**ISL Project**

**Human Activity Recognition With Smart Phones**

Vineeth Reddy Kottam

ID: 16263561

**Data set description:**

A group of 30 volunteers with ages ranging from 19 to 48 years were selected for this task. Each person was instructed to follow a protocol of activities while wearing a waist-mounted Samsung Galaxy S II smartphone. The six selected ADL were standing, sitting, laying down, walking, walking downstairs and upstairs. Each subject performed the protocol twice: on the first trial the smartphone was fixed on the left side of the belt and on the second it was placed by the user himself as preferred. There is also a separation of 5 seconds between each task where individuals are told to rest, this facilitated repeatability (every activity is at least tried twice) and ground trough generation through the visual interface. The tasks were performed in laboratory conditions, but volunteers were asked to perform freely the sequence of activities for a more naturalistic dataset.

**Approaches tried:**

* Used logistic regression but the logic didn't work out well as our problem is a multiclass classification problem.

**Final Approach:**

* Random Forrest is a ensemble model. It takes multiple trees together to build final model

**Technical code:**

The R code and results can be found below:

load the train data and test data

> train<-read.csv("C:/Users/sunny/OneDrive/Desktop/train.csv/train.csv",header = T, sep=",")

> test<-read.csv("C:/Users/sunny/OneDrive/Desktop/test.csv/test.csv",header = T, sep=",")

split the test into test and validation

> nrow(train)

[1] 7352

> nrow(test)

[1] 2947

> rows=seq(1,nrow(test),1)

> set.seed(22222)

> partition = sample(rows,0.66\*nrow(test))

> testdata = test[partition,]

> valdata = test[-partition,]

>

>

> nrow(testdata)

[1] 1945

> nrow(valdata)

[1] 1002

checking missing values in the data set

> sum(is.na(train))

[1] 0

> sum(is.na(testdata))

[1] 0

> sum(is.na(valdata))

[1] 0

Check For class imbalance in Target Variable(Activity)

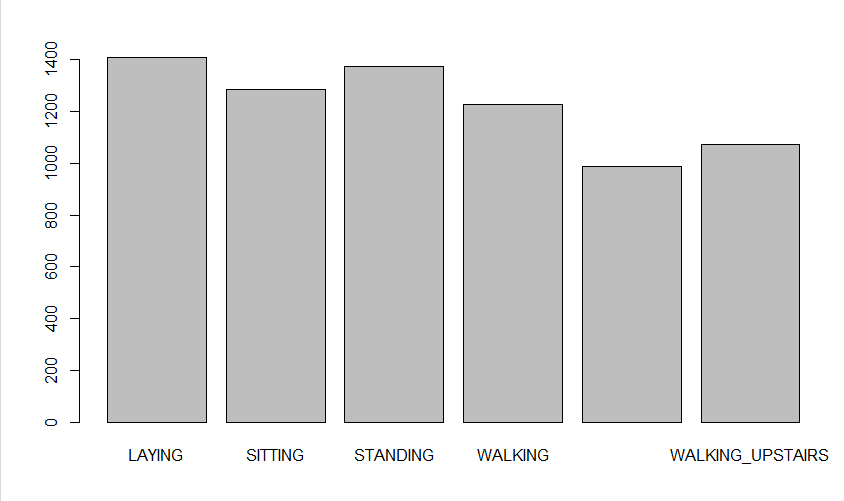
> ClassImbalance<-table(train$Activity)

> barplot(ClassImbalance)

> (tabulate(train$Activity)/nrow(train))\*100

[1] 19.13765 17.49184 18.68879 16.67573 13.41132 14.59467

>



check the structure of the train data and test data and validation data

> dim(train) ## 7352 563

[1] 7352 563

> dim(testdata) ## 1945 563

[1] 1945 563

> dim(valdata) ## 1002 563

[1] 1002 563

remove the subject column

> train<-train[,-562]

> testdata<-testdata[,-562]

> valdata<-valdata[,-562]

remove the activity column

> train\_without\_label<-train[,-562]

> testdata\_without\_label<-testdata[,-562]

> valdata\_without\_label<-valdata[,-562]

**Principal component analysis** (**PCA):**

The main idea of **principal component analysis** (**PCA**) is to reduce the dimensionality of a data set consisting of many variables correlated with each other, either heavily or lightly, while retaining the variation present in the dataset, up to the maximum extent. ... As a layman, it is a method of summarizing data.

Lets do the PCA on the train data set

> train<-train[,-562]

> testdata<-testdata[,-562]

> valdata<-valdata[,-562]

>

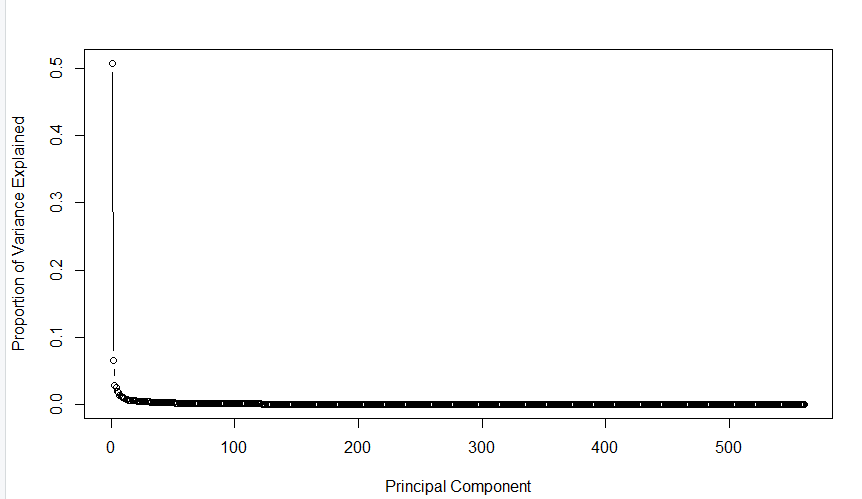
> train\_without\_label<-train[,-562]

> testdata\_without\_label<-testdata[,-562]

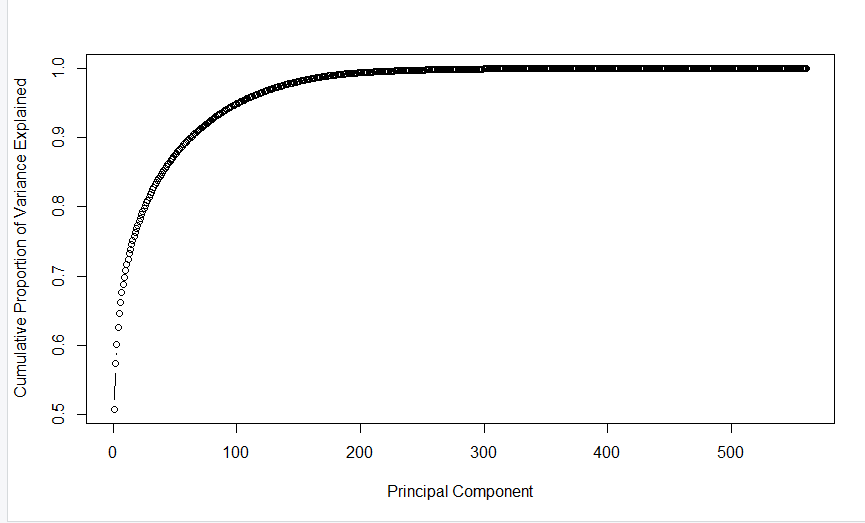
> valdata\_without\_label<-valdata[,-562]

>

> prcomp\_train<-prcomp(train\_without\_label,scale. = T,center = T)



cumulative screen plot



as it is hard to infer the number of principal components from graph do a

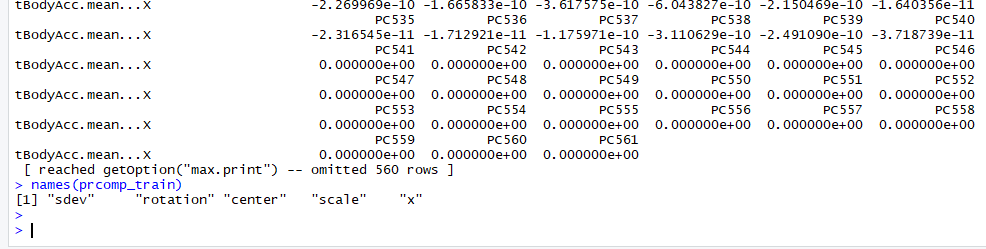
"sum of variance explained" as below

sum(prop\_varex[1:179])



by taking 179 principal components I can explain 0.9901927 it means I can explain 99% variation in the data by 179 principal components

observe the output of prcomp as below



get the label back on train

> dim(train\_data\_pca)

[1] 7352 561

>

> train\_data\_pca\_with\_label<-train\_data\_pca[,c(1:179)]

> train\_data\_pca\_with\_label$Activity<-train$Activity

>

> dim(train\_data\_pca\_with\_label)

[1] 7352 180

>

get the validation data and test data

>valdata\_pca<-predict(prcomp\_train, newdata = valdata\_without\_label)

>valdata\_pca <- as.data.frame(valdata\_pca)

#valdata\_pca\_with\_label$Activity<-valdata$Activity

>valdata\_pca\_with\_label<-valdata\_pca[,1:179]

>valdata\_pca\_with\_label$Activity<-valdata$Activity

>testdata\_pca<-predict(prcomp\_train,newdata=testdata\_without\_label)

>testdata\_pca<-as.data.frame(testdata\_pca)

#testdata\_pca\_with\_label$Activity<-testdata$Activity

>testdata\_pca\_with\_label<-testdata\_pca[,1:179]

>testdata\_pca\_with\_label$Activity<-testdata$Activity

>str(testdata\_pca\_with\_label) ### TEST DATA AFTER PCA

>head(testdata\_pca\_with\_label)

>dim(testdata\_pca\_with\_label)

>str(valdata\_pca\_with\_label) ### VALIDATION DATA AFTER PCA

>head(valdata\_pca\_with\_label)

>dim(valdata\_pca\_with\_label)

>dim(train\_data\_pca\_with\_label) ### TRAIN DATA AFTER PCA

>head(train\_data\_pca\_with\_label)

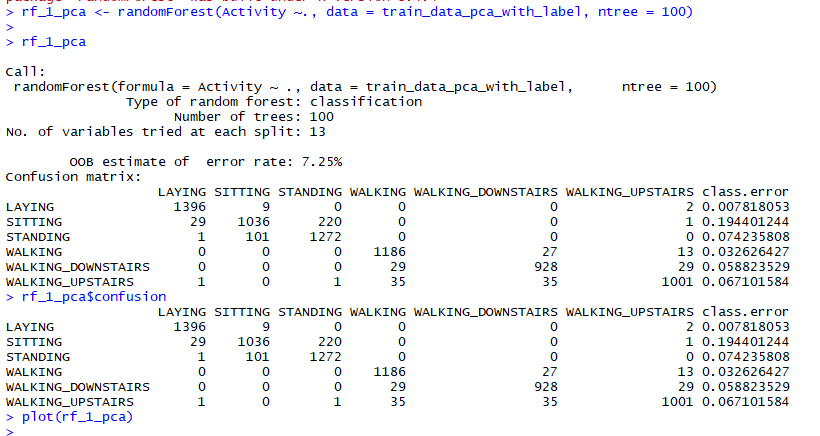
Now that the compressed data is ready. let build a randome forest on the compressed data

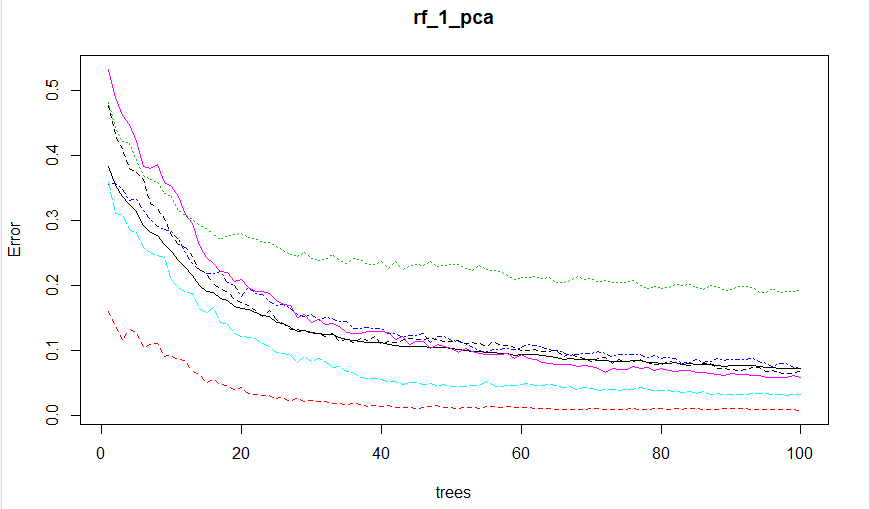
ROUND -- 1

> rf\_1\_pca <- randomForest(Activity ~., data = train\_data\_pca\_with\_label, ntree = 100)

>

> rf\_1\_pca





## round 1 metrics for train and validation

> sum(rf\_1\_pca$confusion)

[1] 7352.455

> accuracy\_train\_round1\_pca <- sum(diag(rf\_1\_pca$confusion))/sum(rf\_1\_pca$confusion)

> accuracy\_train\_round1\_pca

[1] 0.9243172

round 1 validation

> val\_matrix\_round1\_pca<-table(predict(rf\_1\_pca,valdata\_pca\_with\_label), valdata\_pca\_with\_label$Activity)

> accuracy\_val\_round1\_pca <- sum(diag(val\_matrix\_round1\_pca))/sum(val\_matrix\_round1\_pca)

> accuracy\_val\_round1\_pca

[1] 0.8772455

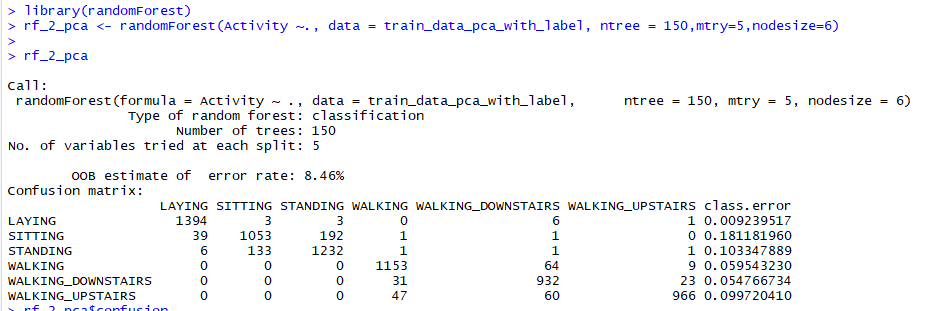
Round 2

> library(randomForest)

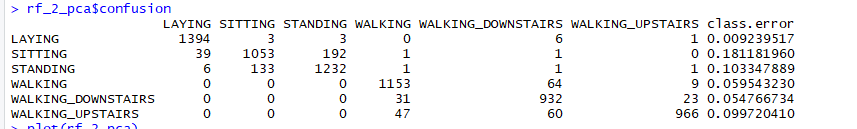
> rf\_2\_pca <- randomForest(Activity ~., data = train\_data\_pca\_with\_label, ntree = 150,mtry=5,nodesize=6)

>

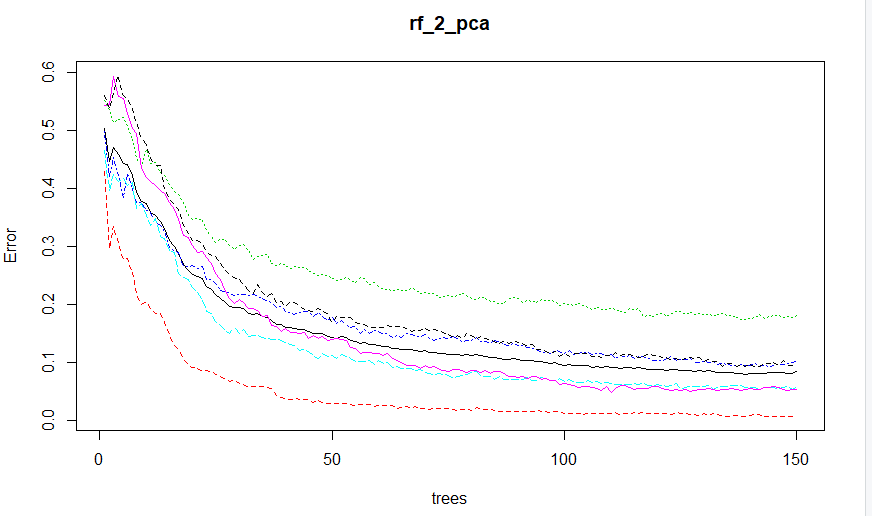
> rf\_2\_pca



> rf\_2\_pca$confusion



> plot(rf\_2\_pca)

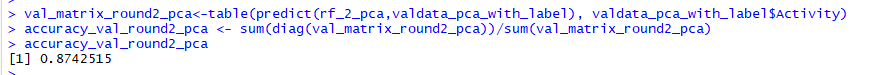


> val\_matrix\_round2\_pca<-table(predict(rf\_2\_pca,valdata\_pca\_with\_label), valdata\_pca\_with\_label$Activity)

> accuracy\_val\_round2\_pca <- sum(diag(val\_matrix\_round2\_pca))/sum(val\_matrix\_round2\_pca)

> accuracy\_val\_round2\_pca

[1] 0.8742515



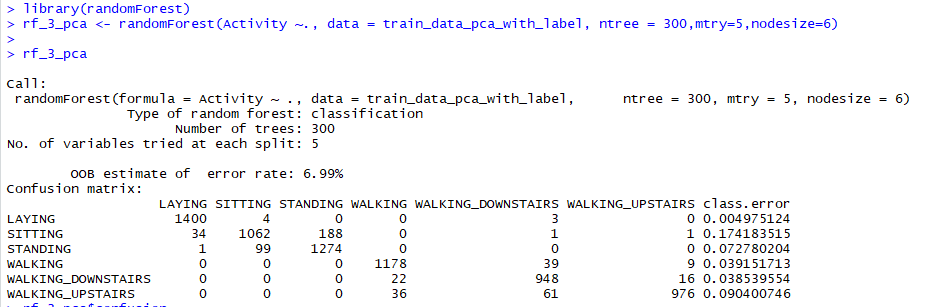
Round 3

> library(randomForest)

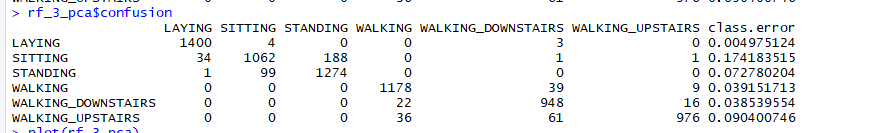
> rf\_3\_pca <- randomForest(Activity ~., data = train\_data\_pca\_with\_label, ntree = 300,mtry=5,nodesize=6)

>

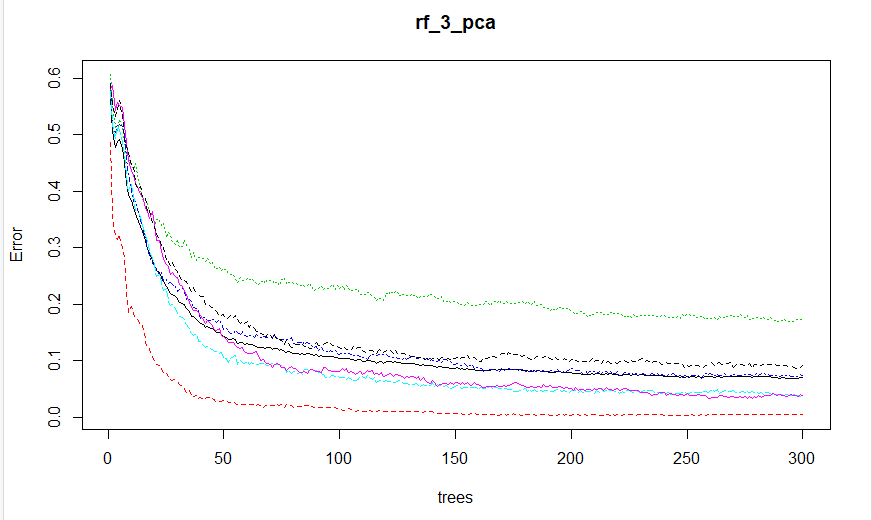
> rf\_3\_pca



> rf\_3\_pca$confusion



> plot(rf\_3\_pca)

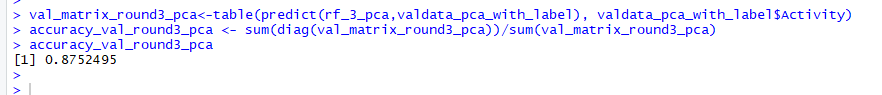


> val\_matrix\_round3\_pca<-table(predict(rf\_3\_pca,valdata\_pca\_with\_label), valdata\_pca\_with\_label$Activity)

> accuracy\_val\_round3\_pca <- sum(diag(val\_matrix\_round3\_pca))/sum(val\_matrix\_round3\_pca)

> accuracy\_val\_round3\_pca

[1] 0.8752495



> val\_matrix\_round3\_pca<-table(predict(rf\_3\_pca,valdata\_pca\_with\_label), valdata\_pca\_with\_label$Activity)

> accuracy\_val\_round3\_pca <- sum(diag(val\_matrix\_round3\_pca))/sum(val\_matrix\_round3\_pca)

> accuracy\_val\_round3\_pca

[1] 0.8752495

>

> test\_matrix\_round3\_pca<-table(predict(rf\_3\_pca,testdata\_pca\_with\_label), testdata\_pca\_with\_label$Activity)

> accuracy\_test\_round3\_pca <- sum(diag(test\_matrix\_round3\_pca))/sum(test\_matrix\_round3\_pca)

> accuracy\_test\_round3\_pca

[1] 0.9017995

>