```
%Name: David George
%StudentID: 251004930


%Part a)
    T = input_data();
    ClucAnalysisSingleCentroid = T(:, [4:9]);
    vnames = ClucAnalysisSingleCentroid.Properties.VariableNames;
    Eng = table2array(rmmissing(T(:, [4:9])));

    %Inverse-varience weights should be used for the PCA
    % As the varibles are in differnt units.

    [wcoeff,score,latent,tsquared,explained] = pca(Eng, ...

  'VariableWeights','variance');
        pc1 = score(:,1);
        pc2 = score(:,2);

        %Visuauallly Showing results of PCA
        figure
        scatter(pc1, pc2, 10, 'MarkerFaceColor','blue')
        alpha(0.2)
        xlabel('1st Principal Component')
        ylabel('2nd Principal Component')
        grid()
        set(gca,'FontSize',20);


    var_by_2_first = sum(explained(1:2));
    fprintf("The first 2 principal components explain %d prct of the
 variance.", var_by_2_first);
    %The first two PC account for 92% of the vareince, this close to
    %100%,this is enough.

%Part b)
    figure
    coef_norm = inv(diag(std(Eng)))* wcoeff;
    biplot(coef_norm(:,1:2),'Scores',score(:,1:2), ...
    'Varlabels',vnames);
    %Interpretation:
    % the direction and length of the vector indicate how each
 variable
    % contributes to the two principal components in the plot.

    % The largest coefficients in the first principal component
    % correspond to the variables `Weight` and `HorsePower`.

    % The second principal component, on the vertical axis,
    % has POSITIVE coefficients for the variables acceleratio, weight,
 cylinders,
    % and NEGATIVE coefficients for HorsePower and MPG.
```

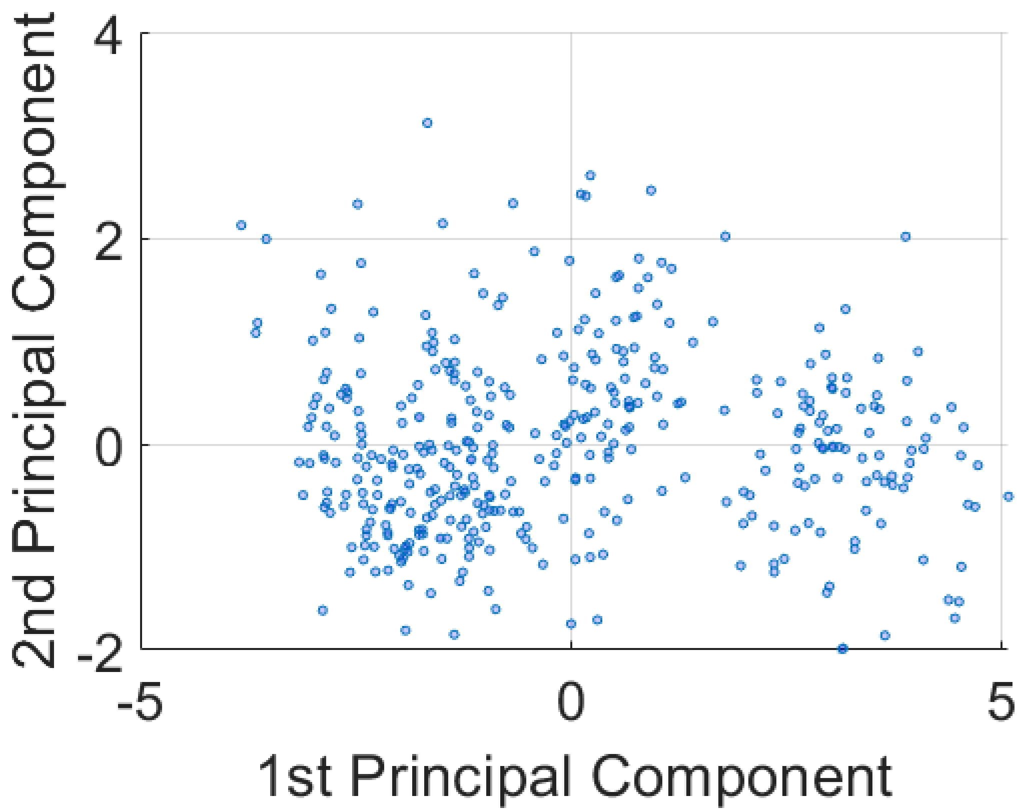
```
    % This indicates that the second component distinguishes among
 cars
    % that have high values for the first set of variables (Weight,
 Acceleration ,...)
    % and low for the second (MPG, Horsepower,...)
    % and cars that have the opposite.

%Part c)
    %Clasical multi-dimesnional scaling (MDS)
    D = squareform(pdist(Eng));
    [tmp ev] = cmdscale(D);
    %Values after classical multi-dimensional scaling
    % It returns 2 outputs:
    % 1) the matrix tmp of the coordinates in the lower dimension
 space
    %    that tries to preserve the original distances
    % 2) the eigenvalues (ev)  of the spectral decomposition used
 inside the MDS, that
    %    indicates how large the lower dimension space should be.


 %Part D)
    MDS  = cmdscale(D, 2);
    figure

     %MDS with labels for manufacturer and year number
     gscatter(MDS(:,1), MDS(:,2),(rmmissing(T).Mfg));
      title("MDS with Manufacturer label");

     figure
      gscatter(MDS(:,1), MDS(:,2),(rmmissing(T).Model_Year));

      title("MDS with Model Year label");


%Part E) Clustering Analysis
    % We are choosing, a priori, 3 clusters in total.

    [idxCluster, centroids] = kmeans(MDS,3);



    figure
    hold on

    % Color the data points wih their respective cluster:
    scatter(MDS(:,1), MDS(:,2),10, idxCluster,'Filled')
    title("k-means Clustering",'FontSize',10)
    alpha(0.5); grid()
```

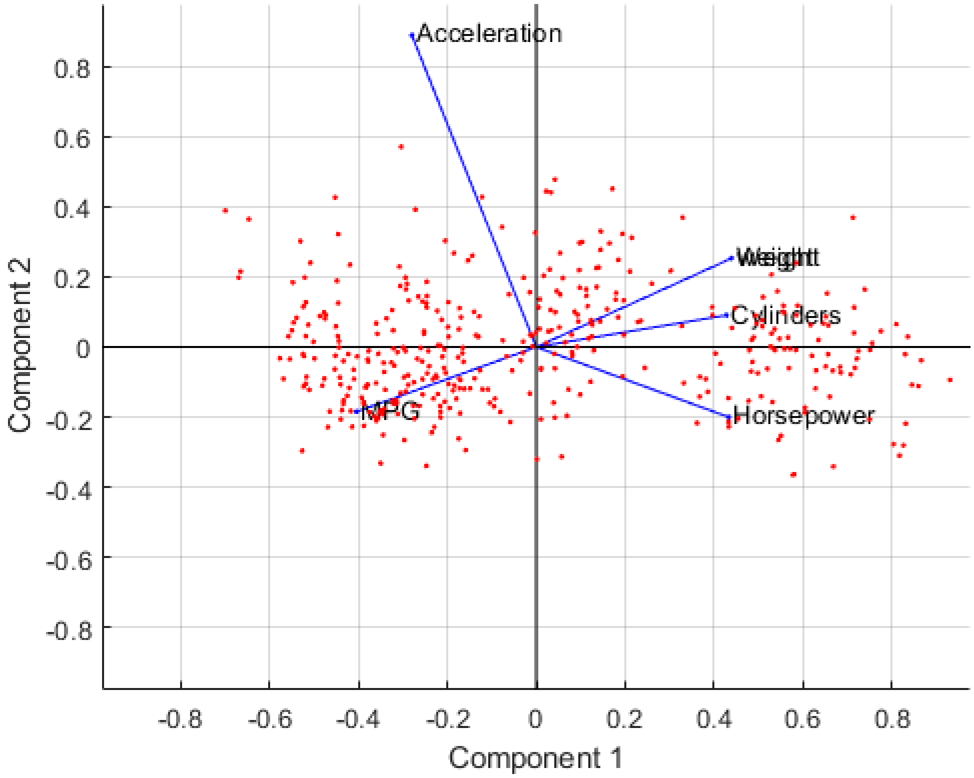
```
Model = string(rmmissing(T).Mfg);

%The following code will be used to label 5 points from 5
clusters
%Iterate over the cluster Varible and seperate the specifc
indexis
%into the corresponsign group array
%Generate 5 random numbers, to choose five random indexes for
each
%group
% Use this to annotate

indexOne = [];
indexTwo = [];
indexThree = [];
CountOne = 1;
CountTwo = 1;
CountThree = 1;
for idx = 1:numel(idxCluster)

    if idxCluster(idx) == 1

        indexOne(CountOne) = idx;
        CountOne = CountOne + 1;
    end

     if idxCluster(idx) == 2
         indexTwo(CountTwo) = idx;
        CountTwo = CountTwo + 1;
     end

     if idxCluster(idx) == 3
         indexThree(CountThree) = idx;
        CountThree = CountThree + 1;
    end


    %text(MDS(random(idx),1),MDS(random(idx),2),Model{idx} );
end
  random = randi([1,80],1,5);
    while(length(random) ~= length(unique(random)))

         random = randi([1,80],1,5);

    end


for idx = 1:5
    text(MDS(
 indexOne(random(idx)),1),MDS(indexOne(random(idx)),2),Model{idx});

text(MDS(indexTwo(random(idx)),1),MDS(indexTwo(random(idx)),2),Model{idx});
```

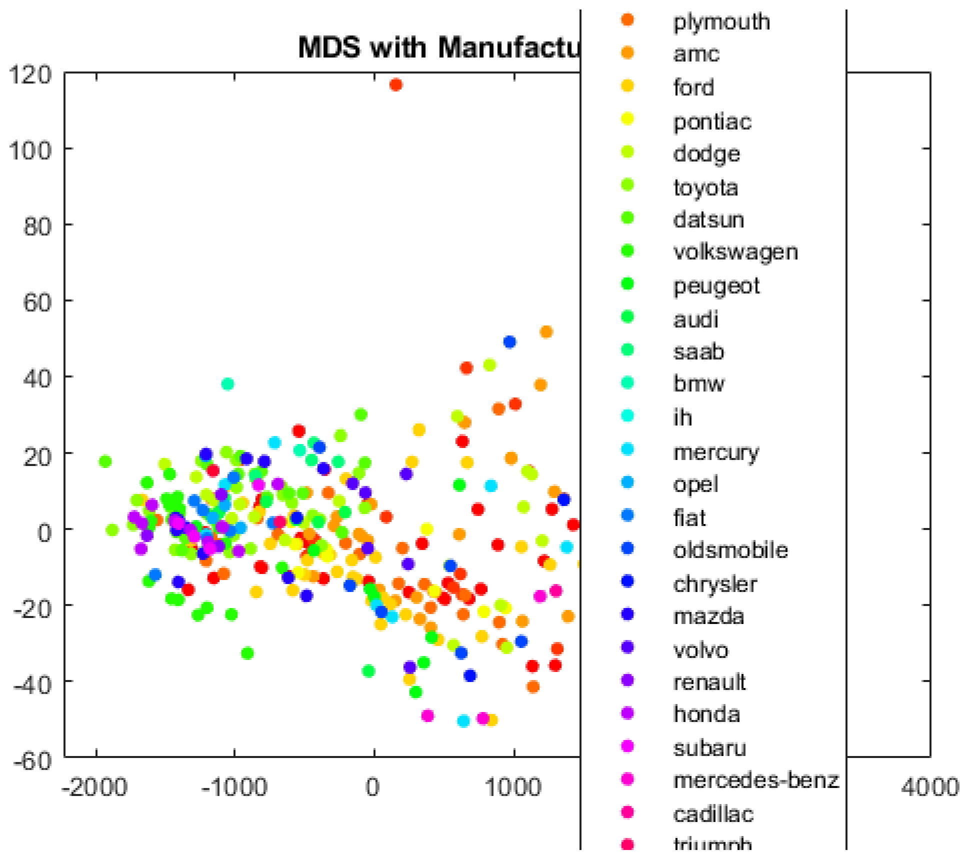
```
text(MDS(indexThree(random(idx)),1),MDS(indexThree(random(idx)),2),Model{idx});
     end

    % text(MDS(random(idx),1),MDS(random(idx),2),Model{idx} );


 %Part F) Clustering analysis using two


   %Single Clustering
   ClusterAnalsys = linkage(MDS,"single");

   figure
   subplot(2, 2,1);
   dendrogram(ClusterAnalsys,0, 'Orientation','left')
   grid()
   title("Single");

   ClucAnalysisSingleCentroid = cluster(ClusterAnalsys,'Maxclust',3);
   subplot(2, 2,2);
   scatter(MDS(:,1), MDS(:,2), 10,
ClucAnalysisSingleCentroid, 'filled')
   title("Single");

   %Centroid Clustering
   ClusterAnalsys = linkage(MDS,"centroid");
   subplot(2, 2,3);
   dendrogram(ClusterAnalsys,0, 'Orientation','left')
   grid()
   title("Centroid");

   ClucAnalysisSingleCentroid = cluster(ClusterAnalsys,'Maxclust',3);
   subplot(2, 2,4);
   scatter(MDS(:,1), MDS(:,2), 10,
ClucAnalysisSingleCentroid, 'filled')
   title("Centroid");

       %No the clustering is not similar, and this can be atributed
to the
       %specfic methods themselves. They have differnt min max
distances, thus the linkages
       %between the two are differnt, leading to clustering which is
not
       %similar.


function T = input_data()

%Function to covnert types in data table to the correct units
T = readtable("cars.csv");
T.Model_Year = double(T.Model_Year);
T.Acceleration =  double(T.Acceleration);
T.Cylinders = double(T.Cylinders);
```

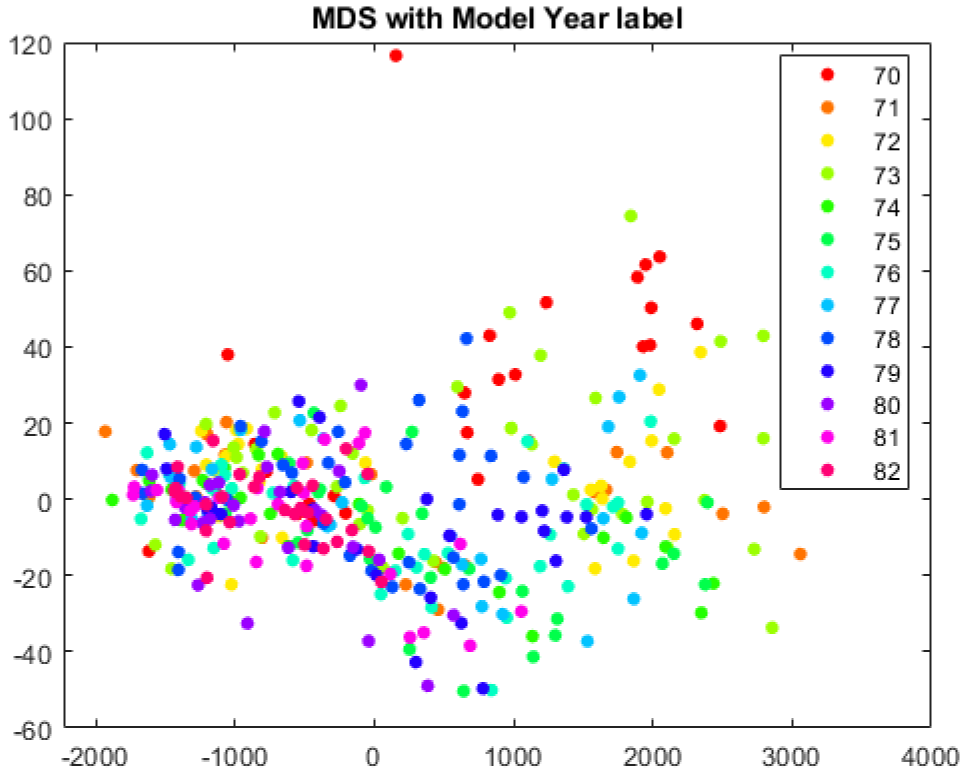
```
T.Horsepower  = double(T.Horsepower);
T.MPG = double(T.MPG);
T.weight = double(T.Weight);
end
```

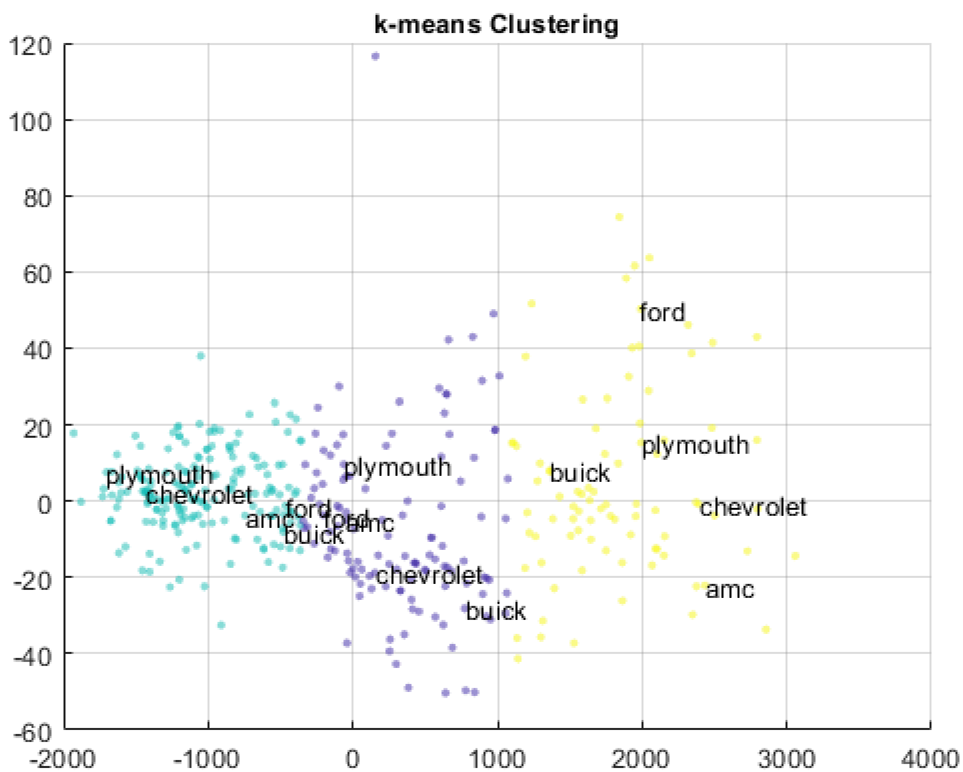
```
Warning: Columns of X are linearly dependent to within machine
 precision.
Using only the first 5 components to compute TSQUARED.
The first 2 principal components explain 9.229243e+01 prct of the
 variance.Warning: Non-monotonic cluster tree -- the centroid linkage
 is probably not
appropriate.
```

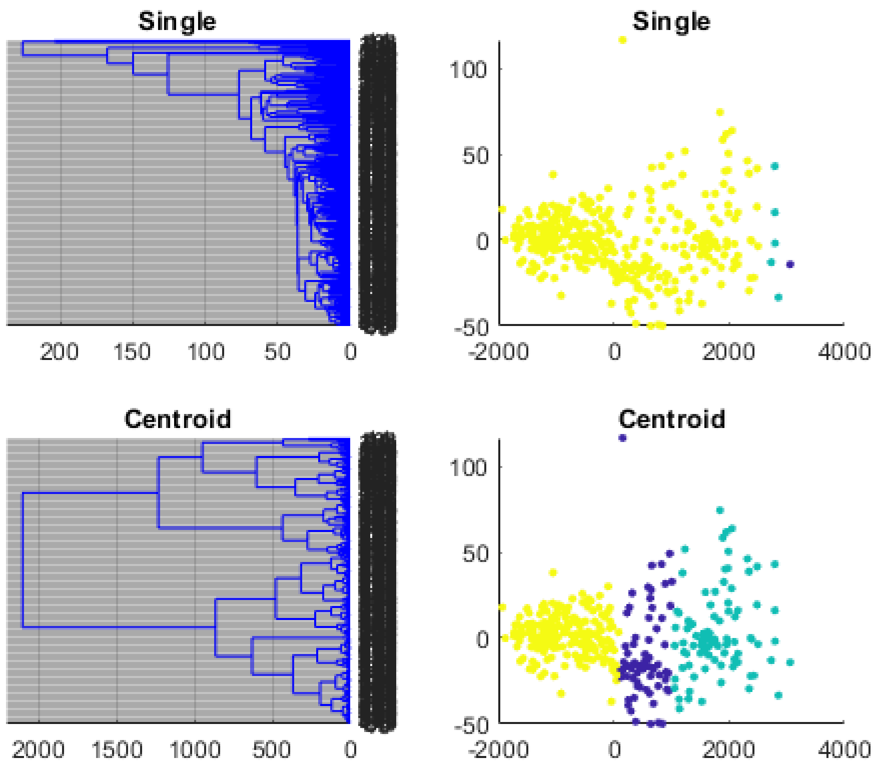



Acceleration
Weight
Cylinders
Horsepower
MPG
Component 2
Component 1


MDS with Manufactu
plymouth
amc
ford
pontiac
dodge
toyota
datsun
volkswagen
peugeot
audi
saab
bmw
ih
mercury
opel
fiat
oldsmobile
chrysler
mazda
volvo
renault
honda
subaru
mercedes-benz
cadillac


MDS with Model Year label
70
71
72
73
74
75
76
77
78
79
80
81
82


k-means Clustering
ford
plymouth
buick
chevrolet
amc
plymouth
chevrolet
amc
ford
buick
ford
amc
plymouth
chevrolet
buick



Published with MATLAB® R2019b