ENDG 319
Group 91
Shichao Han (30126712)
Syed Mohammed Abbas Kazmi (30125715)
Kaushik Natarajan (30118209)
Valeria Navarrete (30101778)

CURE Project Deliverable 3
Submission Data: Nov 29, 2021

Task 1:

We used the same data from deliverable 3 to develop the knn-classifier model. Please see the python file for the detailed machine learning model.

##MinMaxScaler

In [8]:
```
y_train = le.transform(sy_train)
```

In [9]:
```
dataX_train.to_numpy()

n1 = preprocessing.MinMaxScaler()
n1 = n1.fit(dataX_train.to_numpy())

X_train = n1.transform(dataX_train.to_numpy())
X_train
```

Out[9]:
```
array([[4.73684211e-01, 3.60268173e-01, 5.70633273e-01, 2.67826300e-01,
        1.71408250e-01, 5.55084224e-01],
       [7.71929825e-01, 3.55681158e-01, 4.22000000e-01, 5.80948810e-01,
        4.40967283e-02, 3.69633711e-01],
       [7.10526316e-01, 1.12208938e-01, 4.90727273e-01, 5.18324308e-01,
        3.55618777e-02, 3.96036118e-01],
       [7.45614035e-01, 2.82639441e-01, 4.22000000e-01, 5.54300086e-01,
        4.33854908e-02, 3.79402602e-01],
       [3.24561404e-01, 9.91531614e-01, 6.18181818e-01, 1.65894930e-01,
        5.51920341e-01, 5.94758242e-01],
       [8.77192982e-02, 7.43471998e-01, 8.23309091e-01, 1.28691821e-02,
        1.00000000e+00, 9.95749212e-01],
       [1.22807018e-01, 8.55681126e-01, 8.05505818e-01, 1.47883868e-02,
        9.09672831e-01, 9.95749212e-01],
       [6.84210526e-01, 3.98729939e-02, 4.90909091e-01, 5.13660782e-01,
        1.99146515e-02, 4.05805009e-01],
       [5.43859649e-01, 7.47354003e-01, 5.31701818e-01, 3.71756325e-01,
        6.89900427e-02, 5.13993276e-01],
       [1.92982456e-01, 3.02399495e-01, 7.07352364e-01, 4.99729789e-02,
```

In [10]:
```
X_train.shape
```

Out[10]:
```
(87, 6)
```

In [11]:
```
from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier(n_neighbors = 45)
knn = knn.fit(X_train, y_train)

y_test = le.transform(sy_test.to_numpy())
sy_test, y_test

y_test = le.transform(sy_test.to_numpy())
X_test = n1.transform(dataX_test.to_numpy())

knn.predict(X_test)

y_test
```

Out[11]:
```
array([1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1,
       0, 0, 0, 1, 1, 1, 1], dtype=int64)
```

In [12]:
```
knn.score(X_test, y_test)
```

Out[12]:
```
0.9310344827586207
```

In [13]:
```
knn.score(X_train, y_train)
```

Out[13]:
```
0.9885057471264368
```

In [14]:
```
knn.predict(X_test)
```

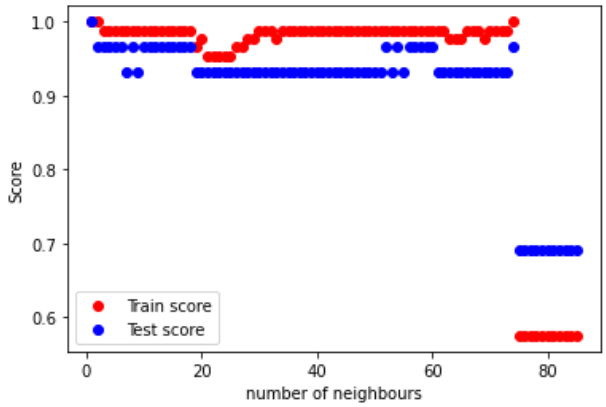
Out[14]:
```
array([1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1,
       1, 0, 0, 1, 1, 1, 1], dtype=int64)
```

In [15]:
```
y_test
```

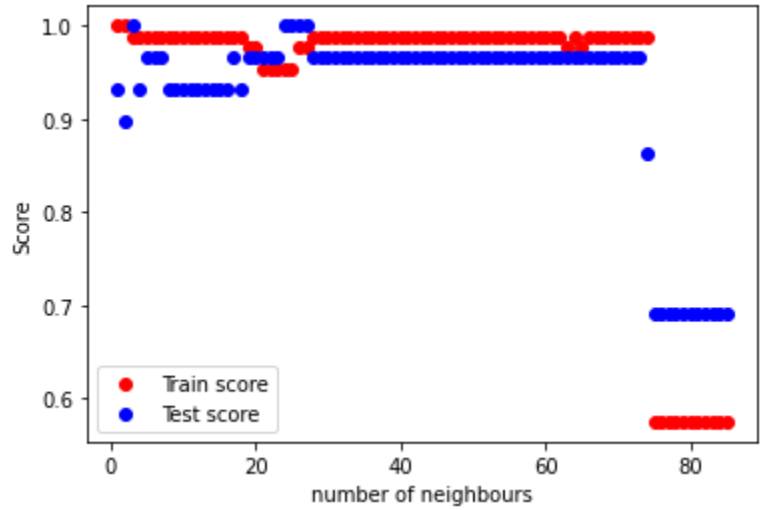
Out[15]:
```
array([1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1,
       0, 0, 0, 1, 1, 1, 1], dtype=int64)
```

Task 2:

Graph for model accuracy for training and test dataset against k with MinMaxScaler:



Graph for model accuracy for training and test dataset against k with StandardScaler:



Task 3:

We find the best value of k is 45 using StandardScaler for our final model.

Confusion matrix is used to show how well the machine learning model can identify the class of instances correctly. The Y-axis is the true label of each instance, and the X-axis is the predicted label from the machine learning model. Instance laid on the diagonal of the matrix means the machine learning model has successfully predicted the instance correctly. Any instance off from the diagonal of the matrix means the machine learning model has failed to predict the instance label correctly.

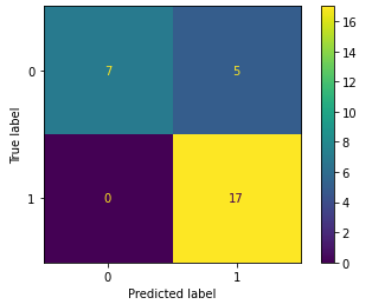
The confusion matrix and plot show below:

```
In [70]: #confusion matrix
         import seaborn as sns
         knn = KNeighborsClassifier(n_neighbors = 45)
         knn = knn.fit(X_train_p, y_train)

         from sklearn.metrics import confusion_matrix,ConfusionMatrixDisplay


         y_act=pd.Series(y_test,name='actual')
         y_prid=pd.Series(knn.predict(X_test),name='Predict')
         df_confusion=pd.crosstab(y_act,y_prid)
         df_confusion
```

Out[70]:

actual	Predict 0	1
0	7	5
1	0	17

```
In [71]: from sklearn.datasets import make_classification
         from sklearn.svm import SVC


         cm = confusion_matrix(y_act, y_prid, labels=knn.classes_)
         disp = ConfusionMatrixDisplay(confusion_matrix=cm,display_labels=knn.classes_)
         disp.plot()
         plt.show()
```



Task 4:

For our new instance we went back to the souse where we find the data and obtained the next day data, show below:

Date	Temperature	Humidity	Windspeed	Pressure	Visibility
117	8.43	0.6545	26.45367	1011.243343	6.43846

From the source, this day had high precipitation, therefore this instance should be a class 1 instance. After we run the test, we find the predicted value matches with the true value.

```
In [76]: new_example_rawdata = np.array([[117 , 8.433 , 0.6545 , 26.45367 , 1011.243343 , 6.43846]])
         X_new = n1.transform(new_example_rawdata)
         X_new

Out[76]: array([[1.00869565, 0.19016941, 0.409     , 0.76355259, 0.54731405,
                 0.16954394]])

In [77]: knn.predict(X_new)

Out[77]: array([1], dtype=int64)
```

Date used:

Number of days	Temperature	Humidity	Wind Speed	Pressure	Visibility	Precipitation_classes
1	9.472222	0.95	8.0012	1015.13	15.8746	0
2	9.355556	0.94	8.00154	1015.63	15.8746	0
3	9.377778	0.93	8.1102	1015.94	15.8263	0
4	8.288889	0.92	8.11029	1016.41	15.8263	0
5	8.755556	0.92122	8.112039	1016.51	15.8263	0
6	9.222222	0.929128	8.11304	1016.66	15.8263	0
7	7.733333	0.91	8.2393	1016.72	15.8263	0
8	8.772222	0.91829	8.2413	1016.84	15.8263	0
9	10.82222	0.918929	8.24232	1017.37	15.8263	0
10	13.77222	0.9	8.243453	1017.22	15.8263	0
11	16.01667	0.901029	8.24454	1017.42	15.8263	0
12	17.14444	0.90282	8.31254	1017.74	15.8263	0
13	17.8	0.89	8.32423	1017.59	15.8263	0
14	17.33333	0.890183	8.33533	1017.48	15.8263	0
15	18.87778	0.890239	8.345242	1017.17	15.8263	0
16	18.91111	0.893028	8.35892	1016.47	15.8263	0
17	15.38889	0.88	8.41242	1016.15	15.8263	0
18	15.55	0.882129	8.42534	1016.17	15.8263	0
19	14.25556	0.883949	8.43542	1015.82	15.0052	0
20	13.14444	0.884958	8.44564	1015.83	14.9569	0
21	11.55	0.87	8.446743	1015.85	14.9569	0
22	11.18333	0.871829	8.5682	1015.77	14.9569	0
23	10.11667	0.829394	9.0965	1015.4	14.9569	0

24	10.2	0.839044	9.2092	1015.51	14.9569	0
25	10.42222	0.834858	9.8049	1014.4	14.9569	0
26	9.911111	0.9	9.9015	1014.2	14.9569	0
27	11.18333	0.821029	10.4006	1008.71	14.9086	0
28	7.155556	0.822939	10.4006	1014.47	14.0553	0
29	6.111111	0.823443	10.465	1014.45	11.4471	0
30	6.788889	0.81	10.7548	1014.49	11.4471	0
31	7.261111	0.812993	10.8192	1014.52	11.4471	0
32	7.8	0.812342	11.0446	1014.16	11.27	0
33	9.872222	0.813848	11.0768	1014.24	11.27	0
34	12.22222	0.812839	11.1573	1014.25	11.27	0
35	15.09444	0.80192	11.1734	1013.96	11.27	0
36	17.35556	0.80203	11.2056	1013.85	11.27	0
37	19.00556	0.803404	11.3183	1013.04	11.27	0
38	20.04444	0.804096	11.3344	1012.22	11.27	0
39	21.05	0.79	12.0106	1011.44	11.27	0
40	21.18333	0.791122	12.3648	1010.52	11.27	0
41	20.11667	0.792343	12.3648	1009.83	11.2056	0
42	20.21667	0.793434	12.5258	1009.26	11.2056	0
43	20	0.794556	12.8156	1008.76	11.2056	0
44	17.8	0.781211	13.0088	1008.36	11.2056	0
45	16.06111	0.782322	13.7494	1008.11	11.2056	0
46	15.02222	0.783224	13.8299	1008.15	11.2056	0
47	14.42222	0.784642	13.8943	1007.85	11.0768	1
48	14.25556	0.77	13.9587	1007.89	11.0768	1
49	13.77222	0.771323	14.1036	1007.36	11.0285	1

50	13.28333	0.772329	14.1197	1007.26	11.0285	1
51	8.633333	0.773488	14.1519	1005.1	11.0285	1
52	11.25	0.774594	14.2646	1007.01	10.8997	1
53	11.18333	0.76	14.3612	1006.73	10.8836	1
54	10.69444	0.761232	14.4095	1006.59	10.8514	1
55	11.11111	0.762939	14.4095	1006.34	10.8514	1
56	11.11111	0.763848	14.4739	1006.09	10.8192	1
57	12.16667	0.764858	14.9086	1005.97	10.8192	1
58	12.75556	0.75	15.1984	1005.63	10.8031	1
59	13.83889	0.751029	15.3755	1005.83	10.7548	1
60	16.18333	0.752931	15.5848	1005.97	10.6743	1
61	17.51667	0.753848	15.6331	1005.64	10.5455	1
62	17.38333	0.74	15.7297	1005.2	10.5455	1
63	17.36111	0.74125	16.5025	1005.1	10.3523	1
64	17.20556	0.742436	16.9855	1004.65	10.3523	1
65	15.63333	0.743738	17.0499	1004.04	10.0464	1
66	13.57778	0.74442	17.0982	1004.08	10.0464	1
67	10.91111	0.73	17.2109	1004.61	10.0303	1
68	8.8	0.731282	17.4363	1004.99	9.982	1
69	8.961111	0.7317	17.549	1004.85	9.982	1
70	8.2	0.73582	17.5651	1004.96	9.982	1
71	7.688889	0.73412	17.6456	1005.14	9.982	1
72	7.766667	0.73314	17.8549	1005.05	9.982	1
73	8.2	0.73182	18.1125	1004.8	9.982	1
74	8.177778	0.7285	19.2878	1004.89	9.982	1
75	7.311111	0.7256	19.3039	1007	9.982	1

76	7.644444	0.7241	19.7869	1004.3	9.982	1
77	6.622222	0.7265	19.8352	1003.68	9.982	1
78	6.683333	0.7223	20.0123	1003.86	9.982	1
79	6.088889	0.7211	20.0445	1003.57	9.982	1
80	6.066667	0.72	20.4148	1003.96	9.982	1
81	6.144444	0.7183	20.447	1004.1	9.982	1
82	7.133333	0.7231	20.5114	1004.26	9.982	1
83	7.205556	0.7199	20.5275	1004.18	9.982	1
84	7.566667	0.7145	20.6885	1004.23	9.982	1
85	8.9	0.7203	20.9622	1004.47	9.982	1
86	9.961111	0.7134	21.3808	1004.52	9.982	1
87	9.888889	0.6821	21.3969	1004.29	9.982	1
88	11.06667	0.7014	21.413	1003.93	9.982	1
89	10.11667	0.6997	21.9443	1004	9.982	1
90	11.03889	0.6821	22.0409	1004.3	9.982	1
91	10.65	0.6792	22.3951	1004	9.982	1
92	10.05	0.6758	22.7815	1004.39	9.982	1
93	9.9	0.66	23.2162	1005.48	9.982	1
94	8.794444	0.6674	23.4255	1005.89	9.982	1
95	7.827778	0.671	23.6992	1006.37	9.982	1
96	7.855556	0.6603	23.8924	1006.56	9.982	1
97	7.316667	0.6547	25.0355	1007.07	9.9015	1
98	7.244444	0.6674	25.3092	1007.37	8.05	1
99	5.438889	0.6553	25.3414	1012.23	7.9695	1
100	7.2	0.6547	25.4219	1007.28	7.6153	1
101	6.688889	0.6314	25.6956	1007.18	6.6976	1

102	6.211111	0.587	25.6956	1007.39	6.6976	1
103	6.111111	0.6436	26.5006	1007.67	6.3434	1
104	6.111111	0.6134	26.5328	1007.98	6.1985	1
105	6.172222	0.6031	26.9031	1008.57	6.1663	1
106	7.222222	0.5893	27.8691	1008.87	6.118	1
107	7.288889	0.5856	27.9818	1009.44	6.118	1
108	7.405556	0.5778	28.0945	1010.03	5.9731	1
109	7.961111	0.5772	28.1267	1010.22	5.6833	1
110	8.033333	0.5778	28.336	1010.39	5.5382	1
111	9.077778	0.5735	28.3682	1010.24	5.4383	1
112	9.05	0.574	28.4809	1010	4.8271	1
113	9.05	0.5689	28.5131	1009.98	4.7392	1
114	9.183333	0.5693	29.8333	1010.05	4.693493	1
115	9.072222	0.5674	30.8637	1010.1	4.5221	1
116	9.102812	0.45	32.1678	1010.39	4.512	1