National Technical University of Athens
MSc - Data Science and Machine Learning

Data Driven Models
Assignment 3

MSc student
Vasileios Depastas
A.M: 03400131
vasileiosdepastas@mail.ntua.gr

July 2023

1 Introduction - Extreme Driving Detection

In the present assignment, a dataset consisted of driving data from a number of drivers collected using the accelerometer, gyroscope and GPS of the driver's smartphone is used to detect extreme turning movement. The signals received are re-oriented to match the driving direction. Since we have no knowledge of anomalies (ie. extreme turning movements), we will approach the problem in an unsupervised manner, in the context of anomalies detection (or outliers detection).

The dataset at hand includes 7 features, as presented in the table below and 103891 records/rows.

Feature Category	**Measuring Device**	**Feature Name**	**Units**
Rotated Accelerations	Accelerometer	NewAccelX	km/h/s
		NewAccelY	
		NewAccelZ	
Rotation Rates	Gyroscope	NewRotRateX	rad/s
		NewRotRateY	
		NewRotRateZ	
Location Speed	GPS	locationSpeed	km/h

Next, we proceed to present some descriptive statistics about the dataset in scope.

	NewAccelX	NewAccelY	NewAccelZ	NewRotRateX	NewRotRateY	NewRotRateZ	locationSpeed
count	103891.000000	103891.000000	103891.000000	103891.000000	103891.000000	103891.000000	103891.000000
mean	-0.000079	0.000867	0.009910	-0.000357	-0.000180	0.000291	16.333149
std	0.058077	0.060980	0.063359	0.134682	0.128994	0.146620	12.691258
min	-1.322000	-2.163000	-1.415000	-9.102000	-12.628000	-13.615000	0.000000
25%	-0.022000	-0.022000	-0.015000	-0.013000	-0.013000	-0.010000	6.040000
50%	0.000000	0.001000	0.011000	0.000000	0.000000	0.000000	13.080000
75%	0.022000	0.024000	0.035000	0.013000	0.013000	0.010000	29.000000
max	1.224000	1.033000	1.185000	7.147000	7.212000	6.953000	47.660000

Also, we present in the next picture x, y and z directions of the vehicle's movement. The driving direction of the vehicle is the y, while the x direction defines the right and left steering directions and, finally, z is the vertical direction that is not of particular interest in this problem.

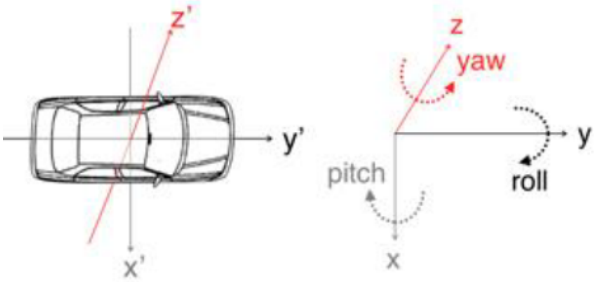

Figure 1: Vehicle movement directions

2 Feature engineering - selection

The determination of the features to be used for any machine learning task is crucial. It has to be performed in conjunction with the problem specifics at hand. This necessity becomes more important in this unsupervised setup since no supervised technique can be devised that would perform automatic feature selection. Hence, we select features that are closely related to the physics of the problem. The features that seem to be most relevant are NewAccelX (a_x), which is the acceleration towards the left and right directions, NewRotRateZ (w) which is the angular velocity that concerns the turning of the vehicle according to the rule of the right thumb and, finally, locationSpeed (u), which is the velocity of the vehicle. We will combine these variables into variable T through software engineering. T is an indicator of turning intensity and is defined by

$$T(u, a_x, w) = \frac{1}{g}\sqrt{u^2 w^2 + a_x^2}.$$

In the illustration below, we can see how the T is defined based on the vertical force due to gravity and the horizontal force on the vehicle's tires. We will use feature T in the subsequent analysis to detect outliers.

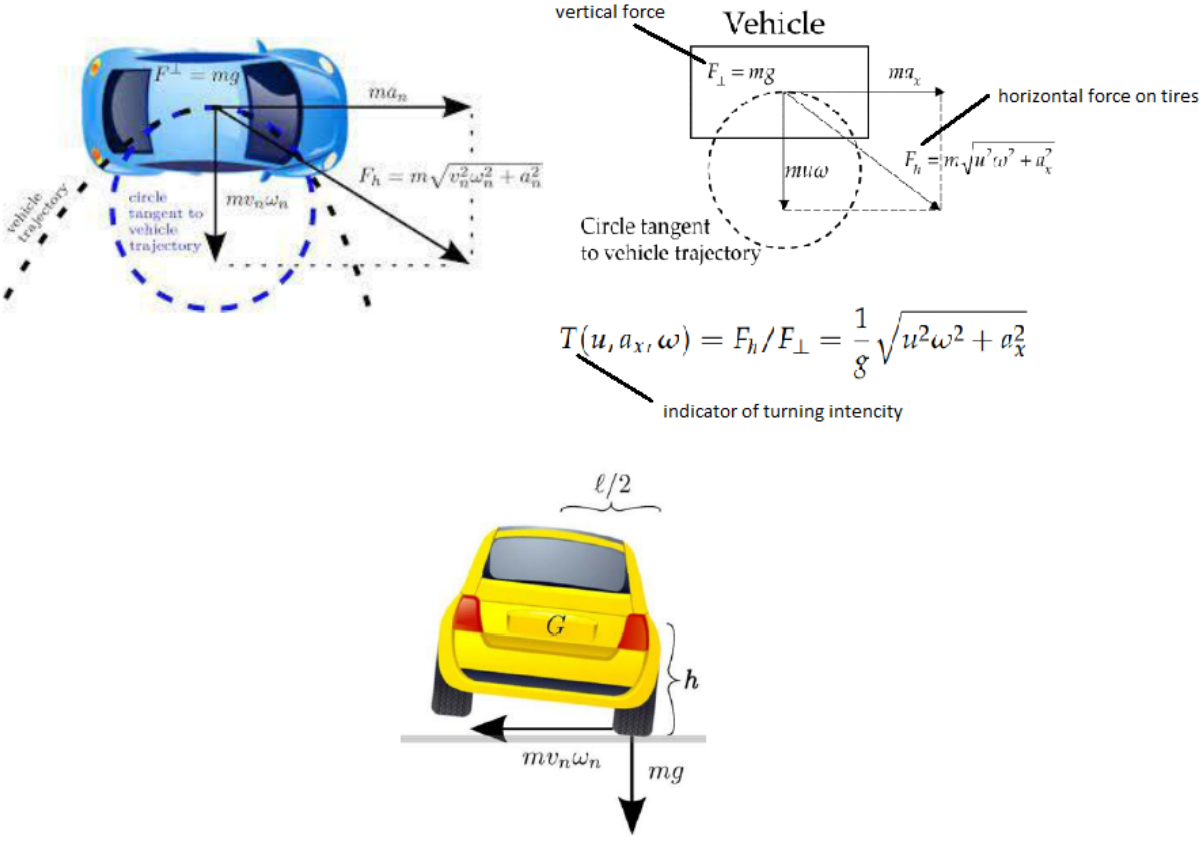

Figure 2: Vehicle forces

3 Outliers detection

For the detection of outliers, we will utilise a series of algorithms and methods and then we compare the results. The first one is a simple box plot, then a z-score approach and subsequently a distance-based method with kNN, a density-based method with Local Outlier Factor (LOF), a tree-based method with Isolation Forest and finally a simple time-series approach identifying peaks.

Box plot

This is one of the simplest outlier detection methods, considering points as outliers that are outside the $[Q_1 - 1.5 \cdot IQR, Q_3 + 1.5 \cdot IQR]$ range, where Q_1 and Q_3 are the first and third quartiles respectively. This means that 50% of the observations will have T values within the $[Q_1, Q_3]$ range and 25% below Q_1 and another 25% above Q_3 and $IQR = Q_3 - Q_1$. The method essentially classifies as outliers the points that are at least a few standard deviations away from the median value. The method is only applied to one variable. Below, we can see the region that a point should lie in for it to be classified as outlier. A point should have a T value of almost more than 3 deviations away from the median value for this to happen.

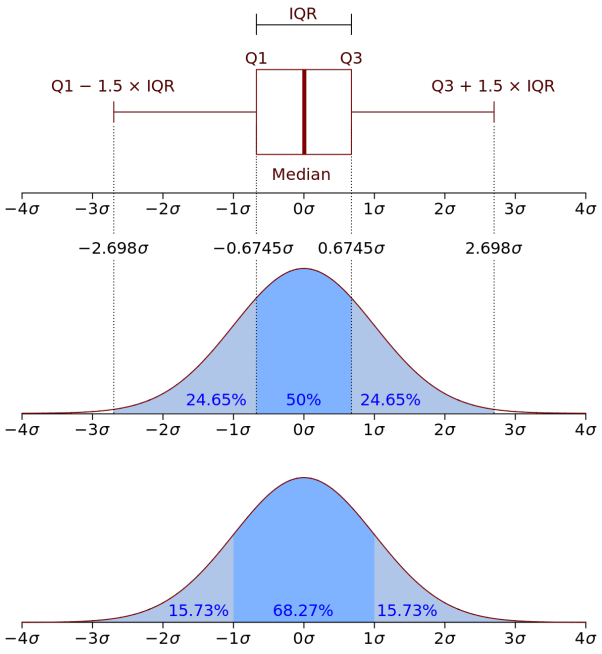

Figure 3: Box plot IQR curves

Performing the required calculations programmatically, we get

$$Q_1 = 0.0052154 \ , \ Q_3 = 0.0469025$$

$$IQR = 0.0416871 \ , \text{ and}$$

$$\text{Outliers number} = 11028.$$

We visualize these outliers in a 3D plot with its axes being the 3 variables that have been used in the definition of the T variable.

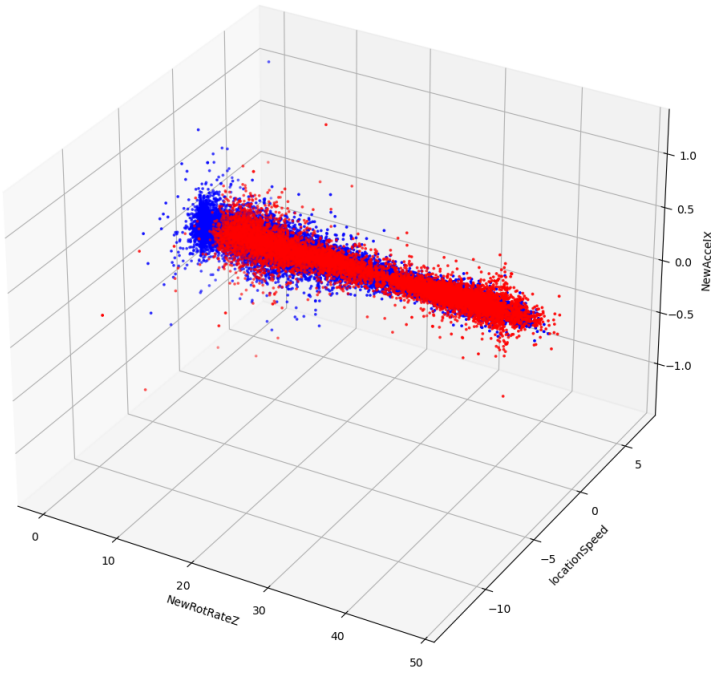

Figure 4: Box plot outliers 3D

Z-score

Another simple method for detecting outliers is the Z-score method. It quantifies the distance of a data point from the remaining of the dataset. It is defined as

$$\text{Z-score} = \frac{|T - mean_T|}{std_T}$$

and quantifies the number of standard deviations a particular observation is away from the mean. A threshold value is defined in this case and points that have a Z-score over the threshold are considered as outliers. Please note that here, the distance from the mean is used for the determination of outliers contrary to the box plot's usage of the median value instead.
We can plot the z-scores versus the index in order to visualize the z values.

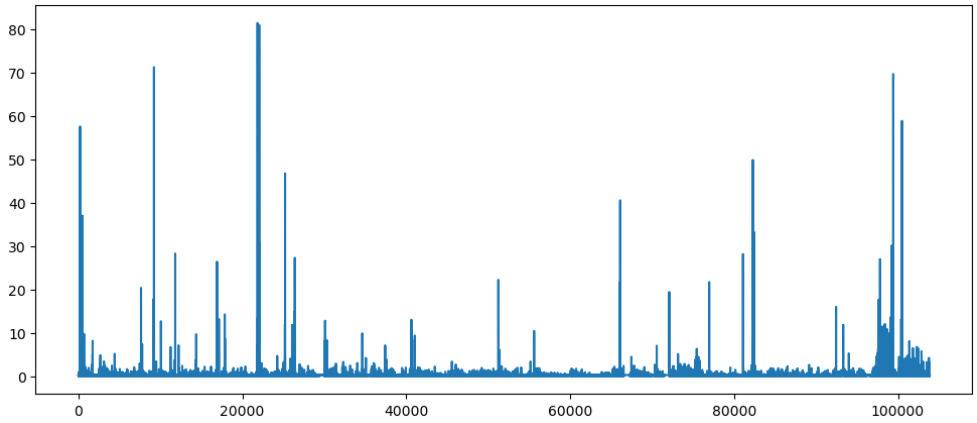

Figure 5: Z-score plot

In the plot, we can see some highly visible peaks as well as smaller ones that can potentially be classified as outliers. The threshold selected will highly influence the detection. Having that in mind, we create a range of potential threshold values programmatically and then we test the number of identified outliers for each one and compare them with an arbitrarily selected number of assumed outliers equal to 0.5% of the dataset records, i.e. $0.5\% \cdot 103891$ records $= 519$ records. The previous plot help manually define a suitable range of threshold values to be tested. The range selected was from 1.5 to 10 with a step of 0.25. The optimal threshold and outliers found were

$$\text{Z-threshold} = 4.25$$

$$\text{Outliers number} = 514.$$

Next, we visualize the outliers on the plot as well as in a 3D plot with the 3 variables that make up the T variable.

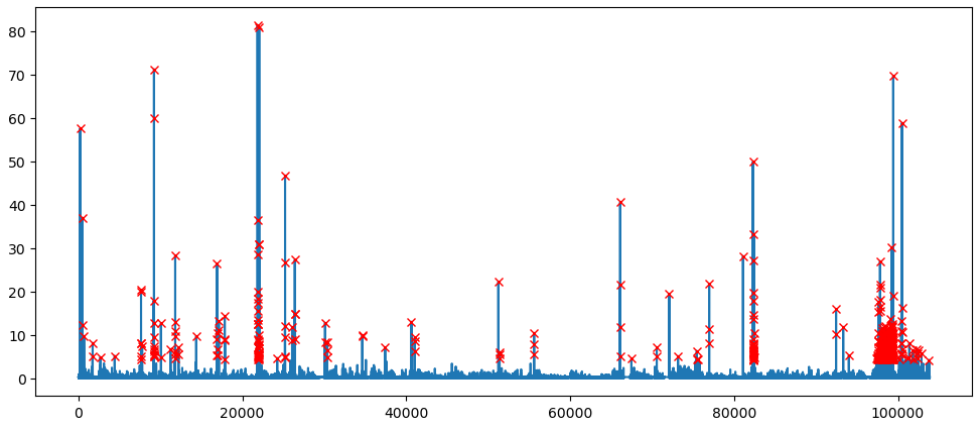

Figure 6: Z-score plot and outliers

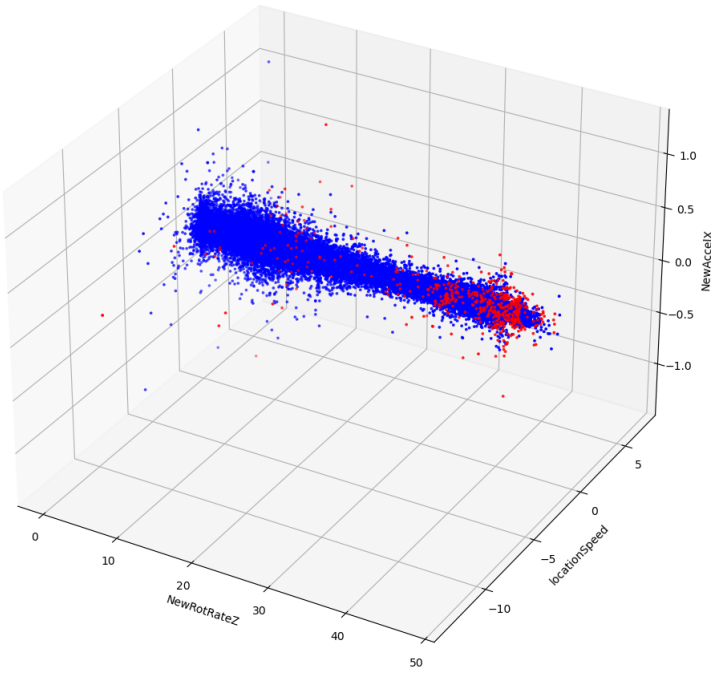

Figure 7: Z-score outliers 3D

Unlike the box plot method, with the Z-score method we can control the number of outliers by selecting an appropriate threshold.

kNN

We now employ a distance-based outlier detection algorithm, the k-Nearest Neighbors one. In this method, we compute the distance between each pair of data points first and then there are several ways to define outliers. The variables selected here are NewAccelX, NewRotRateZ, locationSpeed, and T. We selected the following way of outlier definition. For each data point, we locate the distances of its 3-nearest neighbors and then calculate the mean distance from those 3 neighbors for all data points. Then, similarly to the previous method, we define a range of threshold values to test for which we identify the outliers detected and compare with the 0.5% of the dataset's records. In the plot below, we can see the measured mean distances and the detected outliers that were derived by taking a range between 0 and 0.2 with a step of 0.0001, and finally we got

$$\text{Mean-distance threshold} = 0.147$$

$$\text{Outliers number} = 518.$$

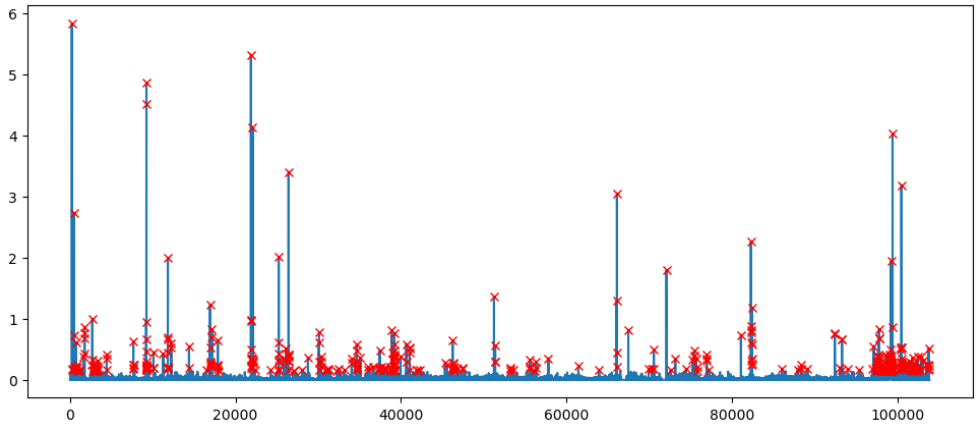

Figure 8: 3-NN plot and outliers

Below, we plot the outliers detected by the 3-NN method in two 3D plots in order to visualize the effect of the three variables.

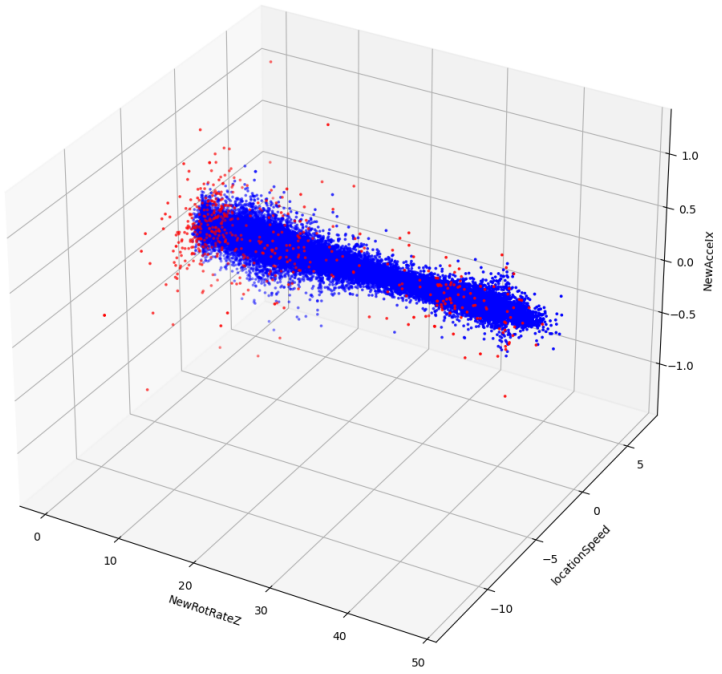

Figure 9: 3-NN outliers 3D - first view

We can already see from the 3D plot that there is some overlap on the outliers identified by 3-NN and Z-score.

Local Outlier Factor (LOF)

Next, we employ a density-based outlier detection method, where outliers are considered to be objects of low density. The same variables as with kNN are used. The LOF method quantifies the local density deviation of a data point with respect to nearby points, and if the density of a point is not similar to its neighbors but much smaller, then it is considered an outlier. There is a simple rule to identify the outliers which is LOF(k) > 1, where k is a chosen parameter for the number of nearest neighbors.

Similarly to the previous methods, a range of k values from 2 to 20 is used in order to identify a number of outliers close to 0.5% of the dataset's records. Upon programmatically testing this, we end up with

$$k = 7$$

$$\text{Outliers number} = 3565.$$

Below, we plot the 3D visualization of the detected outliers.

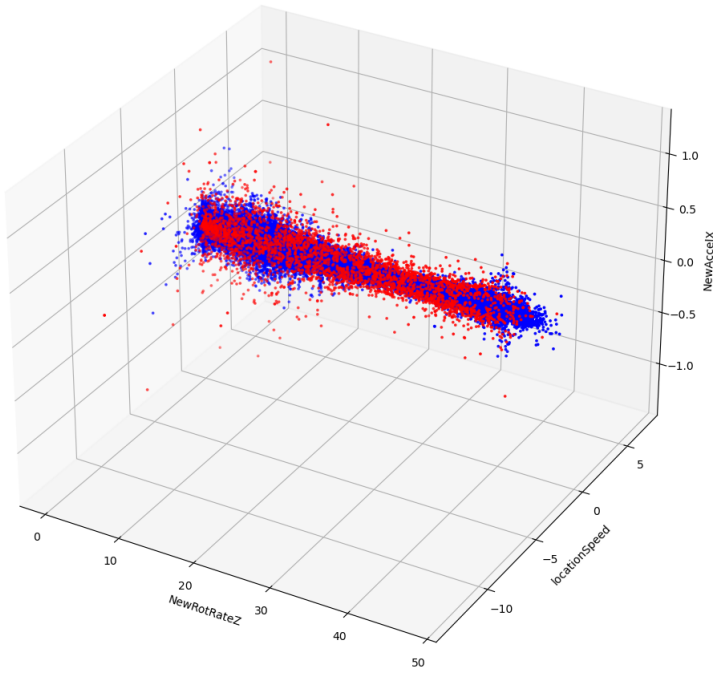

Figure 10: LOF outliers 3D

Isolation Forest

Another tree-based outlier detection method is tested, the Isolation Forest method, where a tree-based model is fitted (same variables as with kNN and LOF) and data with few splits to be isolated are considered outliers. In the sklearn's IsolationForest model, there is an important parameter, contamination, that has to be set. The amount of contamination of the data set is essentially the proportion of outliers in the data set. It is used when fitting to define the threshold on the scores of the samples and we defined a range of contamination values from 0.0025 to 0.5 with a step of 0.0025 for which the model is fitted and the outliers detected. We then keep the value for which the outliers number is closer to the 0.5% of the dataset's size. Finally, we get

$$\text{Contamination} = 0.005$$

$$\text{Outliers number} = 520.$$

Below is the 3D plot of the outliers vs the 3 variables from which T is calculated.

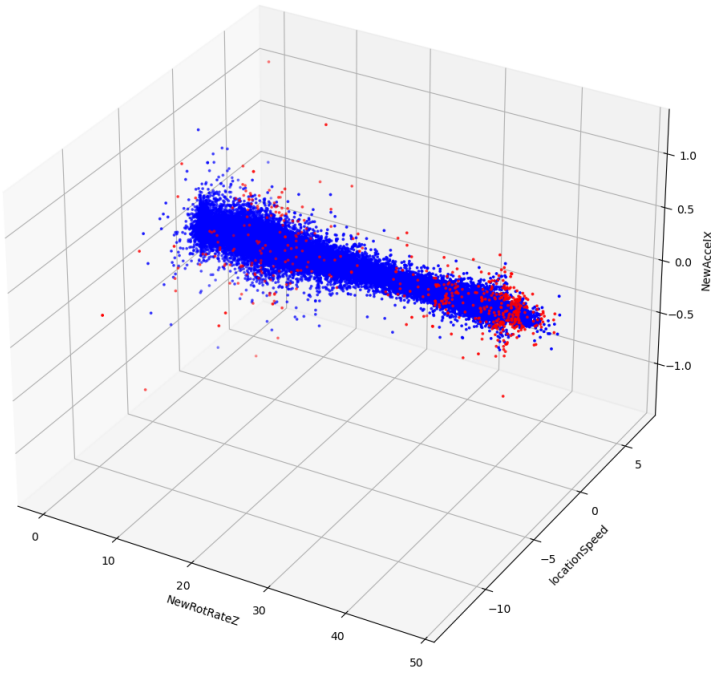

Figure 11: Isolation Forest outliers 3D

Time series peaks

Finally, we utilize a time-series approach to detect outliers based on the time-series peaks. We already saw the time-series plot of the T variable in Figure 12 in the Z-score subsection. Here, we utilize scipy's package that can identify time-series peaks, by setting up a height threshold above which a point is considered an outlier. Similarly to our previous approach, a range of heights from 0.05 to 10 with a step of 0.05, we iteratively calculate the number of peaks/outliers detected and keep the iteration and outliers detected that are closer to 0.5% of the dataset's length. As a result, we get

$$\text{Height threshold} = 0.25$$

$$\text{Outliers number} = 565.$$

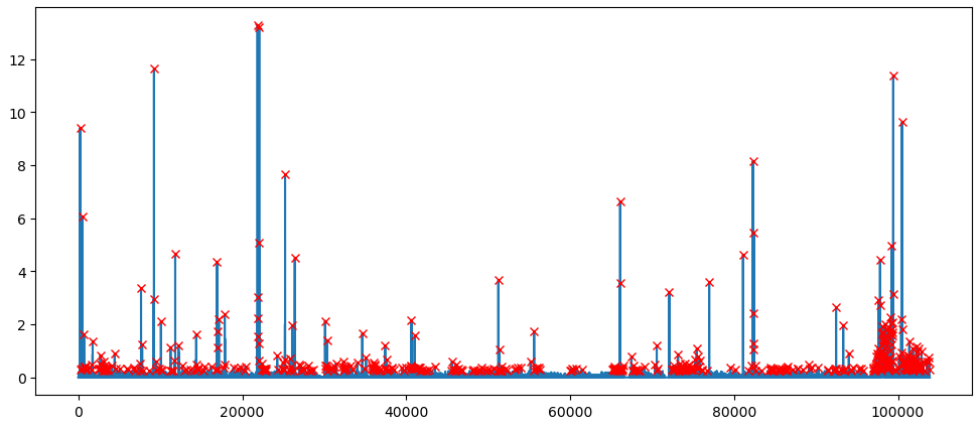

Figure 12: Time-series outliers plot

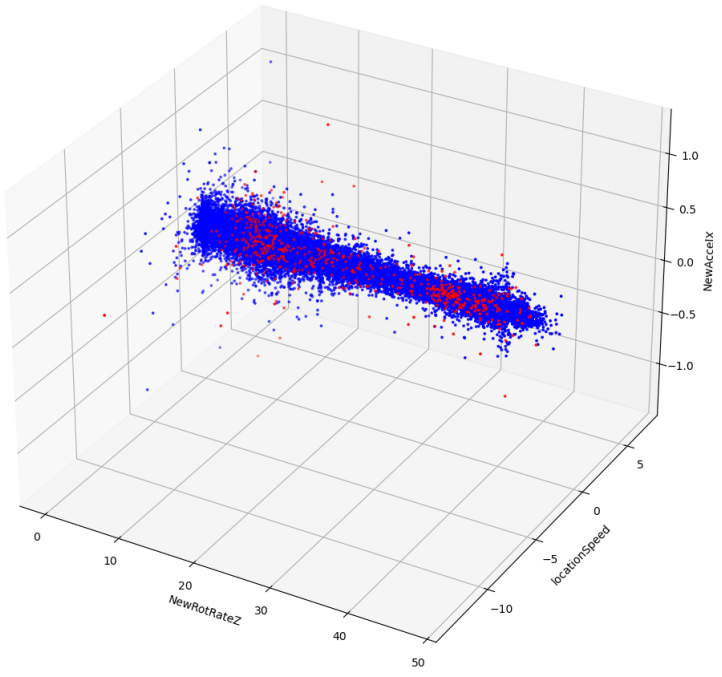

Figure 13: Time series peaks outliers 3D

4 Results

Having employed 6 different algorithms (Box plot, Z-score, kNN, LOF, Isolation Forest, Time-series peaks) to identify outliers, we now proceed in comparing the results from those. Since, there is no labels for the ground-truth regarding the outliers, we will seek more confidence in the methods' outlier classification based on their agreement on those. First, we take the results in pairs of two methods and present their agreement in a heatmap. We note also in the table below the number of outliers detected by each method.

Method	**Outliers Number**
Box Plot	11028
Z-score	514
kNN	518
LOF	3565
Isolation Forest	520
Time Series Peaks	565

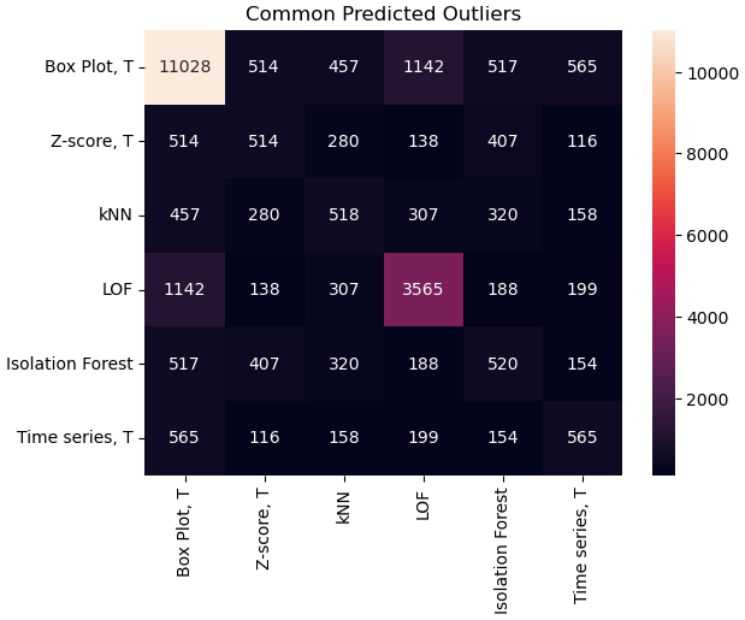

Figure 14: Heatmap of outliers

From the heatmap, we can derive the following conclusions:

- The box plot method seems to be coinciding with most of the methods (Z-score, kNN, Isolation Forest, Time series) for most of their outliers, given that the box plot is generating a big number of outliers. It coincides with the LOF method in only a third of the LOF outliers.
- The Z-score method coincides with approximately 80% of the Isolation forest outliers.
- kNN coincides in about 300 outlier cases cases with Z-score, LOF and Isolation Forest.

- LOF doesn't seem to coincide in more than 300 cases with the methods that generate approximately 500 outliers, which probably means that most of its identified 3565 outliers are false positives.
- The isolation forest method has an agreement of about 65% with kNN.
- The time series method does not seem to agree for more than 40% of the cases with most methods.

Then, we can remove the Box plot and LOF methods from the heatmap in order to concentrate on the methods that identified approximately 500 outliers in the dataset.

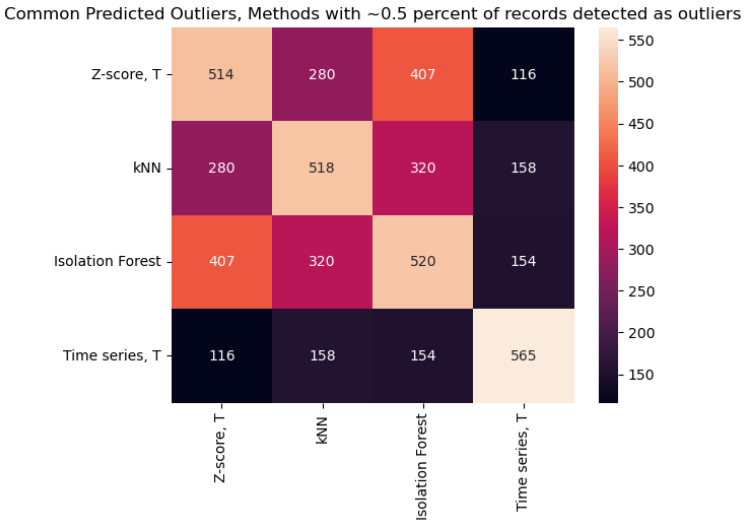

Figure 15: Reduced heatmap of outliers

From the last heatmap, we can see that there is a high level of agreement (more than 60%) between the three methods of Z-score, kNN and Isolation Forest. Based on that result, we conclude to creating a majority vote from these three methods in order to classify records as outliers. Upon performing this programmatically, we end up with 501 records identified as outliers based on majority voting (i.e. at least two of the three methods agree that it's an outlier). Also, there are 253 records that all 3 methods agree are outliers, hence those points have even higher confidence of being outliers. The results of the outlier detection based on the majority voting is exported in a csv file (outliers.csv). Also, we present the 3D plot of the majority vote outliers.

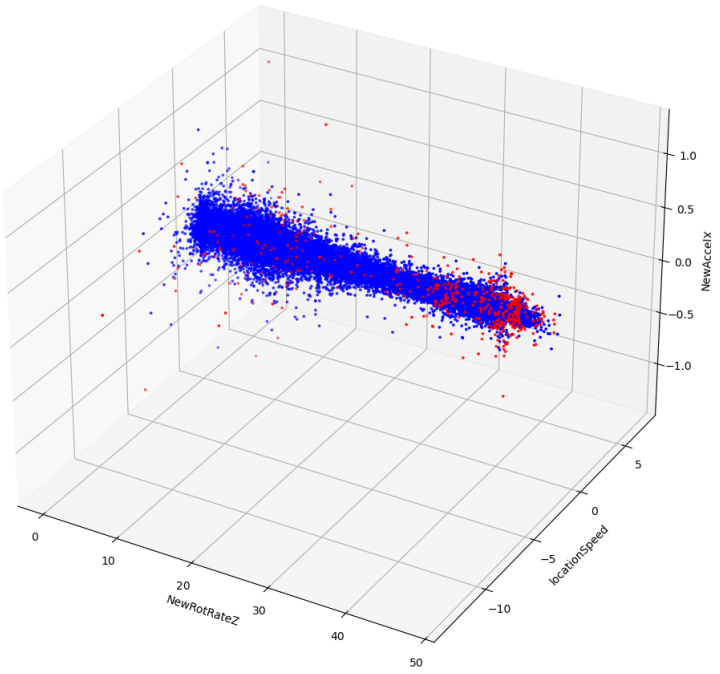

Figure 16: Majority vote outliers 3D

As a final note, the results are highly dependent on the assumption made initially about circa 0.5% of the dataset records being outliers. Any results can be far off the truth, if this assumption doesn't hold.