I first tried using the RNN, since the lesson said that it worked better with temporal data and our ClimateWins dataset is from the 1960s to the 2000s. However, no matter how I adjusted the parameters the accuracy was down around 5 percent. The best I was able to get was 8 percent. I went back and used CNN instead.

First Attempt: I started with the parameters shown below. The accuracy with the first epoch was 8.31 percent with a loss of 21,417. My loss continued to grow with my accuracy never getting higher than 13 percent. I ended at 12.5 percent accuracy with a very large loss. The confusion matrix shows that the model was able to identify 14 of the 15 stations with these parameters.

```
#01 RNN Keras model
epochs = 50
batch_size = 16
n_hidden = 160

timesteps = len(X_train[0])
input_dim = len(X_train[0][0])
n_classes = len(y_train[0])

model = Sequential()
model.add(Conv1D(n_hidden, kernel_size=2, activation='relu', input_shape=(timesteps, input_dim)))
model.add(Dense(16, activation='relu'))
model.add(MaxPooling1D())
model.add(Flatten())
model.add(Dense(n_classes, activation='softmax'))
```

```
2]: #05 print matrix
print(confusion_matrix(y_test, model.predict(X_test)))
```

```
180/180 ━━━━━━━━━━━━━━━━━━━━ 0s 1ms/step
Pred        BASEL  BELGRADE  BUDAPEST  DUSSELDORF  HEATHROW  KASSEL  \
True
BASEL         116       622       878           2       835      48
BELGRADE        0       421       248           0       225       5
BUDAPEST        0        48        57           0        55       0
DEBILT          0        27         4           0        28       0
DUSSELDORF      0         5         1           0        10       0
HEATHROW        0         6         9           0        36       1
KASSEL          0         4         1           0         5       0
LJUBLJANA       0        10        19           0        17       0
MAASTRICHT      0         1         2           0         2       0
MADRID          1        28        80           0       163      10
MUNCHENB        0         4         2           0         0       0
OSLO            0         0         1           0         0       0
STOCKHOLM       0         1         2           0         0       0
VALENTIA        0         0         0           0         0       0

Pred        LJUBLJANA  MAASTRICHT  MADRID  MUNCHENB  OSLO  SONNBLICK  \
True
BASEL               6          62     467       196   101          6
BELGRADE            0           1      57         2    16          0
BUDAPEST            0           1      15         0     2          0
DEBILT              0           0       0         0     2          0
DUSSELDORF          0           0       2         0     2          0
HEATHROW            0           0       8         0     5          0
KASSEL              0           0       0         0     1          0
LJUBLJANA           0           0      12         0     0          0
MAASTRICHT          0           1       2         1     0          0
MADRID              0           0      97         3    24          0
MUNCHENB            0           0       0         1     1          0
OSLO                0           0       0         0     0          0
STOCKHOLM           0           0       0         0     1          0
VALENTIA            0           0       0         0     0          0

Pred        STOCKHOLM  VALENTIA
True
```

Second Attempt

I made the batch size slightly smaller and increased the hidden layers to 200. Each epoch took longer to run with the smaller batch size. The loss continued to get larger and larger as it ran. The final accuracy was 8.9 percent with a huge loss.

```
#01 RNN Keras model
epochs = 50
batch_size = 10
n_hidden = 200

timesteps = len(X_train[0])
input_dim = len(X_train[0][0])
n_classes = len(y_train[0])

model = Sequential()
model.add(Conv1D(n_hidden, kernel_size=2, activation='relu', input_shape=(timesteps, input_dim)))
model.add(Dense(16, activation='relu'))
model.add(MaxPooling1D())
model.add(Flatten())
model.add(Dense(n_classes, activation='softmax'))
```

Third attempt

Changed batch size back to 16, epoch at 50, hidden layers remained at 200 but changed activation to sigmoid. The loss still exponentially rose throughout the running of the model. The accuracy improved to 16.55 percent and it only identified a single station.

```
]: #05 print matrix
   print(confusion_matrix(y_test, model.predict(X_test)))
```

```
180/180 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
Pred        BASEL
True
BASEL        3682
BELGRADE     1092
BUDAPEST      214
DEBILT         82
DUSSELDORF     29
HEATHROW       82
KASSEL         11
LJUBLJANA      61
MAASTRICHT      9
MADRID        458
MUNCHENB        8
OSLO            5
STOCKHOLM       4
VALENTIA        1
```

Fourth attempt

Kept everything the same, except used tanh as the activation. The time the loss was a lot less, along with accuracy. The model never got any better with each epoch. It identified seven of the stations

```
1076/1076 - 2s - 2ms/step - accuracy: 0.0318 - loss: 23.3053
Epoch 42/50
1076/1076 - 2s - 2ms/step - accuracy: 0.0318 - loss: 23.3053
Epoch 43/50
1076/1076 - 2s - 2ms/step - accuracy: 0.0318 - loss: 23.3053
Epoch 44/50
1076/1076 - 2s - 2ms/step - accuracy: 0.0318 - loss: 23.3053
Epoch 45/50
1076/1076 - 2s - 2ms/step - accuracy: 0.0318 - loss: 23.3053
Epoch 46/50
1076/1076 - 2s - 2ms/step - accuracy: 0.0318 - loss: 23.3053
Epoch 47/50
1076/1076 - 2s - 2ms/step - accuracy: 0.0318 - loss: 23.3053
Epoch 48/50
1076/1076 - 2s - 2ms/step - accuracy: 0.0318 - loss: 23.3053
Epoch 49/50
1076/1076 - 3s - 3ms/step - accuracy: 0.0318 - loss: 23.3053
Epoch 50/50
1076/1076 - 2s - 2ms/step - accuracy: 0.0318 - loss: 23.3053
538/538 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - accuracy: 0.0309 - loss: 23.3991
Loss: 23.305326461791992 Accuracy 0.03178015351295471
```

Final attempt

I went back to the softmax activation. I increased the number of hidden layers to 200 in hopes of identifying all 15 stations. Was able to get all 15 stations but the loss was huge and the accuracy was only at 4.3 percent.

```python
#01 RNN Keras model
epochs = 50
batch_size = 16
n_hidden = 200

timesteps = len(X_train[0])
input_dim = len(X_train[0][0])
n_classes = len(y_train[0])

model = Sequential()
model.add(Conv1D(n_hidden, kernel_size=2, activation='relu', input_shape=(timesteps, input_dim)))
model.add(Dense(16, activation='relu'))
model.add(MaxPooling1D())
model.add(Flatten())
model.add(Dense(n_classes, activation='softmax'))
```

```
1076/1076 - 2s - 2ms/step - accuracy: 0.1152 - loss: 827534656.0000
Epoch 45/50
1076/1076 - 2s - 2ms/step - accuracy: 0.1114 - loss: 880139648.0000
Epoch 46/50
1076/1076 - 2s - 2ms/step - accuracy: 0.1134 - loss: 935311744.0000
Epoch 47/50
1076/1076 - 2s - 2ms/step - accuracy: 0.1106 - loss: 992528768.0000
Epoch 48/50
1076/1076 - 2s - 2ms/step - accuracy: 0.1125 - loss: 1052101504.0000
Epoch 49/50
1076/1076 - 2s - 2ms/step - accuracy: 0.1124 - loss: 1114025984.0000
Epoch 50/50
1076/1076 - 2s - 2ms/step - accuracy: 0.1145 - loss: 1178277632.0000
538/538 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - accuracy: 0.0440 - loss: 1237124992.0000
Loss: 1210752000.0 Accuracy 0.04369044676423073
```

[81]:
```
#04 define stations
print(confusion_matrix(y_test, model.predict(X_test)))
```

```
180/180 ━━━━━━━━━━━━━━━━━━━━ 0s 1ms/step
```

Pred / True	BASEL	BELGRADE	BUDAPEST	DEBILT	DUSSELDORF	HEATHROW	KASSEL
BASEL	44	131	718	3	31	807	89
BELGRADE	0	8	234	0	1	225	17
BUDAPEST	0	0	29	0	0	59	4
DEBILT	0	0	15	0	0	24	2
DUSSELDORF	0	0	2	0	0	14	0
HEATHROW	0	0	2	0	0	31	3
KASSEL	0	0	1	0	0	4	1
LJUBLJANA	0	0	10	0	0	18	1
MAASTRICHT	0	0	3	0	0	2	0
MADRID	0	3	35	1	1	152	38
MUNCHENB	0	0	3	0	0	0	0
OSLO	0	0	1	0	0	1	0
STOCKHOLM	0	0	1	0	0	0	0
VALENTIA	0	0	0	0	0	1	0

Pred / True	LJUBLJANA	MAASTRICHT	MADRID	MUNCHENB	OSLO	SONNBLICK
BASEL	779	4	664	56	232	8
BELGRADE	456	0	74	0	50	0
BUDAPEST	79	0	13	0	20	0
DEBILT	26	0	2	0	8	0
DUSSELDORF	5	0	3	0	4	0
HEATHROW	17	0	11	0	16	0
KASSEL	4	0	0	0	1	0
LJUBLJANA	18	0	10	0	2	0
MAASTRICHT	2	0	2	0	0	0
MADRID	62	0	95	0	57	0
MUNCHENB	4	0	0	0	1	0
OSLO	0	0	0	0	3	0
STOCKHOLM	2	0	0	0	1	0
VALENTIA	0	0	0	0	0	0

Pred / True	STOCKHOLM	VALENTIA
BASEL	78	38
BELGRADE	27	0
BUDAPEST	10	0
DEBILT	5	0