**Equations**

In this section we introduce formulation of algorithms and loss functions used. Most of the parameters for every algorithm have been evaluated in this study.

**A. PCA**: Principle component analysis is used to convert high dimensional data to low dimensional data

Principal Component Analysis Standardization

**B. Support Vector Machine:** SVM is a supervised machine learning model based on classification algorithm for two groups. It tries to find hyper-plane in N-dimensional space (N – number of features)

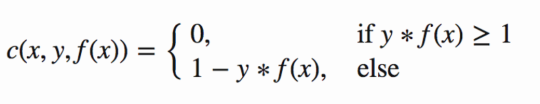
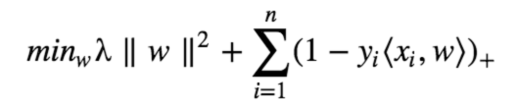


Figure 1: hinge loss for SVM

We also add regularization parameter the cost function. The objective of the regularization parameter is to balance the margin maximization and loss. After adding regularization term, cost function likes this-



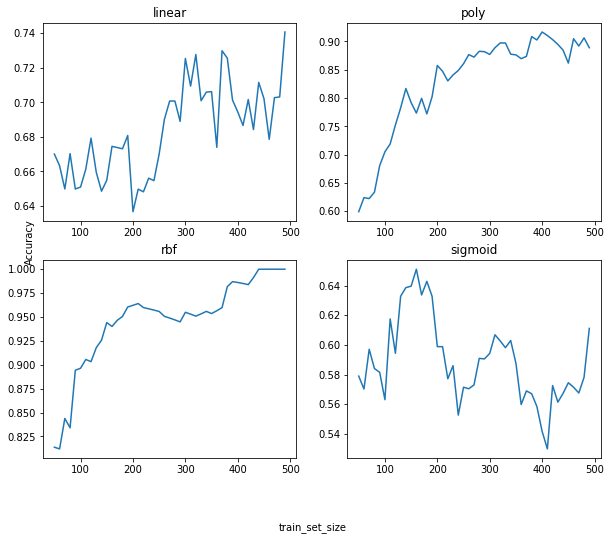
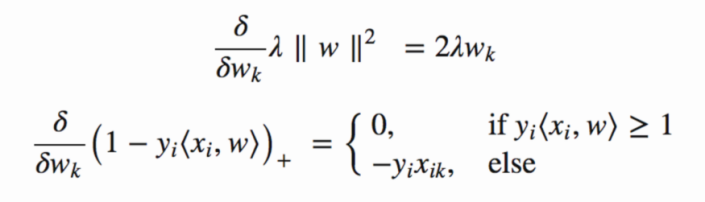
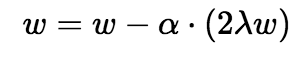
****

Figure 2: validation accuracy of SVM model with different kernel parameters . x-axis: samples in trainng data, y-axis: accuracy

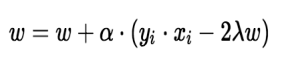
Figure 3 hinge loss with weight regularization for k-nearest number.



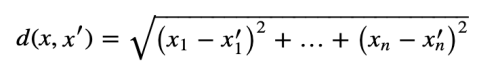
Loss minimisation:

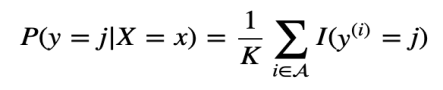


Correct classification

Gradient update:

wrong classification

 **C. K-Nearest Neighbours:** KNN is a distance based algorithm and used make classification based on mean of



nearest neighbours. Distance metric used for this study is Euclidian distance.

**Experiments**

Principal Component Analysis feature vectorIn this section, we introduce experiment steps done and results of every experiment. As our data contains information from 8 accelerometers and reading of each sensor was taken for some fixed amount of time, we choose 8868 units of time reading.

1. **PCA analysis**1-

Component PCA analysis shows that sensors data has a huge overlap i.e. not all algorithms will converge over this data.

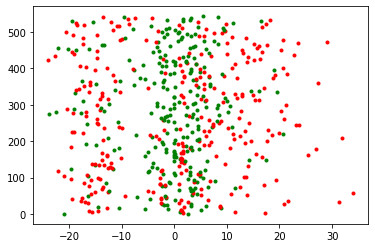


Figure 4: 1-Component PCA, x-axis: pca value,

y-axis: data sample number

1. **Support Vector Machine ( SVM )**

Different kernel for SVM is tested and most promising results came up with “rbf” kernel i.e. ~99.9% with execution time of **2ms**, as the size of training data set increases SVM(‘rbf’) becomes more accurate. These are the stats for these kernels:

Table 1: SVM kernel vs Accuracy

|  |  |
| --- | --- |
| Kernel | Accuracy |
| Linear | 74.07% |
| Poly | 91.44% |
| rbf | ~ 99.9% |
| sigmoid | 65.01% |

1. **K-Nearest Neighbours ( KNN )**

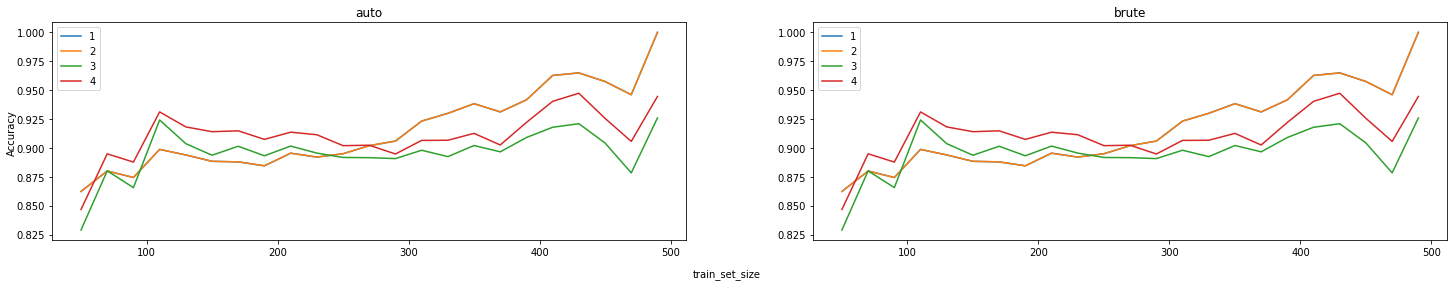
****KNN found second most promising algorithms in this study and we found its accuracy around 95.94% for 1 and 2 nearest neighbours. Accuracy of KNN decreased gradually after 2 nearest neighbours.

Figure 5 k-NN accuracy chart for validation data, x-axis: sample in training data, y-axis: accuracy. Plot for N nearest neighbours

In this experiment we found that accuracy for 1, 2 NN is exactly same. In future work we can take advantage of this fact.

Table: k-NN accuracy, execution time vs N neighbours 2

|  |  |  |
| --- | --- | --- |
| Nearest neighbours | Accuracy | Execution time |
| 1-NN | ~99.8% | 8ms |
| 2-NN | ~99.8% | 8ms |
| 3-NN | 92.59 | 10ms |
| 4-NN | 94.73 | 10ms |