [7]:
```
df_pct_change = df.iloc[:, 1:].pct_change().dropna()
df_pct_change
```

Out[7]:

	MSFT	F	BAC
1	-0.001675	0.025300	0.006375
2	-0.005703	-0.010390	-0.008012
3	0.009784	0.014436	0.003944
4	0.001002	0.007762	0.000935
5	-0.010013	-0.007702	0.001495
...	...	...	...
2513	-0.004878	0.004839	0.002662
2514	0.000632	-0.005618	0.000443
2515	-0.004583	-0.011299	-0.012384
2516	-0.001429	-0.001633	-0.014778
2517	-0.012083	-0.008177	0.004545

2517 rows × 3 columns

The one-class SVM approach differs from parametric and proximity-based methods primarily in its non-reliance on the underlying data distribution. Unlike parametric methods, it does not assume a normal distribution, making it suitable for datasets with complex structures. Its ability to handle multi-dimensional data, as seen in the stock price dataset, allows for more nuanced detection of outliers in complex datasets where relationships between variables are not linear. This allows it to capture outliers in a global context rather than through local density.

The analysis involved calculating daily percentage changes in stock prices. The classifier was trained to identify data points that deviates from norm. A 3D scatterplot was created, where outliers (approx. 5.01% of the dataset) are marked in red and inliers in blue.

The method, shows outliers without assuming a specific distribution, making it suitable for financial datasets like the stock prices where patterns are not easy to see from data.

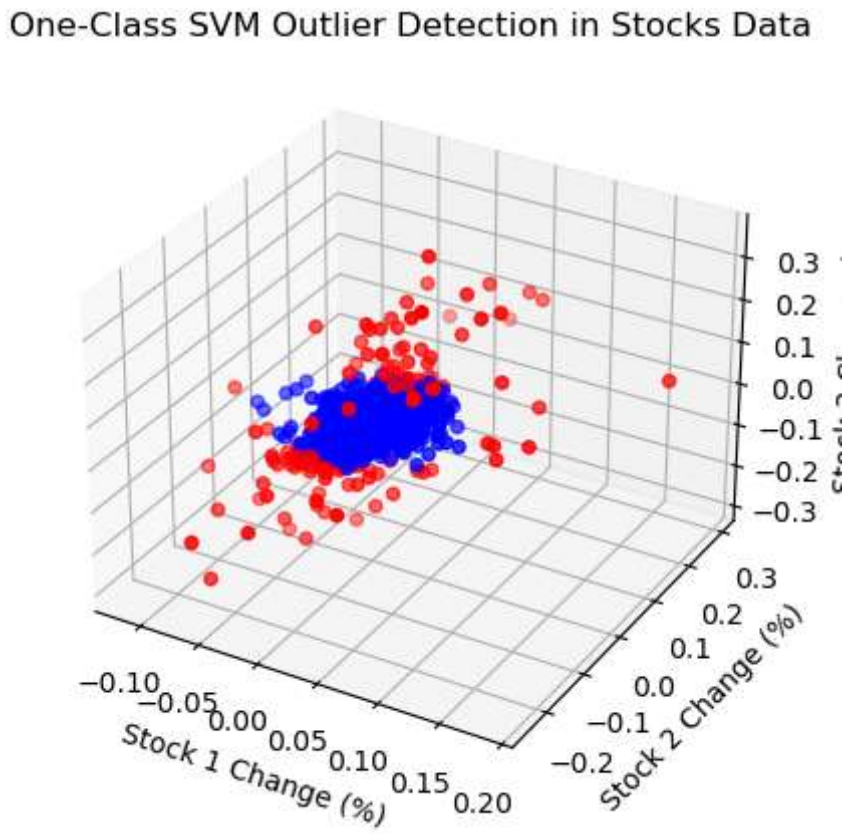
In [8]:
```
oc_svm = OneClassSVM(nu=0.05, kernel="rbf", gamma=0.1)
oc_svm.fit(df_pct_change)
outliers = oc_svm.predict(df_pct_change)
outliers
```

Out[8]:
```
array([1, 1, 1, ..., 1, 1, 1], dtype=int64)
```

In [9]:
```
x = df_pct_change.iloc[:, 0]
y = df_pct_change.iloc[:, 1]
z = df_pct_change.iloc[:, 2]
colors = np.where(outliers == -1, 'r', 'b')
```

In [10]:
```
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.scatter(x, y, z, c=colors)
ax.set_xlabel('Stock 1 Change (%)')
ax.set_ylabel('Stock 2 Change (%)')
ax.set_zlabel('Stock 3 Change (%)')
```

```
plt.title("One-Class SVM Outlier Detection in Stocks Data")
plt.show()
```



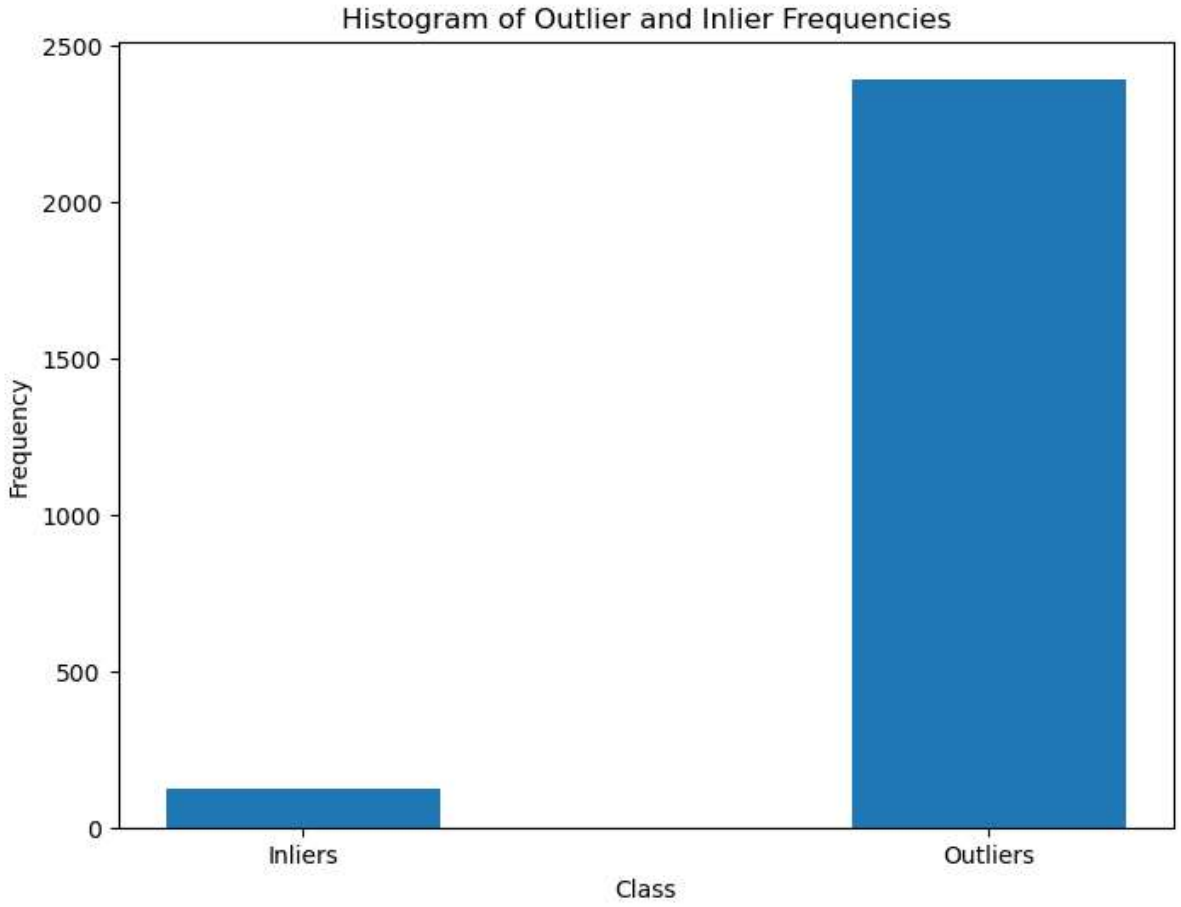
In [11]:
```
outliers_count = np.sum(outliers == -1)
total_count= len(outliers)
percentage_outliers = (outliers_count/ total_count) * 100
inliers_count= (total_count-outliers_count)

print('Outlier Frequency:', outliers_count)
print('Inlier Frequency:',inliers_count)
print('Percentage of Outliers:',percentage_outliers)
```

```
Outlier Frequency: 126
Inlier Frequency: 2391
Percentage of Outliers: 5.005959475566151
```

In [12]:
```
labels, counts = np.unique(outliers, return_counts=True)

plt.figure(figsize=(8, 6))
plt.bar(labels, counts, tick_label=['Inliers', 'Outliers'])
plt.xlabel('Class')
plt.ylabel('Frequency')
plt.title('Histogram of Outlier and Inlier Frequencies')
plt.show()
```



In [13]:
```
outlier_freq = np.sum(outliers == -1)
inlier_freq = np.sum(outliers == 1)

outlier_percentage = (outlier_freq / len(outliers)) * 100
print(f'Outlier Frequency: {outlier_freq}')
print(f'Inlier Frequency: {inlier_freq}')
print(f'Percentage of Outliers: {outlier_percentage}%')
```

```
Outlier Frequency: 126
Inlier Frequency: 2391
Percentage of Outliers: 5.005959475566151%
```

(3a) Inspect the HTML code of the above URL and provide a short report on the various tags present in the code. What is the function of each unique tag present in the HTML code?[6 marks]

https://eecs.qmul.ac.uk/~emmanouilb/income_table.html Upon examining the HTML code at the provided URL, the structure of the document is organised with a unique HTML tags:

the root of the HTML document.

contains meta-information, such as its title and links to scripts, stylesheets and the