SAN JOSÉ STATE UNIVERSITY

DEPARTMENT OF ELECTRICAL ENGINEERING

EE259 – AN INTRODUCTION TO STATISTICAL LEARNING

PROFESSOR: Dr. BIRSEN SIRKECI

SPRING 2018

PROJECT REPORT

WALL-FOLLOWING ROBOT NAVIGATION DATA

Praneeth Varma Alluri - 012406277
Yogeswara Sarat Bankapalli - 012431692

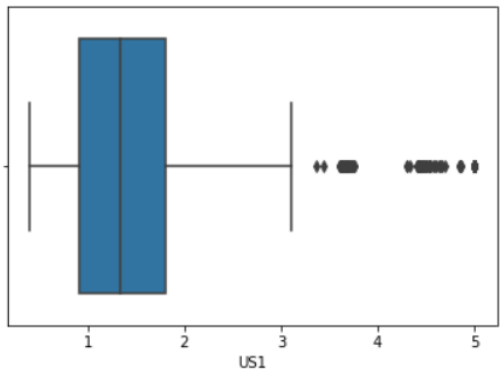
DATA SET DESCRIPTION:

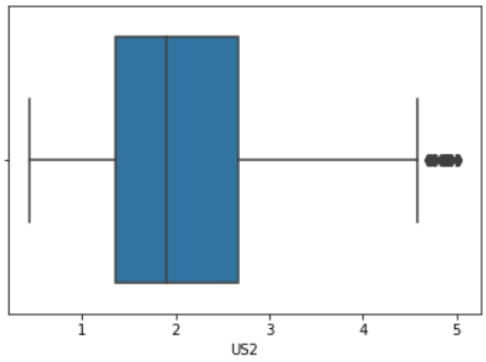
The data set comprises of 24 input variables and one output variable. The output variable is a categorical variable with four different categories which give the direction of navigation of the wall-following robot. The output variable categories are Move-Forward, Slight-Right-Turn, Sharp-Right-Turn, Slight-Left-Turn. The 24 input variables are the raw measurements of the ultrasound sensors with different reference angles. The reference angles range from -180 A^o to +180 A^o. The input variables are thus named as US1 (ultrasound sensor 1), US2 (ultrasound sensor 2) and so on till US24 (ultrasound sensor 24) All the input variables are of integer data type. The input variable data is collected by recording all the ultra sound sensor data while the robot is navigating and is in turn used to determine the direction of navigation. The dataset is free from missing values and have 5456 datapoints in total. The underlying problem in the dataset is a classification problem. The dataset has few considerable issues. It has few outliers and the scale of each input variable is different from that of others. Feature scaling technique can be used in the data cleaning phone to effectively deal with this issue.

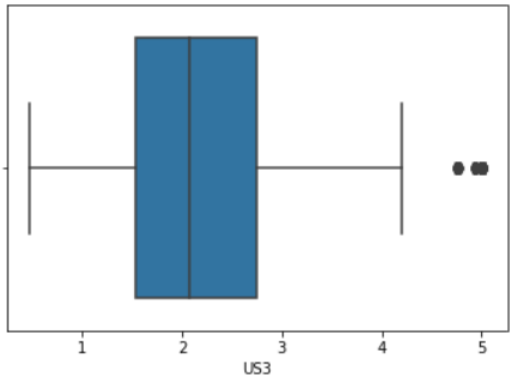
DATA SET VISUALIZATION:

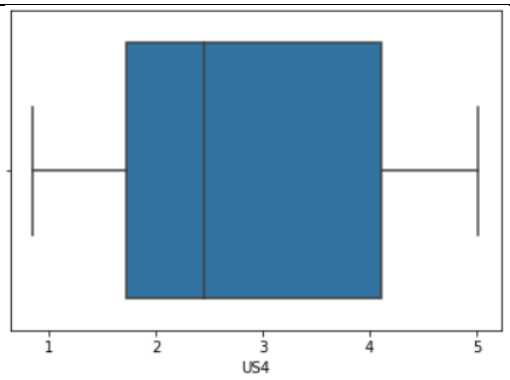
Firstly, since there is no separate test data to this data set, the available data set needs to be divided into training and test data. The training data is then used for model development and test data is used for verifying the accuracy of the model. The data is divided in such a way that 85 percent of the data points fall in training data and remaining fall in test dataset. Train_test_split package has been imported from the model_selection package to perform the split. The resulting training dataset is named as dataset train.

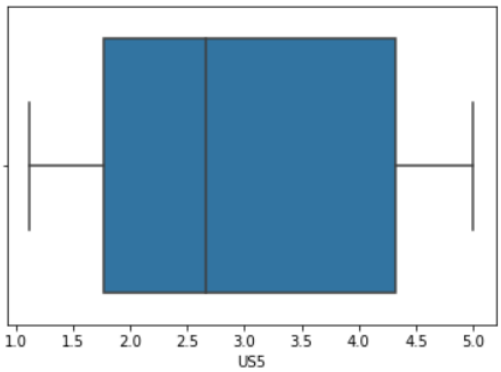
Data set visualization is a very important step in machine learning which can be used to effectively understand the data deeply. Boxplot is one of the visualization techniques which enables us to visually understand how the values in the feature are distributed. As the name suggests the box plot gives a box like plot with the end points of the box ranging from the 25^{th} percentile to 75^{th} percentile value. Box plot helps us determine if the feature has outlier values or not. The features of the dataset are plotted using boxplot to determine if the dataset contains outliers.

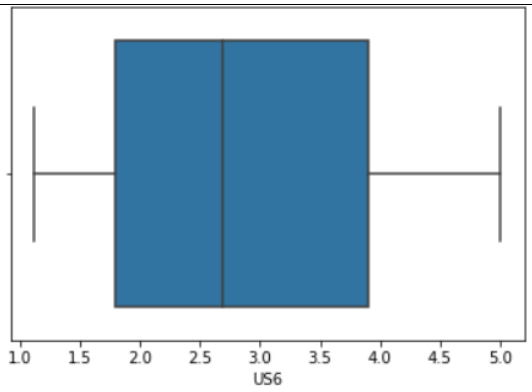

1
2
3
4
5
US1

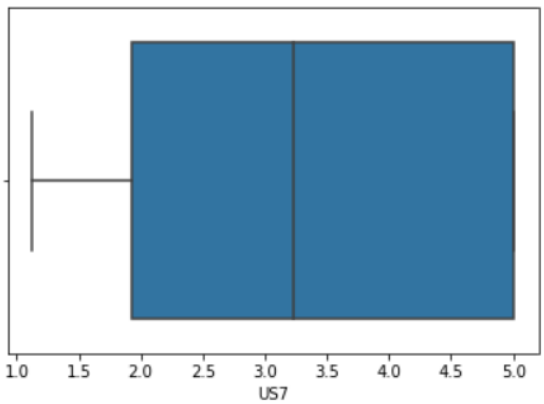

1
2
3
4
5
US2

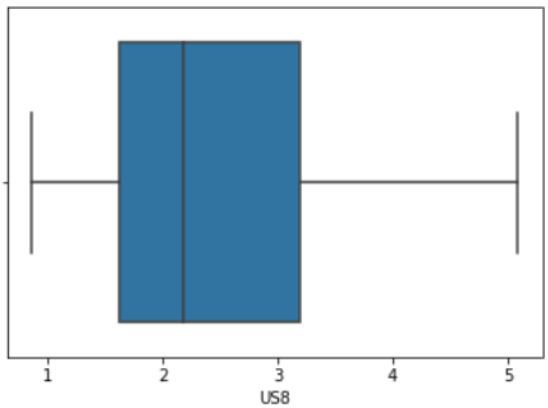

1
2
3
4
5
US3

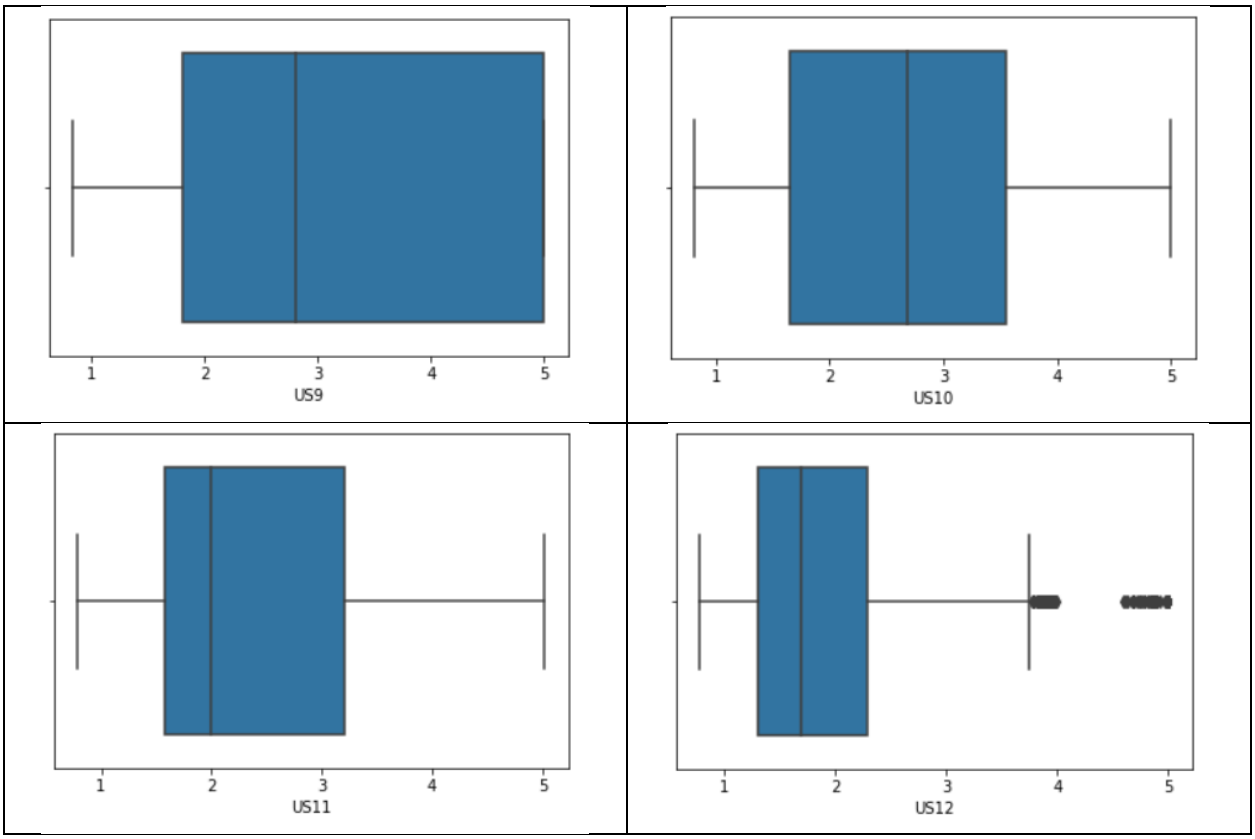

1
2
3
4
5
US4

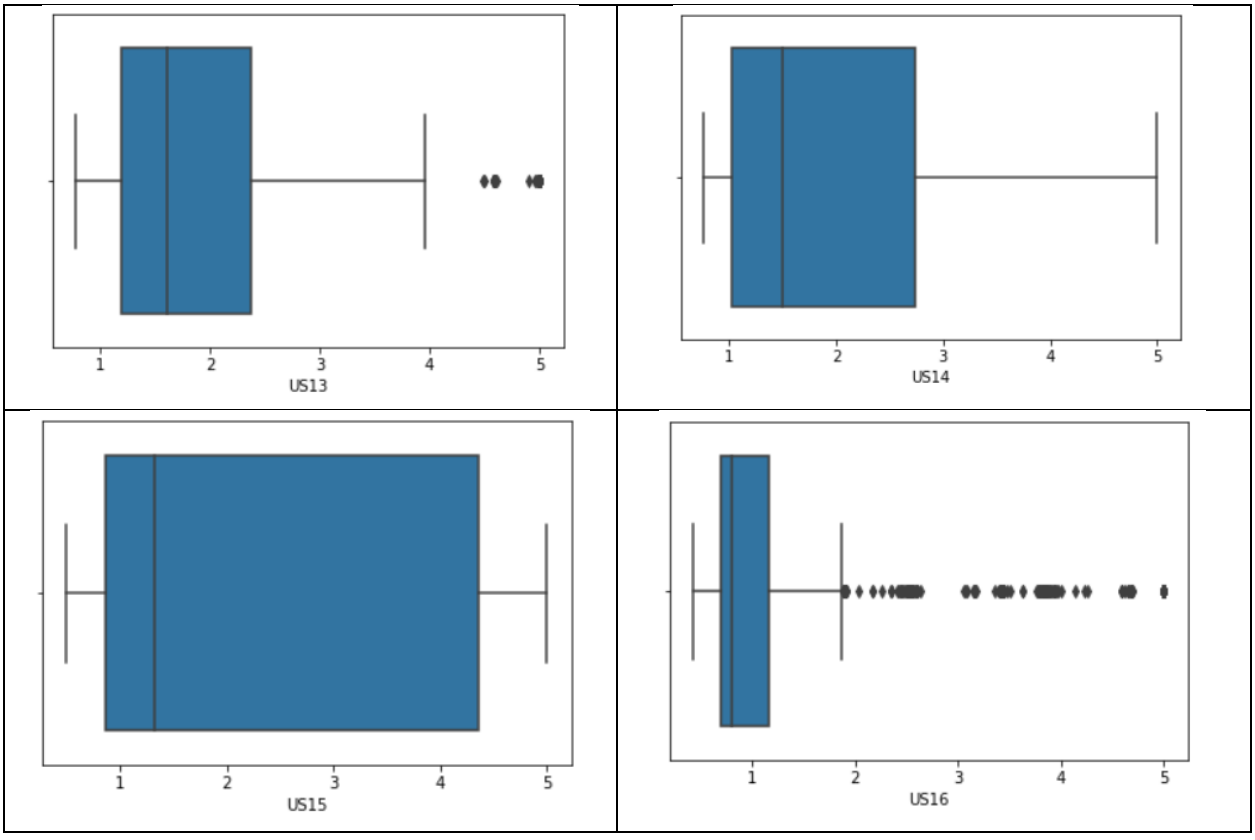

1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
US5


1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
US6


1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
US7


1
2
3
4
5
US8


1
2
3
4
5
US9
US10
US11
US12


1
2
3
4
5
US13
US14
US15
US16

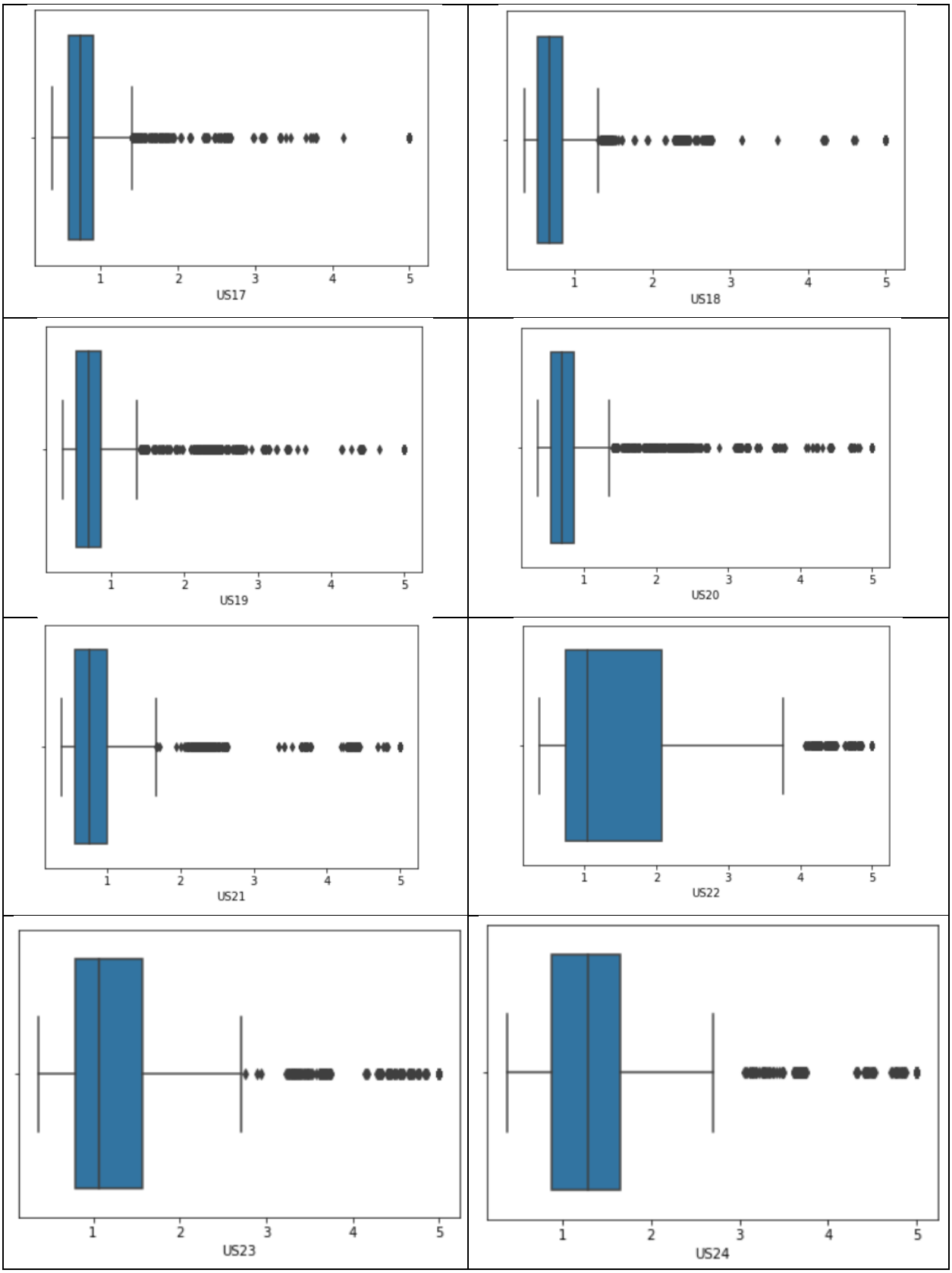
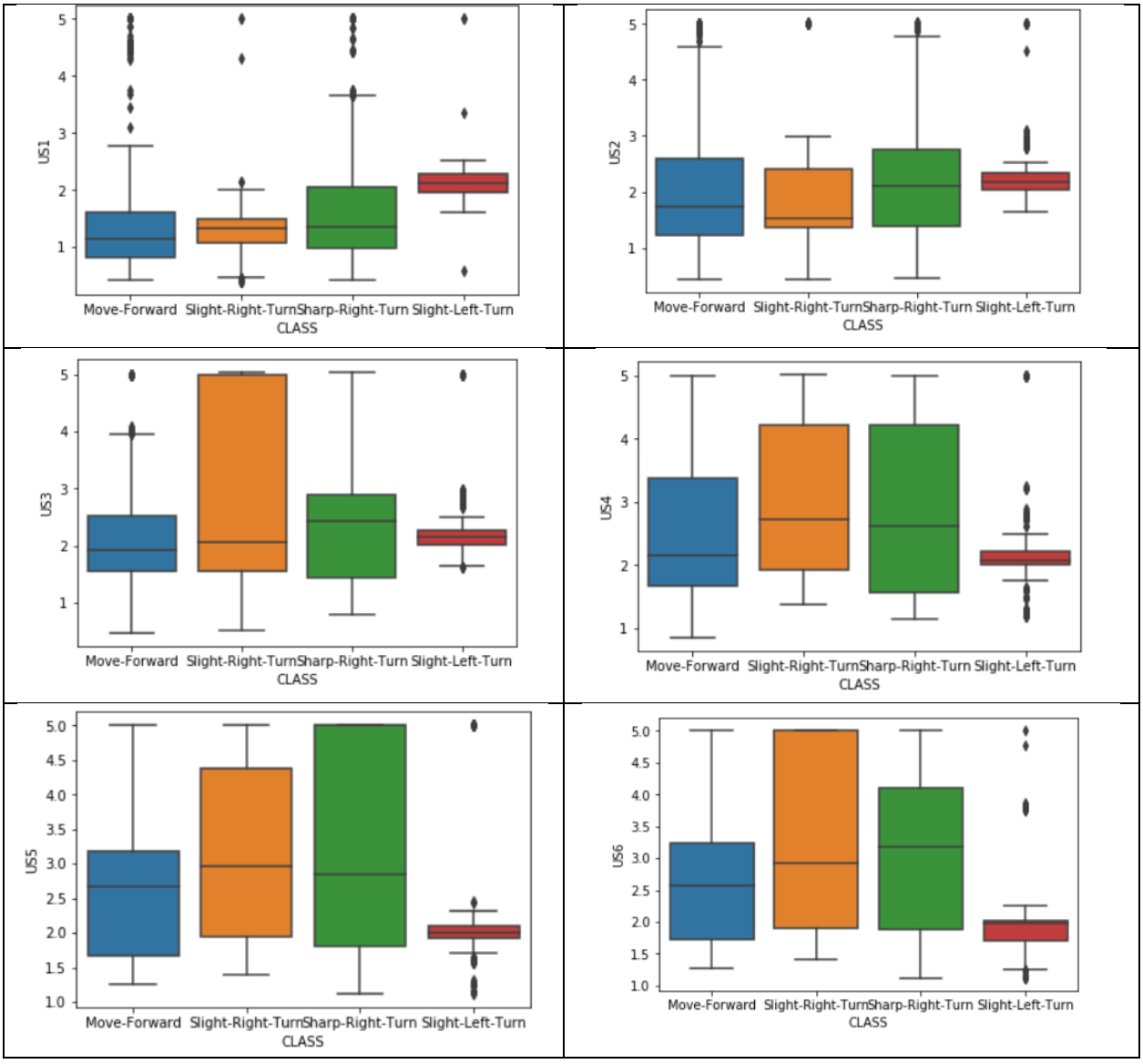
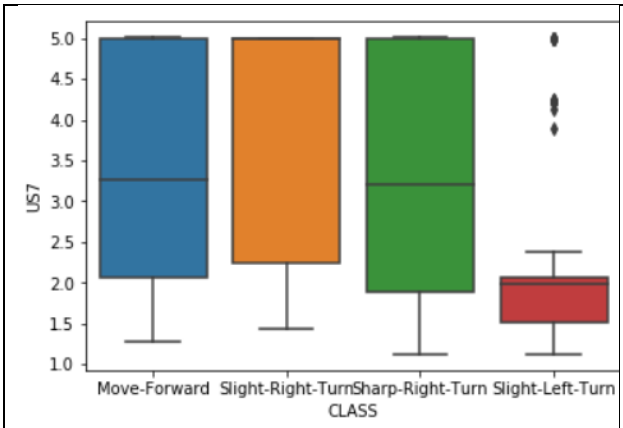
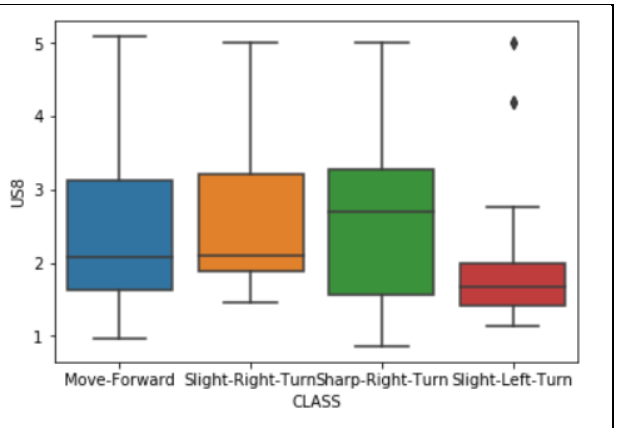


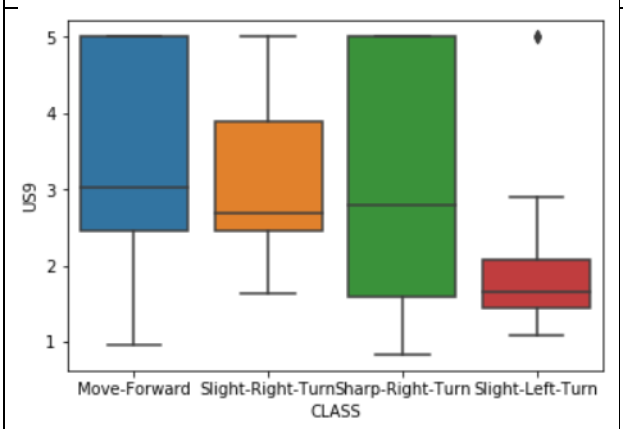
Table: 1 Box plots of all input features

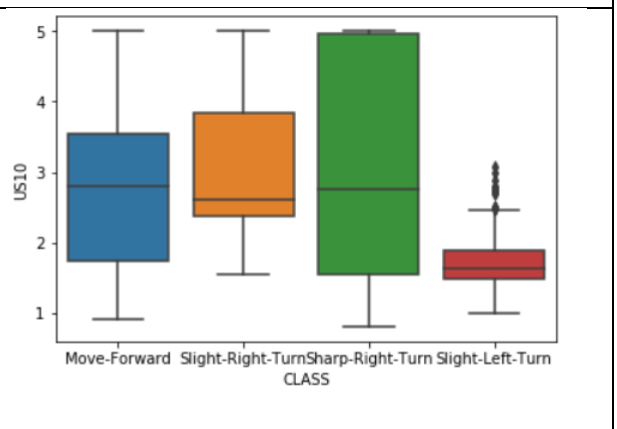
Table 1 provides the boxplots obtained for all input variables. We can observe that some box plots have a few points lying outside the whisker. These points are considered as outliers. By observing the above table, we can conclude that features US1, US2, US3, US12, US13, US16, US17, US18, US19, US20, US21, US22, US23, US24 have outliers in them.

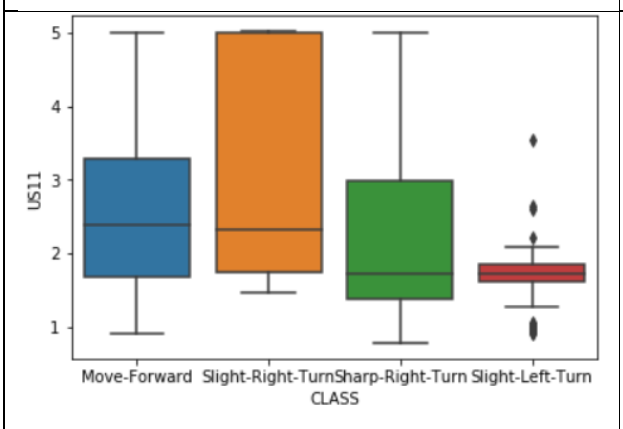


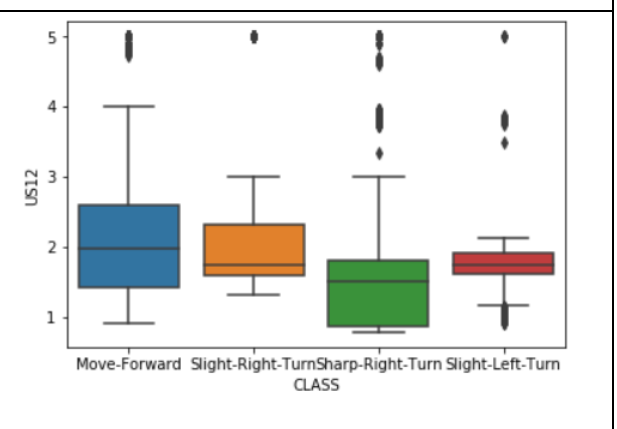

US7
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
Move-Forward
Slight-Right-Turn
Sharp-Right-Turn
Slight-Left-Turn
CLASS

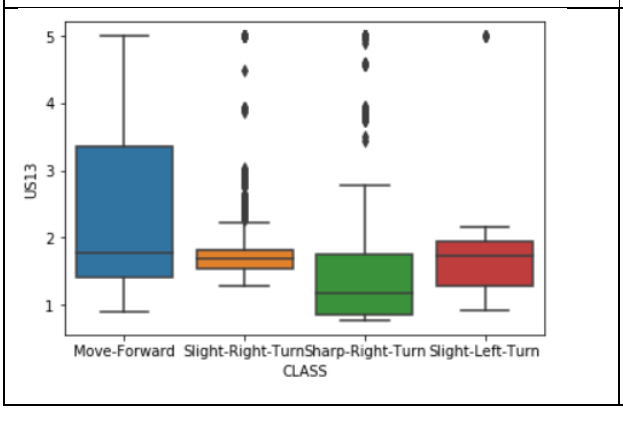

US8
5
4
3
2
1
Move-Forward
Slight-Right-Turn
Sharp-Right-Turn
Slight-Left-Turn
CLASS

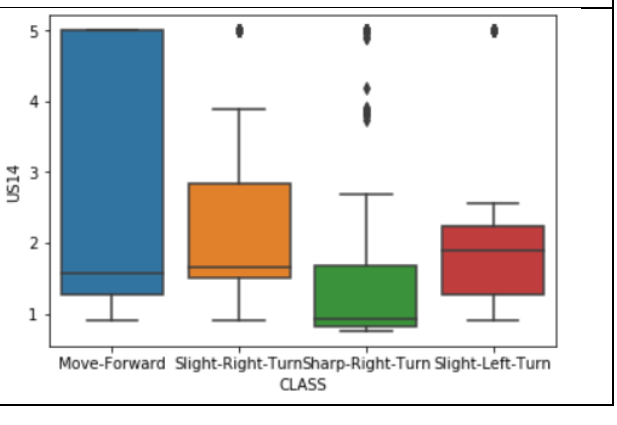

US9
5
4
3
2
1
Move-Forward
Slight-Right-Turn
Sharp-Right-Turn
Slight-Left-Turn
CLASS

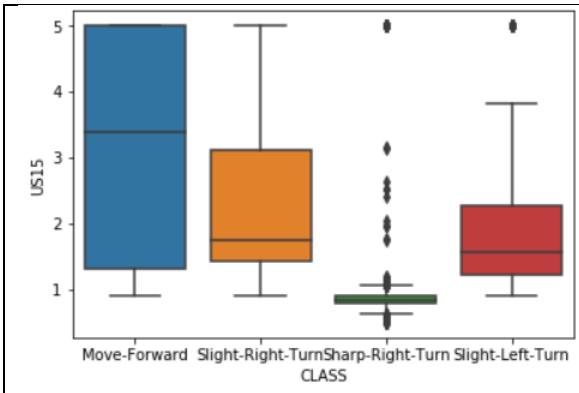

US10
5
4
3
2
1
Move-Forward
Slight-Right-Turn
Sharp-Right-Turn
Slight-Left-Turn
CLASS

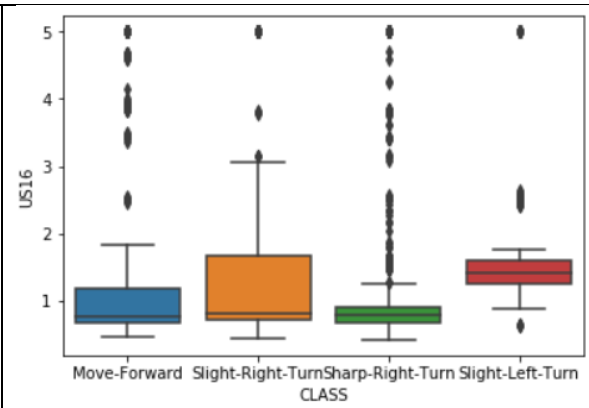

US11
5
4
3
2
1
Move-Forward
Slight-Right-Turn
Sharp-Right-Turn
Slight-Left-Turn
CLASS

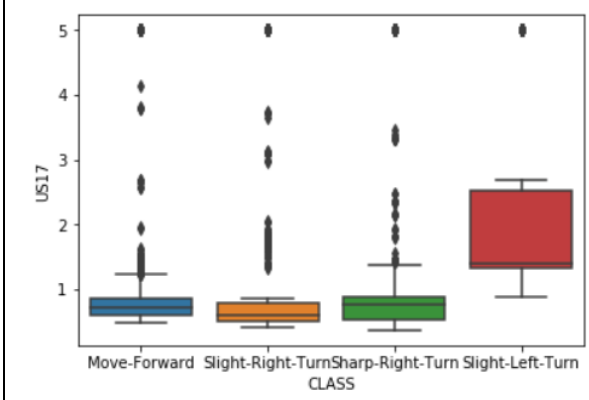

US12
5
4
3
2
1
Move-Forward
Slight-Right-Turn
Sharp-Right-Turn
Slight-Left-Turn
CLASS

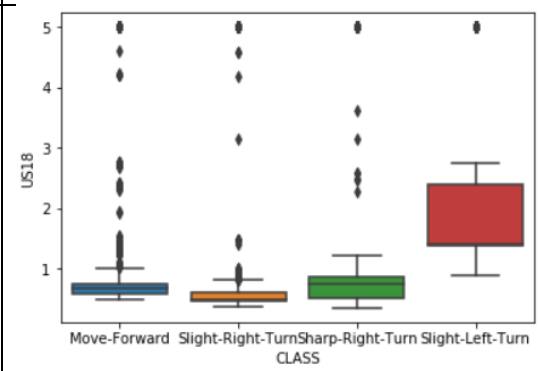

US13
5
4
3
2
1
Move-Forward
Slight-Right-Turn
Sharp-Right-Turn
Slight-Left-Turn
CLASS

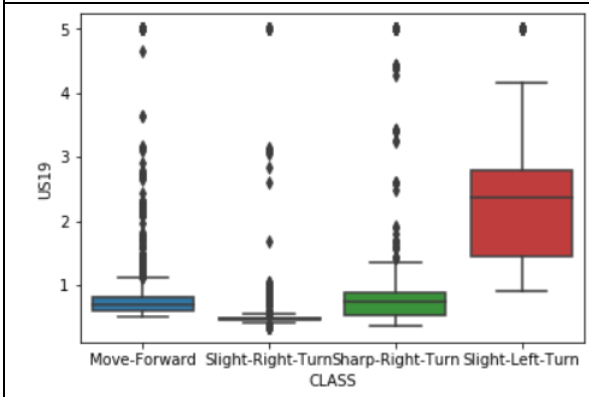

US14
5
4
3
2
1
Move-Forward
Slight-Right-Turn
Sharp-Right-Turn
Slight-Left-Turn
CLASS

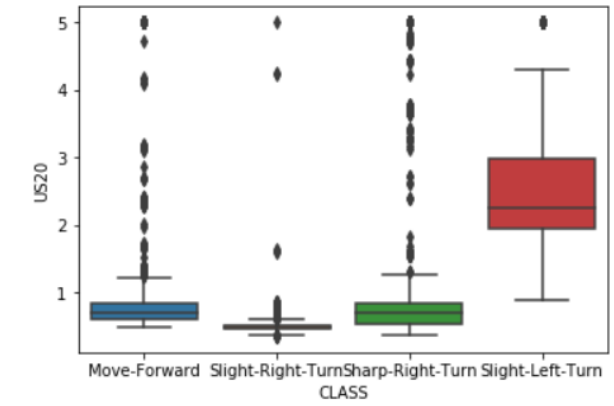

US15
5
4
3
2
1
Move-Forward
Slight-Right-Turn
Sharp-Right-Turn
Slight-Left-Turn
CLASS

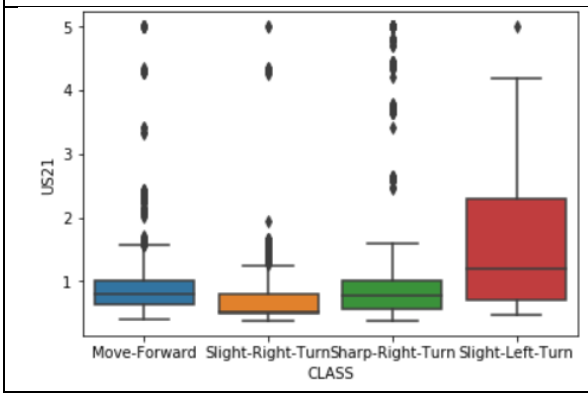

US16
5
4
3
2
1
Move-Forward
Slight-Right-Turn
Sharp-Right-Turn
Slight-Left-Turn
CLASS

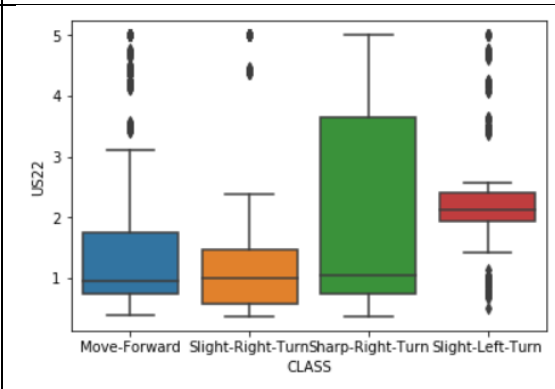

US17
5
4
3
2
1
Move-Forward
Slight-Right-Turn
Sharp-Right-Turn
Slight-Left-Turn
CLASS


US18
5
4
3
2
1
Move-Forward
Slight-Right-Turn
Sharp-Right-Turn
Slight-Left-Turn
CLASS


US19
5
4
3
2
1
Move-Forward
Slight-Right-Turn
Sharp-Right-Turn
Slight-Left-Turn
CLASS


US20
5
4
3
2
1
Move-Forward
Slight-Right-Turn
Sharp-Right-Turn
Slight-Left-Turn
CLASS


US21
5
4
3
2
1
Move-Forward
Slight-Right-Turn
Sharp-Right-Turn
Slight-Left-Turn
CLASS


US22
5
4
3
2
1
Move-Forward
Slight-Right-Turn
Sharp-Right-Turn
Slight-Left-Turn
CLASS

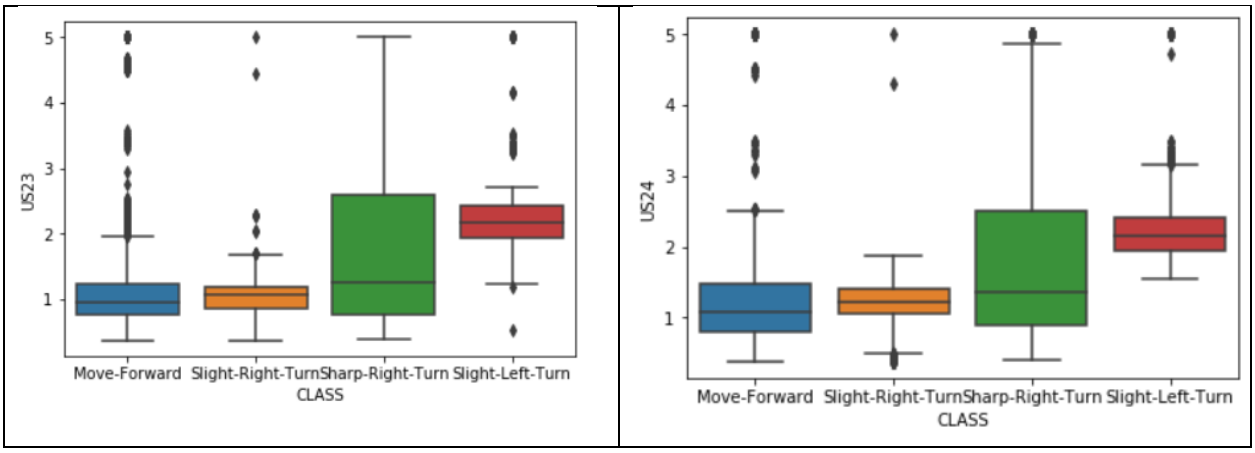


Table 2: Relation between input variables and output

Table 2 provides a set of boxplots which can help determine the relation between various inputs and the output categorical variable. Each plot above has 4 box plots representing the distribution of input variable values of belonging to each output category. Let us consider the plot of input variable US1. The boxplots of each output category are distinct to each other which indicates that this input parameter is helpful in determining the output category. For plots whose box plots are similar, we can conclude that the input parameter need not be considered, and its values are not helpful in determining the output category. This box plot visualization helps in eliminating unimportant feature set thereby reducing the number of features. If THE PLOT OF us17 and US 18 are considered, it is observed that the boxplots of the first three categories of this plot are almost similar. That is, these features are not good enough to determine the output value, hence they can be eliminated from the feature set.

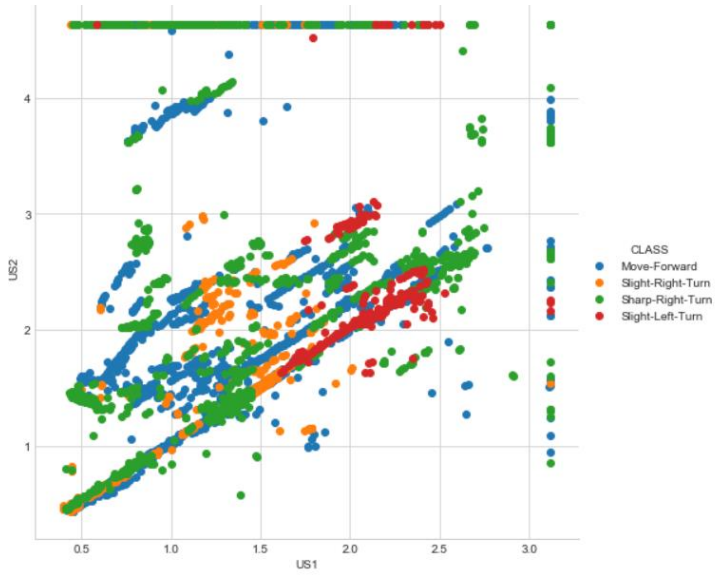


Fig 1: US1 VS US2

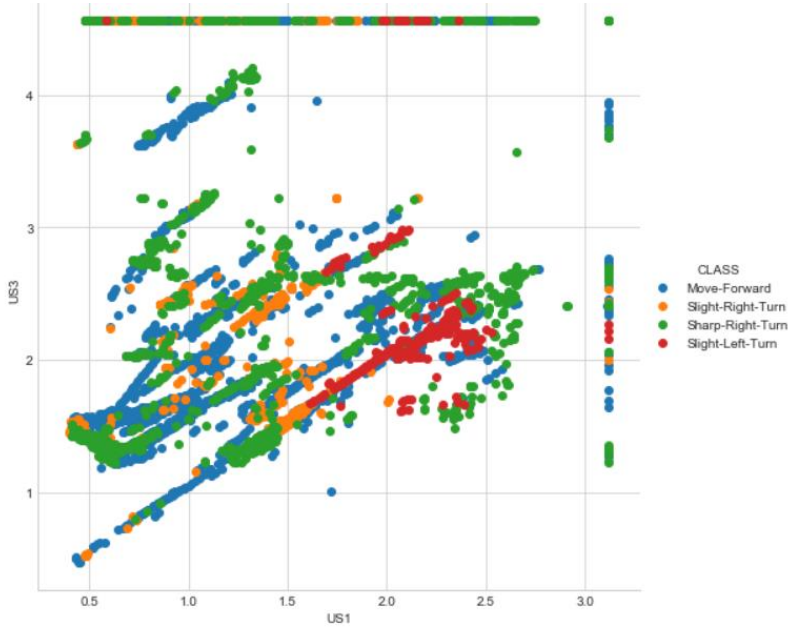


Fig2 : US1 VS US3

The above figures Fig 1 and Fig 2 gives the scatter plot of the datapoints of features US1, US2 and US1, US3. The points are colored based on the output category they belong to. These plots help us determine that the decision boundary. These plots are two of the examples to show the type of relation two features has in this dataset. The relation between any 2 features is highly non-linear.

DATA SET CLEANING:

Data set cleaning is an important phase where we clean the data and make it ready for model creating and fitting. There are many activities involved in data set cleaning. They are:

Handling missing values: The input variables may have missing values among them. It is very important to either eliminate them or replace them with some estimated value. Elimination can be done if the data set is very big and the missing values are very less. The usual procedure followed is replacing the missing values with mean, median or mode of the feature data.

Handling categorical data: The input feature can be a categorical variable. This categorical variable can be converted into numerical data by performing one hot encoding. Most of the machine leaning models work better on numerical data, hence the categorical data needs to be converted for better results.

Handling outliers: Outliers effect the mean and standard deviation of the feature data; hence it is very important to get rid of the any outliers present in the data.

Feature Scaling: Feature scaling is used to change the scales of the input parameters so that all of them have the same scale. The machine learning models tend to work better when feature scaling is applied than when it is not applied. Feature scaling avoids the dominance of feature in greater ranges over the features in smaller ranges.

The dataset considered doesn't contain any missing values or categorical input variables. The outliers can be handles either by eliminating the datapoints which contain outliers or by changing the outlies to some appropriate values. The number of datapoints have drastically reduced to 730 from 4636 when outlier elimination has been implemented. So, the other method of outlier handling is adopted to deal with the outliers. The first quartile (Q1) and third quartile (Q3) values are calculated for every feature. Then Q1- IQR value has been considered as lower limit and Q3 + IQR value has been considered as upper limit of each feature data. The feature data which is lower than lower limit has been changed so that they are equal to lower limit. The feature data which is higher than upper limit has been changed to upper limit. This method effectively removed the outliers in the dataset without reducing the number of datapoints.

	Min	Q1	Q2	Q3	Max
US1	0.4	0.922	1.332	1.8	5.0
US2	0.437	1.363	1.905	2.67025	5.025

US3	0.47	1.54	2.0675	2.749	5.029
US4	0.848	1.73175	2.4565	4.10375	5.017
US5	1.12	1.772	2.6665	4.3185	5.0
US6	1.114	1.789	2.692	3.899	5.005
US7	1.122	1.93275	3.229	5.0	5.008
US8	0.859	1.626	2.1775	3.19525	5.087
US9	0.836	1.803	2.802	5.0	5.0
US10	0.81	1.645	2.6795	3.54675	5.0
US11	1.254375	1.58175	1.992	3.20225	3.529625
US12	0.778	1.30275	1.695	2.286	3.760875
US13	0.77	1.1905	1.61	2.36825	4.134875
US14	0.7559	1.026	1.492	2.7335	3.89449
US15	0.495	0.86	1.3185	4.3635	5.0
US16	0.424	0.69	0.805	1.16525	5.0
US17	0.373	0.58	0.738	0.913	5.0
US18	0.354	0.52875	0.683	0.84	5.0
US19	0.34	0.52275	0.687	0.858	5.0
US20	0.355	0.54	0.691	0.862	5.0
US21	0.38	0.56575	0.7645	1.009	5.0
US22	0.37	0.74275	1.035	2.06925	5.0
US23	0.367	0.79375	1.073	1.56825	5.0
US24	0.377	0.88375	1.287	1.657	5.0

Table 3: minimum, Q1, Q2, Q3, maximum values of input variables before handling outliers

	Minimum	Q1	Q2	Q3	Maximum
US1	0.4	0.922	1.332	1.8	3.117
US2	0.437	1.363	1.905	2.67025	4.631125
US3	0.47	1.54	2.0675	2.749	4.5625
US4	0.848	1.73175	2.4565	4.10375	5.017
US5	1.112	1.772	2.6665	4.3185	5.0
US6	1.114	1.789	2.692	3.899	5.005
US7	1.122	1.93275	3.229	5.0	5.008
US8	0.859	1.626	2.1775	3.19525	5.087
US9	0.836	1.803	2.802	5.0	5.0
US10	0.81	1.645	2.6795	3.54675	5.0
US11	1.254375	1.58175	1.992	3.20225	3.529625
US12	0.778	1.30275	1.695	2.286	3.760875
US13	0.77	1.1905	1.61	2.36825	4.134875

US14	0.7559	1.026	1.492	2.7335	3.89449
US15	0.495	0.86	1.3185	4.3635	5.0
US16	0.424	0.69	0.805	1.16525	1.876625
US17	0.373	0.58	0.738	0.913	1.4125
US18	0.354	0.52875	0.683	0.84	1.306875
US19	0.34	0.52275	0.687	0.858	1.360875
US20	0.355	0.54	0.691	0.862	1.345
US21	0.38	0.56575	0.7645	1.009	1.673875
US22	0.37	0.74275	1.035	2.06925	4.059
US23	0.367	0.79375	1.073	1.56825	2.73
US24	0.377	0.88375	1.287	1.657	2.816875

Table 4: Minimum, Q1, Q2, Q3, Maximum values of input variables after handling outliers

Table 3 and 4 give a brief description of input variables before and after handling the outliers. It is observed that the Q1, Q2, Q3 values remain same in both cases and the minimum and maximum values of the features have been updated accordingly.

Feature scaling is then performed on the training set. StandardScaler class has been used to perform this feature scaling which is imported from sklearn.preprocessing package. Table 5 shows the description of the scaled feature values.

	Minimum	Q1	Q2	Q3	Maximum
US1	-1.5845	-0.77	-0.14	0.582	2.62
US2	-1.419	-0.700	-0.279	0.315	1.838
US3	-1.754	-0.796	-0.323	0.286	1.910
US4	-1.489	-0.8145	-0.261	0.9964	1.693
US5	-1.375	-0.888	-0.22	1.1012	1.5211
US6	-1.392225	-0.866	-0.1626	0.77	1.6395
US7	-1.584	-1.011	-0.0955	1.156	1.16
US8	-1.518	-0.828	-0.33	0.583	2.285
US9	-1.690	-0.977	-0.241	1.378	1.3789
US10	-1.55	-0.9125	-0.120	0.5428	1.655
US11	-1.22	-0.833	-0.346	1.089	1.478
US12	-1.247	-0.68	-0.256	0.382	1.975
US13	-1.0828	-0.7077	-0.333	0.342	1.9188
US14	-1.0248	-0.7933	-0.3939	0.670	1.665
US15	-0.99257	-0.779	-0.510	1.27	1.6426
US16	-1.3357	-0.6877	-0.411	0.463	2.18

US17	-1.479	-0.768	-0.2259	0.374	2.0899
US18	-1.497	-0.814	-0.211	0.4025	2.227
US19	-1.4339	-0.806	-0.2435	0.343	2.0682
US20	-1.4383	-0.7896	-0.260	0.339	2.033
US21	-1.3065	-0.7965	-0.2509	0.420	2.245
US22	-0.981	-0.691	-0.464	0.338	1.882
US23	-1.316	-0.7225	-0.334	0.354	1.970
US24	-1.505	-0.7552	-0.157	0.3901	2.108

Table 5: minimum, Q1, Q2, Q3, maximum values of input variables after feature scaling

RELATED WORK:

The paper in which this dataset is used is about the case study focusing on the influence of short-term memory mechanisms on neural classifiers. For this purpose, the authors of this paper have collected the sensor data of the robot using a heuristic classifier. Then they used the data set obtained in training four neural network architectures which are Logistic Perceptron, Multilayer Perceptron, Mixture of experts and Elman network. These neural network architectures are trained to make decisions on which the movement of the robot depends. The goal is to mimic the heuristic classifier in the best way possible. Without the involvement of the short-term memory mechanisms this problem can be described as a pure classification problem where the decision boundary is highly non-linear. The performance of all the four neural networks are evaluated and studied deeply. It is observed that without the involvement of short-term memory MLP perform better than all the other networks involved. But the simple wall-following navigation is indeed involved a very complex decision making. The use of the short-term memory mechanisms in the form of the recurrent loops allowed a multilayered neural architecture to reduce the number of hidden neurons without compromising the discriminative ability of the classifier. This dataset acted as a perfect classification problem they required to study the influence of the STM mechanisms on neural classifiers and the decision-making process completely changed once the STM mechanisms came into consideration. The result they obtained was MLP is a better network when STM is not considered where when LP worked comfortably when STM is considered.

TRAINING MODELS:

During the data visualization phase, it has been determined that the decision boundary of the output variable is not linear. Hence models like logistic regression, linear discriminant

analysis which has a linear decision boundary doesn't work well on this data set. The models selected to be trained on the data set are Decision tree, random forest and support vector machine classifier.

A decision tree model was fitted to the data set by importing DecisionTreeClassifier function from sklearn.tree package. An object of this function is created and fitted to the training data points. This object is now used to predict the results of the test data points. A random forest model was fitted by importing RandomForestClassifier class from sklearn.ensemble package. An object of this class is created and fitted to the training data points and then in turn used to predict the results of test points. A support vector classifier model is fitted by importing svm class from sklearn package. An object of this class is created and fitted to the training data set and is sed to obtain the predicted values of the test data points.

K-fold cross validation has been performed with k value as 10. The cross-validation scores obtained for decision tree, random forest and svm are 99.63, 95.425 and 89.62 respectively. From the above obtained scores, it can be concluded that decision tree classifier is the model which best suits this data set.

FINE TUNING THE MODELS AND FEATURE SET:

The above created models are now run on the test data to obtain the predicted values and test error of the models. The test error obtained for decision tree, random forest and support vector classifier are 0.2 percent, 5.1 percent and 19.5 percent respectively. So, when test error is considered decision tree classifier turns out be a better performer.

	Test error
Decision Tree Classifier	0.2
Random Forest Classifier	5.1
Support vector Classifier	19.5

Table 6: Test errors of the three models chosen

Feature elimination is done on the existing features. The number od features are reduced to 22 from 24. The features to be eliminated are decide by observing the box plots obtained in the data visualization phase. After observing the box plots, US17 and US18 features are eliminated to further optimize the feature set.

	Test error
Decision Tree Classifier	0.6
Random Forest Classifier	0.2
Support vector Classifier	19.91

Table 7: Test errors of the fine tuning the model on the optimized feature set.

The decision tree, random forest and support vector classifier are fitted and fine to the new training data and are fine-tuned using GridSearchCV class. This class is imported from the sklearn model_selection package. Then an object is created of this class. The class attributes we need to provide are type of estimator, parameter list and number of folds for k-fold cross validation. This GridSearch then calculates the test errors through all possible estimators obtained from the combination of the parameter provided. The best estimator which gives the least error is picked and returned. In this way through cross validation we can obtain the best possible estimator so as to get the least test error. The test error obtained for decision tree, random forest and support vector classifiers are 0.6, 0.2 and 19.9 respectively. This fine tuning of the model on the optimized feature set can result in the best model possible.

PERFORMANCE:

There are various performance metrics on which the performance of classifier models is compared. They are accuracy, precision, recall.

	Accuracy	Precision	Recall
Decision Tree classifier	1.0	1.0	1.0
Random forest classifier	0.979	0.979	0.979
Support vector classifier	0.933	0.934	0.934

Table 8: Accuracy, precision and recall values obtained over training data

	Accuracy	Precision	Recall
Decision Tree classifier	0.997	0.998	0.998

Random forest classifier	0.948	0.949	0.949
Support vector classifier	0.799	0.835	0.80

Table 9: Accuracy, precision and recall values obtained over test data.

The above tabular information tells that the performance of the models over the training data is always superior when compared to that of the test data. The precision, accuracy and recall values obtained for training data are more than that of the test data for every classifier. When the test results are considered, decision tree classifier out performs both the random forest and support vector classifier. The accuracy and precision obtained through decision tree classifier are near perfect values. Random forest also performed better and achieved a good accuracy and precision values of about 95 percent. When we consider the test-error obtained on the set of features which are not optimized, decision tree classifier performs better than the other two.

	Accuracy	Precision	Recall
Decision Tree classifier	0.993	0.994	0.994
Random forest classifier	0.997	0.998	0.998
Support vector classifier	0.799	0.829	0.80

Table 10: Accuracy, precision and recall values obtained over the feature set which is completely optimized.

The performance of decision tree and support vector classifiers is almost the same, there has been no much enhancement in the performance of these two classifiers after fine tuning the models and feature set. However, the performance of random forest classifier has improved considerably. There is approximately a 4.95 percent increase in the accuracy, precision and recall values of random forest model. The test error had reduced from 5.1 percent to 0.4 percent. There has been a decrease in the performance of other two classifiers, but that change is negligible. If the fine tune models are considered, then random forest out performs the other two classifier and can be considered as best fit model for this data set.

RESULTS AND CONCLUSIONS:

Random forest classifier turned out to be the best model out the three models selected. The data has a classification problem with a non-linear decision boundary. Data visualization and analysis is used to study how the data points of different features are distributed and what features are useful in determining the output. Feature scaling is performed on both test and training data so that a few features don't get to dominate a certain set of features. This scaling of features enhances the performance of the classification model. Since the decision boundary is non-linear, the models which depend on linear boundary are avoided and the feature set is optimized by eliminating the unwanted features. The fine tuning of the models is performed on this optimized feature set to obtain the best model possible. After fine tuning the performance of random forest has improved making it the best model possible for this dataset.

CITATIONS:

[1] Ananda L. Freire, Guilherme A. Barreto, Marcus Veloso and Antonio T. Varela (2009), 'Short-Term Memory Mechanisms in Neural Network Learning of Robot Navigation Tasks: A Case Study'. Proceedings of the 6th Latin American Robotics Symposium (LARS'2009), ValparaÃso-Chile, pages 1-6, DOI: 10.1109/LARS.2009.5418323

[2] hackernoon "https://hackernoon.com/what-is-one-hot-encoding-why-and-when-do-you-have-to-use-it-e3c6186d008 "available : 2017

[3] http://scikit-learn.org/stable/modules/classes.html