Weather Project

This is a project applying machine learning method to predict whether in Helsinki, Finland. The dataset is a set of measurements such as temperature, atmospheric pressure, collected by the weather station 2978 in Helsinki from January 2010 to December 2018. The data have adapted from a public dataset avaliable at Aalto University. The original data has been resampled by day and simplified as following:

– “T" is the air temperature (in degrees Celsius)

– “Po" is the atmospheric pressure at weather station level (in millimeters of mercury)

– “P" is the atmospheric pressure reduced to mean sea level (in millimeters of mercury.

– “Ff" is the mean wind speed (in meters per second).

– “Tn" is the minimum air temperature over the past day (in degrees Celsius,). – “Tx" is the maximum air temperature, over the past day (in degrees Celsius) – “W" is the horizontal visibility, in km.

– “Td" is the dewpoint temperature (in degrees Celsius).

– “U" is the relative humidity (in percentage)

For the all the attributes (excluding the OBSERVED variable), the mean and the variance values for each day are provided. For example, “T mu" indicates the mean of the air temperature, and “T var" indicates the variance of the air temperature, which constitutes for 17 independent variables and 2 target variables.

✓ the first attribute “datetime", is an index indicating the date in the format YYYY-MM-DD.

✓ all the other attributes are continuous

✓ the 18th attribute (target attribute) – variable OBSERVED – is a binary variable, where 0 (not dry) indicates that the amount of precipitation was more than 0.3 millimeters, and 1 (dry) indicates that the amount of precipitation was less than 0.3 millimeters.

✓ the 19th attribute (target attribute) is the relative mean humidity (variable U_mu) and represents a target continuous variable

This project will execute the whole process of data mining and build models for prediction in Matlab programming language. The first model is based on regression method to predict the relative mean of humidity based on set of predictors matrix. The second model applied classification and ROC curve method to anticipate whether the weather is dry or not on a specific day.

Data exploration and preparation

Input data

```
rng('default');
T=readtable('weatherHelsinki.xlsx')
```

```
T = 3202×19 table
```

...

	datetime	T_mu	Po_mu	P_mu	Ff_mu	Tn_mu	Tx_mu	W_mu
1	01-Jan-2010	-12.8500	755.5250	755.9000	4.2500	-14	-10.5000	19.3875

	datetime	T_mu	Po_mu	P_mu	Ff_mu	Tn_mu	Tx_mu	W_mu
2	02-Jan-2010	-17.3875	762.0250	762.4000	3.6250	-18.6500	-15.2500	27.8750
3	03-Jan-2010	-12.6375	757.7000	758.0875	2.5000	-21.8000	-10.4500	19.3375
4	04-Jan-2010	-3.9286	750.6500	751.0167	3.4286	-7.7500	-2.4000	29.1143
5	05-Jan-2010	-9.5000	752.6125	752.9750	3.7500	-11.3500	-4.1500	34.1250
6	06-Jan-2010	-11.1750	760.5375	760.9000	1.7500	-13.4500	-9.9000	17.4625
7	07-Jan-2010	-8.6500	757.6500	758.0125	4.7500	-9.9500	-7.4000	6.9875
8	08-Jan-2010	-16.1125	765.1125	765.5125	4	-17	-12.0500	21.2500
9	09-Jan-2010	-15.4250	775.0625	775.4625	2.1250	-17.9000	-12.3500	17.8750
10	10-Jan-2010	-12.5625	777.2750	777.6625	1.7500	-15.5500	-12.4500	6.6625
11	11-Jan-2010	-10.2250	772.3625	772.7375	2.7500	-13.8500	-8.3000	15
12	12-Jan-2010	-3.2125	767.8500	768.2375	3.2500	-5.0500	-2.8000	8.8750
13	13-Jan-2010	-4.0375	767.1625	767.5250	1.8750	-4.3500	-3.1500	8.6750
14	14-Jan-2010	-4.6875	768.5125	768.8625	2.8750	-5.2500	-4.4000	14.6500
15	15-Jan-2010	-3.9250	773.3500	773.6750	2.1250	-4.7500	-3.3500	21.8750
16	16-Jan-2010	-3.9375	779.5875	779.9250	2.5000	-4.7000	-2.9000	27.5000
17	17-Jan-2010	-6.5500	778.8625	779.2625	3.3750	-7.8000	-4.9500	29.7500
18	18-Jan-2010	-7.3375	776.8875	777.2250	4.3750	-9	-5.5500	27.1250
19	19-Jan-2010	-7.7500	777.0375	777.4250	3	-8.8000	-7	29.5000
20	20-Jan-2010	-10.5500	780.2750	780.6625	3.3750	-11.6500	-7.5000	28.5000
21	21-Jan-2010	-14.8857	784.6286	785.0429	3.2857	-16.3000	-12.9000	11
22	22-Jan-2010	-11.9375	784.9250	785.3000	3	-13.6500	-9.7000	13.3750
23	23-Jan-2010	-15.1375	784.0750	784.4625	2.5000	-16.2000	-12.7000	15.5000
24	24-Jan-2010	-16.5429	782.4429	782.8143	1.5714	-22.5500	-13.8000	6.3143
25	25-Jan-2010	-9.2250	780.9500	781.3250	2.2500	-11.6500	-7.7000	16.5625
26	26-Jan-2010	-16.4571	782.4857	782.8714	2.7143	-17.8000	-11.7000	22.4286
27	27-Jan-2010	-17.5375	767.2000	767.5750	4	-20.7000	-16.8500	14.5000
28	28-Jan-2010	-16.0125	748.1375	748.5125	6.6250	-18	-15	4.0625
29	29-Jan-2010	-12.0750	741.4500	741.8000	4	-14.4500	-11.2500	3.3500
30	31-Jan-2010	-2.8125	746.4250	746.7750	4.3750	-5.8000	-1.4500	5.7875
31	01-Feb-2010	-5.5750	754.4125	754.7125	3	-9.5500	-3.7500	8.0750
32	02-Feb-2010	-5.7750	749.6250	749.9875	2.7500	-7	-4.3000	12.5875
33	03-Feb-2010	-4.1857	758.5143	758.8714	3.1429	-5.6000	-2.4500	37.7143
34	04-Feb-2010	-3.7125	767.0500	767.4250	4	-4.8000	-3	14.1250

	datetime	T_mu	Po_mu	P_mu	Ff_mu	Tn_mu	Tx_mu	W_mu
35	05-Feb-2010	-2.9375	772.3125	772.6875	3.2500	-3.6500	-2.4000	7.5625
36	06-Feb-2010	-6.0125	772.8875	773.2375	2.3750	-7.0500	-3.9500	8.1125
37	07-Feb-2010	-4.1625	766.7125	767.0875	3.2500	-6.0500	-3.9000	4.9125
38	08-Feb-2010	-2.9250	760.4000	760.7500	3.7500	-3.6000	-1.5000	6.1875
39	09-Feb-2010	-8.0375	761.3625	761.7500	3.2500	-9.1000	-5	7.6000
40	10-Feb-2010	-11.9125	765.1375	765.5250	2	-13.3500	-9.6500	7.7875
41	11-Feb-2010	-10.9250	767.0875	767.4625	1.5000	-12.7500	-9.5500	8.6250
42	12-Feb-2010	-8.8375	768.6375	769.0125	4.1250	-11.8500	-8.3000	7.0250
43	13-Feb-2010	-8.3625	765.4375	765.7625	2.8750	-9.1500	-7.2500	19.3750
44	14-Feb-2010	-10.4500	762.1000	762.4625	1.7500	-16.7000	-7.3500	11.3750
45	15-Feb-2010	-10.0125	758.8500	759.2250	2.7500	-14.5000	-7.5500	12.7500
46	16-Feb-2010	-8.2125	758.7750	759.1500	1.7500	-9.0500	-7.3000	10.5250
47	17-Feb-2010	-9.1375	760.8625	761.2125	4.2500	-9.9000	-7.3000	13.6125
48	18-Feb-2010	-12.8375	758.2125	758.6000	2.3750	-14.1000	-10.8500	16.8250
49	19-Feb-2010	-14.4857	755.6000	755.9857	5.8571	-16	-13.5000	15.1571
50	20-Feb-2010	-18.3250	755.8125	756.1875	6.1250	-21	-15.1000	35.2500
51	21-Feb-2010	-16.6500	749.9500	750.3125	6	-19.1500	-16.2000	7.0500
52	22-Feb-2010	-11.6625	749.2125	749.6000	4.7500	-14.0500	-9.0500	10.1000
53	23-Feb-2010	-10.2143	745.6143	745.9857	3.8571	-15.1000	-9.1000	6.1857
54	24-Feb-2010	-12.4125	753.3250	753.6875	4	-15.6500	-8.5500	21.2375
55	25-Feb-2010	-6.2375	759.1000	759.4375	2.7500	-10.1500	-4.5500	14.1000
56	26-Feb-2010	-3.0500	753.6750	754	2.7500	-6.5000	-2.3000	19
57	28-Feb-2010	1.0250	750	750.3250	3.3750	0.4500	1.2500	4
58	01-Mar-2010	0.3571	744.7429	745.1000	4.1429	0.3500	1.1000	5.0714
59	02-Mar-2010	0.3250	741.8125	742.1500	3	0.1500	1.1500	15.3375
60	03-Mar-2010	-3.9250	746.9875	747.3000	3.7500	-4.5500	-1.3000	12.3625
61	04-Mar-2010	-3.2286	752.5429	752.9143	3.7143	-4.9000	-2.1000	23
62	05-Mar-2010	-6.1875	760.0500	760.4375	4	-8.3500	-3.5500	40.6250
63	06-Mar-2010	-6.4500	765.2375	765.5875	3.2500	-11.4500	-4.3000	39.1750
64	07-Mar-2010	-7.2625	771.8250	772.2250	2.5000	-10.2000	-3.8000	45
65	08-Mar-2010	-3.0500	765.3875	765.7500	5.2500	-7.7500	-2.5500	39.8750
66	09-Mar-2010	-2.2625	763.2750	763.6250	4.2500	-7.2500	0.8500	43.7500
67	10-Mar-2010	2.3125	759.5000	759.8625	5	-0.9500	3.4000	48.1250

	datetime	T_mu	Po_mu	P_mu	Ff_mu	Tn_mu	Tx_mu	W_mu
68	11-Mar-2010	0.0250	755.1000	755.4750	5.3750	-0.7000	2.6500	20.7500
69	12-Mar-2010	-1.7750	751.4375	751.7875	3.1250	-3.5000	0.6000	28.0500
70	13-Mar-2010	-3.5750	748.3625	748.7000	2.6250	-7.5000	-1.4500	43.5000
71	14-Mar-2010	-5.8750	745.5375	745.9125	4.1250	-7.5000	-2.2500	45.6250
72	15-Mar-2010	-7.4125	750.6875	751.0500	3.3750	-11.8000	-4.6500	37.0625
73	16-Mar-2010	-7.6875	755.6125	755.9750	2.1250	-11.9000	-5.1000	33.6250
74	17-Mar-2010	-7.3625	759.1250	759.5000	2.1250	-13.5500	-4.7000	23
75	18-Mar-2010	-3.1750	758.9375	759.2750	3.5000	-7.4000	-1.7000	22.4500
76	19-Mar-2010	1.4750	752.1833	752.5167	2.7500	-0.1500	2.7000	9.1375
77	20-Mar-2010	1.4000	745.2625	745.6125	3.8750	0.9000	1.8000	6.4250
78	21-Mar-2010	0.2500	748.4714	748.8000	3.1250	-0.9500	1.2000	19.2250
79	22-Mar-2010	-1.6875	756.4750	756.8250	5.2500	-5.4000	0.7000	29.7875
80	23-Mar-2010	-1.5875	758.1375	758.4750	3.1250	-5.6500	0.5500	30.3125
81	24-Mar-2010	-0.5750	761.9625	762.3000	2.1250	-4	2.1500	35.3750
82	25-Mar-2010	-0.3000	767.8500	768.2125	2.1250	-2.0500	0.6000	35.7500
83	26-Mar-2010	1.4625	759.4625	759.8375	3	-0.4000	2.6500	27.2875
84	29-Mar-2010	1.2750	750.6000	750.9625	2.7500	0.4500	1.9000	14.5875
85	30-Mar-2010	1.6625	753.9000	754.2375	3.1250	0.7500	2.8500	10.4125
86	31-Mar-2010	1.6000	758.2000	758.5625	4.6250	-0.0500	2.9500	13.7250
87	01-Apr-2010	4.9750	756.9750	757.3125	4.2500	3.0500	5.7000	12.8750
88	02-Apr-2010	3.3625	761.9250	762.2625	1.5000	2.1500	6.1500	10.6250
89	05-Apr-2010	2.9375	765.3750	765.7125	3	1.2500	4.1000	15.8250
90	07-Apr-2010	1.2571	769.5857	769.9571	2.5714	0.3000	3.5000	24.9286
91	08-Apr-2010	2.5750	769.4875	769.8375	3.3750	0.6000	3.3000	18.2000
92	09-Apr-2010	3.9250	767.3250	767.7000	2.6250	3	5.0500	26.2500
93	10-Apr-2010	4.8000	768.5000	768.8875	2.6250	2.5500	6.7500	25
94	11-Apr-2010	5.7250	772.5750	772.9625	3.3750	3.5000	7.4500	22.3750
95	12-Apr-2010	5.9625	771.4500	771.8000	3.6250	4	8.5000	40
96	13-Apr-2010	6.2000	764.7500	765.1000	2	3.8000	8.5000	19.8750
97	14-Apr-2010	7.0875	763.6250	763.9500	3	4.2000	10.3000	36.5000
98	15-Apr-2010	4.1500	761.7500	762.1000	2.1250	1.9000	7.1000	36.1250
99	17-Apr-2010	3.4375	753.6375	753.9625	3.5000	1.1500	5.8500	43
100	18-Apr-2010	4.6125	748.3000	748.6375	5.5000	2.5500	6.3500	31.2500

⋮

Summarise the table

```
summary(T)
```

```
Variables:

    datetime: 3202×1 datetime

        Values:

            Min       01-Jan-2010 00:00:00
            Median    21-Jul-2014 12:00:00
            Max       31-Dec-2018 00:00:00

    T_mu: 3202×1 double

        Values:

            Min       -22.675
            Median        6.4
            Max        26.146

    Po_mu: 3202×1 double

        Values:

            Min         729.5
            Median     759.21
            Max        790.42

    P_mu: 3202×1 double

        Values:

            Min        729.85
            Median     759.54
            Max        790.81

    Ff_mu: 3202×1 double

        Values:

            Min        1.1667
            Median      3.375
            Max        10.792

    Tn_mu: 3202×1 double

        Values:

            Min        -26.2
            Median       4.7
            Max        23.65

    Tx_mu: 3202×1 double

        Values:

            Min          -21
            Median     8.125
```

```
        Max        28.75

W_mu: 3202×1 double

    Values:

        Min        0.625
        Median      23.5
        Max           50

Td_mu: 3202×1 double

    Values:

        Min       -25.913
        Median     3.0625
        Max        21.462

T_var: 3202×1 double

    Values:

        Min       0.014286
        Median      2.5911
        Max         66.571

Po_var: 3202×1 double

    Values:

        Min         0.005
        Median      1.444
        Max        89.849

P_var: 3202×1 double

    Values:

        Min         0.005
        Median      1.457
        Max        89.808

Ff_var: 3202×1 double

    Values:

        Min             0
        Median    0.95471
        Max        17.571

Tn_var: 3202×1 double

    Values:

        Min             0
        Median       0.98
        Max        93.845

Tx_var: 3202×1 double

    Values:

        Min             0
        Median          2
```

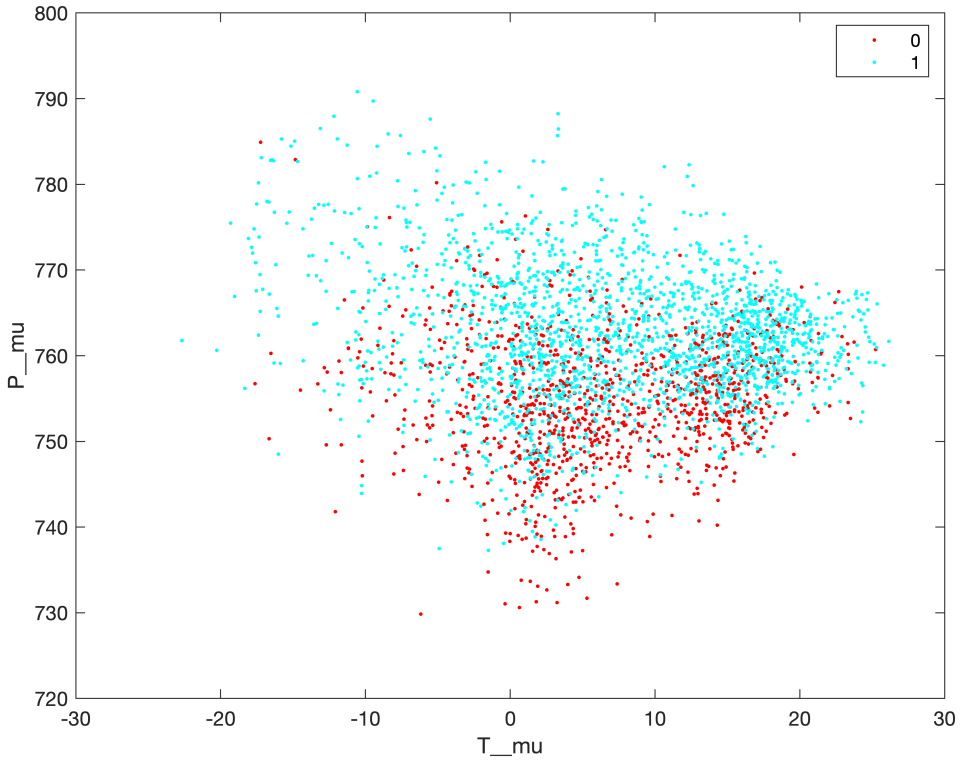
```
        Max        105.12

    W_var: 3202×1 double

        Values:

            Min             0
            Median     41.312
            Max        645.76

    Td_var: 3202×1 double

        Values:

            Min       0.013333
            Median      1.2886
            Max          77.45

    OBSERVED: 3202×1 double

        Values:

            Min          0
            Median       1
            Max          1

    U_mu: 3202×1 double

        Values:

            Min        34.286
            Median      83.25
            Max        99.875
```

Data Visualization

Visualise relationship between attributes to better understand data.

First the relationship between the air temperature (T) and atmospheric pressure (P) with the OBSERVED variable is checked in the following graph. .

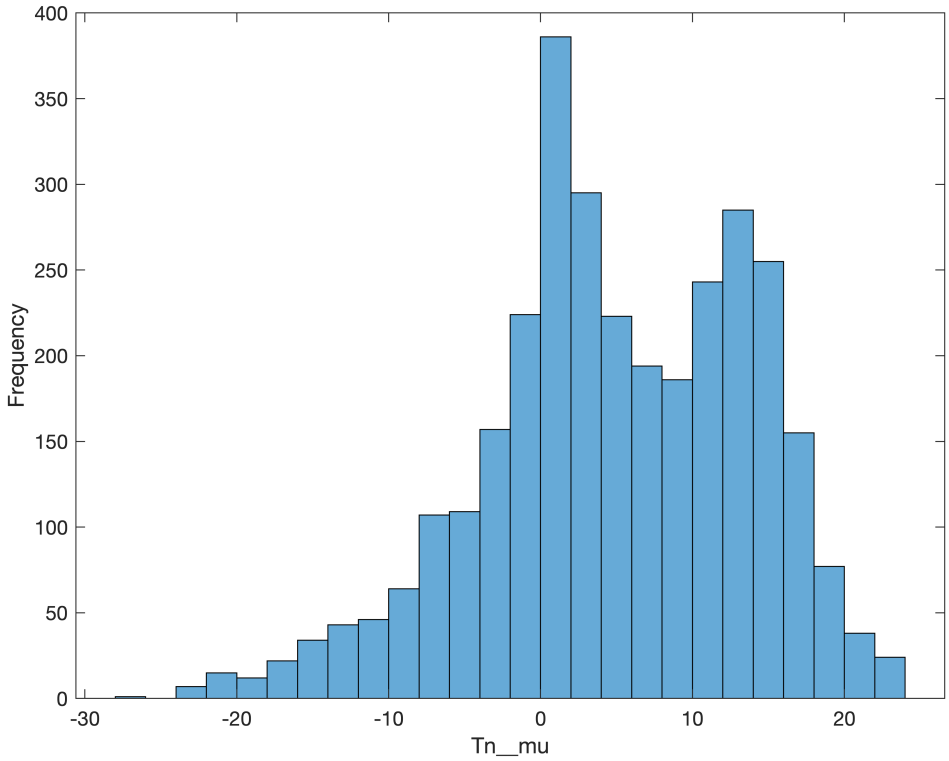
```
gscatter(T.T_mu,T.P_mu,T.OBSERVED)
xlabel('T__mu');
ylabel('P__mu');
```



The 2D draft shows that days with high pressure and low temperature is usually dry. In contrast, high temperature and low pressure tend to have not dry whather.

Check the minimum air temperature over the past day (in degree Celsius)

```
histogram(T.Tn_mu)
xlabel('Tn__mu')
ylabel('Frequency')
```



The distribution is right-skewed. The most frequent minimum temperature is around 0 degree Celsius.

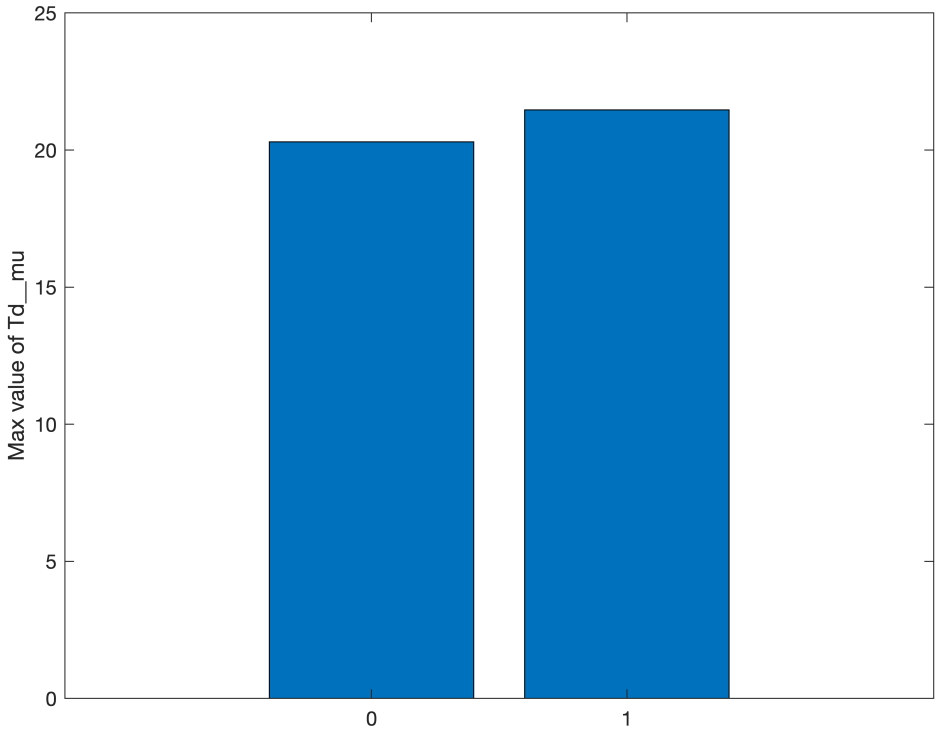
The dataset is split into 2 groups: dry and not dry.

```
[grpObs,ObsVals] = findgroups(T.OBSERVED);
maxTd_mu = splitapply(@max,T.Td_mu,grpObs)
```

```
maxTd_mu = 2×1
   20.3000
   21.4625
```

The average maximum dew point temperature of "not dry" group (0) is 20.3 degrees Celsius and 21.46 degrees for "dry" group (1).

```
bar(maxTd_mu)
xticklabels(ObsVals)
ylabel("Max value of Td__mu")
```



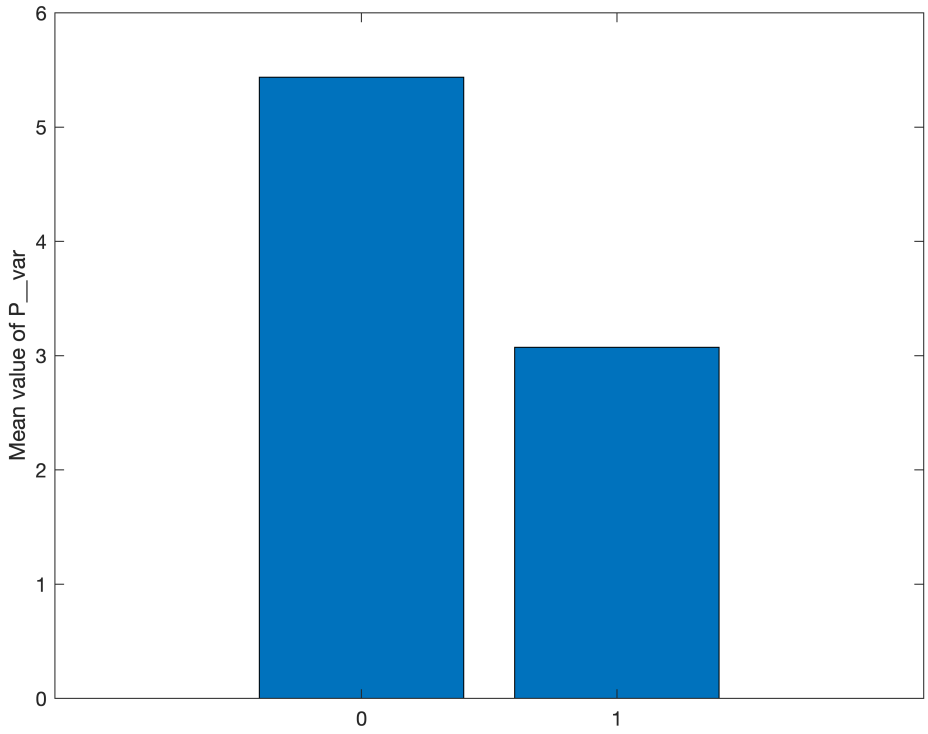
The same process is applied to atmospheric pressure:

```
meanP_var = splitapply(@mean,T.P_var,grpObs)
```

```
meanP_var = 2×1
    5.4376
    3.0735
```

Mean atmospheric pressure of group 0 (not dry) is much higher than group 1 (dry).

```
bar(meanP_var)
xticklabels(ObsVals)
ylabel("Mean value of P__var")
```



Separate datetime to year, month and day variables.

```
[T.YY,T.MM,T.DD] = ymd(T.datetime);
T.datetime = [];
```

The dataset is split into independent variables and dependent variables for the next part of model building. Y1 variable is the target vavriable for binary logistic regression and Y2 variable is for multivariate linear regression. The other 17 variables formed a matrix X as indepedent variables.

```
Y1 = T.OBSERVED;
Y2 = T.U_mu;

X = table2array(T);
X(:,17:18)=[];
```

Principal Component Analysis (PCA) and clustering

PCA Analysis:

Pairwise correlation of all variables:

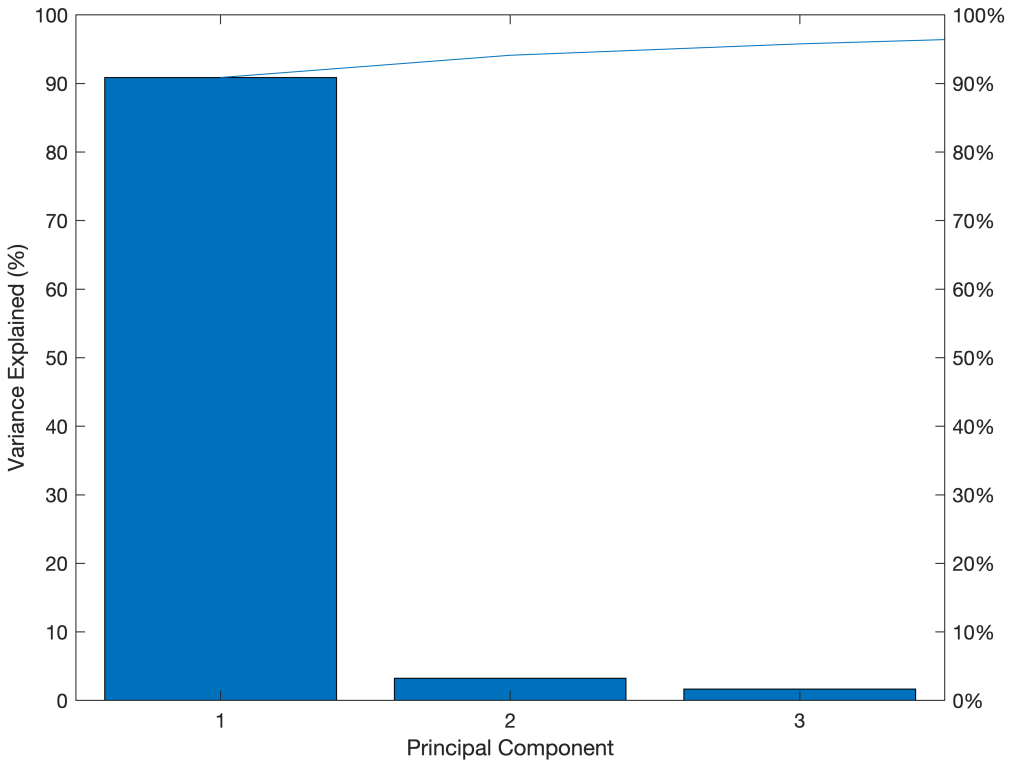
```
C = corr(X)
```

```
C = 19×19
    1.0000   -0.0752   -0.0765   -0.1474    0.9927    0.9945    0.1805    0.9565 ···
   -0.0752    1.0000    1.0000   -0.3173   -0.1021   -0.0575    0.0656   -0.1513
   -0.0765    1.0000    1.0000   -0.3171   -0.1035   -0.0588    0.0653   -0.1526
```

```
   -0.1474   -0.3173   -0.3171    1.0000   -0.1227   -0.1670    0.0895   -0.1635
    0.9927   -0.1021   -0.1035   -0.1227    1.0000    0.9815    0.1517    0.9627
    0.9945   -0.0575   -0.0588   -0.1670    0.9815    1.0000    0.2085    0.9366
    0.1805    0.0656    0.0653    0.0895    0.1517    0.2085    1.0000    0.0210
    0.9565   -0.1513   -0.1526   -0.1635    0.9627    0.9366    0.0210    1.0000
    0.1386    0.2141    0.2140   -0.2422    0.0495    0.1938    0.1987    0.0045
   -0.1909   -0.2343   -0.2338    0.3516   -0.1835   -0.1919    0.0222   -0.1683
     ⋮
```

Based on the table above, we can see that some variables highly correlated to others such as T, Tn, Tx variables have the correlation of 0.99. Therefore, we have to implement PCA:

```
[coeff,score,latent,~,explained]= pca(X);
figure
    pareto(explained)
    xlabel('Principal Component')
    ylabel('Variance Explained (%)')
```



Based on the PCA analysis, the first 2 principal components explained more than 94% of variance. So we choose them to perform the PCA analysis.

K-means Clustering:

Testing with different clusters:

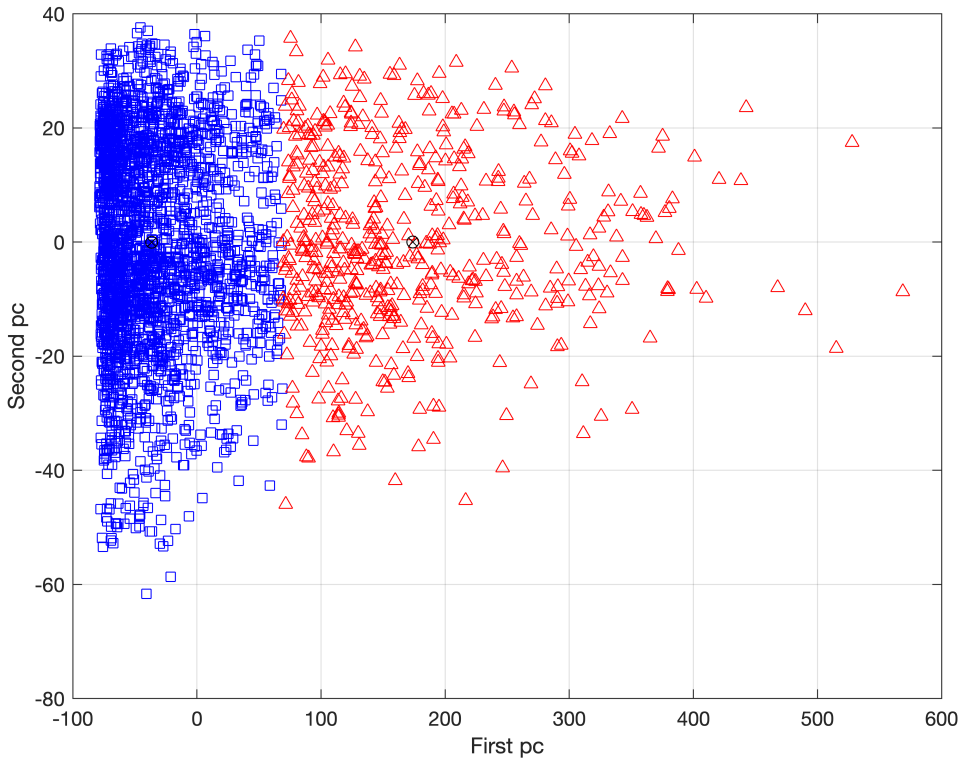
```
T1 = score(:,1:2);
```

2 clusters:

```
[cidx2,cmeans2] = kmeans(T1,2,'replicates',10,'display','final');
```

```
Replicate 1, 6 iterations, total sum of distances = 8.90381e+06.
Replicate 2, 7 iterations, total sum of distances = 8.90382e+06.
Replicate 3, 7 iterations, total sum of distances = 8.90382e+06.
Replicate 4, 8 iterations, total sum of distances = 8.90381e+06.
Replicate 5, 8 iterations, total sum of distances = 8.90381e+06.
Replicate 6, 6 iterations, total sum of distances = 8.90382e+06.
Replicate 7, 6 iterations, total sum of distances = 8.90381e+06.
Replicate 8, 7 iterations, total sum of distances = 8.90381e+06.
Replicate 9, 7 iterations, total sum of distances = 8.90381e+06.
Replicate 10, 4 iterations, total sum of distances = 8.90382e+06.
Best total sum of distances = 8.90381e+06
```

```
ptsymb = {'bs','r^','md','go','c+'};
for i = 1:2
    clust = find(cidx2==i);
    plot(T1(clust,1),T1(clust,2),ptsymb{i});
    hold on
end
plot(cmeans2(:,1),cmeans2(:,2),'ko');
plot(cmeans2(:,1),cmeans2(:,2),'kx');
hold off
xlabel('First pc');
ylabel('Second pc');
grid on
```

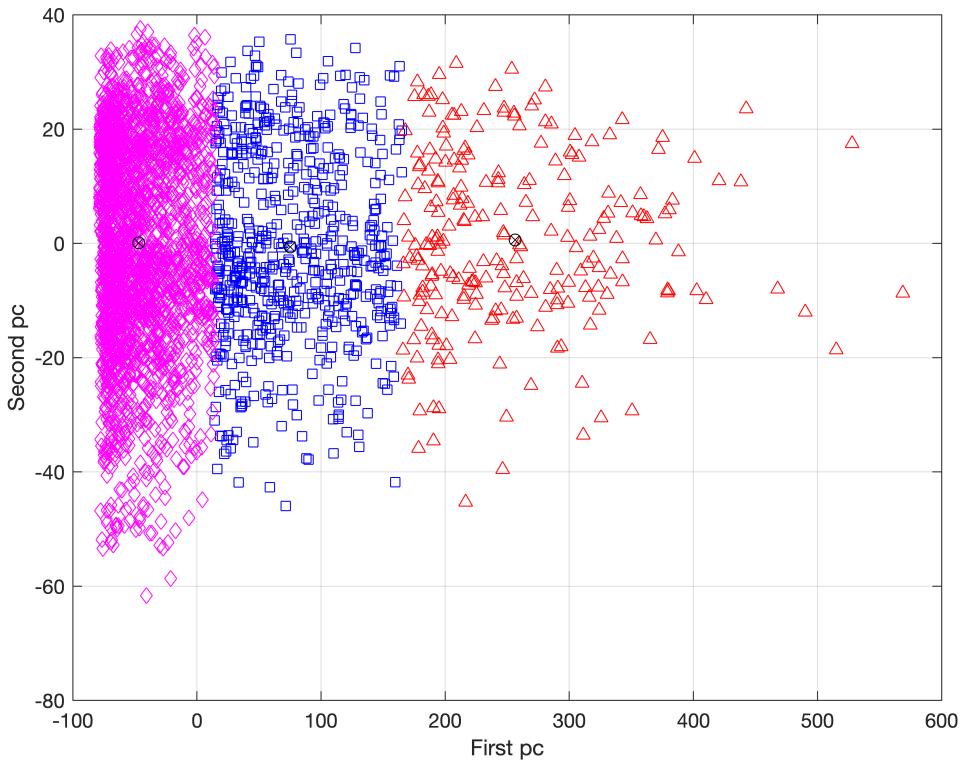


3 clusters:

```
[cidx3,cmeans3] = kmeans(T1,3,'replicates',10,'display','final');
```

```
Replicate 1, 14 iterations, total sum of distances = 4.98909e+06.
Replicate 2, 27 iterations, total sum of distances = 4.98902e+06.
Replicate 3, 25 iterations, total sum of distances = 4.98902e+06.
Replicate 4, 18 iterations, total sum of distances = 4.98909e+06.
Replicate 5, 20 iterations, total sum of distances = 4.98909e+06.
Replicate 6, 7 iterations, total sum of distances = 4.98902e+06.
Replicate 7, 12 iterations, total sum of distances = 4.98902e+06.
Replicate 8, 30 iterations, total sum of distances = 4.98902e+06.
Replicate 9, 27 iterations, total sum of distances = 4.98902e+06.
Replicate 10, 33 iterations, total sum of distances = 4.98902e+06.
Best total sum of distances = 4.98902e+06
```

```
ptsymb = {'bs','r^','md','go','c+'};
for i = 1:3
    clust = find(cidx3==i);
    plot(T1(clust,1),T1(clust,2),ptsymb{i});
    hold on
end
plot(cmeans3(:,1),cmeans3(:,2),'ko');
plot(cmeans3(:,1),cmeans3(:,2),'kx');
hold off
xlabel('First pc');
ylabel('Second pc');
grid on
```

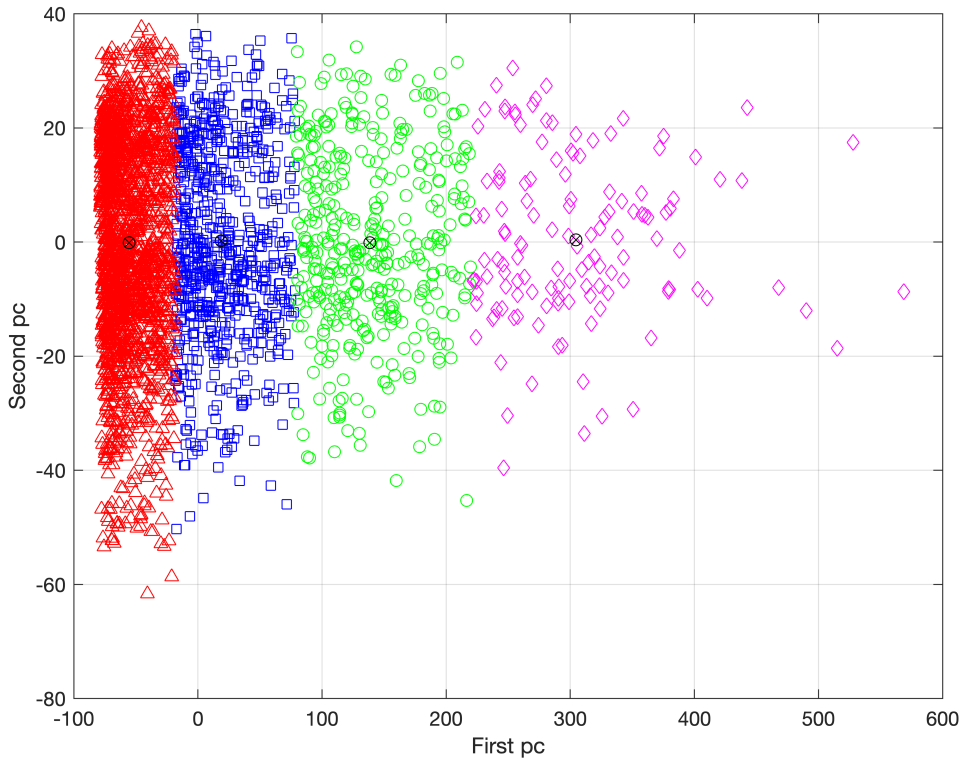


4 clusters:

```
[cidx4,cmeans4] = kmeans(T1,4,'replicates',10,'display','final');
```

```
Replicate 1, 7 iterations, total sum of distances = 3.26765e+06.
Replicate 2, 31 iterations, total sum of distances = 3.26768e+06.
Replicate 3, 6 iterations, total sum of distances = 3.26768e+06.
Replicate 4, 13 iterations, total sum of distances = 3.26768e+06.
Replicate 5, 27 iterations, total sum of distances = 3.26947e+06.
Replicate 6, 15 iterations, total sum of distances = 3.26947e+06.
Replicate 7, 20 iterations, total sum of distances = 3.26768e+06.
Replicate 8, 14 iterations, total sum of distances = 3.26947e+06.
Replicate 9, 37 iterations, total sum of distances = 3.26768e+06.
Replicate 10, 16 iterations, total sum of distances = 3.26768e+06.
Best total sum of distances = 3.26765e+06
```

```
ptsymb = {'bs','r^','md','go','c+'};
for i = 1:4
    clust = find(cidx4==i);
    plot(T1(clust,1),T1(clust,2),ptsymb{i});
    hold on
end
plot(cmeans4(:,1),cmeans4(:,2),'ko');
plot(cmeans4(:,1),cmeans4(:,2),'kx');
hold off
xlabel('First pc');
ylabel('Second pc');
grid on
```

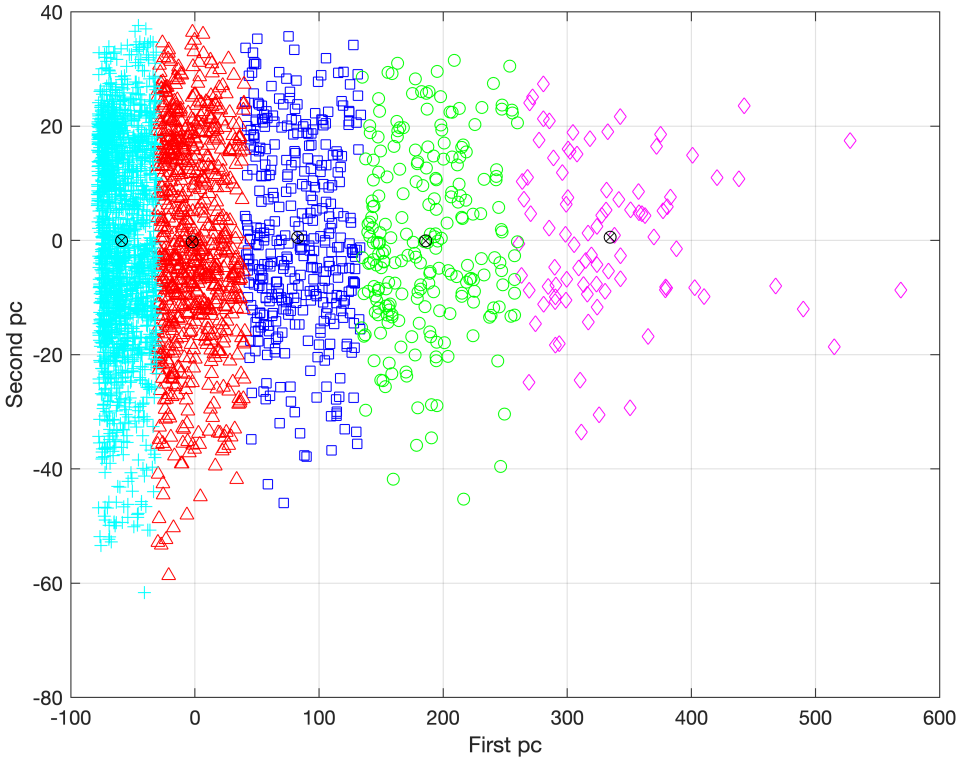


5 clusters:

```
[cidx5,cmeans5] = kmeans(T1,5,'replicates',10,'display','final');
```

```
Replicate 1, 34 iterations, total sum of distances = 2.53701e+06.
Replicate 2, 35 iterations, total sum of distances = 2.53701e+06.
Replicate 3, 18 iterations, total sum of distances = 2.53701e+06.
Replicate 4, 40 iterations, total sum of distances = 2.53701e+06.
Replicate 5, 14 iterations, total sum of distances = 2.52679e+06.
Replicate 6, 14 iterations, total sum of distances = 2.53701e+06.
Replicate 7, 35 iterations, total sum of distances = 2.79688e+06.
Replicate 8, 33 iterations, total sum of distances = 2.53701e+06.
Replicate 9, 18 iterations, total sum of distances = 2.52635e+06.
Replicate 10, 15 iterations, total sum of distances = 2.79608e+06.
Best total sum of distances = 2.52635e+06
```

```
ptsymb = {'bs','r^','md','go','c+'};
for i = 1:5
    clust = find(cidx5==i);
    plot(T1(clust,1),T1(clust,2),ptsymb{i});
    hold on
end
plot(cmeans5(:,1),cmeans5(:,2),'ko');
plot(cmeans5(:,1),cmeans5(:,2),'kx');
hold off
xlabel('First pc');
ylabel('Second pc');
grid on
```



Choose the optimal number of clusters:

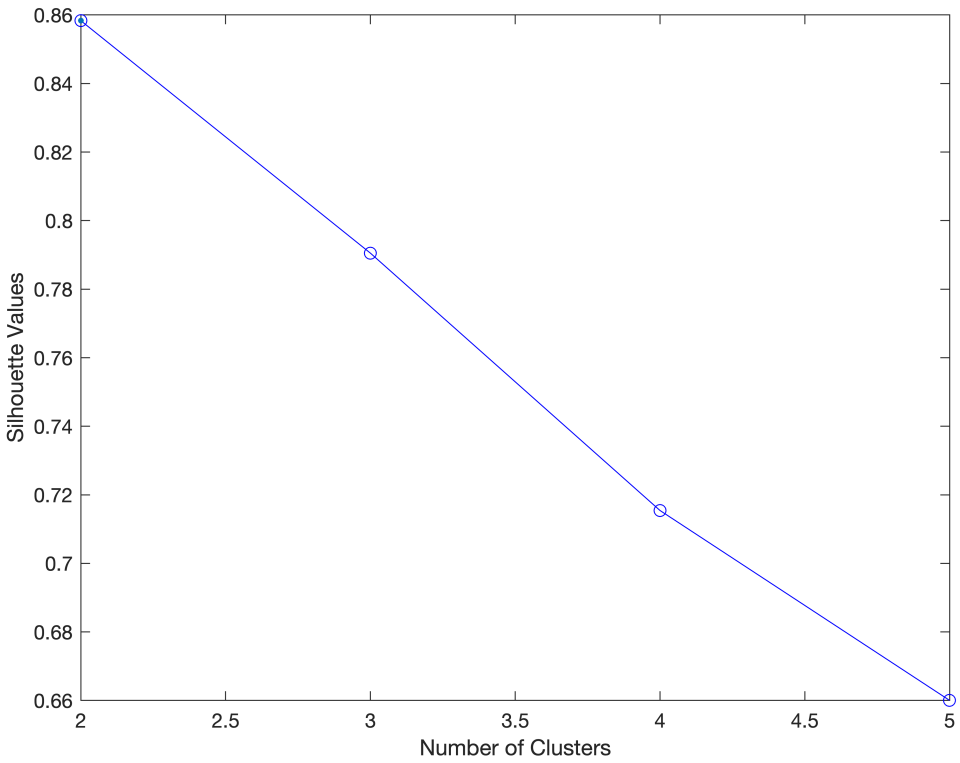
```
E = evalclusters(T1,'kmeans','silhouette','klist',[2:5])
```

```
E =
  SilhouetteEvaluation with properties:

    NumObservations: 3202
         InspectedK: [2 3 4 5]
    CriterionValues: [0.8583 0.7905 0.7154 0.6600]
           OptimalK: 2
```

```
kbest = E.OptimalK
```

```
kbest = 2
```

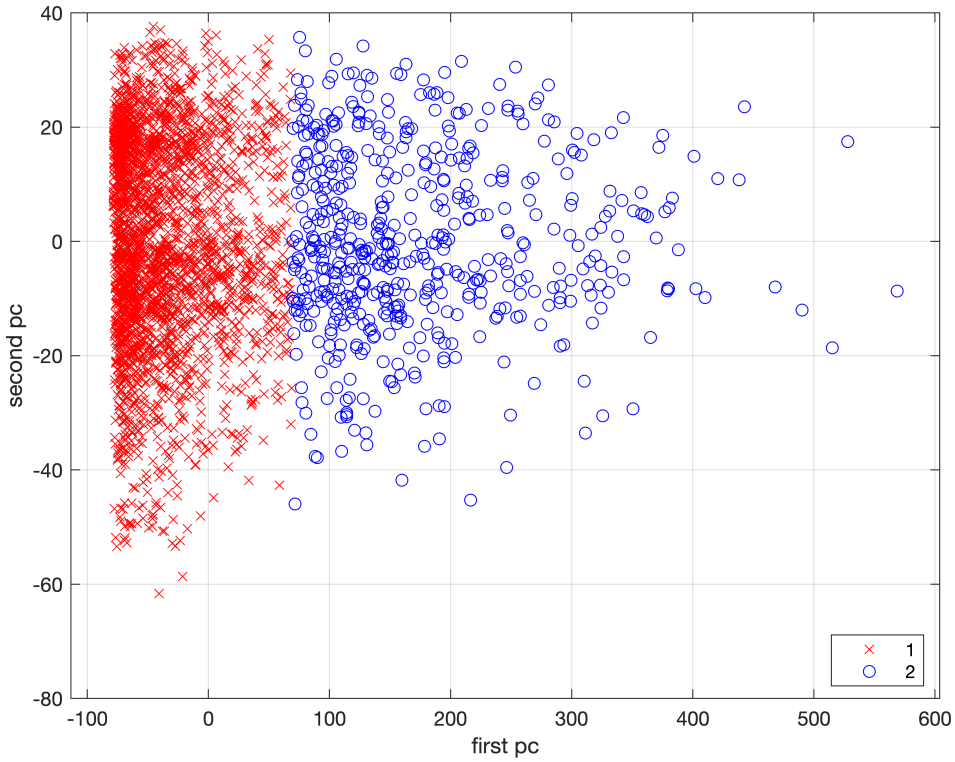
```
figure
plot(E)
```



The plot shows that the highest silhouette value occurs at two clusters, suggesting that the optimal number of clusters is 2.

We can create a scatter plot (2D) to visually examine the suggested clusters.

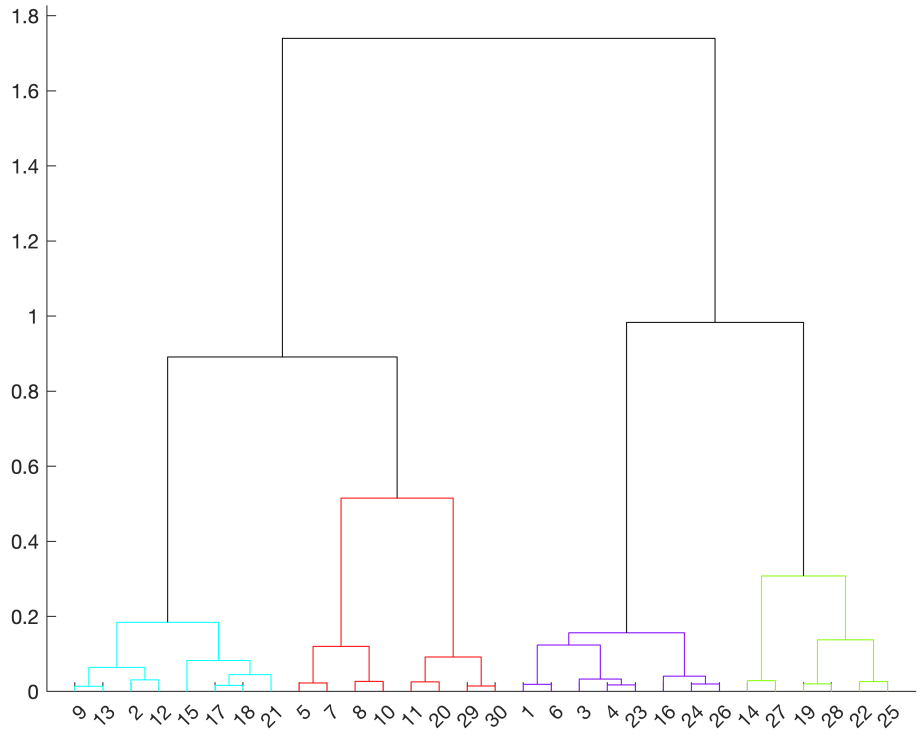
```
figure;
gscatter(T1(:,1),T1(:,2),E.OptimalY,'rbg','xod')
xlabel('first pc');
ylabel('second pc');
grid on
```



Hierarchical Clustering:

At linkage 0.6, there is 4 clusters.

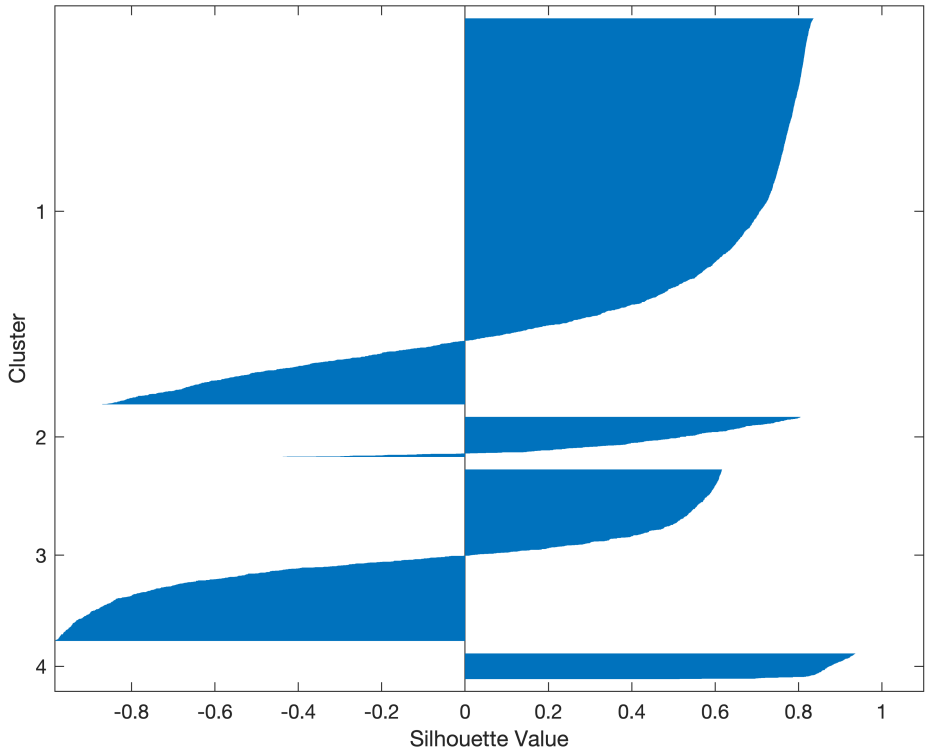
```
hidx = cluster(clustTreeCos,'Maxclust',4);
hidxbis=clusterdata(T1,'maxclust',4','linkage', 'average');
cutoff = 0.6;
dendrogram(clustTreeCos,'ColorThreshold',cutoff)
```



Evaluate this clustering:

We use the silhoutte coefficient to evaluate the quality of this cluster solution.

```
[silh4tree,h] = silhouette(T1,hidx);
```




```
mean(silh4tree)
```

```
ans = 0.3292
```

The mean Silhouette value is 0.3292. The mean and the graph tell us that this is not the optimal clustering.

GMM Clustering:

Predictors of the dataset are reduced and transformed (according to the principal components in the previous part). GMM (Gaussian Mixture Model) clustering is adopted with 3 clusters and 1000 maximum iterations (for the Expectation-Maximization algorithm).

```
k = 3; % Number of GMM components
options = statset('MaxIter',1000);
gm = fitgmdist(T1,k,'Options',options);
P = posterior(gm,T1)
```

```
P = 3202×3
    0.0000    0.9998    0.0002
    0.3661    0.0177    0.6162
    0.0000    1.0000    0.0000
         0    1.0000    0.0000
    0.0000    0.0731    0.9269
    0.0000    0.8673    0.1327
    0.0000    0.0413    0.9587
    0.5945    0.0120    0.3936
    0.8153    0.0068    0.1779
    0.7939    0.0082    0.1979
```

```
P(8,:)
```

```
ans = 1×3
    0.5945    0.0120    0.3936
```

As we can see, the probability of the 8th observation in 1st cluster is 0.5952 in 2nd cluster is 0.3928 and 3rd is 0.012. Hence, it is the most optimal compared to others.

Multivariate Regression Model

The goal of this task is to predict the relative mean humidity (variable Y2, i.e. variable U_mu), based on predictors matrix X.

The dataset is randomly divided into train and test data, considering 70% of the data for the training data and 30% for the test data (non-stratified partition). Specifically, divide your predictors matrix (X) in Xtrain and Xtest. Similarly, divide your target variable of interest Y2 in Y2train and Y2test.

```
rng('default');
cv = cvpartition(height(T),'holdout',0.30);
Xtrain = X(training(cv),:);
Y1train = Y1(training(cv),:);
Y2train = Y2(training(cv),:);

Xtest = X(test(cv),:);
Y1test = Y1(test(cv),:);
Y2test = Y2(test(cv),:);
```

A linear regression model is trained using all the available predictors in X (use training data)

```
mdl=fitlm(Xtrain,Y2train,'linear')
```

```
mdl =
Linear regression model:
    y ~ 1 + x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 + x9 + x10 + x11 + x12 + x13 + x14 + x15 + x16 + x17 + x

Estimated Coefficients:
                    Estimate        SE          tStat       pValue
                   __________    __________    ________    __________

    (Intercept)        132.55        24.943       5.3141    1.1787e-07
    x1                -4.3021      0.068169      -63.109             0
    x2                 1.5314          1.35       1.1344       0.25676
    x3                -1.5706        1.3494      -1.1639       0.24457
    x4               -0.27974      0.032686      -8.5584    2.0994e-17
    x5                0.27852      0.043519       6.3998    1.8912e-10
    x6               -0.34298      0.041211      -8.3226    1.4829e-16
    x7              -0.066836     0.0035604      -18.772     4.283e-73
    x8                 4.4723       0.01993        224.4             0
    x9                0.056931      0.013773       4.1335    3.7052e-05
    x10                0.2093       0.24464      0.85555       0.39234
    x11               -0.20611       0.24443      -0.84326       0.39917
    x12              -0.0030109      0.030421    -0.098975       0.92117
    x13               0.051664     0.0058543       8.8249    2.1711e-18
```

```
    x14             -0.020704      0.007825      -2.6458     0.0082068
    x15              0.0024234    0.00035835       6.7626    1.7272e-11
    x16                0.02746     0.0086608       3.1706     0.0015421
    x17             -0.0018835      0.012244     -0.15383       0.87776
    x18               0.018421     0.0098327       1.8735      0.061132
    x19              -0.002303        0.0035       -0.658       0.51061


Number of observations: 2242, Error degrees of freedom: 2222
Root Mean Squared Error: 1.43
R-squared: 0.986,  Adjusted R-Squared: 0.986
F-statistic vs. constant model: 8.12e+03, p-value = 0
```

Some variables has p-value that is not significant at 5% level. Those will be removed and then, we build a second model.

```
Xtrain1 = Xtrain;
Xtrain1(:,[2,3,10,11,12,17,18,19]) = [];
mdl1=fitlm(Xtrain1,Y2train,'linear')
```

```
mdl1 =
Linear regression model:
    y ~ 1 + x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 + x9 + x10 + x11

Estimated Coefficients:
                    Estimate        SE         tStat       pValue
                    ________    __________    _______    __________

    (Intercept)       98.184       0.14556     674.53             0
    x1               -4.4138      0.068039    -64.871             0
    x2               -0.17182     0.028614    -6.0046    2.2327e-09
    x3                0.31934      0.04395     7.2659    5.0982e-13
    x4               -0.31318     0.041788    -7.4945    9.5432e-14
    x5               -0.064486    0.0035349   -18.243     2.061e-69
    x6                  4.5276     0.018496     244.79             0
    x7                0.052474     0.013938     3.7647    0.00017104
    x8                0.055584    0.0059166     9.3945     1.372e-20
    x9               -0.019814    0.0079474    -2.4932      0.012732
    x10              0.0028179   0.00035653     7.9036    4.2192e-15
    x11               0.033808    0.0087582     3.8602    0.00011651


Number of observations: 2242, Error degrees of freedom: 2230
Root Mean Squared Error: 1.46
R-squared: 0.985,  Adjusted R-Squared: 0.985
F-statistic vs. constant model: 1.35e+04, p-value = 0
```

```
Xtest1 = Xtest;
Xtest1(:,[2,3,10,11,12,17,18,19]) = [];
```

Applying the model on the test data, we predict the target variable.

```
Y2predicted_test= mdl.predict(Xtest)
```

```
Y2predicted_test = 960×1
   89.6914
   87.5950
   84.5713
   90.8698
   91.3966
   92.1759
```

```
   95.8211
   87.3122
   86.6713
   85.8967
     ⋮
```

```
Y2predicted_test1= mdl1.predict(Xtest1)
```

```
Y2predicted_test1 = 960×1
   89.5111
   87.2742
   84.8841
   91.2562
   91.8013
   92.5289
   96.2429
   88.2260
   87.4211
   86.7890
     ⋮
```

```
MSEtest = immse(Y2test,Y2predicted_test)
```

```
MSEtest = 1.9483
```

```
MSEtest2 = immse(Y2test,Y2predicted_test1)
```

```
MSEtest2 = 1.9677
```

Compare MSE of 2 models, we see that there is not much difference between them.

Principal component regression (PCR):

Apply Principal Component Analysis over training predictors and train 3 linear regression models (use training data):

```
Xtrain; Y2train;
Xtest; Y2test;

[coeff,scoreTrain,~,~,explained,mu] = pca(Xtrain);
```

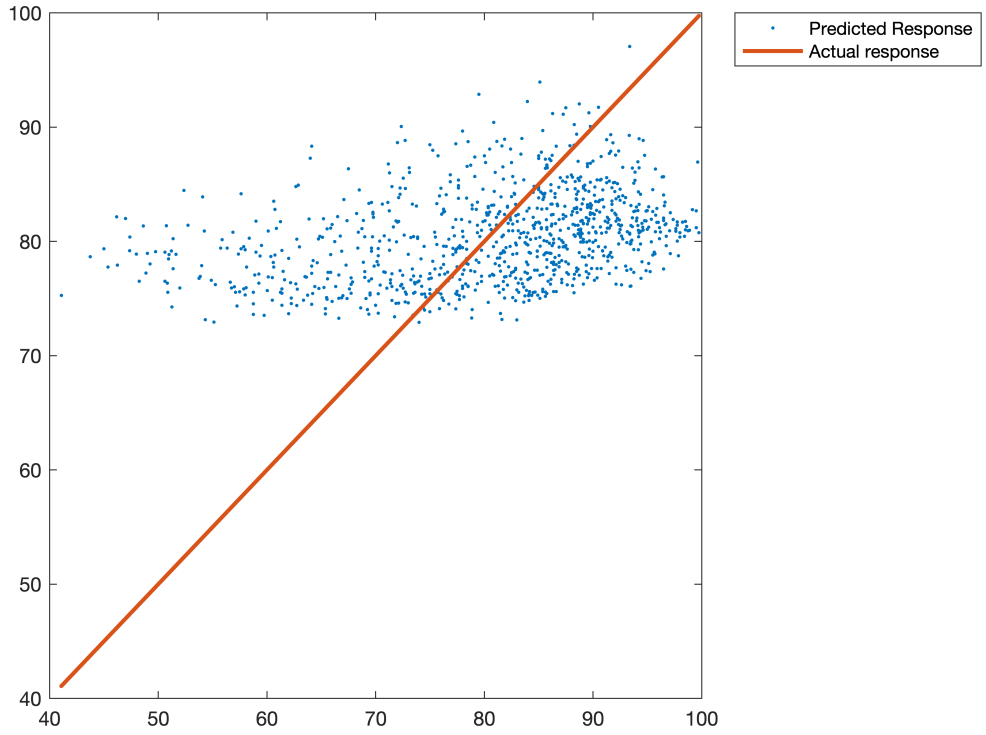
PCR with 2 PCs:

```
scoreTrain2comp = scoreTrain(:,1:2);
mdl2 = fitlm(scoreTrain2comp,Y2train);
scoreTest2comp = (Xtest-mu)*coeff(:,1:2);
YTest_predicted2 = predict(mdl2,scoreTest2comp);
```

Let's graph the predicted value and the real value to compare

```
figure
plot(Y2test,YTest_predicted2,".")
hold on
plot(Y2test,Y2test,'Linewidth',2)
```

```
hold off
legend('Predicted Response','Actual response','location','bestoutside')
```

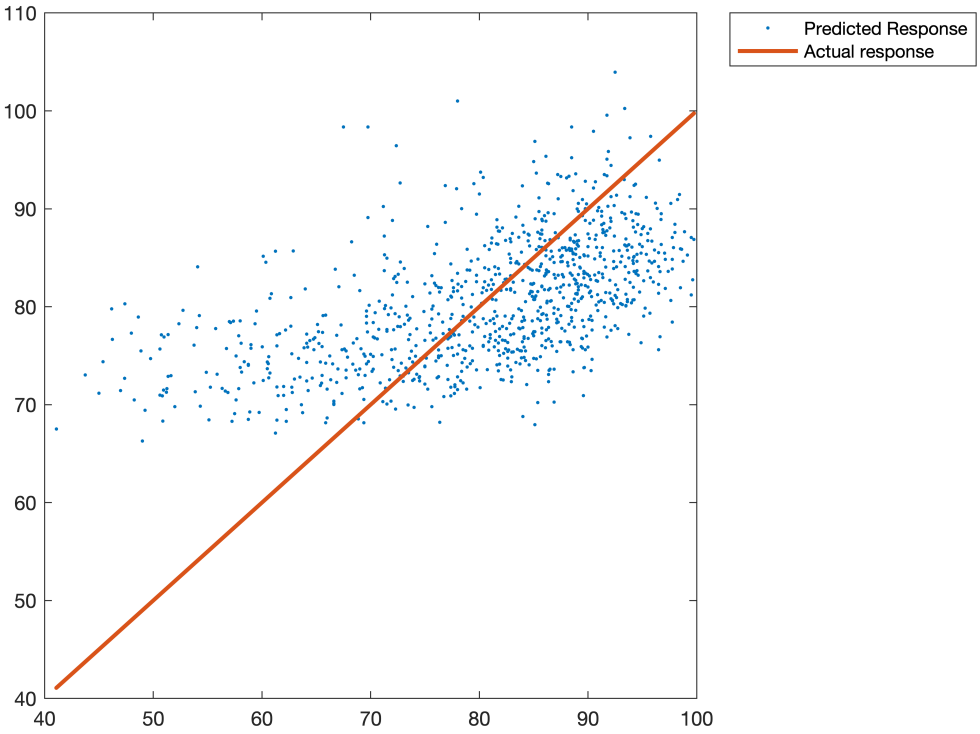


PCR with 3 PCs:

```
scoreTrain3comp = scoreTrain(:,1:3);
mdl3 = fitlm(scoreTrain3comp,Y2train);
scoreTest3comp = (Xtest-mu)*coeff(:,1:3);
YTest_predicted3 = predict(mdl3,scoreTest3comp);
```

Let's graph the predicted value and the real value to compare

```
figure
plot(Y2test,YTest_predicted3,".")
hold on
plot(Y2test,Y2test,'Linewidth',2)
hold off
legend('Predicted Response','Actual response','location','bestoutside')
```

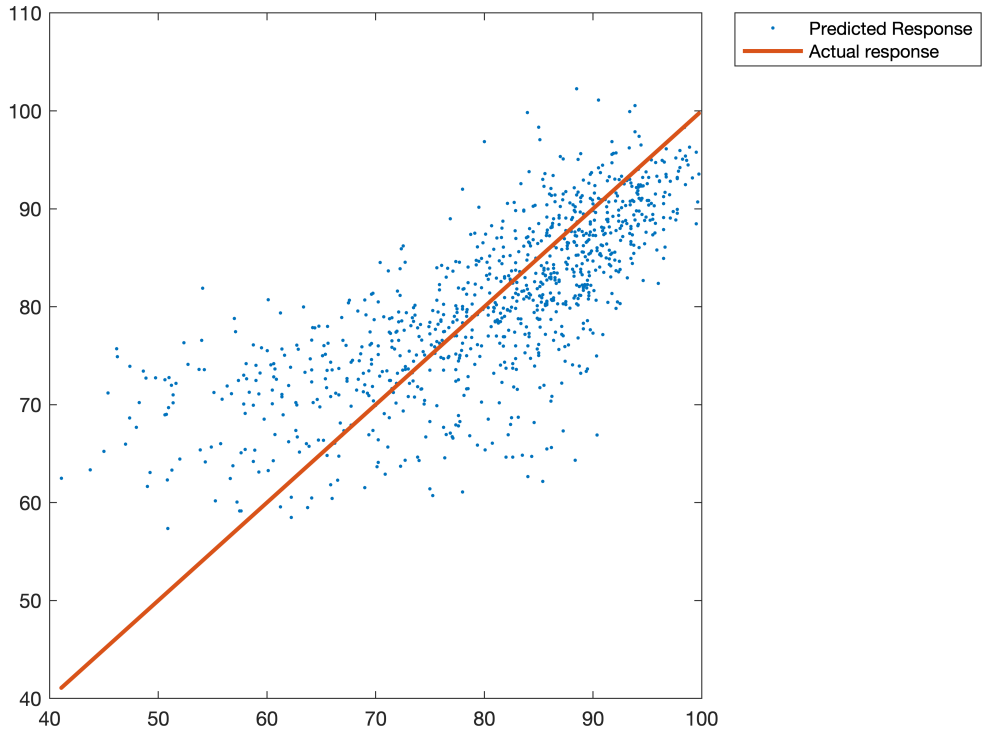


PCR with 4 PCs:

```
scoreTrain4comp = scoreTrain(:,1:4);
mdl4 = fitlm(scoreTrain4comp,Y2train);
scoreTest4comp = (Xtest-mu)*coeff(:,1:4);
YTest_predicted4 = predict(mdl4,scoreTest4comp);
```

Let's graph the predicted value and the real value to compare

```
figure
plot(Y2test,YTest_predicted4,".")
hold on
plot(Y2test,Y2test,'Linewidth',2)
hold off
legend('Predicted Response','Actual response','location','bestoutside')
```



Evaluate which models predict most correctly.

```
MSEpcr2comp = immse(Y2test,YTest_predicted2)
```

```
MSEpcr2comp = 123.9894
```

```
MSEpcr3comp = immse(Y2test,YTest_predicted3)
```

```
MSEpcr3comp = 96.5464
```

```
MSEpcr4comp = immse(Y2test,YTest_predicted4)
```

```
MSEpcr4comp = 63.5570
```

The objective is to minimise MSE and the model with first 4pcs seems smallest among 3. Therefore, it is the most fitted model. We can see that through the figures which illustrate each model above.

Classification and ROC Curve:

The goal of this is to classify the weather conditions either as 1(dry) or 0 (not dry) (variable Y1, i.e. variable OBSERVED), based on predictors matrix X. We apply the same holdout non-stratified partition (split intro training and test data) as in regression model. Reuse Xtrain and Xtest and split the variable of interest Y1 into Y1train and Y1test.

```
Y1train; Y1test;
Y1test = logical(Y1test);
```

We train a logistic regression model using all the available predictors in X (use training data)

```
glm = fitglm(Xtrain, Y1train,'Distribution','binomial','link','logit')
```

```
glm =
Generalized linear regression model:
    logit(y) ~ 1 + x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 + x9 + x10 + x11 + x12 + x13 + x14 + x15 + x16 +
    Distribution = Binomial

Estimated Coefficients:
                    Estimate        SE         tStat       pValue
                   _________    _________    _______    _________

    (Intercept)       -240.44       48.163    -4.9923    5.9676e-07
    x1                0.24141      0.13173     1.8326      0.066858
    x2                -4.2362       2.5263    -1.6769      0.093565
    x3                 4.3313       2.5253     1.7151      0.086322
    x4                -0.25491      0.06256    -4.0747     4.607e-05
    x5                 0.36909     0.086367     4.2735    1.9243e-05
    x6                -0.27765     0.078088    -3.5557    0.00037703
    x7                0.062217    0.0075765     8.2119    2.1773e-16
    x8                -0.38771     0.052531    -7.3806    1.5759e-13
    x9                 0.22031     0.032877     6.7011    2.0682e-11
    x10               -0.88531      0.49684    -1.7819      0.074772
    x11                0.87692      0.49639     1.7666      0.077296
    x12               -0.11348     0.056792    -1.9981      0.045707
    x13                0.041964     0.011585     3.6223    0.00029203
    x14               -0.023812     0.015593    -1.5271       0.12674
    x15              -0.0048175    0.00067731    -7.1127     1.138e-12
    x16                 -0.1288     0.019927    -6.4635    1.0228e-10
    x17                0.083123     0.023429     3.5478    0.00038849
    x18               0.0083137     0.017674    0.47039       0.63808
    x19              -0.0049734    0.0065256    -0.76213       0.44598


2242 observations, 2222 error degrees of freedom
Dispersion: 1
Chi^2-statistic vs. constant model: 1.05e+03, p-value = 4.54e-210
```

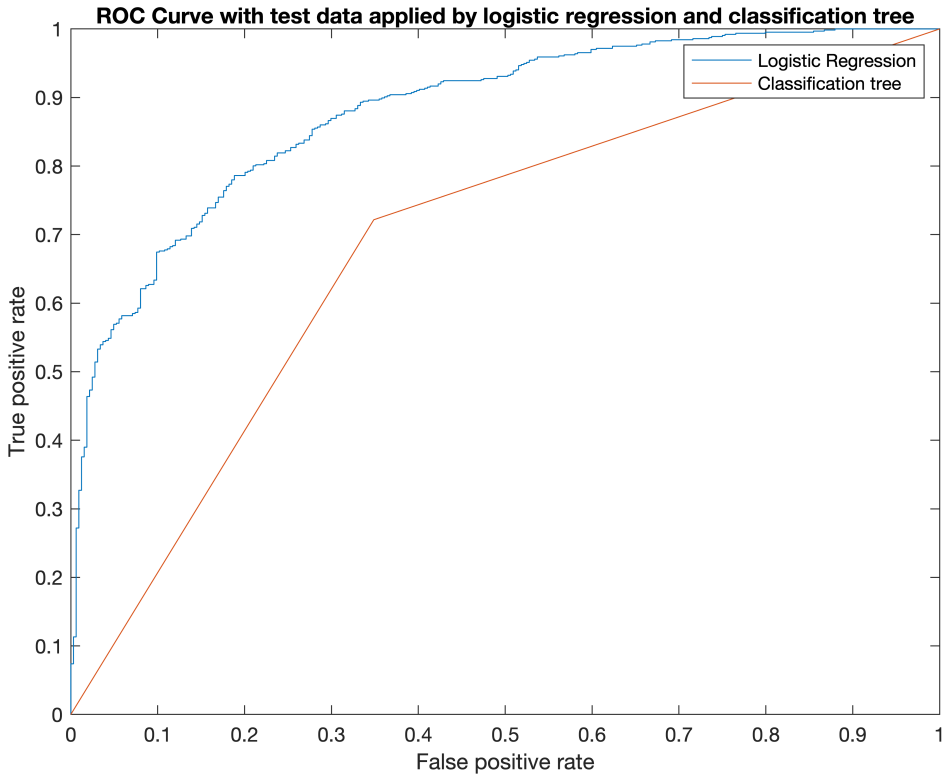
```
Y1_glm = predict(glm, Xtest);
[xxtest,yytest,Tresholds,auctest] = perfcurve(Y1test,Y1_glm,'true');
```

Now, we apply Principal Component Analysis over training predictors and train a classification tree using the first 3 principal components (use training data)

```
[coeff,scoreTrain,~,~,explained,mu] = pca(Xtrain);
scoreTrain3comp = scoreTrain(:,1:3);
[coeff,scoreTest,~,~,explained,mu] = pca(Xtest);
scoreTest3comp = scoreTest(:,1:3);
t = fitctree(scoreTrain3comp,Y1train);
Y1_t = predict(t, scoreTest3comp);
[xxtest1,yytest1,Tresholds1,auctest1] = perfcurve(Y1test,Y1_t,'true');
```

```
figure
plot(xxtest,yytest)
hold on
plot(xxtest1,yytest1)
xlabel('False positive rate'); ylabel('True positive rate');
```

```
legend('Logistic Regression','Classification tree');
title('ROC Curve with test data applied by logistic regression and classification tree
hold off
```



The graph below clearly shows that AUC of logistic regression model is larger than AUC of tree model. Thus, Logistic regression model predicts better than the other.

```
[auctest auctest1]
```

```
ans = 1×2
    0.8799    0.6865
```

Classification and Confusion Matrix

The goal of this is to classify again the weather conditions either as 1(dry) or 0 (not dry) (variable Y1, i.e. variable OBSERVED), based on predictors matrix X. However, now, we use Support Vector Machine (SVM) classifier with K-fold partition (k=3).

```
rng(1);
cvp = cvpartition(Y1,'KFold',3);
```

SVM classifier (3-fold cross validation) is built.

```
mdlSVM = fitcsvm(X,Y1,'Standardize',true,'CVPartition', cvp, 'KernelFunction','RBF','K
cross_validation_missclassificatio_rate_svm= kfoldLoss(mdlSVM)
```

```
cross_validation_missclassificatio_rate_svm = 0.2027
```

```
cross_validation_Accuracy_svm=1-cross_validation_missclassificatio_rate_svm
```

```
cross_validation_Accuracy_svm = 0.7973
```

Next, classification tree is trained based on matrix X.

```
t= fitctree(X,Y1,'CVPartition', cvp);
cross_validation_missclassificatio_rate_t= kfoldLoss(t)
```

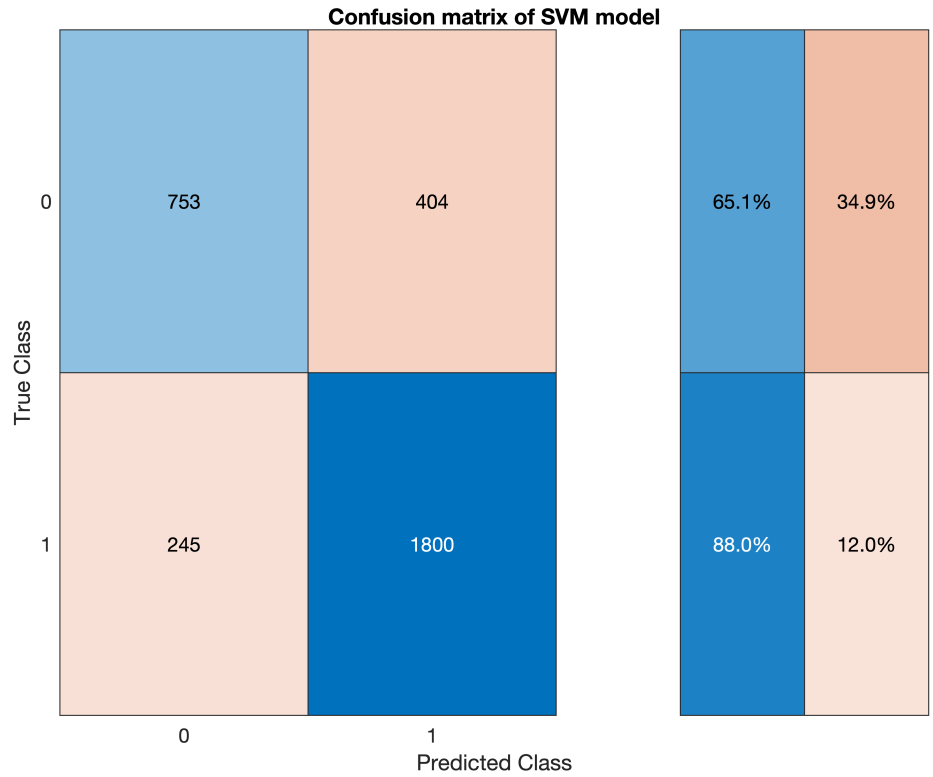
```
cross_validation_missclassificatio_rate_t = 0.2764
```

```
cross_validation_Accuracy_t=1-cross_validation_missclassificatio_rate_t
```

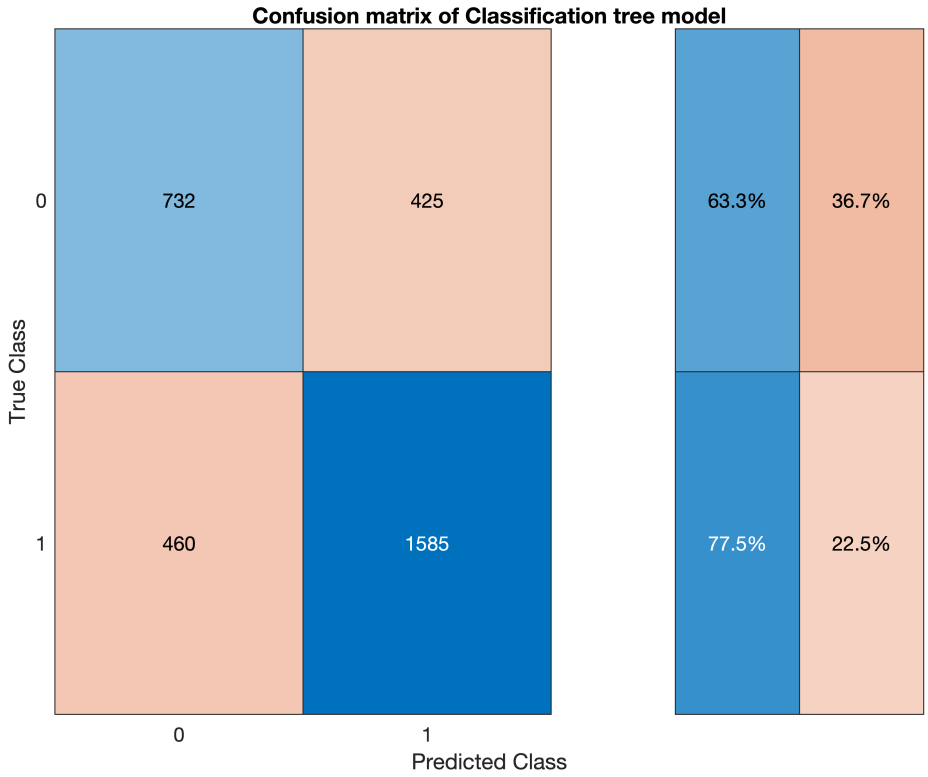
```
cross_validation_Accuracy_t = 0.7236
```

Confusion matrix:

```
% Confusion matrix of SVM classifier
predicted_classes_svm = kfoldPredict(mdlSVM);
C = confusionchart(Y1,predicted_classes_svm,'RowSummary','row-normalized');
title('Confusion matrix of SVM model');
```




```
% Confusion matrix of classification tree
predicted_classes_t = kfoldPredict(t);
C = confusionchart(Y1,predicted_classes_t,'RowSummary','row-normalized');
title('Confusion matrix of Classification tree model');
```




```
A = [cross_validation_Accuracy_svm cross_validation_Accuracy_t]
```

```
A = 1×2
    0.7973    0.7236
```

The accuracy of svm model is 0.7973 while the accuracy of tree model is 0.7264.

TPR of svm model is 85.5% while of the tree model is 77.9%.

TNR of svm model is 73% while of the tree model is 62%.

So the svm model predicts better than the other.

Conclusion

In conclusion, the goal of this project is to build different models for prediction of weather and choose the most fitted one. This report has covered data mining process, carried out principal component analysis. It also built various regression model to predict the mean humidity in Helsinki. They are multivariate regression, stepwise regression and PCR with 2, 3 and 4 PCs. The best model is the adjusted linear regression.

To anticipate whether the weather is dry or not, different models were trained and built binary logistic regression, classification tree and support vector machine classifier. Binary logistic regression and support vector machine are better model than classification tree. Either one of them can be chosen.