STATUS REPORT: DIH SUMMER PROJECT

Probabilistic Approach Based Anomaly Detection Techniques for Real Time Systems

Prepared by:

Shikhar Gupta [15035040]
Rahul Meena [16084012]
Harshit Mishra [16155031]

Supervised by:

Dr. Hari Prabhat Gupta
Dept. of Computer Science

Problem Statement :

- In a **data**-driven culture, companies approach **decision-making** from a quantitative point of view and rely less on gut feeling when **making** major **decisions**. ... Lack of accurate, timely or relevant **data** from across the business is also a major concern among companies with primitive or basic capabilities.

- In statistics, an **outlier** is an observation point that is distant from other observations. ... An **outlier** can cause serious **problems** in statistical analyses. **Outliers** can occur by chance in any distribution, but they often indicate either measurement **error** or that the population has a heavy-tailed distribution.

Anomaly Detection problems

Anomaly detection is applicable in a variety of domains, such as intrusion detection, fraud detection, fault detection, system health monitoring, event detection in sensor networks, and detecting ecosystem disturbances. It is often used in preprocessing to remove anomalous data from the dataset. In supervised learning, removing the anomalous data from the dataset often results in a statistically significant increase in accuracy.

Dataset : Single Chest-Mounted Accelerometer Dataset

Data Set Information:

- The dataset collects data from a wearable accelerometer mounted on the chest
- Sampling frequency of the accelerometer: 52 Hz
- Accelerometer Data are Uncalibrated
- User Activity: Walking
- Data Format: CSV
- Training Instance = 25000

Attribute Information:

sequential number, x acceleration, y acceleration, z acceleration

Data Statistics:

	x-axis	y-axis	z-axis
count	25000.000000	25000.00000	25000.00000
mean	1884.735640	2380.74432	2052.27404
Std	45.984328	88.14063	48.85544
min	1704.000000	2179.00000	1878.00000
25%	1859.000000	2315.00000	2019.00000
50%	1886.000000	2364.00000	2043.00000
75%	1908.000000	2433.00000	2085.00000
max	2110.000000	2678.00000	2248.00000

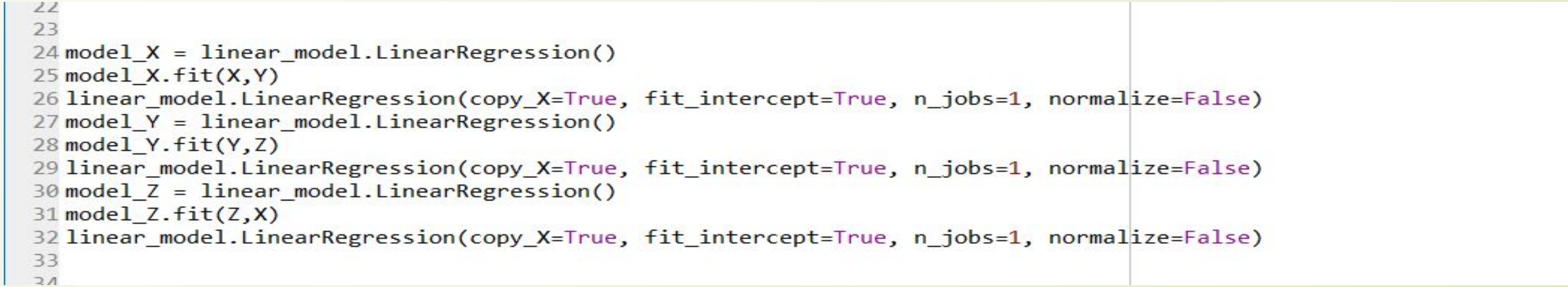
Our approach:

We are using linear regression technique for anomaly detection in real time system data. **linear regression** is a **linear approach** to modelling the relationship between a scalar response (or dependent variable) and one or more explanatory variables (or independent variables). The case of one explanatory variable is called simple **linear regression**.

Implementation:

Importing Libraries :- numpy,matplotlip,pandas, sklearn, linear model

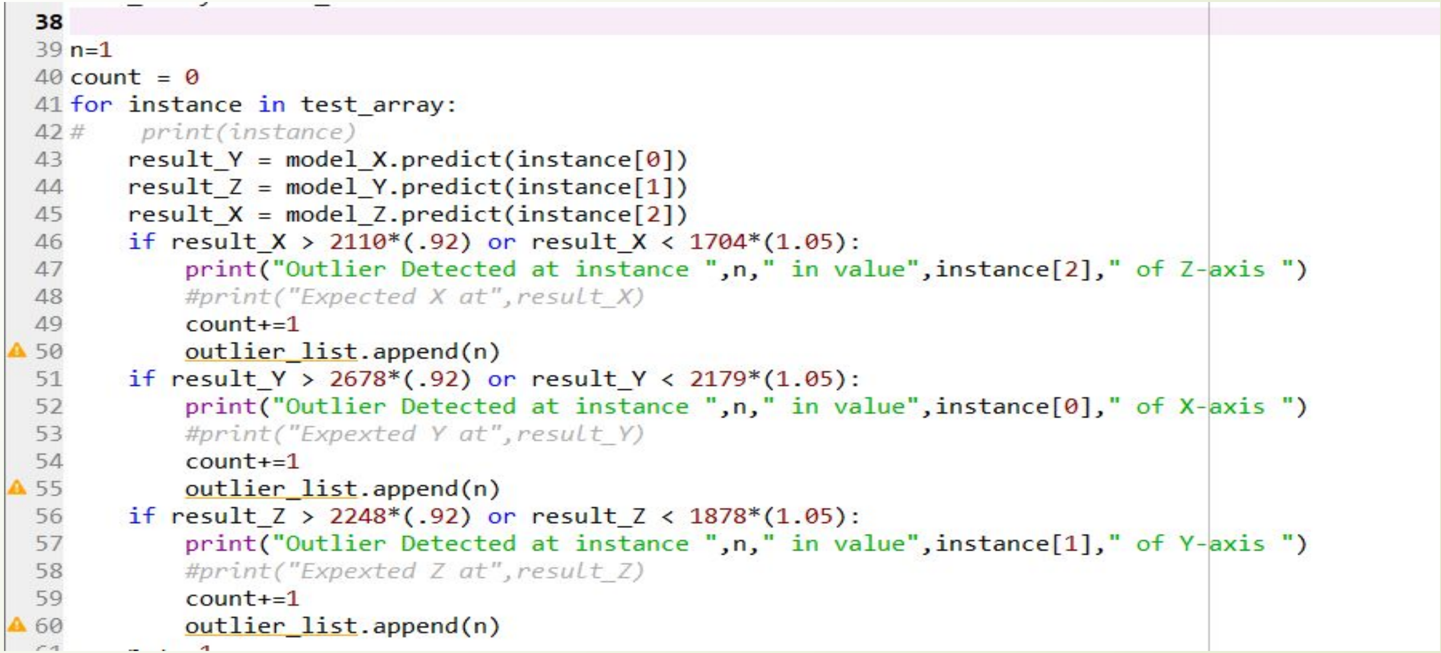
Training data loading
Attribute :- x,y,z
25000 intense



```
model_X = linear_model.LinearRegression()
model_X.fit(X,Y)
linear_model.LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False)
model_Y = linear_model.LinearRegression()
model_Y.fit(Y,Z)
linear_model.LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False)
model_Z = linear_model.LinearRegression()
model_Z.fit(Z,X)
linear_model.LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False)
```

Test data loading
Attribute :- x,y,z
100 intense

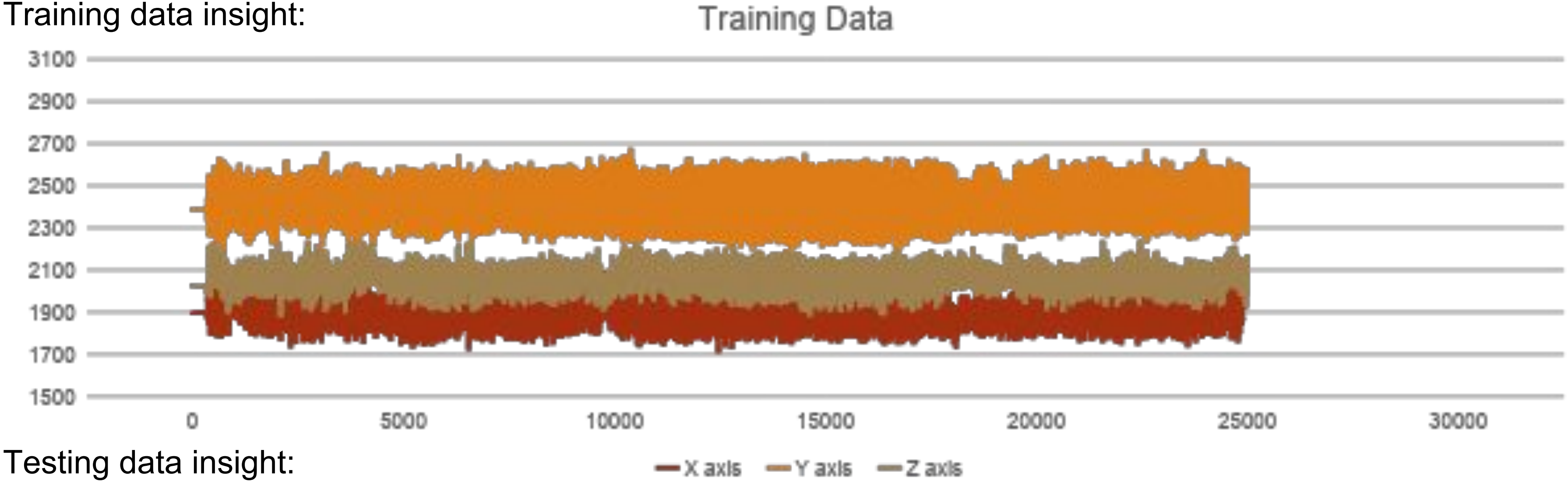
Continue..



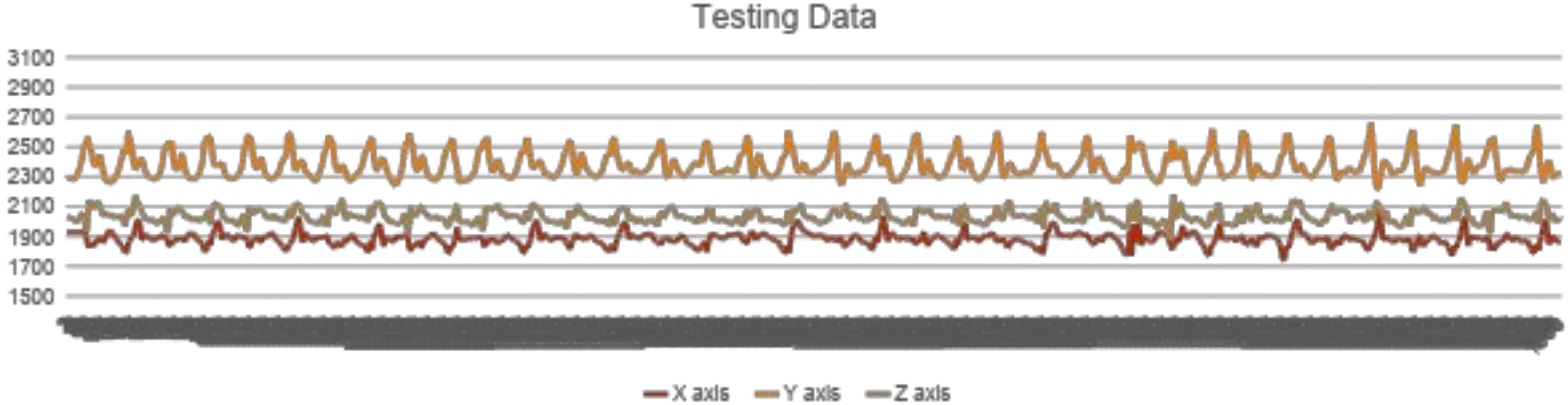
```python

n=1
count = 0
for instance in test_array:
#    print(instance)
    result_Y = model_X.predict(instance[0])
    result_Z = model_Y.predict(instance[1])
    result_X = model_Z.predict(instance[2])
    if result_X > 2110*(.92) or result_X < 1704*(1.05):
        print("Outlier Detected at instance ",n," in value",instance[2]," of Z-axis ")
        #print("Expected X at",result_X)
        count+=1
        outlier_list.append(n)
    if result_Y > 2678*(.92) or result_Y < 2179*(1.05):
        print("Outlier Detected at instance ",n," in value",instance[0]," of X-axis ")
        #print("Expexted Y at",result_Y)
        count+=1
        outlier_list.append(n)
    if result_Z > 2248*(.92) or result_Z < 1878*(1.05):
        print("Outlier Detected at instance ",n," in value",instance[1]," of Y-axis ")
        #print("Expexted Z at",result_Z)
        count+=1
        outlier_list.append(n)
```

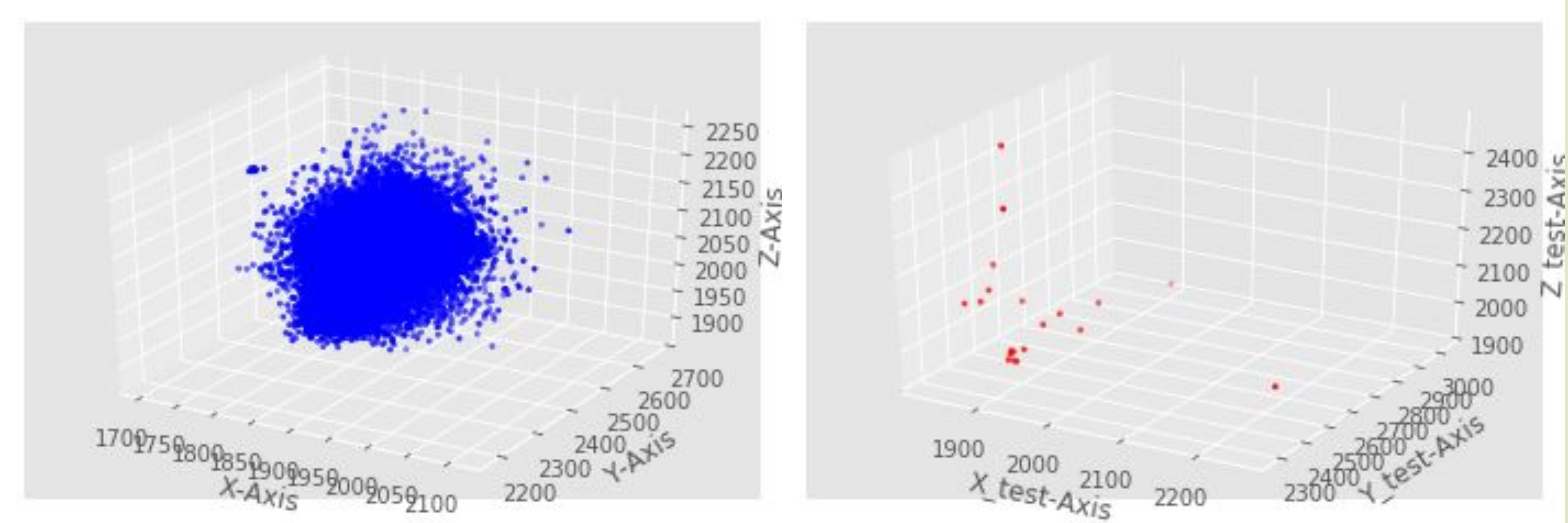
Plotting – 3D graph with X,Y,Z attribute loading of both training and testing data

Training data insight:



Testing data insight:




2250
2200
2150
2100
2050
2000
1950
1900
Z-Axis
2700
2600
2500
2400
2300
2200
Y-Axis
1700
1750
1800
1850
1900
1950
2000
2050
2100
X-Axis
2400
2300
2200
2100
2000
1900
Z test-Axis
3000
2900
2800
2700
2600
2500
2400
2300
Y_test-Axis
1900
2000
2100
2200
X_test-Axis

Thank you