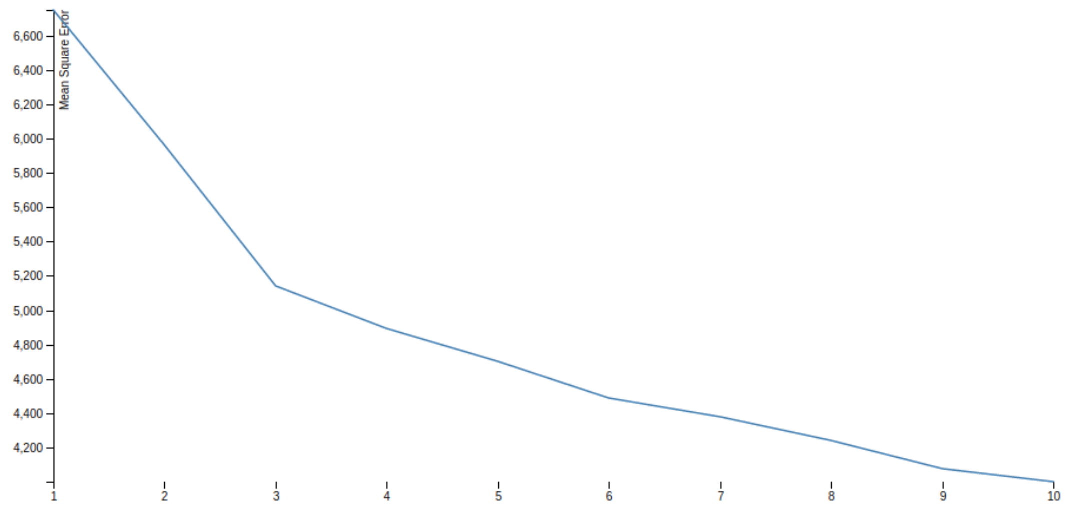
Lab2 Report

Data Set Chose : Flights.csv

1)Data Clustering and Decimation : Data was normalized first.Random sampling and Stratified sampling was done on the dataset.For random sampling python inbuilt function was used .
*rand_smpl=[list_data[i] for i in random.sample(range(len(list_data)),int(settings.samplePercent*len(list_data)))]*
To perform stratified sampling data was divided into clusters first and then random sampling was performed on the clusters to pick a proportionate number of samples.Best K was determined by means of elbow method from the range[1,11).Graph depicts Mean square error vs K

```
for i in range(1,settings.maxK):
        ClustersDict,ClusterCentres=MakeClusters(list_data,i)
        SSE.append(GetSSE(ClustersDict,ClusterCentres))
```

SSE stores the mean sqaured error and k which is then used to plot the graph



From the plot we can observe that elbow has occurred at K=3.
To plot elbow line_chart.js is used.
Two csv files generated by the server after task1 which are saved as RandomSamplingData.csv and StratifiedSamplingData.csv

2)Dimension Reduction : To find the Intrinsic Dimensionality of the decimated data- line graph was plotted as suggested by Prof. on piazza, seperately for RandomSamplingData.csv and StratifiedSamplingData.csv data .From the graphs intrinsic dimensionality of the data is observed.

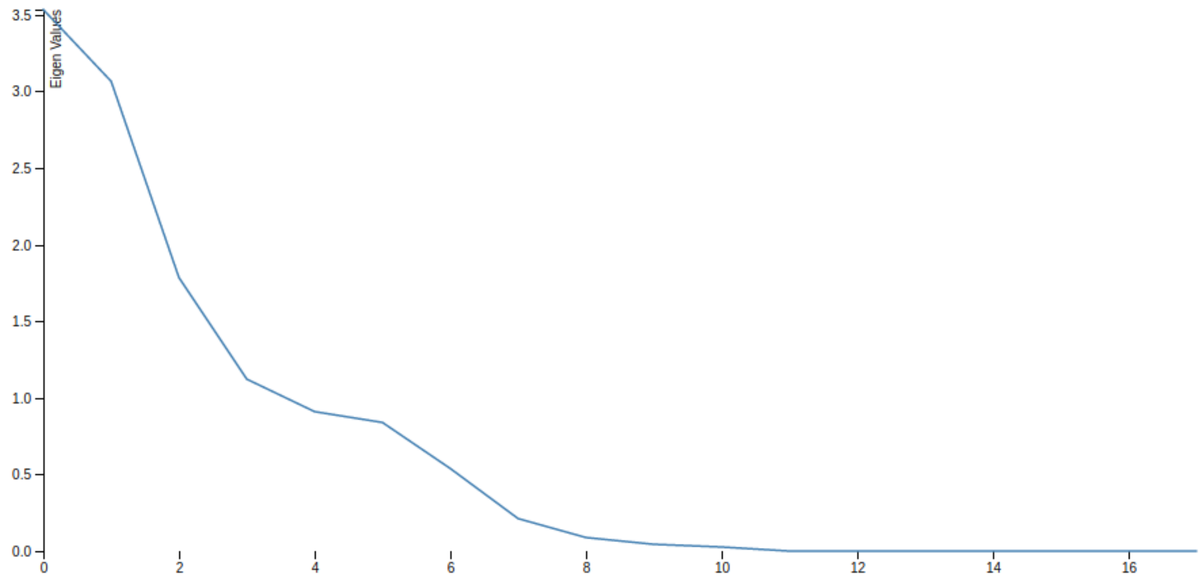
Server end code snippet : *for k in dict_dataColumns.keys()*

```
                list_columnorder .append(k)
                list_dataColumns.append(dict_dataColumns[k])
        for k in list_columnorder:
                dict_dataCovariance[k]=[]
        list_dataCovariance=np.cov(list_dataColumns)
        for i,k in enumerate(list_columnorder):
                dict_dataCovariance[k]=list_dataCovariance[i]
        eigenValues=LA.eigvals(list_dataCovar iance)
```

Eigen values and covariance among the data is used to dimensionally reduce the data.The columns corresponding to the top eigenvalues are chosen as the most contributing columns and the server stores the data in csv files separately for random data and stratified data.
For scree plot visualistaion eigen values of the top components against the componenets is plotted . line_chart.js is used.
Line plot Code snippet :g.append("g")

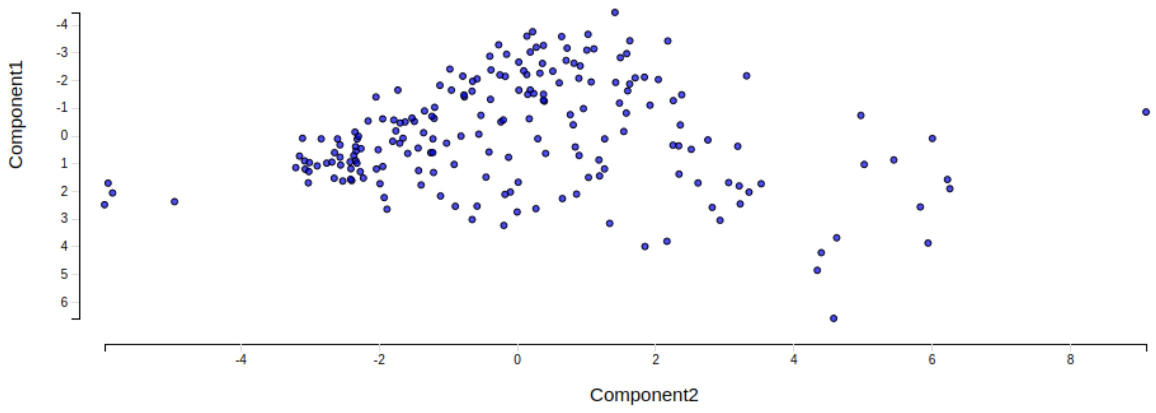
```
      .attr("transform", "translate(0," + height + ")")
      .call(d3.axisBottom(x))
    .select(".domain")
      .remove();

  g.append("g")
      .call(d3.axisLeft(y))
    .append("text")
      .attr("fill", "#000")
      .attr("transform", "rotate(-90)")
      .attr("y", 6)
      .attr("dy", "0.71em")
      .attr("text-anchor", "end")
      .text("Mean Square Error");

  g.append("path")
      .datum(data)
      .attr("fill", "none")
      .attr("stroke", "steelblue")
      .attr("stroke-linejoin", "round")
      .attr("stroke-linecap", "round")
      .attr("stroke-width", 1.5)
      .attr("d", line);
```

For scree plot line chart is plotted.Below attached is the scree plot for stratified sampled data.Similarly scree plot for random sampled data is plotted.



PCA loadings of the attributes with highest loadings are plotted against the attributes separately for random sampled data and stratified sampled data.

3)Data is projected into the top two PCA vectors and is visualised via a 2D scatterplot



Above attached graph is random sampled PCA.
Code snippet at server end :

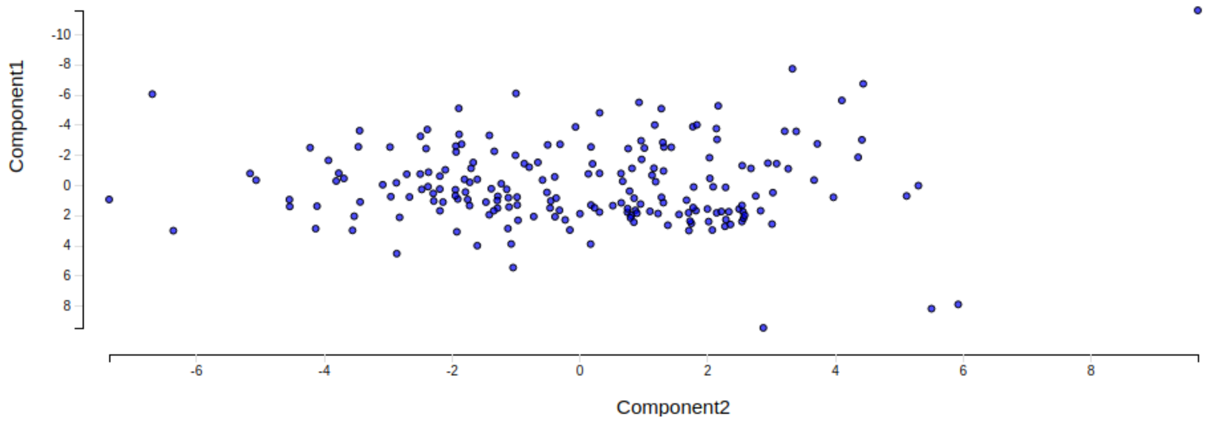
```
def PCAdata(filename):
                list_data , _ = readData(filename)
                list_data = StringtoInt(list_data)
                pca = PCA(n_components=2)
                pcaNumpy=pca.fit_transform(list_data)
                pcaList=[]
                for i in range(len(pcaNumpy)):
                        x=float(pcaNumpy[i][0])
                        y=float(pcaNumpy[i][1])
```

```
                    li=[x,y]
                    pcaList.append(li)
          return pcaList
```

MDS Euclidean and Correlation is plotted for random and stratified sampled data.
Code snippet at server end for MDS Euclidean is:

```
     dis_mat=pairwise_distances(list_data,metric=type_mds)
          mds=MDS(n_components=2,dissimilarity='precomputed')
          mdsNumpy=mds.fit_transform(dis_mat)
          mdsList=[]
          for i in range(len(mdsNumpy)):
                    x=float(mdsNumpy[i][0])
                    y=float(mdsNumpy[i][1])
                    li=[x,y]
                    mdsList.append(li)
          return mdsList
```

Graphs plotted for MDS are scatter plots .
Below attached is a graph for stratified euclidean.



Scatterplot Matrix of the highest three PCA loaded attributes is plotted separately for stratified and random sampled data.This plot provides a better visualisation of the inter dependencies among all three top loaded attributes.

It is observed that stratified sampled data produces a better sampling method.In this method outliers are covered where as in random sampling outliers may be missed.So stratified sampling provides a better insight into the data.From the scree plot attached above(2nd line chart) it can be observed that Intrinsic dimensionality of the data is 3 as only three components have eigenvalues more than 1.