**To detect whether person is running or walking using machine learning in python**

**By Ashish Shinde**

|  |  |  |
| --- | --- | --- |
| **Serial number** | **Description** | **Page number** |
| 1 | objective | 2 |
| 2 | Data dicription | 2 |
| 3 | Introduction | 2 |
| 4 | Loading and preprocessing of data | 3 |
| 5 | Exploratory analysis | 4 |
| 7 | Machine learning approach (parameter tuning of SVC) | 6 |
| 8 | Machine learning approach (training and evaluation of SVC) | 6 |
| 9 | Deep learning approach(exploration) | 7 |
| 10 | Deep learning approach (data preprocessing and modelling) | 8 |
| 11 | Deep learning(evaluation) | 8 |
| 12 | Deep learning  (self-evaluation) | 9 |
| 13 | Conclusion and references | 11 |

**Objective:**

my aim is to detect Whether a person is running or walking based on deep neural network. The sensor data was collected from IOS device. I took this data from Kaggle competition the data is broadly described below.

### Data description:

Currently, the dataset contains a single file which represents 88588 sensor data samples collected from accelerometer and gyroscope from iPhone 5c in 10 seconds interval and ~5.4/second frequency. This data is represented by following columns (each column contains sensor data for one of the sensor's axes):

* acceleration\_x
* acceleration\_y
* acceleration\_z
* gyro\_x
* gyro\_y
* gyro\_z

There is an activity type represented by "activity" column which acts as label and reflects following activities:

* "0": walking
* "1": running

Apart of that, the dataset contains "wrist" column which represents the wrist where the device was placed to collect a sample on:

* "0": left wrist
* "1": right wrist

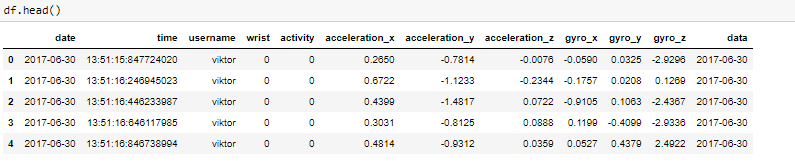
Additionally, the dataset contains "date", "time" and "username" columns.

**Introduction**Smartphones which are almost available to everybody are now offering many other features such as multitasking and deployment of variety of sensors due to which it has become easy to keep a track of our activities, learn them and help in making better decision regarding future actions.

In this project the Data produced by accelerometer and gyroscope will be used to detect human activity(walking/running) for this project I have used two approaches one is machine learning approach and another is deep learning approach(LSTM).

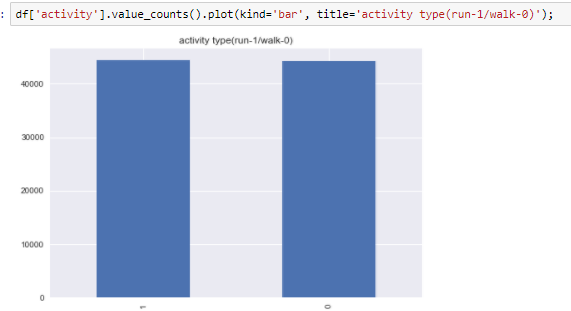
**Loading the data**

* Dataset was available in csv and was loaded with the help of pandas the following fig shows the data set



**Preprocessing**

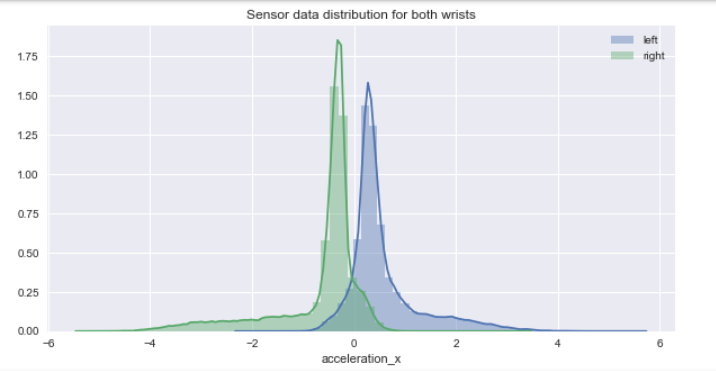
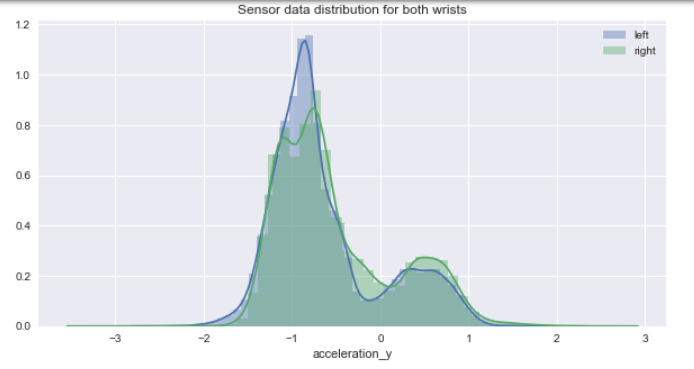
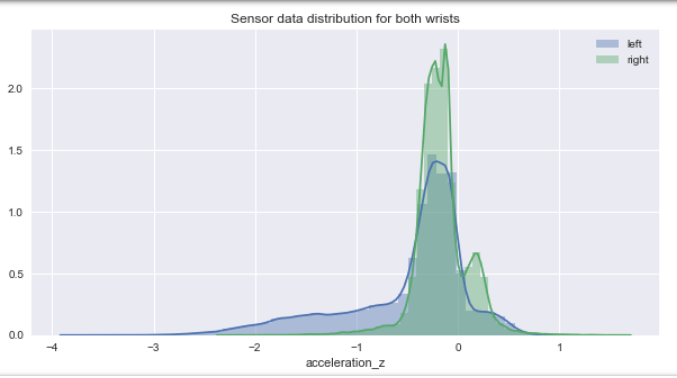
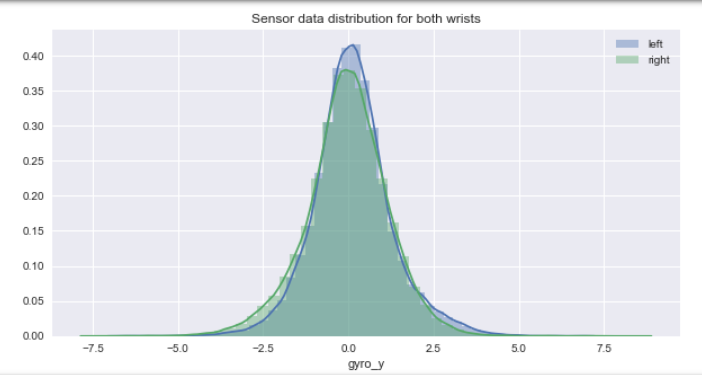
* Since the range of values of raw data does not varies widely scaling is not required
* There were no NAN values present in the data set
* 0 is walk and 1 is run below we can see that the count of both run and walk is nearly equal so there is no need of sampling



* the walk has 44223 values in that 18622 are left wrist and 25601 are right wrist
* the run has 44365 values in that 23708 left wrist and 20657 right wrist
* The only issue is that the number of walk samples recorded on the right wrist is significantly more than the samples recorded on the left wrist.
* However, the most important here is that the total number of walk and run samples is almost the same.

**Exploratory analysis**

* Since we have samples from both left and wright wrists, it makes sense to explore distribution of numerical data separately for both wrists.



The data set analyzed is properly formatted and has no NAN values and has clear values

Regarding the sensor data presented in the dataset, we can conclude that the data presented doesn't suffer from skewing and normally distributed.

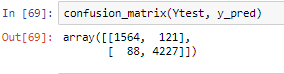
**Machine learning approach**

* Initially I used PCA for future reduction of our input variables since after retaining 99% of variance it gave 6 components therefore PCA is not useful in this case
* Data was split into train and validation set using sklearn.cross\_validation.train\_test\_split(test\_size=0.3)
* Since it is binary classification I used different machine learning algorithms like **SVM,KNN,Logistic regression, Naïve bayes, Random forest** and due to page limit constraint I have documented only Support vector classifier

**Parameter tuning of SVC**

* I used gridsearchcv method for this approach Using GridSearchCV is easy. You just need to import GridSearchCV from sklearn.grid\_search (1)  
  By using this the Best C: 10. Best Kernel: rbf. Best Gamma: 0.01

**Training and evaluation of SVC**

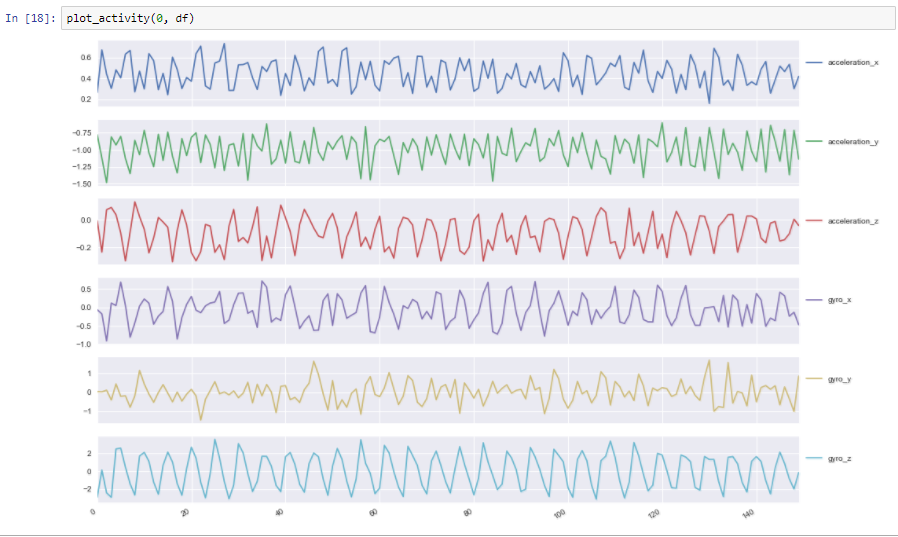
* Using these parameters I trained the model by importing svc from sklearn since the data was huge it was sliced to 20000 observation
* Validation accuracy was measured using metrics.accuracy\_score the the validation accuracy was 0.9651666666666666
* The confusion matrix is
* 
* Precision is 0.98
* Recall is 0.95

**Deep learning approach**

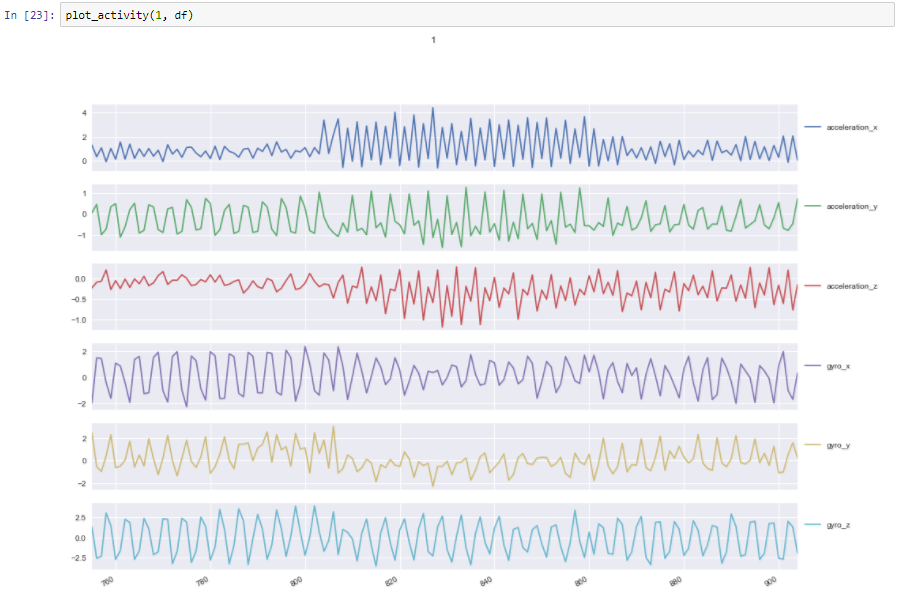
**Exploration**

A plot was plotted which displays observations on the y-axis against equally spaced time intervals on the x-axis

Walking



Running



By the above figures we can easily distinguish between the activities(walking/running) so this data can be used to train LSTM model

**Data preprocessing**

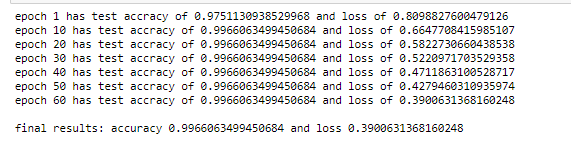
* LSTM model expects fixed length sequences of data so each generated sequence contains 200 training examples
* The new shape of after transformation using stats.mode is (4420, 6, 200) which is far more reduced from original shape (88588, 6)
* This transformed shape is again reshaped into sequence of 200 rows the new shape is (4420, 200, 6)
* Applied 1 hot encoding for labels using pd.get\_dummies function and the data was split into training and testing using test\_train\_split (20%test)

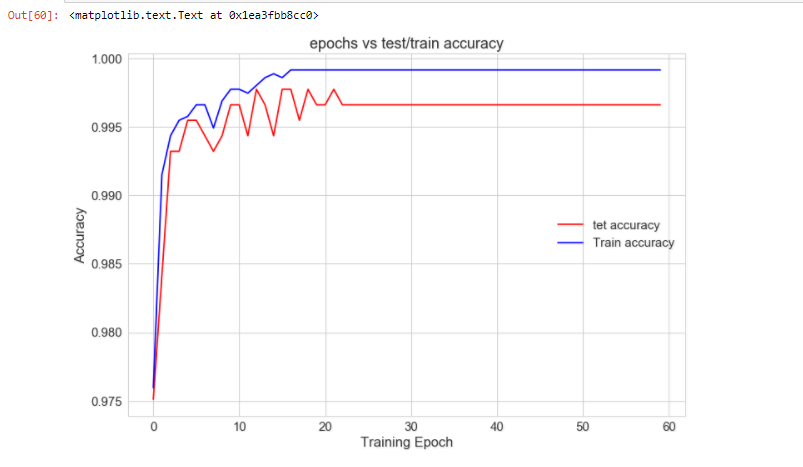
**Modelling**

* This model contains 2 fully-connected and 2 LSTM layers (stacked on each other) with 64 units each. This was modelled in python using tensorflow (3)
* Placeholder were created A placeholder is simply a variable that we will assign data to at a later date. It allows us to create our operations and build our computation graph, without needing the data. In TensorFlow terminology, we then feed data into the graph through these placeholders.(2)
* Tenor was named by which I will obtain predictions and l2 regularization is used which can be noted in loss
* LEARNING\_RATE = 0.0025 and adamoptimiser and relu was used
* trained the model for 60 epochs and keep track of accuracy and error

**Evaluation**

The final test accuracy is 0.996 and loss is 0.39

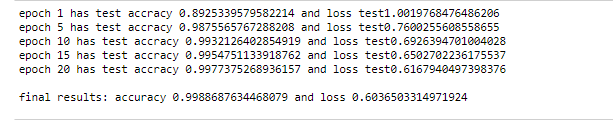


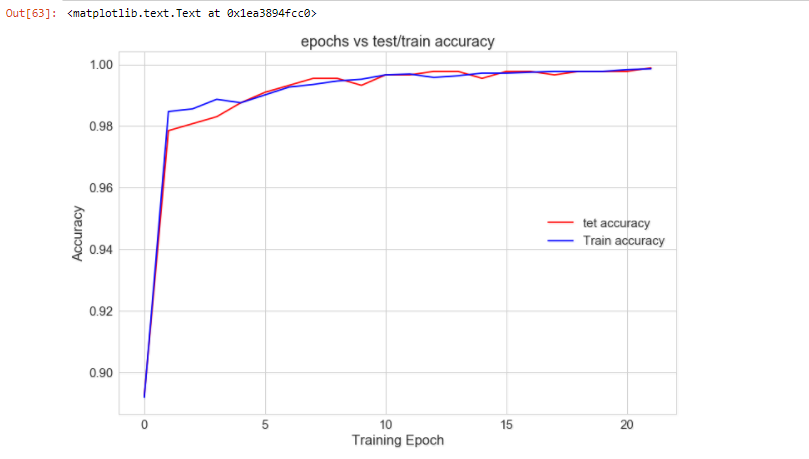


* As seen I the above graph as the epoch increases the accuracy increases and decreases but the test epoch at 22 gives max test accuracy and then model overfits and gives constant accuracy with increasing epoch

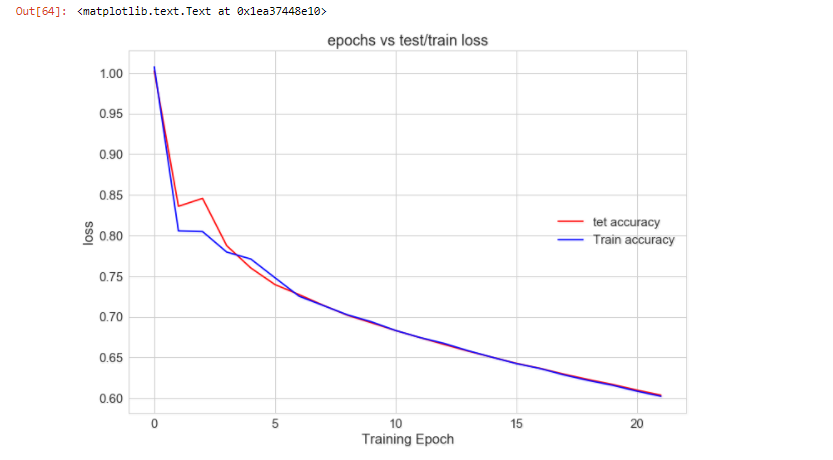
**Self evaluation**

Using epohs ==22 we get more accuracy(test accuracy==99.88%)

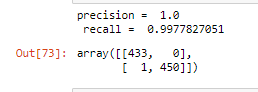




The test accuracy is max at 22nd epoch



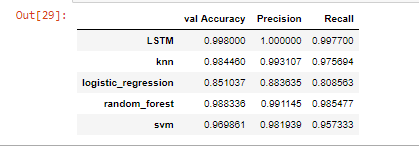
* The loss gradually decreases as number of epoch increases
* Precision, recall and Confusion matrix



* Which gives one only one misclassification and max precision-1.0

**Conclusion**

* The LSTM model(Test accuracy==99.988%) will be my final model with epoch=22. because the test accuracy is more compared to all other models



**References**

1. <https://medium.com/@aneesha/svm-parameter-tuning-in-scikit-learn-using-gridsearchcv-2413c02125a0>
2. <https://learningtensorflow.com/lesson4/>
3. <http://karpathy.github.io/2015/05/21/rnn-effectiveness/>
4. <https://pdfs.semanticscholar.org/f3c8/e586cee14ff1ee33b33d8ca7604d2f5ca31a.pdf>