MLHACK\_01: FALL DETECTION 1

‘

**MLHack\_01: Fall Detection**

**Anand Kumar Singh**

**HCL Technologies**

**(Employee code 51500730)**

MLHACK\_01: FALL DETECTION 2

**Abstract**

Falls are the main cause of injury for OLD People. There are different activities like Standing, Walking, Sitting, Cramps, Running and Falling

There are so many reasons for their fall - High Blood sugar, High BP, High Blood circulation etc.

This paper aims to build an automated fall detection system with the help of motion sensor units data fitted to the person’s body at six different positions. This Sensor device sends all the above mentioned six different positions in real time every 4 seconds with the help of IOT device. We apply statistical analysis and machine learning algorithms to find out insights and predict the fall detection.

‘

MLHACK\_01: FALL DETECTION 3

Contents

[**Prepare Problem** 4](#_Toc25436341)

[ **Load Libraries** 4](#_Toc25436342)

[ **Load datasets** 4](#_Toc25436343)

[ **Total list of columns (attributes)** 4](#_Toc25436344)

[**Summarize Data** 5](#_Toc25436345)

[ **Datatypes of attributes** 5](#_Toc25436346)

[ **Brief About All variables** 5](#_Toc25436347)

[ **Variable types** 6](#_Toc25436348)

[ **Summary Statistics** 6](#_Toc25436349)

[ **Percentage of each event in “Activity” columns** 6](#_Toc25436350)

[ **Find out duplicate entry** 7](#_Toc25436351)

[ **Find out missing value** 7](#_Toc25436352)

[ **Insights** 7](#_Toc25436353)

[**Data Cleaning** 8](#_Toc25436354)

[ **Out Layers** 8](#_Toc25436355)

[ **Used Methods** 8](#_Toc25436356)

[ **Reduce Out layers** 8](#_Toc25436357)

[**EDA Analysis** 11](#_Toc25436358)

[ **Assumption** 13](#_Toc25436359)

[**Normal Distribution** 13](#_Toc25436360)

[ **Insights** 14](#_Toc25436361)

[**Data Splitting** 14](#_Toc25436362)

[**Variable Reduction** 14](#_Toc25436363)

[**Model Building** 15](#_Toc25436364)

[**Model (KNeighborsClassifier)** 15](#_Toc25436365)

[ **Train and test accuracy** 15](#_Toc25436366)

[**Recall Value** 16](#_Toc25436367)

[ **Precision Recall and Fscore** 16](#_Toc25436368)

[ **ROC Curve** 16](#_Toc25436369)

[**Challenges** 16](#_Toc25436370)

[**Future work** 17](#_Toc25436371)

MLHACK\_01: FALL DETECTION 4

# **Prepare Problem**

## **Load Libraries**

We are using below libraries to find out all insights and predict the fall activity.

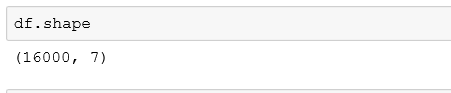
* import numpy as np
* import pandas as pd
* import seaborn as sns
* import matplotlib.pyplot as plt
* from sklearn.preprocessing import MinMaxScaler
* from statsmodels.graphics.gofplots import qqplot
* from sklearn.model\_selection import train\_test\_split
* from sklearn.linear\_model import LogisticRegression
* import statsmodels.formula.api as sm
* %matplotlib inline

## **Load datasets**

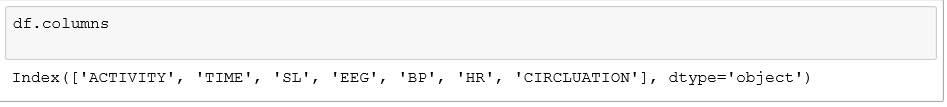
To build the model, we got train and test data sets in form of csv files. We build our model with help of train data sets.

df=pd.read\_csv('Train.csv')

The file has 16000 records and 7 attributes



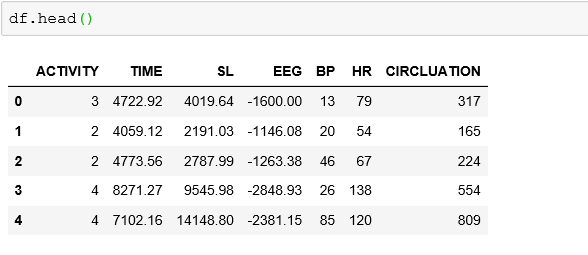
## **Total list of columns (attributes)**



MLHACK\_01: FALL DETECTION 5

# **Summarize Data**

## **Datatypes of attributes**



## **Brief About All variables**

**Activity**

0🡪Is stand for “**Standing**”

1🡪Is stand for “**Walking**”

2🡪Is stand for “**Sitting**”

3🡪Is stand for “**Falling**”

4🡪Is stand for “**Cramps**”

5🡪Is stand for “**Running**”

**Time :** Monitoring Time

**SL**: Sugar Level

**EEG**: Electroencephalograph Monitoring Rate

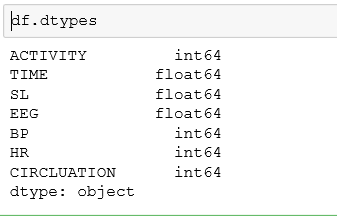
**BP**: Blood Preasure

**HR**: Heart beat rate

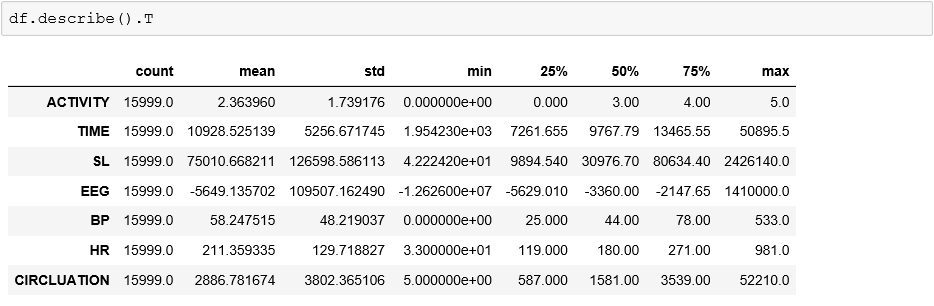
**Circulation**: Blood Circulation

MLHACK\_01: FALL DETECTION 6

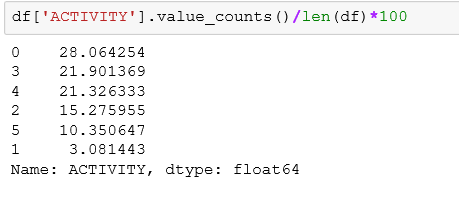
## **Variable types**



## **Summary Statistics**



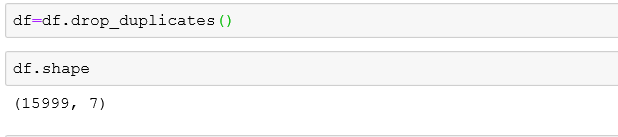
## **Percentage of each event in “Activity” columns**



MLHACK\_01: FALL DETECTION 7

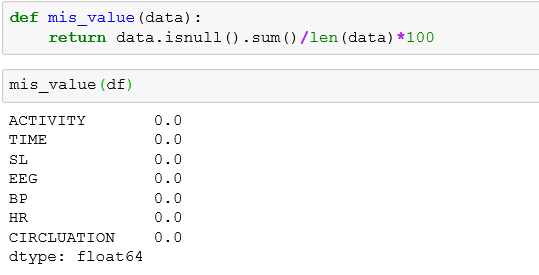
## **Find out duplicate entry**

We observed that we have one duplicate entry in our data as per below out put



## **Find out missing value**

There is no missing value in provided data sets



## **Insights**

* Activity column is having 6 categorical variables
* Our Event rate is ~20%, So it will good to build the model
* There are out layers in all Variable after 75% of records.
* We have one duplicate entry in our data

MLHACK\_01: FALL DETECTION 8

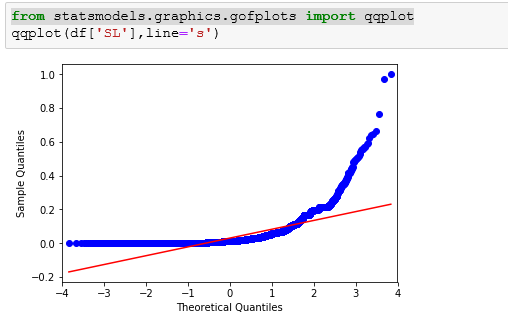
# **Data Cleaning**

## **Out Layers**

Using qq plot to find Out layers.

* Activity columns is categorical data, there are no out layers in Activity columns

## **Used Methods**



From the Graph it is clear that we have upper out layers in ‘SL’ columns.

We checked the same for all the variables and infer that there are out layers in all these variables as specified below

SL🡪it has Upper out layers

EEG🡪it has Upper as well Lower out layers

BP🡪 it has Upper out layers

CIRCLUATION🡪 it has Upper out layers

## **Reduce Out layers**

We are using IQR method to reduce Out layers.

def iqr\_vlue(data):

iqr=data.quantile(0.75)-data.quantile(0.25)

lo=data.quantile(0.25)-3\*iqr

uo=data.quantile(0.75)+3\*iqr

data\_min=data.min()

data\_max=data.max()

return lo,data\_min,uo,data\_max

We created UDF for IQR which will return Lower and Upper IQR value, Min and Max value.

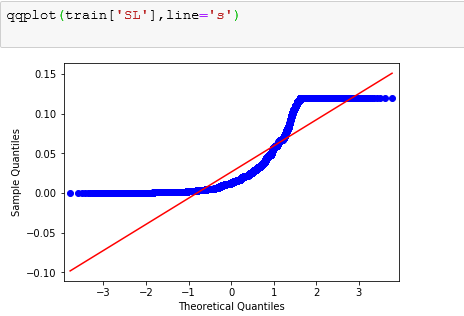
MLHACK\_01: FALL DETECTION 9

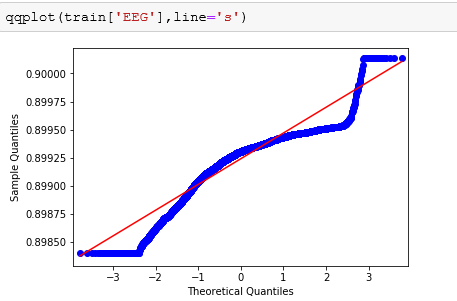
Upper out layers will be replaced with Upper IQR value and Lower out layers will be replaced with Lower IQR value as shown below.

train['SL']=np.where(train['SL']>iqr\_vlue(train['SL'])[2],iqr\_vlue(train['SL'])[2],train['SL'])

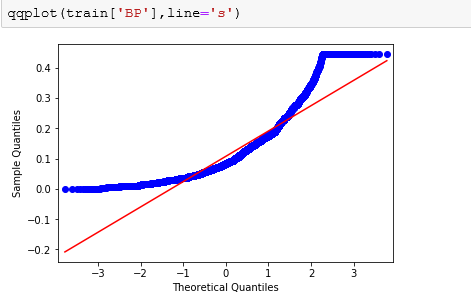
train['EEG']=np.where(train['EEG']<iqr\_vlue(train['EEG'])[0],iqr\_vlue(train['EEG'])[0],train['EEG'])

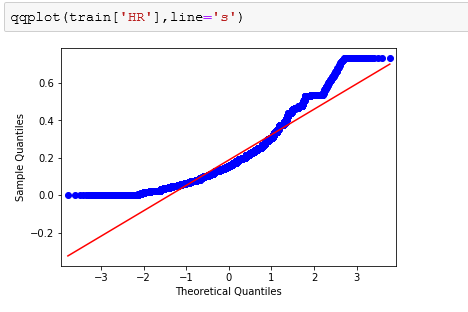
This was done for all the variables and is shown via graphs below .



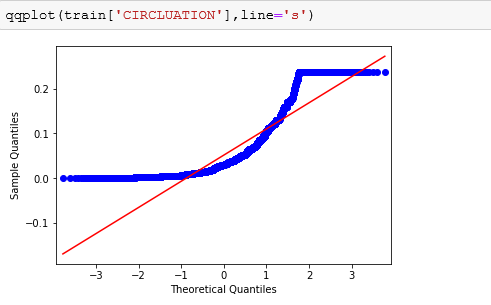


MLHACK\_01: FALL DETECTION 10



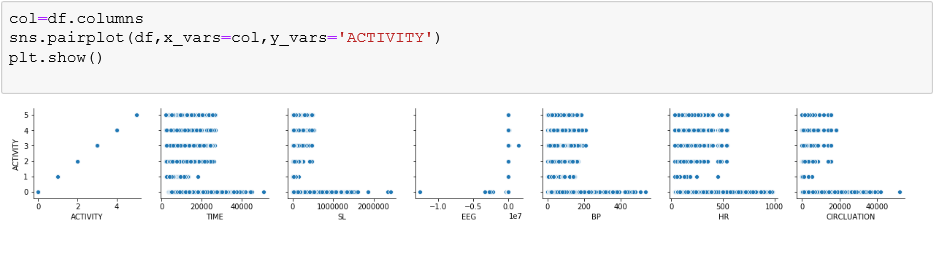


MLHACK\_01: FALL DETECTION 11



# **EDA Analysis**

Plotting all the variables w.r.t Activity columns



Sorting the Time variable to find out what activity happened before Fall and After Fall

Created new data frame with below variables.

* Before -->Activity Before Fall
* Fall-->Activity is equal to Fall
* After-->Activity After Fall

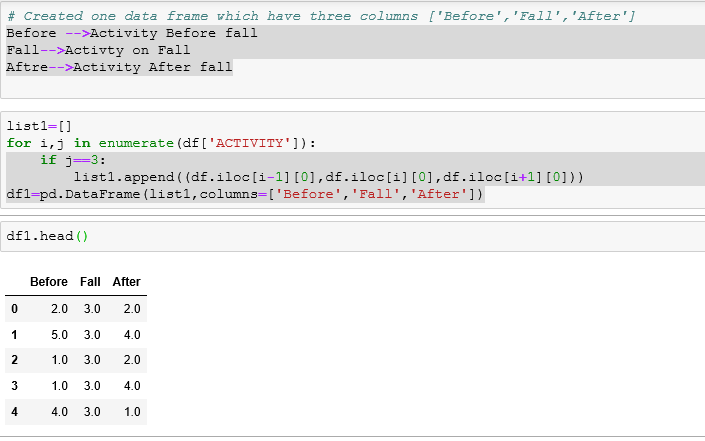
df.sort\_values("TIME", axis = 0, ascending = True,

inplace = True, na\_position ='first')

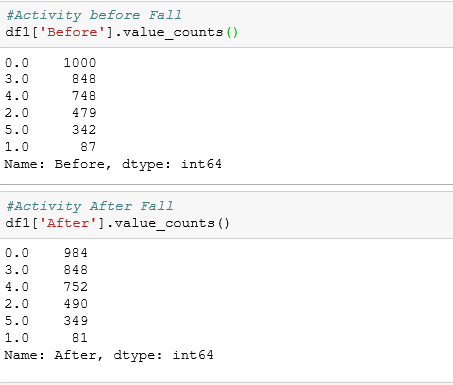
df.reset\_index(drop=True)

df.shape

MLHACK\_01: FALL DETECTION 12



Find out count of Activity Before and After Fall

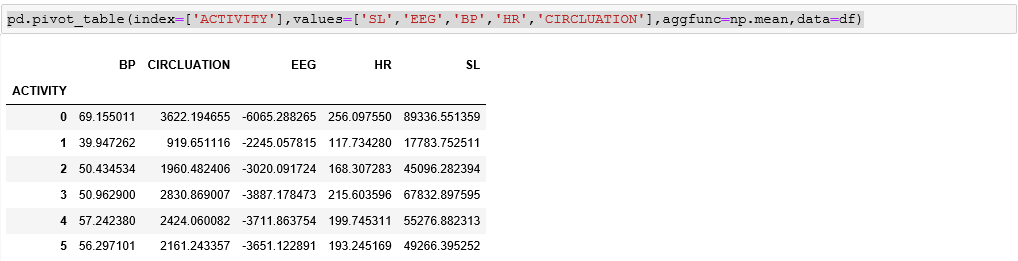


MLHACK\_01: FALL DETECTION 13

Dropping Time Columns, as we have sorted the datasets and have found out Before and After Falling Activity



Mean value of all variables w.r.t Activity column



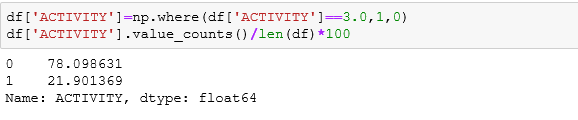
## **Assumption**

We are making our Activity columns as Binary data

1🡪Falling result code 3

0🡪Other then Falling Result code 3

Find out event rate for Activity columns



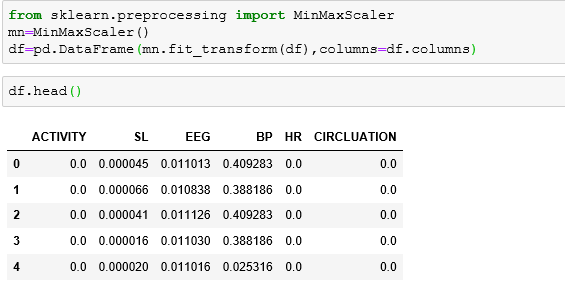
# **Normal Distribution**

Using MinMaxScaler for Normal distribution

As per Normal distribution our DataSets should follow z score rule, As per this rule our data set distribution should be under the range -3 to +3 .

After using MinMaxScaler our full data sets will come under the range 0 to 1.

MLHACK\_01: FALL DETECTION 14



## **Insights**

**1-** As per the above values obtained from EDA analysis, we infer that before falling, the activity counts of Standing and Cramp are more

So,whenever we are getting activity code 0 or 4 ,we need to be alert.

**2-**As per mean value of all variables w.r.t Activity columns, we gather that Falling and Cramp have

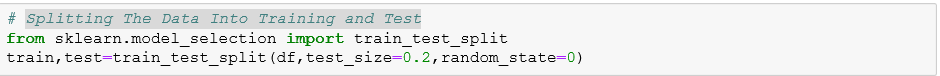
almost close values

So,whenever we are getting activity Cramp ,we need to be alert.

**3-**Since our Event rate is ~21%, it is good to build the model.

# **Data Splitting**

Splitting data into train and test data sets



# **Variable Reduction**

To find the best suited variable, we used VIF technique

VIF= 1/(1-R^2)

R^2= Coefficient of determination

R~ r (R^2 is almost equal as r)

r= Correlation ~0.5

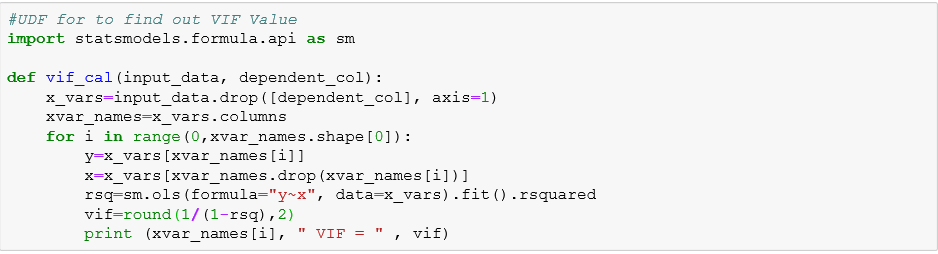
VIF=1/ (1-.5)

VIF~2

This value suggests that the best suitable variables will be those whose VIF value is less than 2 .

MLHACK\_01: FALL DETECTION 15

User Defined Function(UDF) to determine VIF Value

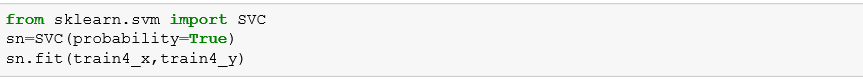




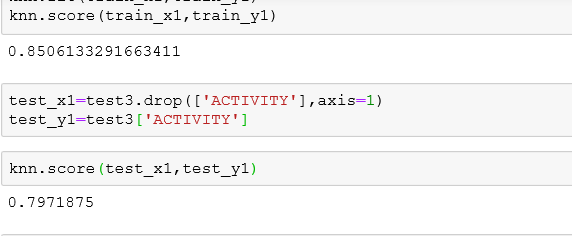
Construct the model with the help of above ‘SL’ and ‘BP’ variables (whose VIF values are less than 2 )

# **Model Building**

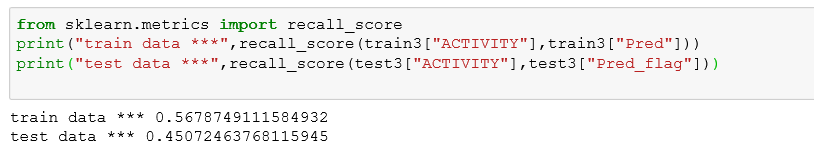
## **Model (KNeighborsClassifier)**



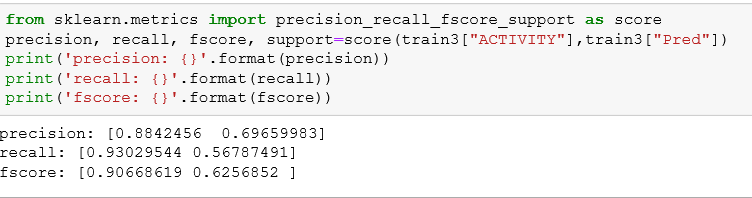
### **Train and test accuracy**



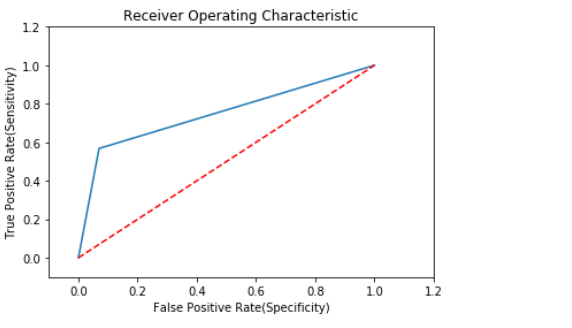
MLHACK\_01: FALL DETECTION 16

**Recall Value** 

### **Precision Recall and Fscore**



### **ROC Curve**



# **Challenges**

* Very limited information about the data sets
* All Variables are not normally distributed

MLHACK\_01: FALL DETECTION 17  
  
  
 **Recommendation**

* If Sugar level increases, then chance of Falling will increase
* If BP level decreases, then chance of Falling will increase
* If we get Cramp Activity code, we need to be alert
* Final equation to be inserted

# **Future work**

* Trying to retune model for better accuracy and other measurement parameters
* There is some issue found in Logistic regression so I removed from the model

I will retune the model and submit it in next release .

* Planning to use KMeans Cluster Algorithm for better accuracy