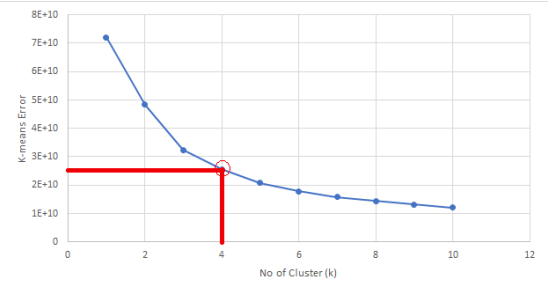
LAB 2 REPORT

1. Chosen Data Set: College.csv
   The data is in csv format and contains no missing/unknown values. It contains the statistics for many US Colleges from the 1995 issue of US News and World Report. The data consist of 777 records/rows and 18 attributes.

2. Tasks:
   a. Data Clustering and Decimation
      Random, and stratified samples are obtained from the main csv data and stored in random_samples.csv and adaptive_samples.csv respectively. The latter needed k-mean clustering to find the optimal value for k (number of clusters) from the elbow curve which is plotted between k-means error and number of clusters.

Snapshot

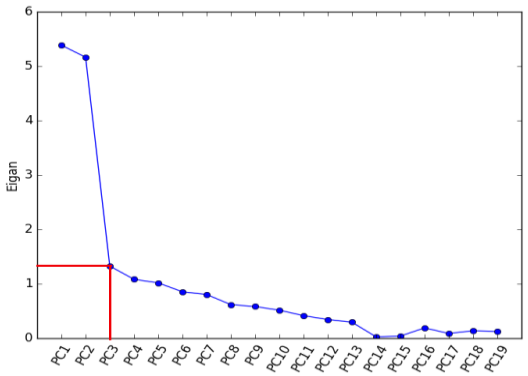


Snippet (Python)

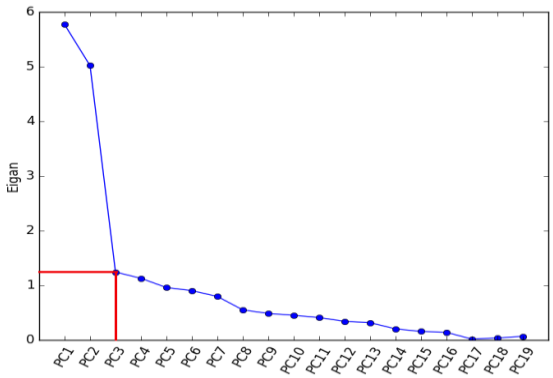
```
def elbowKMeans(inputData, n):
    Ks = list(range(1, n))
    km = [KMeans(n_clusters=i) for i in Ks]
    score = [km[i].fit(inputData).score(inputData) for i in range(len(km))]
    df_Ks = pd.DataFrame(Ks)
    df_Ks.columns = ["K"]
    df_score = pd.DataFrame(score).abs()
    df_score.columns = ["k_means_error"]
    sample = df_Ks.join([df_score])
    sample.to_csv("./data/elbow.csv", sep=',')
    return 1
```

   b. Dimension reduction
      Intrinsic Dimensionality of data is calculated using the PCA plot for both random and adaptive data samples. The Scree plot visualization is produced and intrinsic dimensionality is marked. It comes out to be 3 in both the cases as shown below:

Snapshot (random sampling)



Snapshot (adaptive sampling)

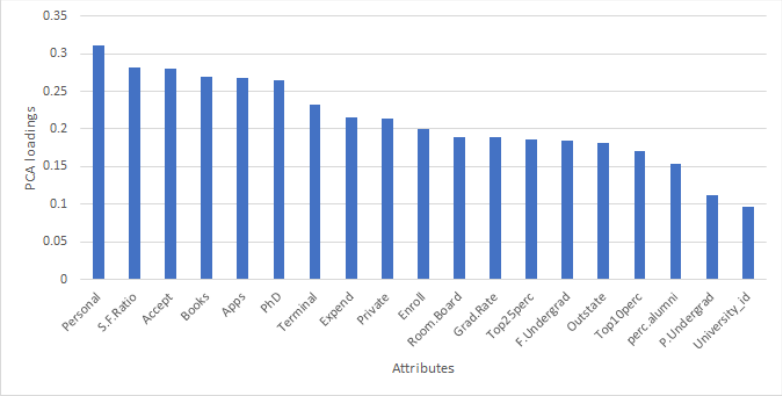


Snippet(Python)

```
def createScreeWithEigan(inputSamples, name):
    X_std = StandardScaler().fit_transform(inputSamples)
    # Eigan values for co-variance matrix
    cov_mat = np.cov(X_std.T)
    eig_vals1, eig_vecs1 = np.linalg.eig(cov_mat)
    # Eigan values for co-relation matrix
    cor_mat1 = np.corrcoef(X_std.T)
    eig_vals, eig_vecs = np.linalg.eig(cor_mat1)
    y = eig_vals
    x = np.arange(len(y)) + 1
    df_eig = pd.DataFrame(eig_vals)
    df_eig.columns = ["eigan_values"]
    df_pca = pd.DataFrame(x)
    df_pca.columns = ["PCA_components"]
    sample = df_eig.join([df_pca])
    sample.to_csv("./data/scree_plot_"+name+".csv", sep=',')
    plt.plot(x, y, "o-")
    plt.xticks(x, ["PC" + str(i) for i in x], rotation=60)
    plt.ylabel("Eigan")
    plt.show()
```

PCA loadings are calculated for all the attributes and arranges in the descending order of loading values.

Snapshot



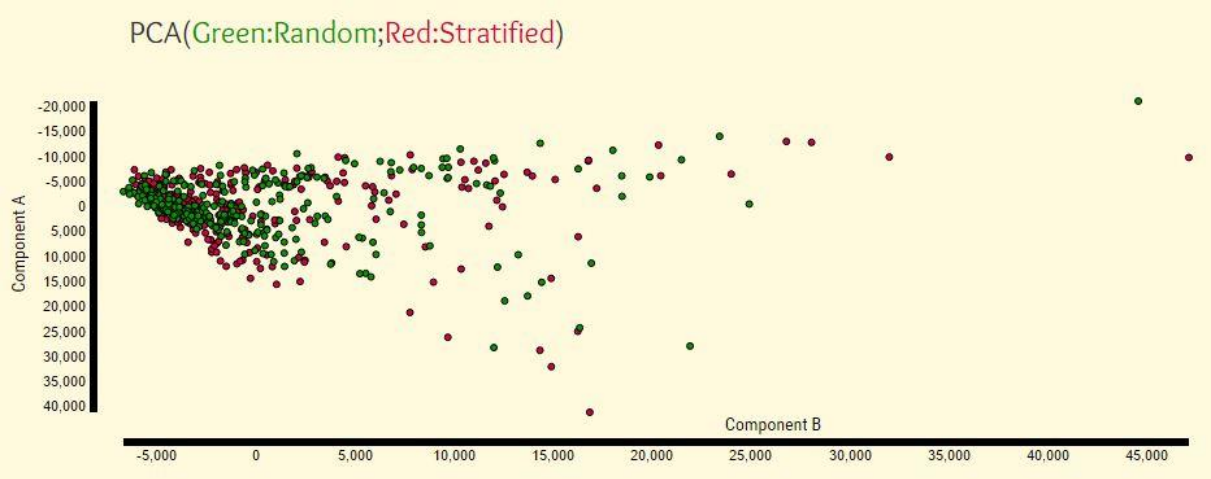
Snippet (Python)

```
def getTop3attributes(squared_loadings):
    top3 = squared_loadings.head(n = 3)
    return top3['attributes'].values.tolist()
     arr = np.array(squared_loadings)
     top3 = arr.argsort()[-3:][::-1]
     return top3
     print(top3)
     print(squared_loadings)
```

c. Visualization

The data is projected into the top two PCA vectors via scatterplot and visualized using 2D scatterplot.

Snapshot



Snippet(D3)

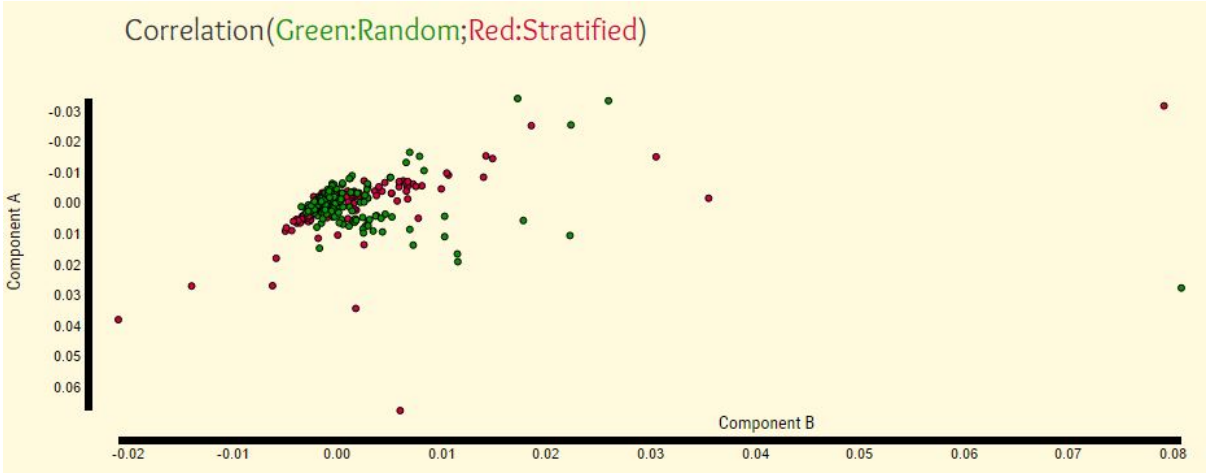
```
<div id="pca" class="address" style="background-color: #fff9dd" >
    <div class="container"><br><br>
            <h3> PCA(<font color="#0c9603">Green:Random</font>;<font color="#d60241">Red:Stratified</font>)</h3>
            <br>
            <script type="text/javascript">
                var w = 940;
                var h = 300;
                var pad = 20;
                var left_pad = 100;

                var xScale = d3.scale.linear().range([left_pad, w-pad]);
                var xAxis = d3.svg.axis().scale(xScale).orient("bottom");
                var yScale = d3.scale.linear().range([pad, h-pad*2]);
                var yAxis = d3.svg.axis().scale(yScale).orient("left");

                var svg = d3.select("#pca")
                            .append("svg")
                            .attr("width", w)
                            .attr("height", h);
                plot_values('pca_output.csv');
            </script>
    </div>
    <div class="container"style="background-color: #fff9dd">
        <p align="right"><a href="data:text/csv;charset=utf-8,'+escape(csv)+'" download="college.csv"><img src=".
        </p><br>
    </div>
```

The data is also visualized via MDS (Euclidian and Correlation distance) in 2D scatterplots as shown below:

Snapshot



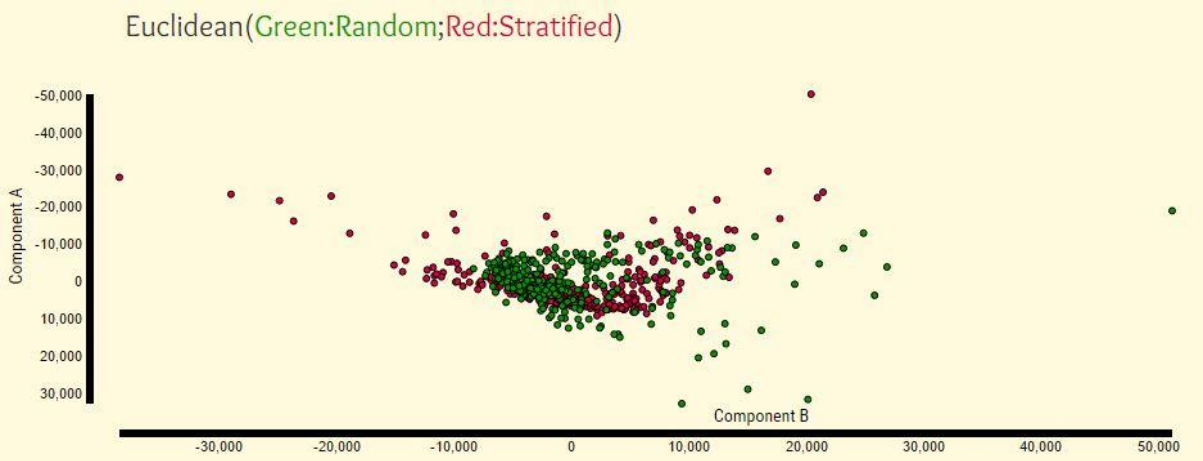
Snippet (D3 for Correlation)

```
<div class="container"><br><br>
        <h3> Correlation(<font color="#0c9603">Green:Random</font>;<font color="#d60241">Red:Stratified</font>)</h3>
        <br>
        <script type="text/javascript">
            var w = 940;
            var h = 300;
            var pad = 20;
            var left_pad = 100;

            var xScale = d3.scale.linear().range([left_pad, w-pad]);
            var xAxis = d3.svg.axis().scale(xScale).orient("bottom");
            var yScale = d3.scale.linear().range([pad, h-pad*2]);
            var yAxis = d3.svg.axis().scale(yScale).orient("left");

            var svg = d3.select("#pca")
                        .append("svg")
                        .attr("width", w)
                        .attr("height", h);
            plot_values('correlation.csv');
        </script>
</div>
```

Snapshot



Snippet (D3 for Euclidean)

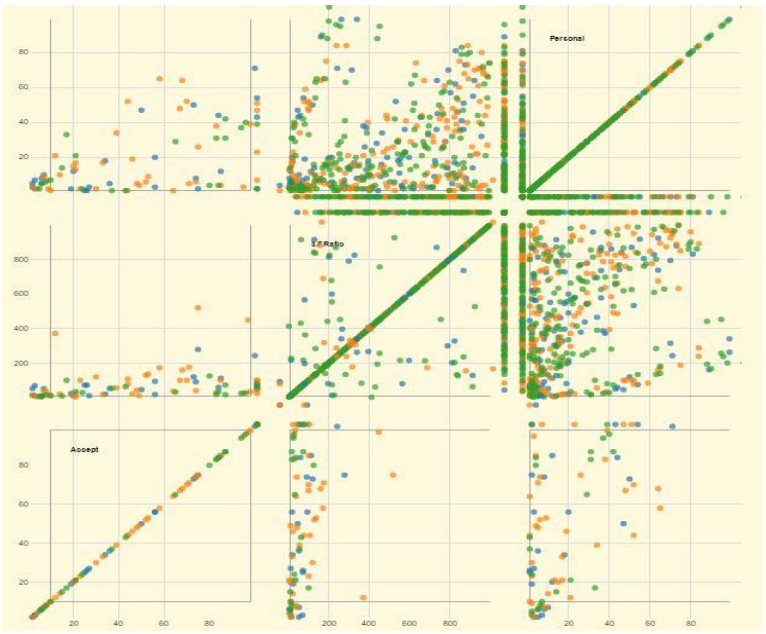
```
<div class="container"><br><br>
        <h3> Euclidean(<font color="#0c9603">Green:Random</font>;<font color="#d60241">Red:Stratified</font>)</h3>
        <br>
        <script type="text/javascript">
            var w = 940;
            var h = 300;
            var pad = 20;
            var left_pad = 100;

            var xScale = d3.scale.linear().range([left_pad, w-pad]);
            var xAxis = d3.svg.axis().scale(xScale).orient("bottom");
            var yScale = d3.scale.linear().range([pad, h-pad*2]);
            var yAxis = d3.svg.axis().scale(yScale).orient("left");

            var svg = d3.select("#pca")
                        .append("svg")
                        .attr("width", w)
                        .attr("height", h);
            plot_values('euclidean.csv');
        </script>
</div>
```

Thereafter, scatterplot matrix is created and visualized for the three highest PCA attributes (which is calculated in the previous step).

Snapshot



3. Project Deliverables:
   URL: http://localhost:port_number/
   (port_number = 4000)

   Submitted Files:

   - 7 HTML files (correlation.html, elbow.html, euclidean.html, index.html, loadings.html, matrix.html, screePlot.html)
   - main dataset (college.csv)
   - 1 python file (backend.py)
   - 7 javascript files (in js folder)
   - 10 csv output files (in data folder)
   - few images (in images folder)
   - 6 css files (in css folder)
   - 2 font files (in fonts folder)

   Other Snapshots: