City Prediction from Weather Data

July 7, 2019

1 Introduction

We have a weather dataset of 26 North American cities from roughly 1960 to 2015, downloaded from the Global Climatology Network (https://www.ncdc.noaa.gov/data-access/land-based-station-data/land-based-datasets/global-historical-climatology-network-ghcn). The dataset consists of the following monthly weather data for each city: 1. average maximum temperature(0.1řC) 2. average minimum temperature(0.1řC) 3. average precipitation(0.1mm) 4. average snowfall(mm) 5. average snow depth(mm)

Using Python and relevant data science libraries, We plan to build and train an accurate machine learning algorithm to classify given unlabelled weather data into their corresponding North American cities. We will try a few different algorithms, validate them, and then choose the one with the best results.

2 Set-up

We need to import Python's data science libraries and read the data file.

```
In [1]: import numpy as np
        import pandas as pd

        data = pd.read_csv('monthly-data-labelled.csv')
        #This is what the data looks like
        data
```

```
Out[1]:          city  year     tmax-01    tmax-02    tmax-03    tmax-04  \
        0     Anchorage  1960  -46.516129  -9.482759  -9.677419  52.400000
        1     Anchorage  1961  -26.096774 -44.571429 -35.064516  58.200000
        2     Anchorage  1962  -59.225806 -31.750000 -18.903226  69.366667
        3     Anchorage  1963  -39.290323 -11.357143  -1.451613  41.700000
        4     Anchorage  1964  -59.129032 -24.655172 -35.096774  45.866667
        5     Anchorage  1965  -81.193548 -73.607143  53.387097  84.266667
        6     Anchorage  1966  -82.387097 -47.285714 -36.419355  57.266667
        7     Anchorage  1967  -93.741935 -50.821429   1.290323  55.266667
        8     Anchorage  1968  -75.580645 -20.310345  17.000000  49.066667
        9     Anchorage  1969 -112.322581 -35.285714  15.677419  79.166667
        10    Anchorage  1970  -87.967742  15.107143  50.354839  59.233333
```

11	Anchorage	1971	-121.838710	-27.750000	-47.225806	41.366667
12	Anchorage	1972	-104.677419	-55.517241	-47.161290	10.400000
13	Anchorage	1973	-119.419355	-58.571429	-6.354839	51.766667
14	Anchorage	1974	-99.838710	-65.142857	1.322581	74.600000
15	Anchorage	1975	-69.548387	-62.642857	-9.483871	35.566667
16	Anchorage	1976	-54.193548	-62.241379	-15.032258	51.033333
17	Anchorage	1977	24.354839	35.750000	-2.129032	53.933333
18	Anchorage	1978	-28.419355	5.214286	24.580645	83.700000
19	Anchorage	1979	-19.967742	-64.035714	30.322581	71.966667
20	Anchorage	1980	-61.677419	9.586207	11.290323	86.900000
21	Anchorage	1981	28.161290	-10.357143	51.935484	72.600000
22	Anchorage	1982	-100.161290	-54.928571	4.322581	47.200000
23	Anchorage	1983	-63.516129	-23.500000	26.258065	66.000000
24	Anchorage	1984	-38.354839	-34.482759	65.322581	78.466667
25	Anchorage	1985	22.645161	-62.928571	12.548387	23.666667
26	Anchorage	1986	-7.161290	-18.785714	2.548387	36.500000
27	Anchorage	1987	-17.548387	0.321429	16.290323	76.866667
28	Anchorage	1988	-44.516129	-13.965517	27.258065	65.200000
29	Anchorage	1989	-116.322581	-37.928571	3.064516	83.100000
...	...	...	...	...	...	...
1129	Winnipeg	1973	-89.000000	-84.142857	46.741935	91.400000
1130	Winnipeg	1974	-170.741935	-111.464286	-50.483871	69.566667
1131	Winnipeg	1975	-108.548387	-108.321429	-56.225806	68.133333
1132	Winnipeg	1976	-121.387097	-57.965517	-39.096774	127.666667
1133	Winnipeg	1977	-160.580645	-65.785714	22.741935	160.133333
1134	Winnipeg	1978	-168.354839	-122.642857	-27.354839	87.133333
1135	Winnipeg	1979	-178.451613	-167.535714	-50.870968	38.800000
1136	Winnipeg	1980	-139.096774	-111.551724	-40.258065	161.000000
1137	Winnipeg	1981	-84.322581	-54.857143	44.741935	118.000000
1138	Winnipeg	1982	-198.806452	-95.321429	-24.645161	91.100000
1139	Winnipeg	1983	-77.935484	-60.750000	-12.322581	90.933333
1140	Winnipeg	1984	-108.483871	-24.586207	-18.709677	141.933333
1141	Winnipeg	1985	-126.032258	-114.321429	18.354839	138.633333
1142	Winnipeg	1986	-80.000000	-103.071429	8.451613	101.266667
1143	Winnipeg	1987	-70.290323	-25.571429	-1.806452	161.433333
1144	Winnipeg	1988	-131.096774	-98.931034	5.612903	135.433333
1145	Winnipeg	1989	-100.451613	-131.964286	-55.129032	82.233333
1146	Winnipeg	1990	-65.741935	-86.285714	3.645161	90.000000
1147	Winnipeg	1991	-131.354839	-54.035714	-5.225806	134.000000
1148	Winnipeg	1992	-67.967742	-53.655172	-1.096774	67.666667
1149	Winnipeg	1993	-95.526316	-94.238095	-19.807692	106.040000
1150	Winnipeg	1994	-186.741935	-130.892857	28.870968	105.800000
1151	Winnipeg	1995	-110.258065	-101.000000	-10.677419	65.500000
1152	Winnipeg	1996	-174.612903	-100.068966	-73.354839	30.833333
1153	Winnipeg	1997	-153.290323	-76.785714	-51.032258	41.766667
1154	Winnipeg	1998	-108.709677	-15.892857	-12.548387	155.833333
1155	Winnipeg	1999	-131.838710	-34.892857	27.193548	132.366667
1156	Winnipeg	2000	-114.774194	-36.344828	60.161290	118.300000

```
1157   Winnipeg  2001  -72.387097 -131.607143 -10.548387  109.600000
1158   Winnipeg  2002  -97.741935  -41.714286 -60.806452   81.833333

         tmax-05     tmax-06     tmax-07     tmax-08  ...     snwd-03  \
0     140.967742  173.166667  180.225806  168.064516  ...  290.903226
1     140.193548  169.633333  178.645161  161.806452  ...  113.096774
2     111.419355  159.633333  187.451613  176.483871  ...  128.645161
3     134.258065  146.200000  185.612903  182.129032  ...   60.645161
4      99.903226  173.566667  182.516129  163.483871  ...  114.793103
5     121.322581  150.200000  188.161290  176.870968  ...  227.741935
6     105.612903  170.966667  183.612903  153.483871  ...  476.967742
7     136.774194  166.766667  190.580645  176.483871  ...  285.935484
8     132.935484  170.233333  189.967742  187.580645  ...   40.806452
9     128.322581  184.066667  185.064516  167.548387  ...  253.322581
10    134.967742  159.266667  170.483871  157.354839  ...   27.806452
11     85.258065  145.933333  165.516129  161.774194  ...  241.838710
12     95.838710  145.666667  196.387097  172.774194  ...  239.225806
13    101.129032  146.566667  181.000000  153.322581  ...  352.322581
14    144.064516  175.566667  181.096774  180.709677  ...  243.290323
15    120.870968  155.466667  183.419355  170.129032  ...  386.741935
16    110.161290  164.966667  189.387097  170.935484  ...  174.483871
17    124.032258  187.200000  213.354839  195.000000  ...   49.870968
18    139.225806  160.733333  185.064516  203.225806  ...  323.677419
19    145.193548  177.600000  192.064516  185.645161  ...  379.354839
20    113.000000  152.233333  174.612903  161.548387  ...   24.483871
21    152.870968  162.533333  169.903226  157.935484  ...   35.290323
22    114.645161  155.200000  169.935484  168.225806  ...   46.580645
23    133.096774  179.700000  185.322581  171.419355  ...   33.483871
24    146.258065  190.733333  192.483871  175.258065  ...  274.677419
25    116.193548  149.733333  182.064516  165.612903  ...  261.290323
26    126.516129  170.200000  184.806452  155.612903  ...   88.451613
27    123.322581  144.366667  172.516129  184.516129  ...   99.903226
28    131.451613  166.133333  183.387097  168.483871  ...  336.903226
29    121.483871  174.300000  193.870968  183.451613  ...  132.032258
...          ...         ...         ...         ...  ...         ...
1129  183.290323  219.400000  244.129032  266.709677  ...   27.096774
1130  135.870968  246.200000  289.483871  226.516129  ...  271.935484
1131  190.064516  229.566667  284.387097  222.580645  ...  234.193548
1132  201.935484  241.700000  266.806452  263.258065  ...  269.032258
1133  244.032258  220.333333  256.548387  207.903226  ...  106.129032
1134  213.580645  227.366667  248.064516  241.548387  ...  133.870968
1135  126.387097  227.066667  274.548387  228.709677  ...  317.419355
1136  238.032258  240.466667  273.387097  225.419355  ...  201.034483
1137  187.645161  223.000000  268.322581  265.903226  ...    1.290323
1138  196.129032  205.900000  259.548387  230.387097  ...   16.451613
1139  159.161290  233.433333  280.580645  296.806452  ...   34.193548
1140  175.225806  223.833333  258.935484  282.387097  ...   20.645161
1141  206.354839  198.600000  257.064516  214.870968  ...  101.290323
```

```
1142  206.032258  231.433333  249.000000  244.064516  ...   87.741935
1143  225.709677  261.500000  257.000000  235.258065  ...  199.677419
1144  224.096774  294.400000  279.741935  270.419355  ...   18.064516
1145  214.161290  228.833333  283.225806  260.774194  ...  160.645161
1146  180.161290  238.466667  254.935484  274.806452  ...   31.612903
1147  202.451613  242.433333  250.967742  279.709677  ...   85.483871
1148  199.387097  207.066667  211.225806  221.967742  ...   39.677419
1149  182.227273  208.600000  222.612903  229.777778  ...  136.538462
1150  197.612903  237.700000  239.806452  232.419355  ...   14.193548
1151  178.516129  270.533333  259.193548  268.193548  ...   96.451613
1152  161.193548  251.366667  251.774194  261.193548  ...  431.200000
1153  154.000000  260.400000  251.516129  245.612903  ...  448.387097
1154  198.677419  205.000000  259.774194  279.806452  ...   60.645161
1155  177.870968  222.233333  254.741935  248.096774  ...   40.645161
1156  186.903226  200.400000  257.451613  256.451613  ...    1.290323
1157  189.612903  223.266667  263.516129  259.580645  ...  157.096774
1158  155.645161  243.166667  274.741935  246.161290  ...  104.838710

         snwd-04   snwd-05  snwd-06  snwd-07  snwd-08   snwd-09    snwd-10  \
0      44.066667  0.000000      0.0      0.0      0.0  0.000000   0.000000
1       8.433333  0.000000      0.0      0.0      0.0  0.000000  45.032258
2       5.866667  0.000000      0.0      0.0      0.0  0.000000   0.000000
3      78.766667  0.000000      0.0      0.0      0.0  0.000000   8.161290
4      53.266667  0.000000      0.0      0.0      0.0  0.000000  15.516129
5       1.666667  0.000000      0.0      0.0      0.0  0.833333  53.161290
6     102.400000  0.000000      0.0      0.0      0.0  0.000000   0.806452
7      35.533333  0.000000      0.0      0.0      0.0  0.000000  13.161290
8      20.233333  0.000000      0.0      0.0      0.0  0.000000  63.096774
9      48.233333  0.000000      0.0      0.0      0.0  0.000000   0.000000
10     10.966667  0.000000      0.0      0.0      0.0  0.000000   8.096774
11    105.766667  0.000000      0.0      0.0      0.0  0.000000  36.032258
12    292.066667  7.322581      0.0      0.0      0.0  0.000000   3.258065
13     63.500000  0.000000      0.0      0.0      0.0  0.000000   8.129032
14      0.833333  0.000000      0.0      0.0      0.0  0.000000   5.677419
15    260.733333  2.451613      0.0      0.0      0.0  0.000000   0.000000
16     88.066667  0.000000      0.0      0.0      0.0  0.000000  42.580645
17     35.533333  0.000000      0.0      0.0      0.0  0.000000  30.290323
18     47.433333  0.000000      0.0      0.0      0.0  0.000000   1.645161
19     89.733333  0.000000      0.0      0.0      0.0  0.000000   1.612903
20      0.000000  0.000000      0.0      0.0      0.0  0.000000  28.709677
21      1.666667  0.000000      0.0      0.0      0.0  0.000000   2.419355
22      2.500000  0.000000      0.0      0.0      0.0  0.000000  79.419355
23      5.900000  0.000000      0.0      0.0      0.0  0.000000  36.870968
24     14.300000  0.000000      0.0      0.0      0.0  0.000000   4.096774
25    213.400000  0.000000      0.0      0.0      0.0  0.000000   0.000000
26    106.600000  0.000000      0.0      0.0      0.0  0.000000   0.000000
27      0.000000  0.000000      0.0      0.0      0.0  0.000000   0.000000
28    134.666667  0.000000      0.0      0.0      0.0  0.000000  10.612903
```

29	5.100000	0.000000	0.0	0.0	0.0	0.000000	60.645161
...	...	...	...	...	...	...	...
1129	0.000000	0.000000	0.0	0.0	0.0	0.000000	0.000000
1130	130.000000	0.000000	0.0	0.0	0.0	0.000000	0.967742
1131	51.666667	0.967742	0.0	0.0	0.0	0.000000	0.967742
1132	4.333333	0.000000	0.0	0.0	0.0	0.000000	0.000000
1133	0.000000	0.000000	0.0	0.0	0.0	0.000000	0.000000
1134	1.000000	0.000000	0.0	0.0	0.0	0.000000	0.000000
1135	139.333333	1.290323	0.0	0.0	0.0	0.000000	0.000000
1136	0.000000	0.000000	0.0	0.0	0.0	0.000000	0.967742
1137	0.333333	0.000000	0.0	0.0	0.0	0.000000	1.935484
1138	2.666667	0.000000	0.0	0.0	0.0	0.000000	0.000000
1139	0.666667	0.000000	0.0	0.0	0.0	0.000000	0.000000
1140	1.666667	0.000000	0.0	0.0	0.0	0.666667	3.870968
1141	0.333333	0.000000	0.0	0.0	0.0	0.000000	7.419355
1142	1.666667	0.000000	0.0	0.0	0.0	0.000000	0.000000
1143	1.000000	0.000000	0.0	0.0	0.0	0.000000	2.903226
1144	0.000000	0.000000	0.0	0.0	0.0	0.000000	8.965517
1145	5.333333	0.000000	0.0	0.0	0.0	0.000000	0.000000
1146	1.666667	0.000000	0.0	0.0	0.0	0.000000	0.000000
1147	5.000000	3.548387	0.0	0.0	0.0	0.000000	4.193548
1148	3.333333	0.000000	0.0	0.0	0.0	0.000000	0.000000
1149	0.000000	0.000000	0.0	0.0	0.0	0.000000	0.000000
1150	3.666667	0.000000	0.0	0.0	0.0	0.000000	0.000000
1151	0.000000	0.000000	0.0	0.0	0.0	0.000000	0.000000
1152	208.666667	0.000000	0.0	0.0	0.0	0.000000	0.645161
1153	263.076923	0.322581	0.0	0.0	0.0	0.000000	0.000000
1154	0.000000	0.000000	0.0	0.0	0.0	0.000000	0.000000
1155	22.666667	0.000000	0.0	0.0	0.0	0.000000	0.000000
1156	0.000000	0.000000	0.0	0.0	0.0	0.000000	0.000000
1157	6.000000	0.000000	0.0	0.0	0.0	0.000000	0.000000
1158	0.333333	2.580645	0.0	0.0	0.0	0.000000	1.290323

	snwd-11	snwd-12
0	29.433333	77.612903
1	98.366667	147.258065
2	10.000000	46.548387
3	34.466667	18.032258
4	148.133333	345.870968
5	110.000000	458.774194
6	96.533333	287.580645
7	12.633333	234.354839
8	228.666667	364.774194
9	17.500000	34.322581
10	32.133333	111.580645
11	152.266667	240.129032
12	60.733333	242.483871
13	61.600000	137.612903

14	54.133333	241.677419
15	20.966667	85.193548
16	54.133333	176.096774
17	154.733333	176.967742
18	39.766667	360.354839
19	35.533333	95.774194
20	15.100000	26.677419
21	103.233333	172.806452
22	149.266667	210.516129
23	144.666667	208.225806
24	5.833333	88.612903
25	1.666667	1.612903
26	2.533333	42.548387
27	143.133333	490.806452
28	77.666667	182.677419
29	98.100000	153.193548
...	...	...
1129	94.666667	207.419355
1130	7.000000	44.193548
1131	7.000000	120.645161
1132	0.000000	74.516129
1133	49.333333	151.612903
1134	97.333333	232.258065
1135	27.000000	74.193548
1136	22.666667	24.193548
1137	1.666667	15.806452
1138	6.000000	22.258065
1139	22.333333	48.387097
1140	7.666667	79.032258
1141	58.000000	126.129032
1142	177.000000	131.612903
1143	0.000000	24.193548
1144	46.666667	85.161290
1145	15.333333	52.903226
1146	9.333333	81.290323
1147	49.333333	159.032258
1148	143.000000	253.333333
1149	16.000000	70.000000
1150	26.000000	75.517241
1151	120.333333	27.096774
1152	149.666667	409.354839
1153	26.666667	10.322581
1154	8.333333	66.774194
1155	4.333333	33.548387
1156	37.000000	208.064516
1157	0.666667	123.870968
1158	9.000000	45.483871

```
[1159 rows x 62 columns]
```

From this dataframe, we need a two-dimensional array consisting of weather data values without the city label, and another array consisting of the corresponding city labels. Machine learning models are trained using this format of dataset.

```
In [2]: X = data.drop('city', 1).values #remove 'city' column
        y = data['city'].values
```

3 Training and Validating

We will try three types of machine learning models: Naive Bayesian, k-Nearest Neighbours, and Support Vector Machine. We will create pipelines to scale each feature's values to lie between 0 and 1 before computing the classifications.

For each model, we will divide the data into training and validation sets. This is how we will find out the accuracy of the trained model - by testing the model on the validation set, which will have never been shown to the model during training.

```
In [3]: from sklearn.model_selection import train_test_split
```

3.1 Naive Bayesian (NB)

```
In [4]: from sklearn.pipeline import make_pipeline
        from sklearn.preprocessing import MinMaxScaler
        from sklearn.naive_bayes import GaussianNB

        n = 30 #number of iterations
        total = 0
        for i in range(n):
            X_train, X_valid, y_train, y_valid = train_test_split(X, y)
            bayes_model = make_pipeline(
                MinMaxScaler(),
                GaussianNB()
                )
            bayes_model.fit(X_train, y_train)
            total += bayes_model.score(X_valid, y_valid)

        print("bayes model score: " + str(total/n))
```

```
bayes model score: 0.6533333333333334
```

The bayes model has a score of roughly 65% accuracy, which is terrible. Why is this the case? NB algorithm assumes input variables to be independent, and also assumes input data to be distributed normally. None of these assumptions would hold true for weather data. For example, temperature is highly correlated with snowfall and snow depth. Moreoever, monthly temperatures throughout the decades have no reason to be normally distributed. This explains the model's low score.

3.2 k-Nearest Neighbours (kNN)

```
In [5]: from sklearn.neighbors import KNeighborsClassifier

        n_nbs = 5 #numer of neighbours to consider
        n = 30 #number of iterations
        total = 0
        for i in range(n):
            X_train, X_valid, y_train, y_valid = train_test_split(X, y)
            knn_model = make_pipeline(
                MinMaxScaler(),
                KNeighborsClassifier(n_neighbors=n_nbs)
                )
            knn_model.fit(X_train, y_train)
            total += knn_model.score(X_valid, y_valid)

        print("knn model score: " + str(total/n))
```

```
knn model score: 0.6677011494252874
```

The result is as bad as the NB approach. One guess as to why this algorithm is not effective is because we have mapped the values of all the features to the same scale. Since kNN measures the Euclidian distance between data points in the feature space to determine the class of the prediction input, scaling as mentioned results in all the features contributing approximately the same weight to the distance calcuation. We have no knowledge of which features most significantly affect classification, and it could be the case that some features should be more heavily weighed than others.

3.3 Support Vector Machine (SVM)

```
In [6]: from sklearn.svm import SVC

        n = 30 #number of iterations
        total = 0
        for i in range(n):
            X_train, X_valid, y_train, y_valid = train_test_split(X, y)
            svc_model = make_pipeline(
                MinMaxScaler(),
                SVC(kernel='rbf', C=5, gamma='scale', decision_function_shape='ovr')
                )
            svc_model.fit(X_train, y_train)
            total += svc_model.score(X_valid, y_valid)

        print("svc model score: " + str(svc_model.score(X_valid, y_valid)))
```

```
svc model score: 0.8482758620689655
```

Our SVM model's accuracy is much better than the other two. SVM is generally more robust to outliers/errors, since it considers all the outliers at the boundary between a pair of classes simultaneously. When more outliers are taken into account simultaneously, they tend to cancel out, which results in a good approximation of the boundary.

4 Conclusion

We trained three machine learning models (Naive Bayes, k-Nearest Neighbours, and Support Vector Machine) using a relatively small decades-long weather dataset with 60 features (5 weather measurements * 12 months). The models were trained to predict cities from weather data, then were evaluated for their accuracy. The SVM model had the top accuracy of around 80%, while the other two models had an accuracy of around 66%.

Our SVM model is far from being practically useful. The model can be improved by adding more rows; cities like Denver has fewer data points than the rest of the cities. Perhaps Denver's (and other cities') missing weather information can be researched and added to the table. Another way to improve the model is to add more relevant features to our data. Adding wind velocity and atmospheric pressure, for example, will significantly improve all our models' accuracy.

```
In [ ]:
```