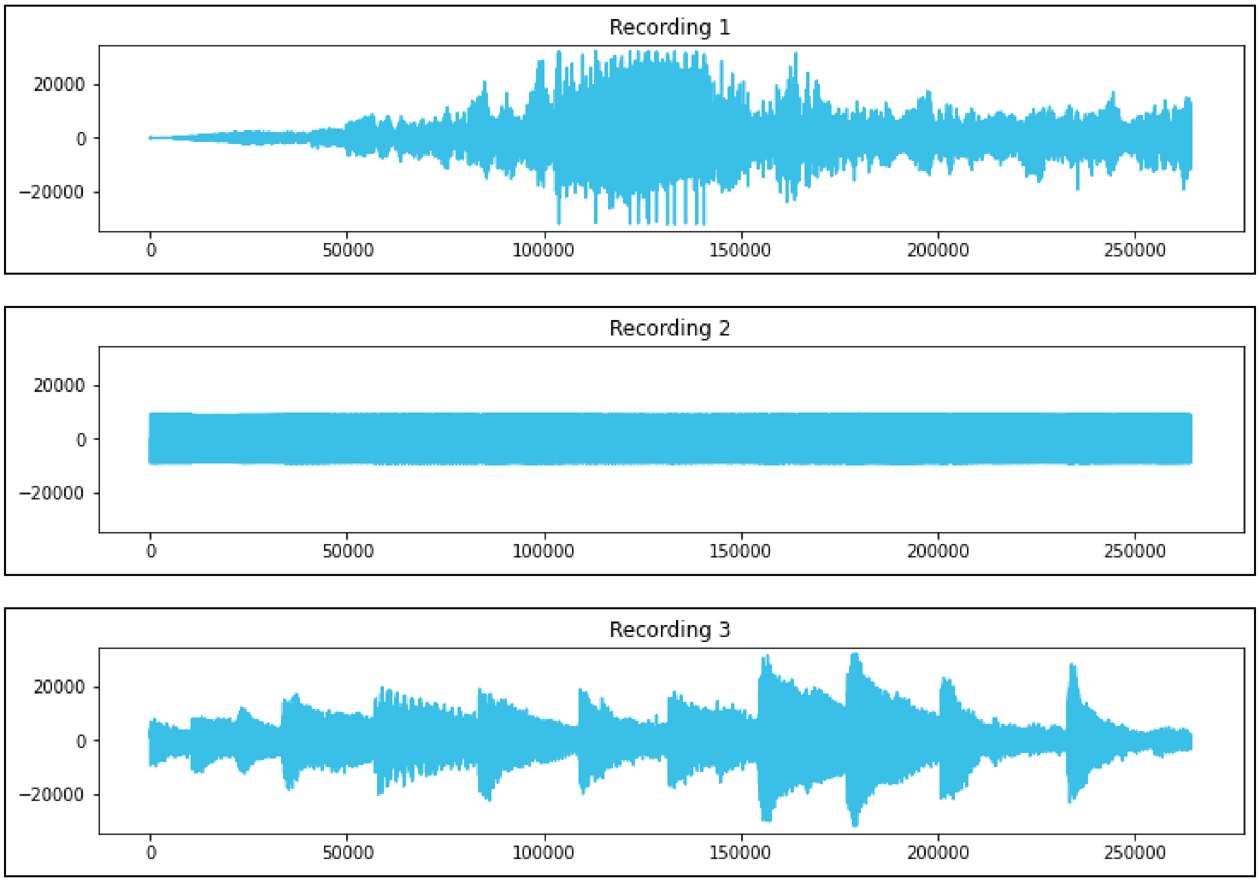
# LAB 8 REPORT

[redacted]

**Q1)**

For the first question, I am using audio files as signals, and mixing them to make mixed signals. For reading audio files, I am using inbuilt libraries such as Ipython and wave. I am basically converting that audio signal into an array that we can operate on. For separating sources from mixed signals ICA is used.The arrays each show a respective signal, and using these 3 arrays(from 3 audio files given to us), we can plot their waveforms. Then these arrays are zipped into a single dataset to form the mixed signal. Before I started to separate these signals I added some noise into the dataset by taking a dot product between the dataset and a 3x3 matrix.

**Original Sources**

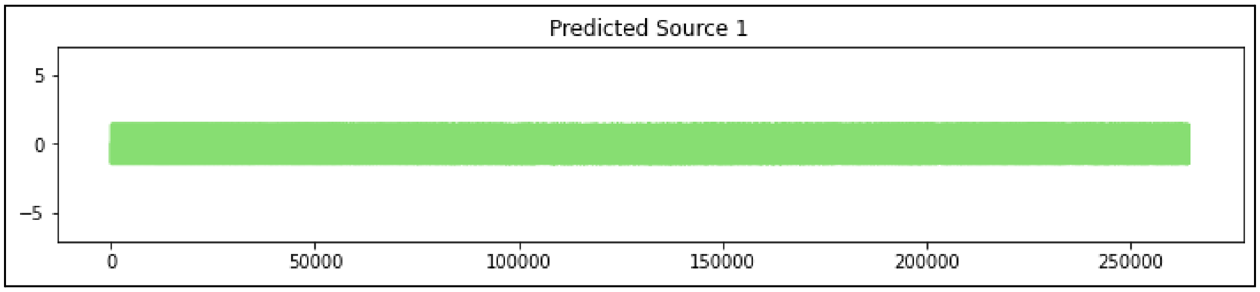


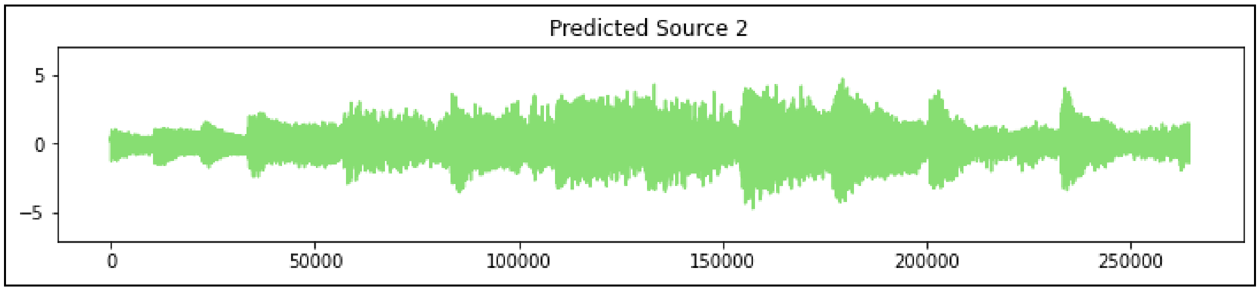
1. Center **x** by subtracting the mean
2. Whiten **x**
3. Choose a random initial value for the de-mixing matrix **w**
4. Calculate the new value for **w**
5. Normalize **w**
6. Check whether algorithm has converged and if it hasn't, return to step 4
7. Take the dot product of **w** and **x** to get the independent source signals

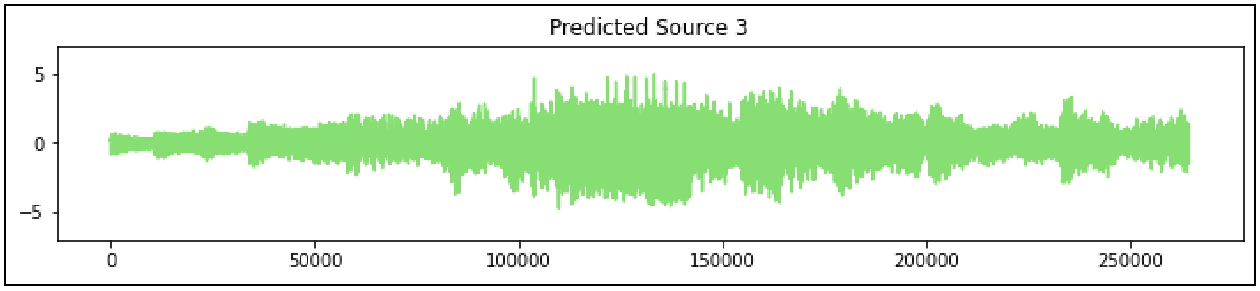
$$S = Wx$$

After using ICA we get the dataset and we can simply separate its 3 components to get independent components. We would get 3 arrays, and using these we can plot waveforms for observed separated signals.

**Predicted Signal From Scratch ICA**



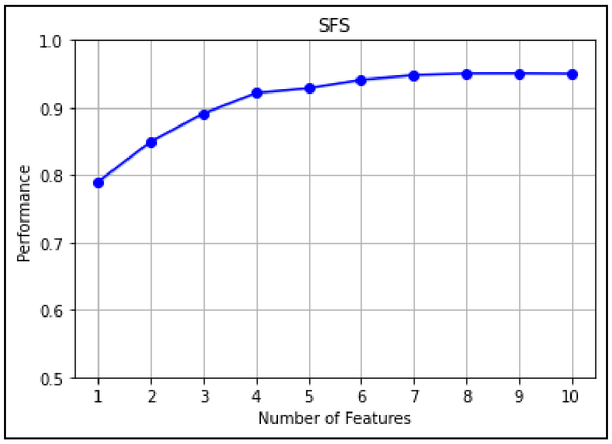


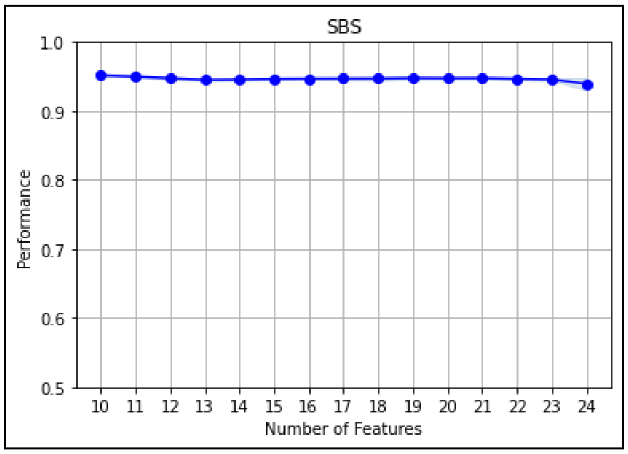


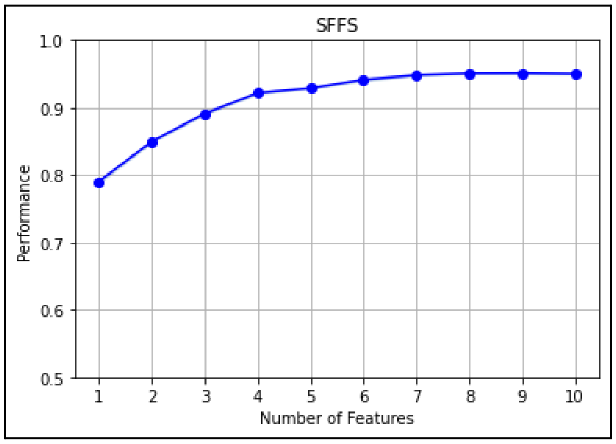
After recovering the signals in array form,they were used to make audio files. Upon listening to the audio files it was noticed that the predicted signal matched quite well with the original audio signals.
When comparing ICA results with Fast ICA it was found that the predicted signal matched pretty well. The only difference observed was that in FastICA amplitude of predicted signal was lower than that of ICA.

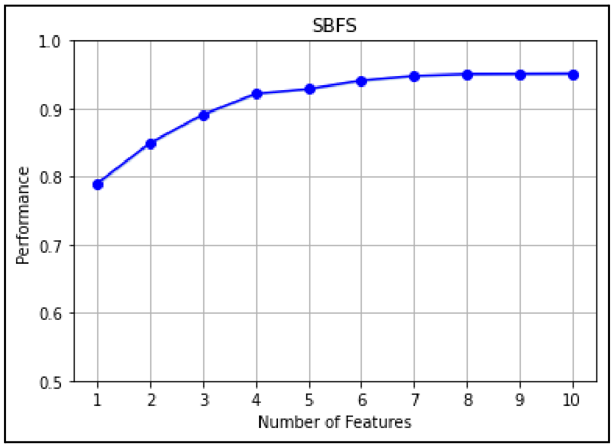
**Q2)**
For second question, I firstly did preprocessing where I removed NaN values and encoded the categorical columns using LabelEncoder.Then I imported the SequentialFeatureSelector from mlxtend and created an object with Decision Tree classifier as the model embedded in it. I used an instance of SFS with 10 features and cv=10. After fitting SFS with the x_train and y_train, a dictionary was received  which contained another dictionary for the number of features going from 1 to 10. We have to work with the key value =10 of the dictionary until that 10 features are selected. With this further dictionary, we can get cv scores and average scores, and the indexes and names of features selected. The best features I received from SFS were Customer Type, Type of Travel, Class, Inflight wifi service, Gate location, Online boarding, Seat comfort, Inflight entertainment, Baggage handling and Inflight service.
Similarly I did SFS(forward True, floating False), SBS (forward False, floating False), SFFS (forward True, floating True), SBFS (forward False, floating True) while keeping cv=4. For all these, an average score of 0.95 and a cv score of

0.94 was achieved . Next I visualized the output of each configuration with the help of ‘get_metric_dict’. Then for better understanding I also plotted output of each configuration.

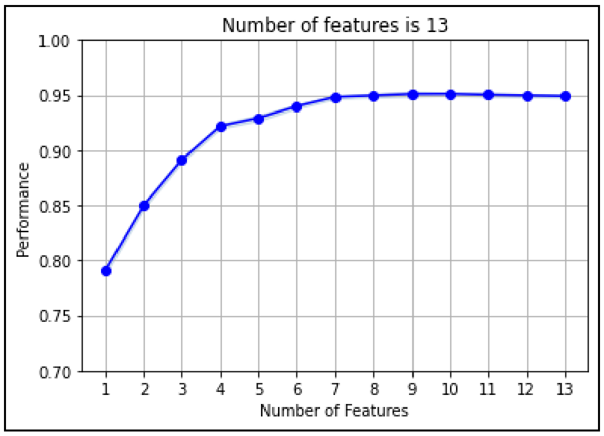


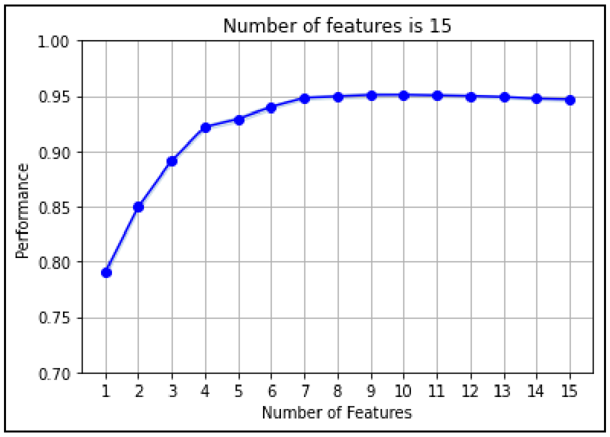


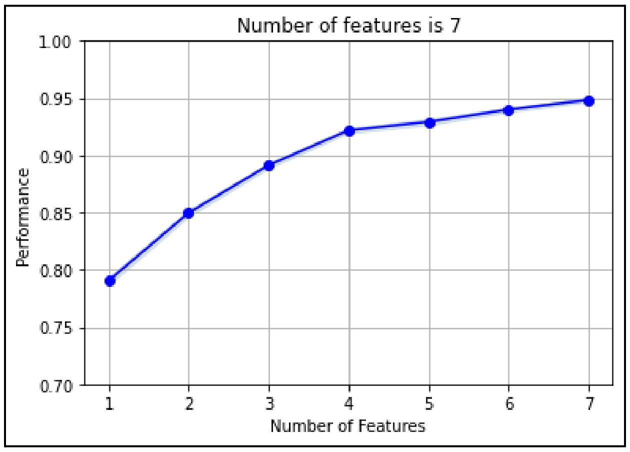


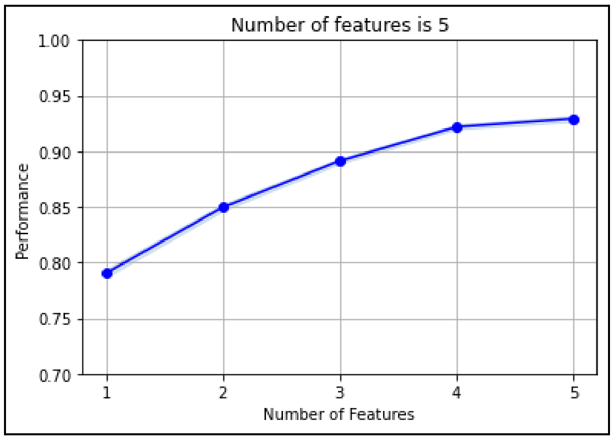


For the last part I tried both increasing(to 13 and15) and decreasing the number (to 5 and 7) of features of SFS. On doing this I observed that in each case SFS achieved the performance increase when the number of features was increased till after that it was seen the performance slowly declined when increasing the number of features above 10.


Number of features is 13
Performance
Number of Features


Number of features is 15
Performance
Number of Features


Number of features is 7
Performance
Number of Features


Number of features is 5
Performance
Number of Features