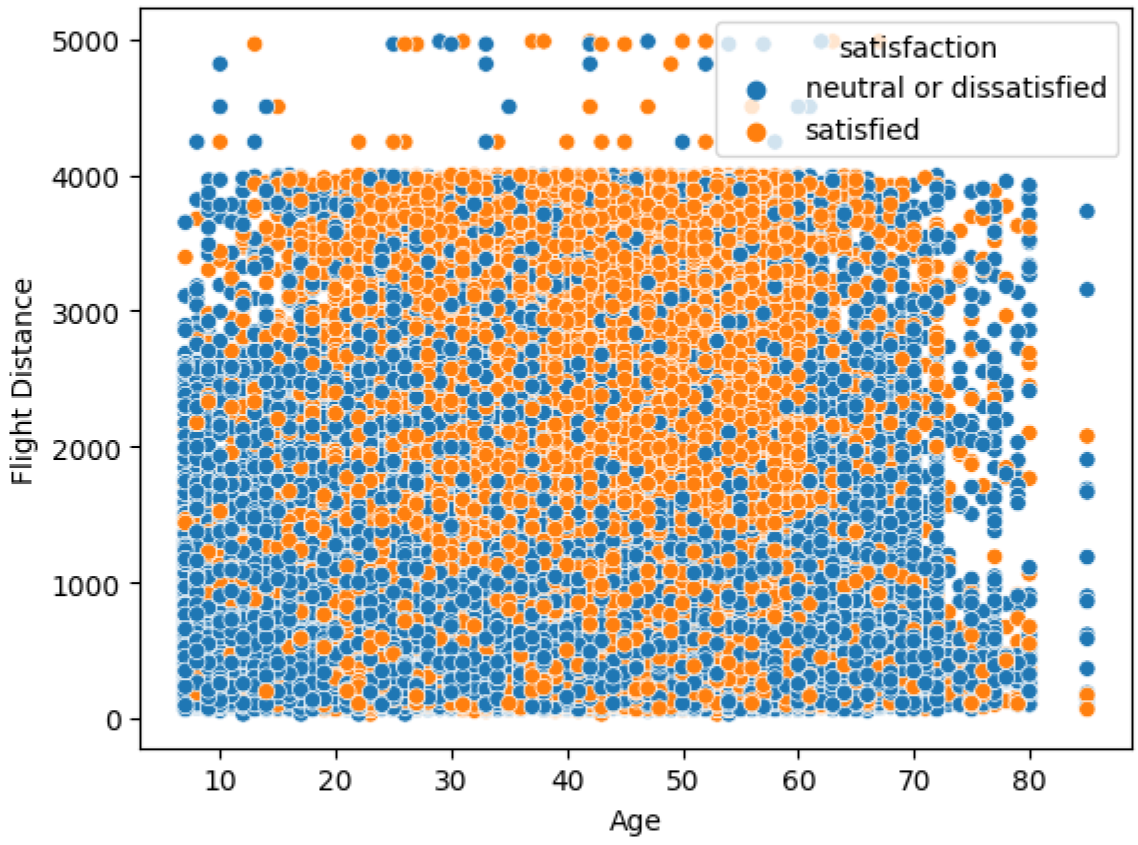
PRML LAB REPORT 8

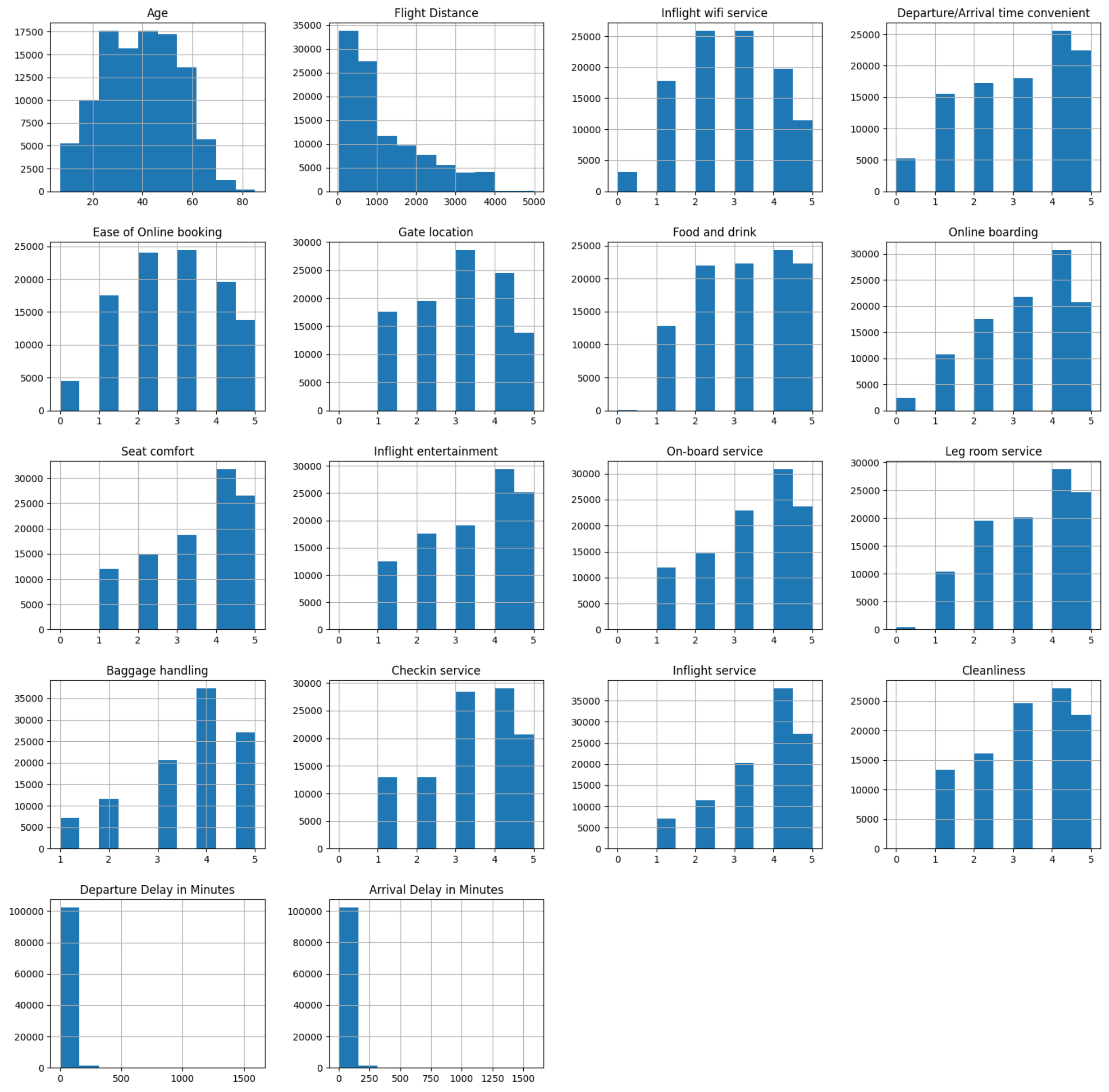
MANISH(B21CS044)

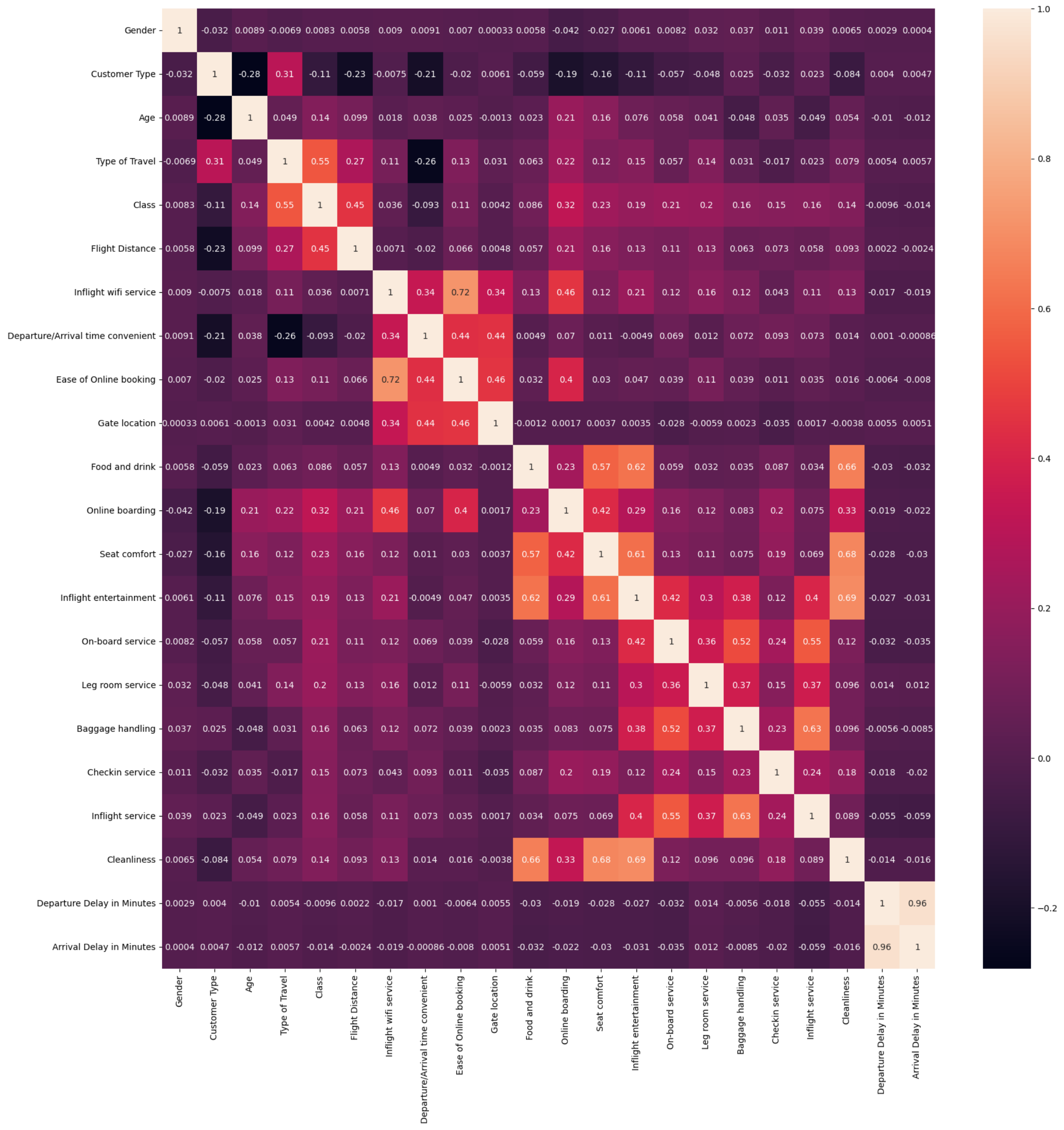
QUESTION 1

Download the dataset from the given link AirlinePassenger and perform the following tasks. There is a separate file for train and testing. Download only the train.csv file.
1) Preprocess, clean and prepare the dataset based on the previous lab experience. Separate features and labels as X and Y respectively. [5 marks]

In this part I have prepared the data as X and Y and preprocessed dropped the unnecessary columns:




Age
Flight Distance
Inflight wifi service
Departure/Arrival time convenient
Ease of Online booking
Gate location
Food and drink
Online boarding
Seat comfort
Inflight entertainment
On-board service
Leg room service
Baggage handling
Checkin service
Inflight service
Cleanliness
Departure Delay in Minutes
Arrival Delay in Minutes

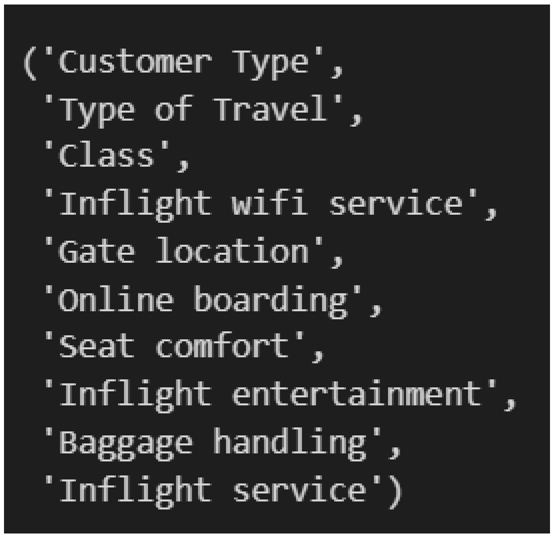


2) Create an object of SFS by embedding the Decision Tree classifier object, providing 10 features, forward as True, floating as False and scoring = accuracy. Train SFS and report accuracy for all 10 features. Also, list the names of the 10 best features selected by SFS. [10 marks]

In this part trained the SFS using Decision Tree Classifier as Classification model and no of features = 10.

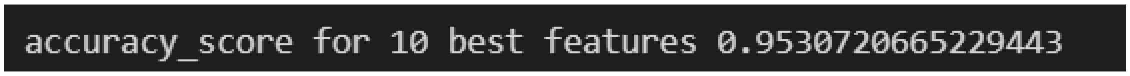
Here are the results for forward= True and floating = False.

List of best 10 selected features is:

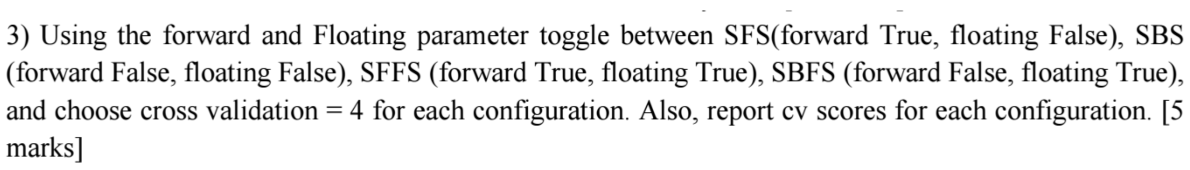


```
('Customer Type',
 'Type of Travel',
 'Class',
 'Inflight wifi service',
 'Gate location',
 'Online boarding',
 'Seat comfort',
 'Inflight entertainment',
 'Baggage handling',
 'Inflight service')
```

Here is the accuracy score for best 10 features:

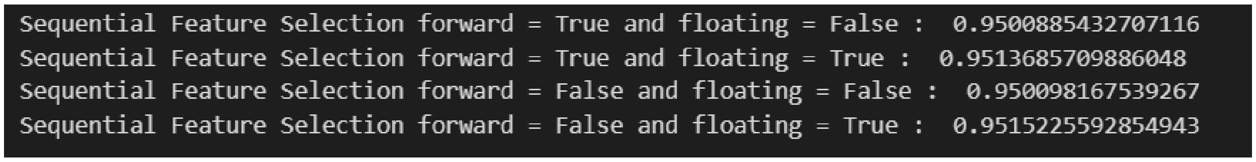


```
accuracy_score for 10 best features 0.9530720665229443
```

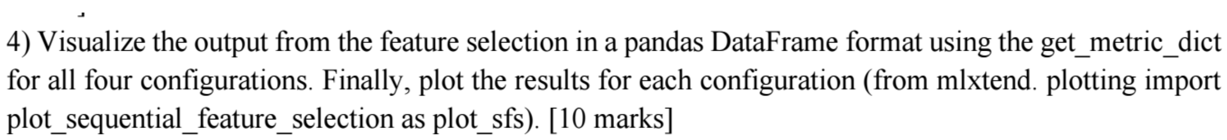


3) Using the forward and Floating parameter toggle between SFS(forward True, floating False), SBS (forward False, floating False), SFFS (forward True, floating True), SBFS (forward False, floating True), and choose cross validation = 4 for each configuration. Also, report cv scores for each configuration. [5 marks]

In this I have performed all four cases of SFS for floating and forward.
Here are the cv scores corresponding to each configuration:

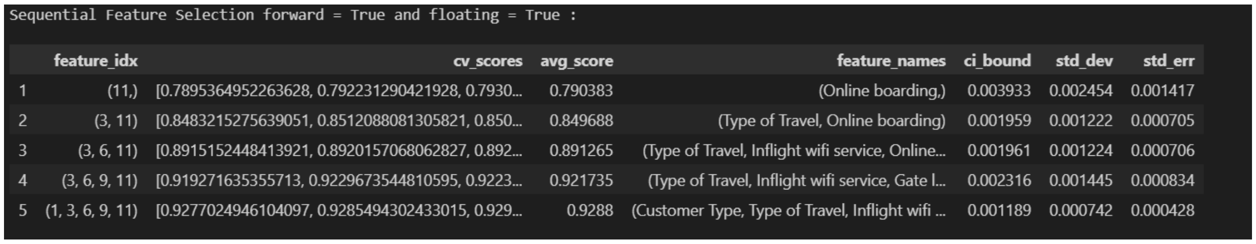


```
Sequential Feature Selection forward = True and floating = False :  0.9500885432707116
Sequential Feature Selection forward = True and floating = True :  0.9513685709886048
Sequential Feature Selection forward = False and floating = False :  0.950098167539267
Sequential Feature Selection forward = False and floating = True :  0.9515225592854943
```



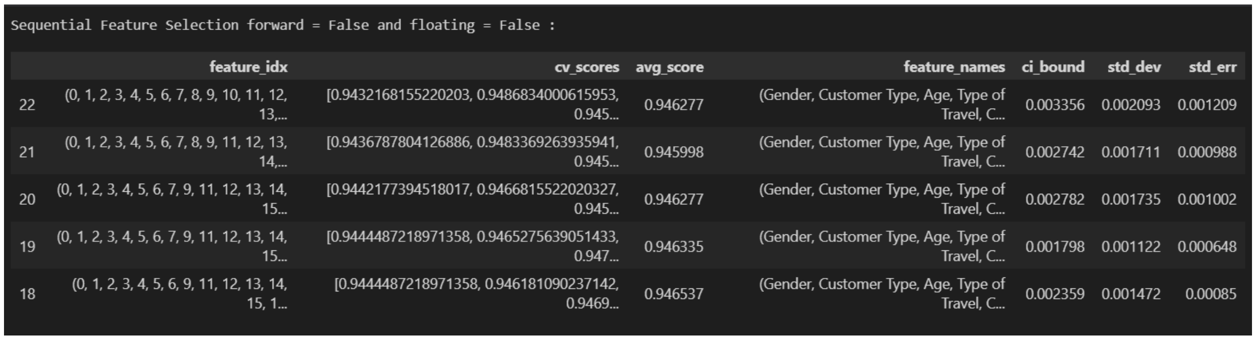
4) Visualize the output from the feature selection in a pandas DataFrame format using the get_metric_dict for all four configurations. Finally, plot the results for each configuration (from mlxtend. plotting import plot_sequential_feature_selection as plot_sfs). [10 marks]

Here are the visualization for the PandaDataframe for get_metric_dict:



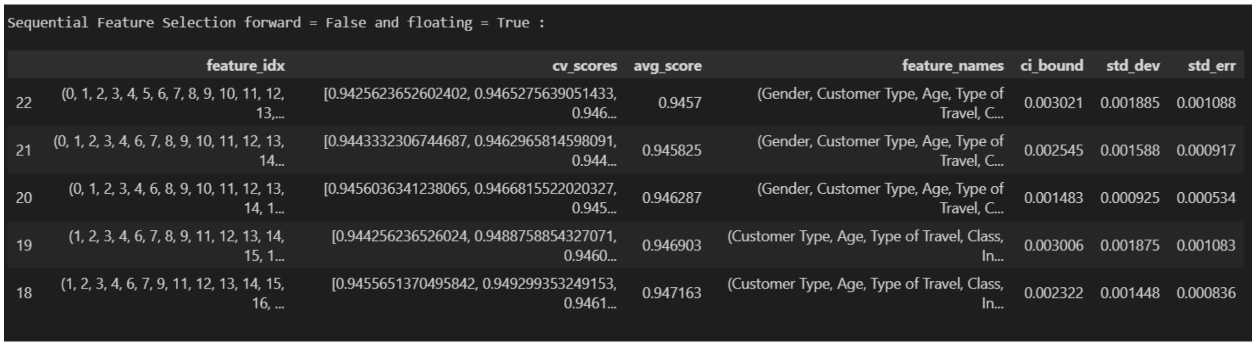
```
Sequential Feature Selection forward = True and floating = True :
```

	feature_idx	cv_scores	avg_score	feature_names	ci_bound	std_dev	std_err
1	(11,)	[0.7895364952263628, 0.792231290421928, 0.7930...	0.790383	(Online boarding,)	0.003933	0.002454	0.001417
2	(3, 11)	[0.8483215275639051, 0.8512088081305821, 0.850...	0.849688	(Type of Travel, Online boarding)	0.001959	0.001222	0.000705
3	(3, 6, 11)	[0.8915152448413921, 0.8920157068062827, 0.892...	0.891265	(Type of Travel, Inflight wifi service, Online...	0.001961	0.001224	0.000706
4	(3, 6, 9, 11)	[0.919271635355713, 0.9229673544810595, 0.9223...	0.921735	(Type of Travel, Inflight wifi service, Gate l...	0.002316	0.001445	0.000834
5	(1, 3, 6, 9, 11)	[0.9277024946104097, 0.9285494302433015, 0.929...	0.9288	(Customer Type, Type of Travel, Inflight wifi ...	0.001189	0.000742	0.000428



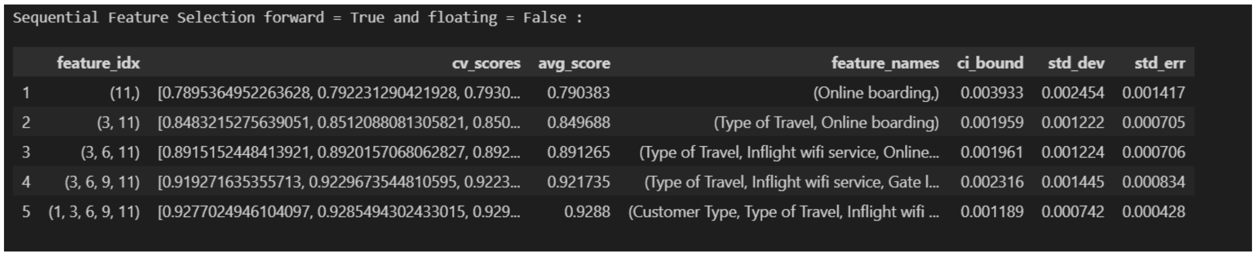
```
Sequential Feature Selection forward = False and floating = False :
```

	feature_idx	cv_scores	avg_score	feature_names	ci_bound	std_dev	std_err
22	(0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13,...	[0.9432168155220203, 0.9486834000615953, 0.945...	0.946277	(Gender, Customer Type, Age, Type of Travel, C...	0.003356	0.002093	0.001209
21	(0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 12, 13, 14,...	[0.9436787804126886, 0.9483369263935941, 0.945...	0.945998	(Gender, Customer Type, Age, Type of Travel, C...	0.002742	0.001711	0.000988
20	(0, 1, 2, 3, 4, 5, 6, 7, 9, 11, 12, 13, 14, 15...	[0.9442177394518017, 0.9466815522020327, 0.945...	0.946277	(Gender, Customer Type, Age, Type of Travel, C...	0.002782	0.001735	0.001002
19	(0, 1, 2, 3, 4, 5, 6, 7, 9, 11, 12, 13, 14, 15...	[0.9444487218971358, 0.9465275639051433, 0.947...	0.946335	(Gender, Customer Type, Age, Type of Travel, C...	0.001798	0.001122	0.000648
18	(0, 1, 2, 3, 4, 5, 6, 9, 11, 12, 13, 14, 15, 1...	[0.9444487218971358, 0.946181090237142, 0.9469...	0.946537	(Gender, Customer Type, Age, Type of Travel, C...	0.002359	0.001472	0.00085



```
Sequential Feature Selection forward = False and floating = True :
```

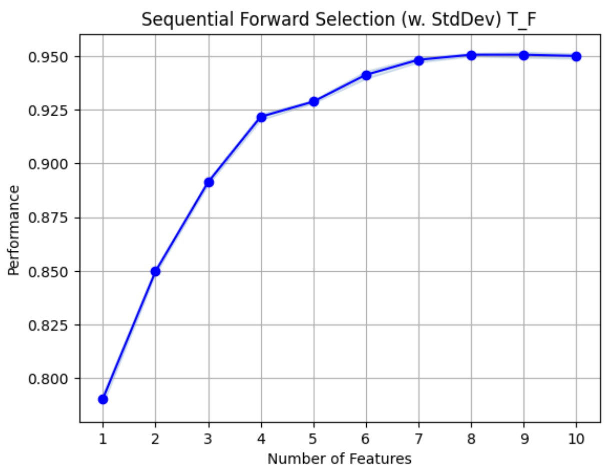
	feature_idx	cv_scores	avg_score	feature_names	ci_bound	std_dev	std_err
22	(0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13,...	[0.9425623652602402, 0.9465275639051433, 0.946...	0.9457	(Gender, Customer Type, Age, Type of Travel, C...	0.003021	0.001885	0.001088
21	(0, 1, 2, 3, 4, 6, 7, 8, 9, 10, 11, 12, 13, 14...	[0.9443332306744687, 0.9462965814598091, 0.944...	0.945825	(Gender, Customer Type, Age, Type of Travel, C...	0.002545	0.001588	0.000917
20	(0, 1, 2, 3, 4, 6, 8, 9, 10, 11, 12, 13, 14, 1...	[0.9456036341238065, 0.9466815522020327, 0.945...	0.946287	(Gender, Customer Type, Age, Type of Travel, C...	0.001483	0.000925	0.000534
19	(1, 2, 3, 4, 6, 7, 8, 9, 11, 12, 13, 14, 15, 1...	[0.944256236526024, 0.9488758854327071, 0.9460...	0.946903	(Customer Type, Age, Type of Travel, Class, In...	0.003006	0.001875	0.001083
18	(1, 2, 3, 4, 6, 7, 9, 11, 12, 13, 14, 15, 16, ...	[0.9455651370495842, 0.949299353249153, 0.9461...	0.947163	(Customer Type, Age, Type of Travel, Class, In...	0.002322	0.001448	0.000836

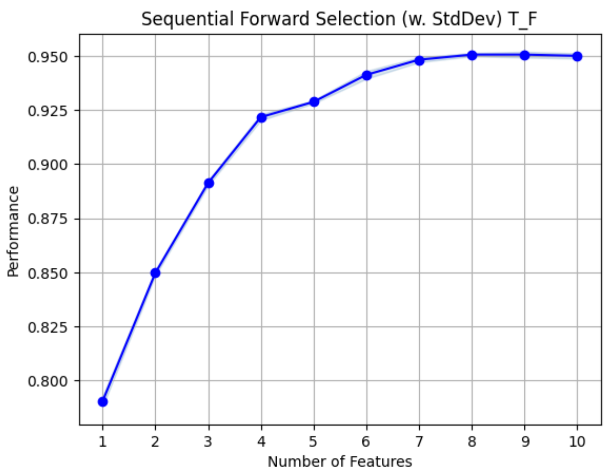


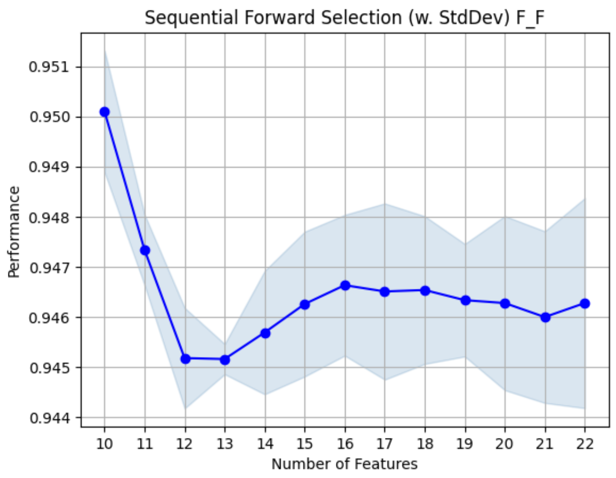
```
Sequential Feature Selection forward = True and floating = False :
```

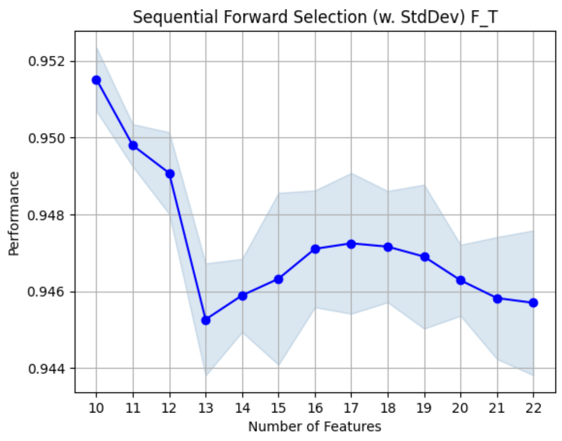
	feature_idx	cv_scores	avg_score	feature_names	ci_bound	std_dev	std_err
1	(11,)	[0.7895364952263628, 0.792231290421928, 0.7930...	0.790383	(Online boarding,)	0.003933	0.002454	0.001417
2	(3, 11)	[0.8483215275639051, 0.8512088081305821, 0.850...	0.849688	(Type of Travel, Online boarding)	0.001959	0.001222	0.000705
3	(3, 6, 11)	[0.8915152448413921, 0.8920157068062827, 0.892...	0.891265	(Type of Travel, Inflight wifi service, Online...	0.001961	0.001224	0.000706
4	(3, 6, 9, 11)	[0.919271635355713, 0.9229673544810595, 0.9223...	0.921735	(Type of Travel, Inflight wifi service, Gate l...	0.002316	0.001445	0.000834
5	(1, 3, 6, 9, 11)	[0.9277024946104097, 0.9285494302433015, 0.929...	0.9288	(Customer Type, Type of Travel, Inflight wifi ...	0.001189	0.000742	0.000428

Here are the plots for each Configuration using mlxtend library:









Here we can see that on floating = True the graphs are similar and on floating = False graphs are similar.

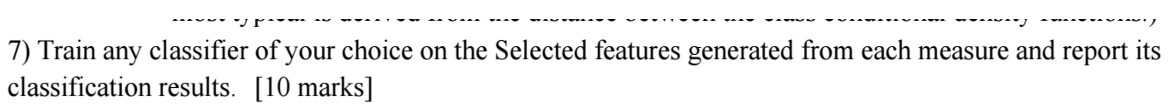
5) Implement Bi-directional Feature Set Generation Algorithm from scratch. It must take a Full Set of features as well as similarity measures as input. [10 marks]

6) Use the function implemented in part 5 and use selection criteria from the following: [10 marks]

- Accuracy Measures: using Decision Tree and SVM Classifiers
- Information Measures: Information gain
- Distance Measure: Angular Separation, Euclidian Distance and City-Block Distance
- Distance Measures. - Measures of separability, discrimination or divergence measures. The most typical is derived from the distance between the class conditional density functions.)

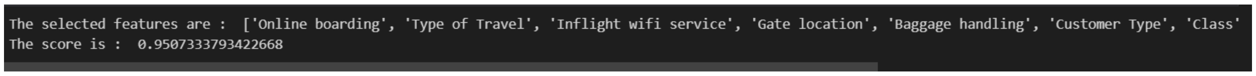
In this part I have implemented the Bi-directional Feature Set Selection From the scratch.

And also used the function for different selection criteria the results for the same are shown in the below part.



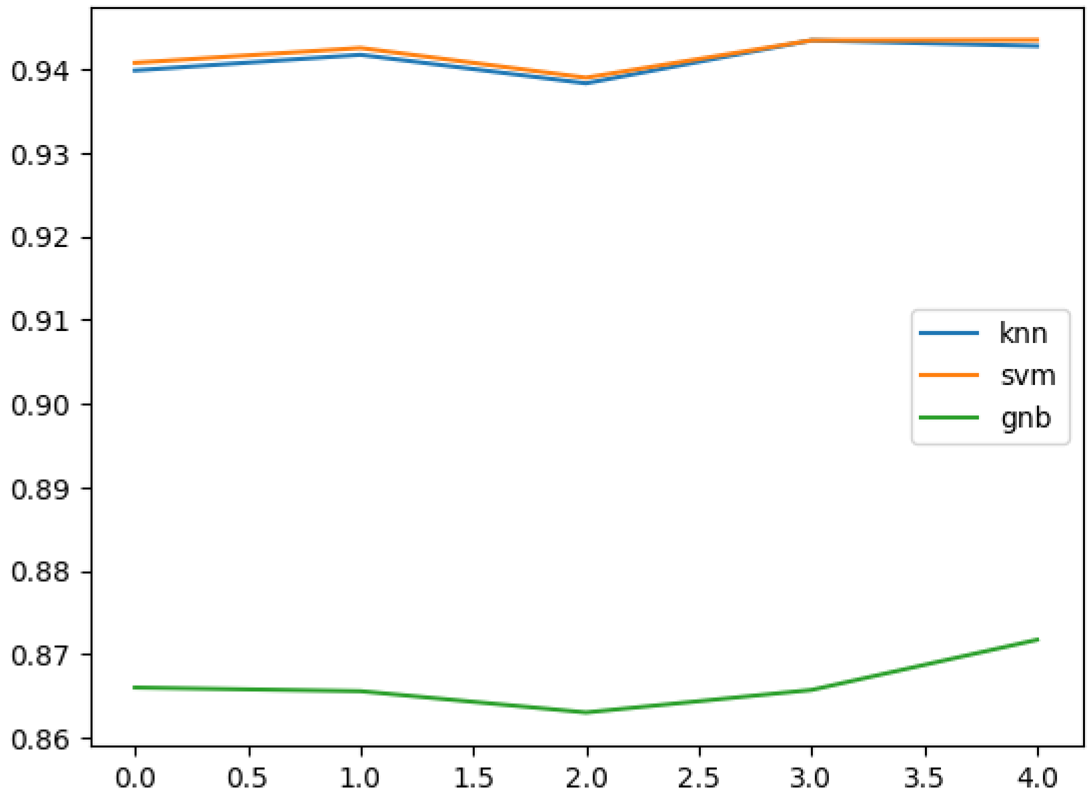
7) Train any classifier of your choice on the Selected features generated from each measure and report its classification results. [10 marks]

In this part I have selected the Decision Tree classifier as the selection criteria and here are the results for the same:



```
The selected features are :  ['Online boarding', 'Type of Travel', 'Inflight wifi service', 'Gate location', 'Baggage handling', 'Customer Type', 'Class'
The score is :  0.9507333793422668
```

Here are the comparison results for the same on different classifier trained on the dtc data:



QUESTION 2

1. Make a Dataset of 1000 points sampled from a zero-centred gaussian distribution with a covariance matrix

$$\sum = \begin{bmatrix} 0.6006771 & 0.14889879 & 0.244939 \\ 0.14889879 & 0.58982531 & 0.24154981 \\ 0.244939 & 0.24154981 & 0.48778655 \end{bmatrix}$$

Label the points as shown below:

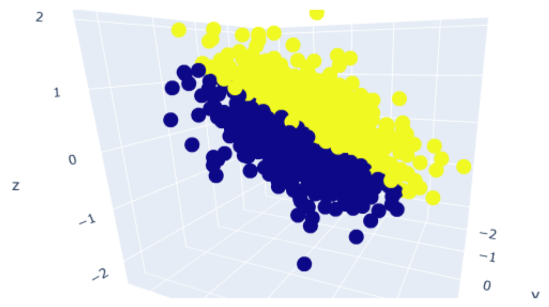
$$class = \begin{cases} 0 & \vec{x}.\vec{v} > 0 \\ 1 & \vec{x}.\vec{v} <= 0 \end{cases} where \vec{v} = \begin{bmatrix} 1/sqrt(6) \\ 1/sqrt(6) \\ -2/sqrt(6) \end{bmatrix}$$

and x is the data point. Visualize the data as a 3D scatter-plot using plotly's scatter_3d function.

[10 marks]

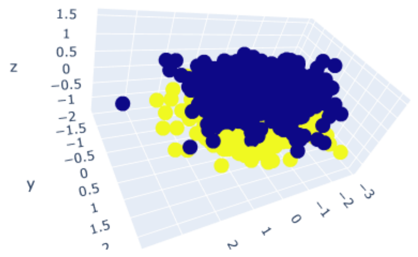
In this part I have generated 1000 data points as given using mean = [0,0,0] and according to the covariance matrix.
After this I have labeled the classes according to the condition given above.

Here is the 3d plot for the data points:



2. Apply Principal Component analysis (using sklearn) with n_components=3 on the input data X and transform the data accordingly. [5 marks]

In this part I have implemented PCA for n_components = 3 and Here is plot after PCA is implemented:



3. Perform Complete FS on the Transformed Data with a number of features in subset =2. Fit a Decision Tree for every subset-set of features of size 2 and plot their decision boundaries superimposed with the data. [20 marks]

In this part I have performed the Feature Selection using Exhaustive Feature Selection and also performed the selection manually and compared the results and Decision Tree is used as Classifier model in the both:

Here are the results for the Exhaustive Feature Selection:

'avg_score': 0.504,

'feature_names': ('x1', 'x2')},

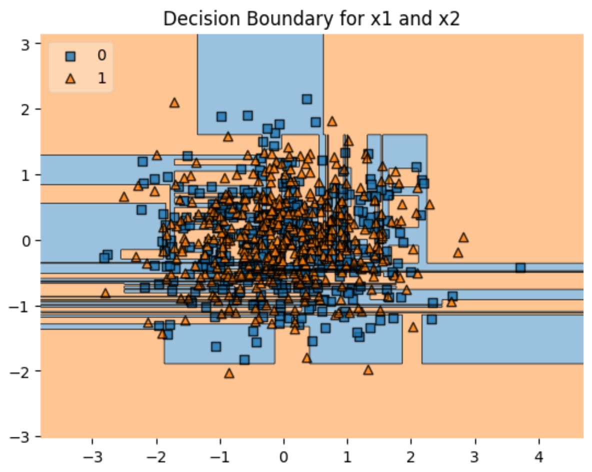
'avg_score': 0.9720000000000001 ,

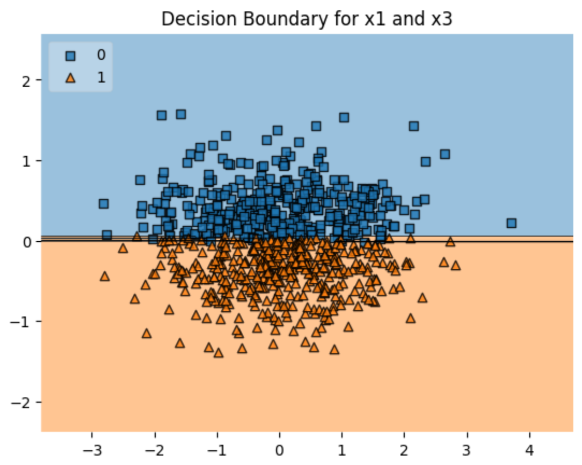
'feature_names': ('x1', 'x3')},

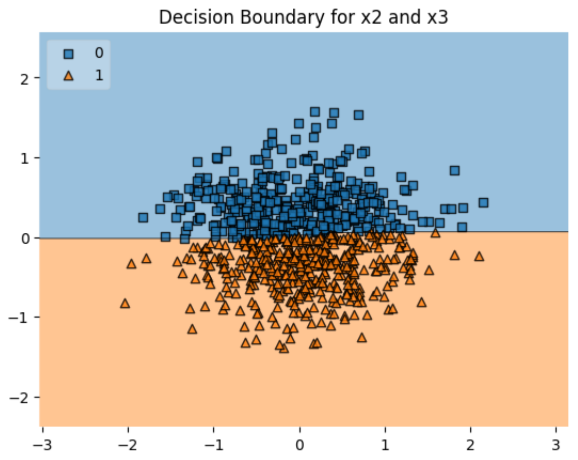
'avg_score': 0.992,

'feature_names': ('x2', 'x3')}}

Here are the Decision Boundaries for the same:



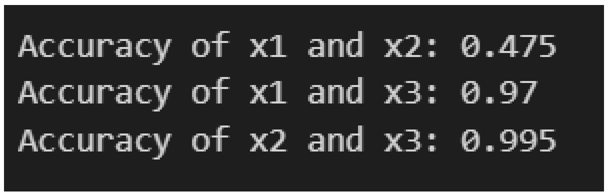




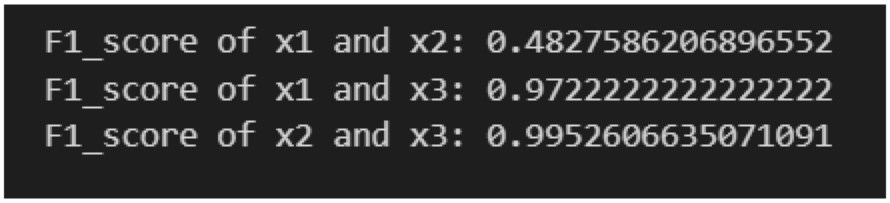
From Decision Boundaries we can see that features X1 and X2 model is performing very poorly as its Decision Boundary is not clear at all. And while for X2 , X3 and X1 , X3. It is perfectly visible.

4. Which of the above feature subsets represents the one that can be obtained by applying PCA(n_components =2)? Explain the difference in the accuracies between this subset and other subsets by running suitable experiments. [5 marks]

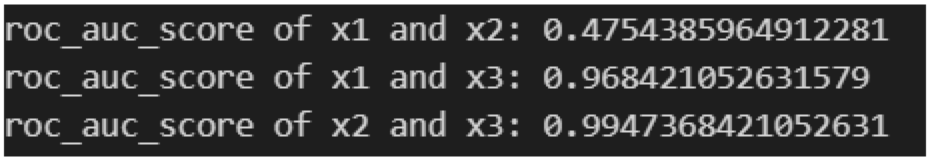
In this part I have calculated different Accuracy Metrics here are the results for the same:



```
Accuracy of x1 and x2: 0.475
Accuracy of x1 and x3: 0.97
Accuracy of x2 and x3: 0.995
```



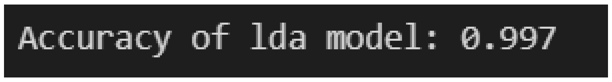
```
F1_score of x1 and x2: 0.4827586206896552
F1_score of x1 and x3: 0.9722222222222222
F1_score of x2 and x3: 0.9952606635071091
```



```
roc_auc_score of x1 and x2: 0.4754385964912281
roc_auc_score of x1 and x3: 0.968421052631579
roc_auc_score of x2 and x3: 0.9947368421052631
```

From the scores we can clearly see that features with high variance are not performing well while features with less variance are performing better. This is because most variant features are not able to check for the relationship between the classes and the features.

For checking So , I have performed LDA on the same data and the result are as follow:



```
Accuracy of lda model: 0.997
```

It demonstrates that variance explanation does not always provide the most accurate data-related information. Because PCA(n_components = 2) was unable to capture the non-linear relationship between the features and the target variable, the subset it produced had the lowest accuracy. However, even with n_components=1, lda yields the best results. This is because the classes get very mixed up when doing PCA.