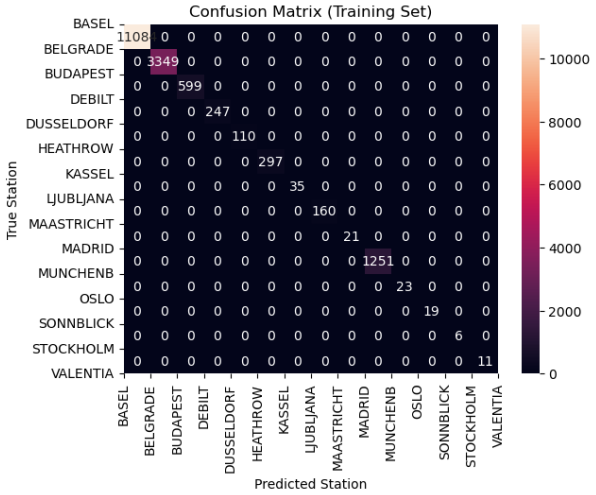
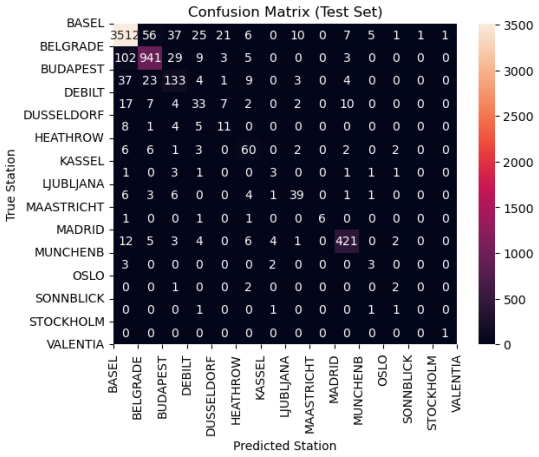
Decision Tree Model Performance





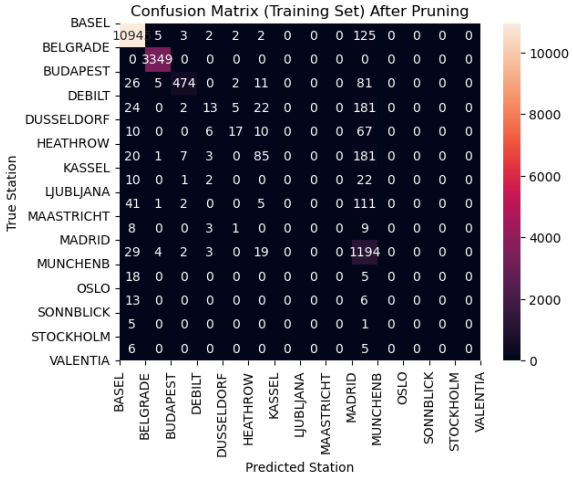
Initial training accuracy: **0.6034**

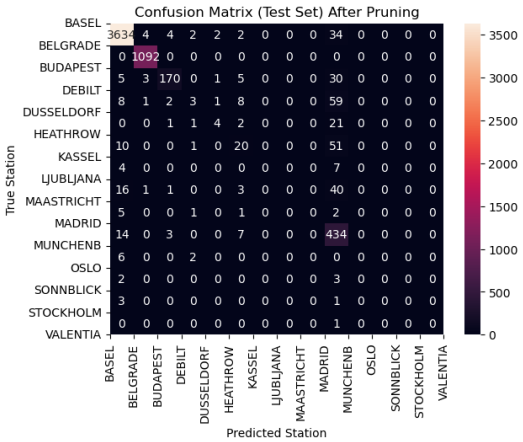
Initial testing accuracy: **0.5469**

Confusion matrix interpretation: First, we can assess the diagonal values that reflect true positives. In both the train and test set we see that **Basel, Belgrade and Madrid** have a lot of true positives. When looking at the matrices by row, any value off of the diagonal tells us there's a **false negative**: it was misclassified as another station. Lastly, looking at the matrices by column, any value off of the diagonal shows a **false positive**: it predicted it to be one station, when in fact it was another.

To prune, or not to prune?

Pruning is a technique that helps prevent overfitting; overfitting is apparent when the training data performs much better than the test data (the model is paying *too much* attention to details in the data which turn out to be *noise*, and not important to the underlying factors). There are two types of pruning: **pre- and post-pruning**. Pre-pruning stops the tree from growing too large and complex in the first place by imposing certain constraints (ie limiting the maximum depth of the tree or the minimum number of samples required to split a node); post-pruning occurs after that fact through removing nodes that contributed little to the overall model performance).





Training accuracy after pre-pruning: **0.9319**

Testing accuracy after pre-pruning: **0.9333**

Confusion matrix interpretation: After pruning, we see a slight improvement in true positives, as indicated by more values present in the diagonal. False negatives improved in some stations, such as **Basel and Madrid**. False positives improved as well, including in stations like **Belgrade**.

Classification Report (Training Set):

	precision	recall	f1-score	support
BASEL	0.98	0.99	0.98	11084
BELGRADE	1.00	1.00	1.00	3349
BUDAPEST	0.97	0.79	0.87	599
DEBILT	0.41	0.05	0.09	247
DUSSELDORF	0.63	0.15	0.25	110
HEATHROW	0.55	0.29	0.38	297
KASSEL	0.00	0.00	0.00	35
LJUBLJANA	0.00	0.00	0.00	160
MAASTRICHT	0.00	0.00	0.00	21
MADRID	0.60	0.95	0.74	1251
MUNCHENB	0.00	0.00	0.00	23
OSLO	0.00	0.00	0.00	19
STOCKHOLM	0.00	0.00	0.00	6
VALENTIA	0.00	0.00	0.00	11
accuracy			0.93	17212
macro avg	0.37	0.30	0.31	17212
weighted avg	0.92	0.93	0.92	17212

Training Accuracy: 0.9340576342086916

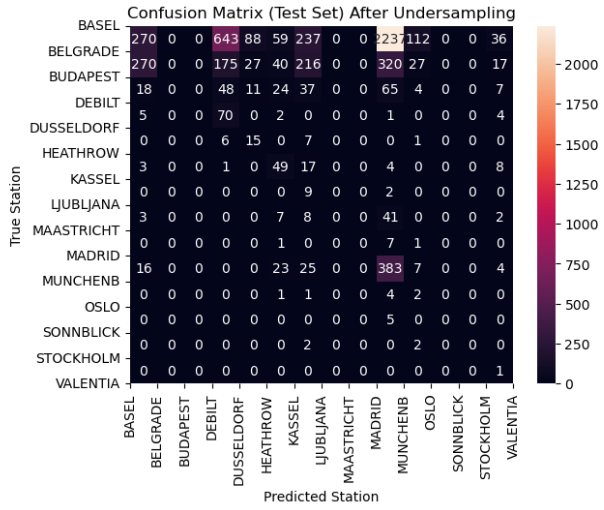
Classification Report (Test Set):

	precision	recall	f1-score	support
BASEL	0.98	0.99	0.98	3682
BELGRADE	0.99	1.00	1.00	1092
BUDAPEST	0.94	0.79	0.86	214
DEBILT	0.30	0.04	0.07	82
DUSSELDORF	0.50	0.14	0.22	29
HEATHROW	0.42	0.24	0.31	82
KASSEL	0.00	0.00	0.00	11
LJUBLJANA	0.00	0.00	0.00	61
MAASTRICHT	0.00	0.00	0.00	9
MADRID	0.64	0.95	0.76	458
MUNCHENB	0.00	0.00	0.00	8
OSLO	0.00	0.00	0.00	5
STOCKHOLM	0.00	0.00	0.00	4
VALENTIA	0.00	0.00	0.00	1
accuracy			0.93	5738
macro avg	0.34	0.30	0.30	5738
weighted avg	0.92	0.93	0.92	5738

Test Accuracy: 0.9336005576856048

Decision tree classification report (after pruning): Overall, the accuracy is strong in both sets, which tells us that overfitting is not occurring. However, as we dive deeper into the performance metrics, we see an inherent **class imbalance**, meaning that stations with fewer samples struggle more than stations with many samples. There's a discrepancy between the **macro and weighted average**, again reflecting the poor performance in smaller classes (thus, a lower macro average), and a high overall weighted average (the

larger classes dominate the overall performance). Other techniques can be applied here, such as undersampling the larger classes to balance the dataset, oversampling for smaller classes, or assigning class weights to give more importance to the smaller classes.

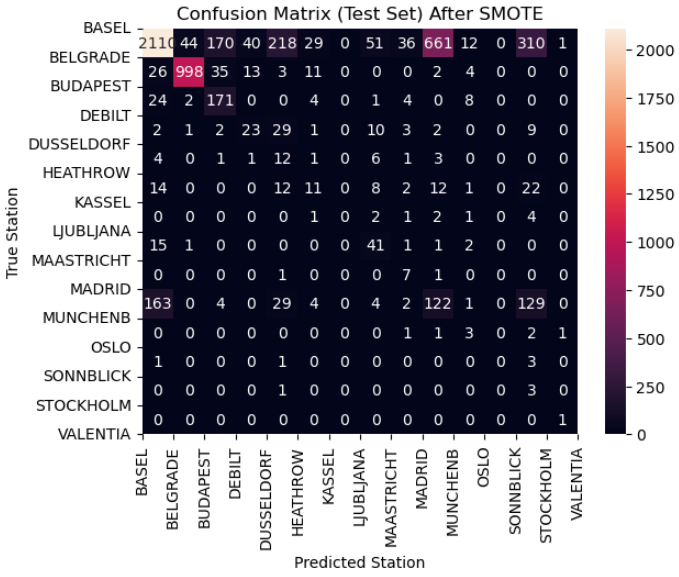



```
Class distribution after undersampling: [6 6 6 6 6 6 6 6 6 6 6 6 6 0 6 6]
Classification Report (Test Set) After Undersampling:

              precision    recall  f1-score   support

       BASEL       0.46      0.07      0.13      3682
    BELGRADE       0.00      0.00      0.00      1092
    BUDAPEST       0.00      0.00      0.00       214
      DEBILT       0.07      0.85      0.14        82
  DUSSELDORF       0.11      0.52      0.18        29
    HEATHROW       0.24      0.60      0.34        82
     KASSEL       0.02      0.82      0.03        11
   LJUBLJANA       0.00      0.00      0.00        61
  MAASTRICHT       0.00      0.00      0.00         9
      MADRID       0.12      0.84      0.22       458
    MUNCHENB       0.01      0.25      0.02         8
        OSLO       0.00      0.00      0.00         5
   STOCKHOLM       0.00      0.00      0.00         4
    VALENTIA       0.01      1.00      0.02         1

    accuracy                           0.14      5738
   macro avg       0.07      0.35      0.08      5738
weighted avg       0.31      0.14      0.11      5738

Test Accuracy after undersampling: 0.13924712443360054
```

Undersampling test accuracy: **0.139**

Overall performance: Much worse. More false positives and negatives occurred, especially in the **Belgrade, Budapest, and Madrid** stations. Let's try something else.

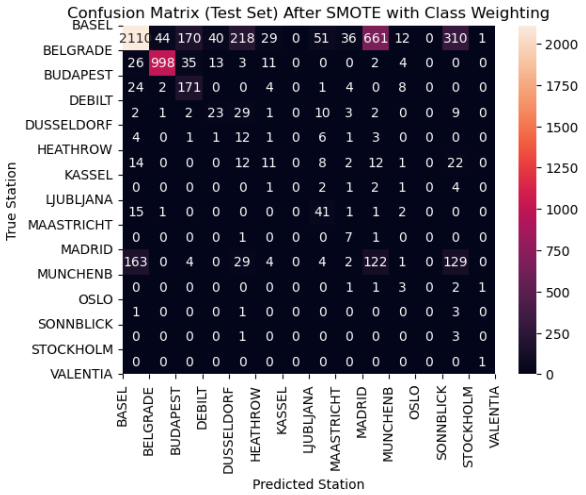
```
Classification Report (Test Set) After SMOTE:

              precision    recall  f1-score   support

       BASEL       0.89      0.57      0.70      3682
    BELGRADE       0.95      0.91      0.93      1092
    BUDAPEST       0.45      0.80      0.57       214
      DEBILT       0.30      0.28      0.29        82
  DUSSELDORF       0.04      0.41      0.07        29
    HEATHROW       0.18      0.13      0.15        82
     KASSEL       0.00      0.00      0.00        11
   LJUBLJANA       0.33      0.67      0.45        61
  MAASTRICHT       0.12      0.78      0.21         9
      MADRID       0.15      0.27      0.19       458
    MUNCHENB       0.09      0.38      0.15         8
        OSLO       0.00      0.00      0.00         5
   STOCKHOLM       0.01      0.75      0.01         4
    VALENTIA       0.33      1.00      0.50         1

    accuracy                           0.61      5738
   macro avg       0.27      0.50      0.30      5738
weighted avg       0.80      0.61      0.67      5738

Test Accuracy after SMOTE: 0.610317183687696
```



SMOTE test accuracy: **0.61**

Overall performance: The model has now misclassified a lot of the majority classes, reducing overall performance. The model did show improvements with minority classes, however, including **Debilt** and **Dusseldorf**. The weighted average F1-score is decent, at

0.67, but the macro average precision is low at 0.27. Perhaps a combo of SMOTE and class weighting will show better performance.

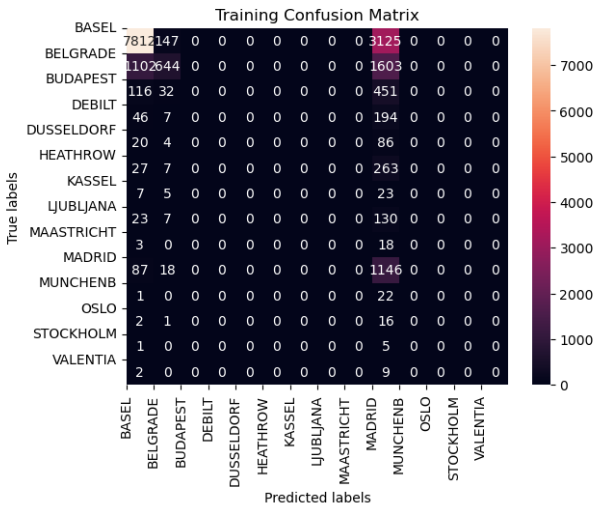



```
Classification Report (Test Set) After SMOTE with Class Weighting:

              precision    recall  f1-score   support

       BASEL       0.89      0.57      0.70      3682
    BELGRADE       0.95      0.91      0.93      1092
    BUDAPEST       0.45      0.80      0.57       214
      DEBILT       0.30      0.28      0.29        82
  DUSSELDORF       0.04      0.41      0.07        29
    HEATHROW       0.18      0.13      0.15        82
      KASSEL       0.00      0.00      0.00        11
   LJUBLJANA       0.33      0.67      0.45        61
  MAASTRICHT       0.12      0.78      0.21         9
      MADRID       0.15      0.27      0.19       458
    MUNCHENB       0.09      0.38      0.15         8
        OSLO       0.00      0.00      0.00         5
   STOCKHOLM       0.01      0.75      0.01         4
    VALENTIA       0.33      1.00      0.50         1

    accuracy                           0.61      5738
   macro avg       0.27      0.50      0.30      5738
weighted avg       0.80      0.61      0.67      5738

Test Accuracy after SMOTE with Class Weighting: 0.610317183687696
```

SMOTE and Class Weight test accuracy: **0.61**

Overall performance: The combo of SMOTE and class weighting has improved the recall for some minority classes like **Budapest** and **Ljubljana**. For majority classes like **Basel** and **Belgrade**, their precision is high but the recall is low, meaning it struggles to identify all instances. The overall accuracy is the same as before and is still below the second iteration, which used a pre-pruning technique. The model still struggles with the tradeoff between minority and majority classes.

Artificial Neural Network Performance

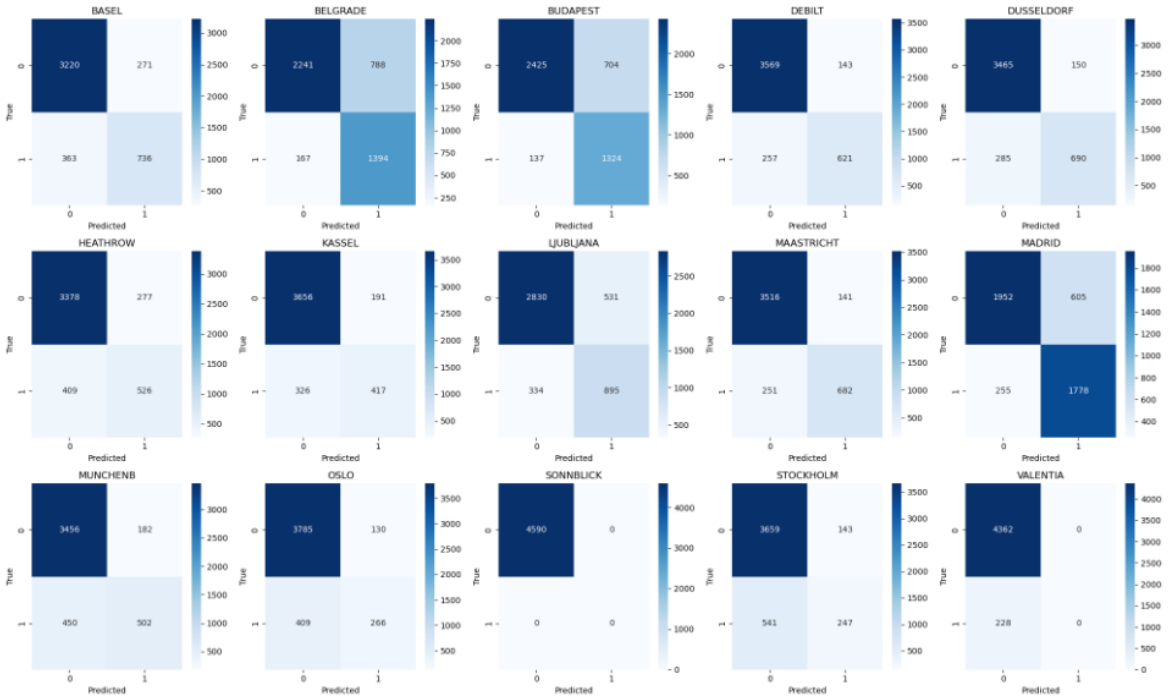



```
Classification Report (Training Set):

              precision    recall  f1-score   support

       BASEL       0.84      0.70      0.77     11084
    BELGRADE       0.74      0.19      0.31      3349
    BUDAPEST       0.00      0.00      0.00       599
      DEBILT       0.00      0.00      0.00       247
  DUSSELDORF       0.00      0.00      0.00       110
    HEATHROW       0.00      0.00      0.00       297
      KASSEL       0.00      0.00      0.00        35
   LJUBLJANA       0.00      0.00      0.00       160
  MAASTRICHT       0.00      0.00      0.00        21
      MADRID       0.16      0.92      0.27      1251
    MUNCHENB       0.00      0.00      0.00        23
        OSLO       0.00      0.00      0.00        19
   STOCKHOLM       0.00      0.00      0.00         6
    VALENTIA       0.00      0.00      0.00        11

    accuracy                           0.56     17212
   macro avg       0.12      0.13      0.10     17212
weighted avg       0.70      0.56      0.57     17212
```

Initial train accuracy: **0.5579**

Initial test accuracy: **0.5627**

Overall performance: Some stations have high precision (Basel) while others have very low values (Madrid); the same is true for recall, with high values for stations like Basel and Belgrade, and values of zero for Ljubljana, Munich, and Valentia. Overall, the model is struggling to classify minority classes, a similar problem to previous decision tree iterations.



Classification Report (Test Set - Unscaled):

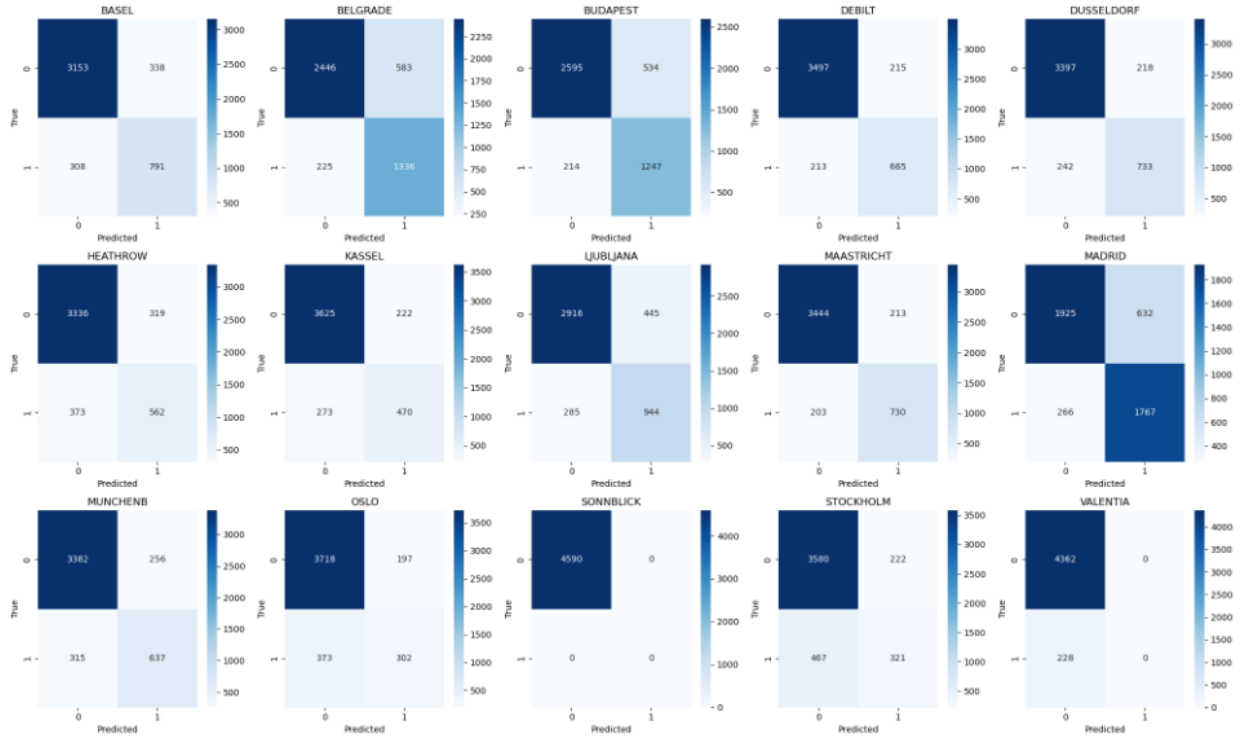
	precision	recall	f1-score	support
BASEL	0.73	0.67	0.70	1099
BELGRADE	0.64	0.89	0.74	1561
BUDAPEST	0.65	0.91	0.76	1461
DEBILT	0.81	0.71	0.76	878
DUSSELDORF	0.82	0.71	0.76	975
HEATHROW	0.66	0.56	0.61	935
KASSEL	0.69	0.56	0.62	743
LJUBLJANA	0.63	0.73	0.67	1229
MAASTRICHT	0.83	0.73	0.78	933
MADRID	0.75	0.87	0.81	2033
MUNCHENB	0.73	0.53	0.61	952
OSLO	0.67	0.39	0.50	675
SONNBLICK	0.00	0.00	0.00	0
STOCKHOLM	0.63	0.31	0.42	788
VALENTIA	0.00	0.00	0.00	228
micro avg	0.70	0.70	0.70	14490
macro avg	0.62	0.57	0.58	14490
weighted avg	0.70	0.70	0.68	14490
samples avg	0.35	0.34	0.33	14490

Unscaled test accuracy: **0.70**

Unscaled vs scaled performance: In this scenario, the unscaled data outperformed the scaled data, with higher precision, recall and F1 scores. Scaling had a particularly negative effect on stations like Belgrade and Budapest. I'm not entirely sure why scaling didn't improve the performance; perhaps because of low numeric variability already present in the data, the simplicity of the original ANN architecture (with 2 hidden layers of 5 neurons each), the compression of outliers with scaling (and those outliers having key features that improve the performance metrics), or other reasons.

ANN Unscaled Round 2: testing multiple configurations

	layers	iterations	tol	train_accuracy	test_accuracy
0	(5, 5)	200	0.00010	0.409967	0.421569
1	(10, 10)	500	0.00010	0.409205	0.419390
2	(50, 50)	500	0.00010	0.372767	0.377124
3	(100, 50, 25)	500	0.00010	0.485730	0.487582
4	(100, 50, 25)	1000	0.00001	0.407516	0.420697
5	(100, 50, 25)	1000	0.00100	0.408007	0.419826

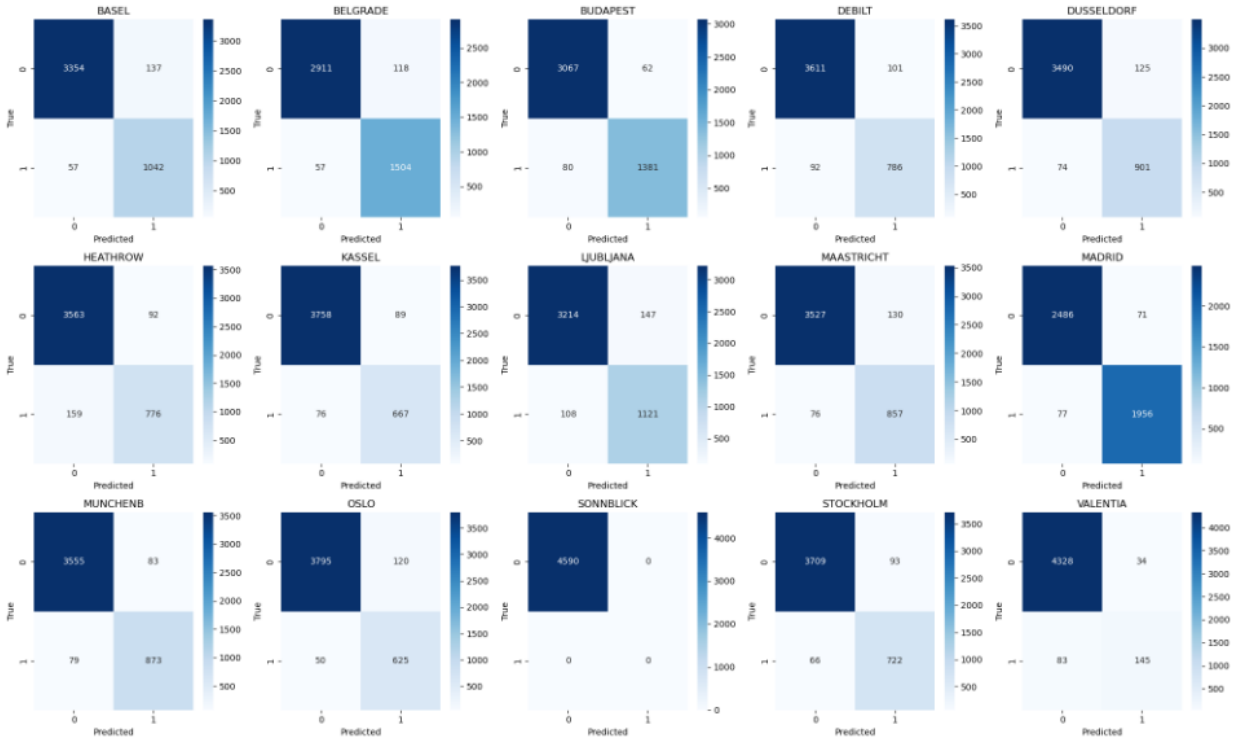


Interpretation: Configuration #3 proved to have the best train (**0.4857**) and test (**0.4875**) accuracy, with a **(100,50,25) architecture, 500 iterations and a tolerance of 0.0001**. This tells us the network had 3 hidden layers, with the first having 100 neurons; the second

having 50 neurons; the third having 25 neurons. 500 iterations mean the model went through the data 500 times to adjust its weights and learn the best patterns. A tolerance of 0.0001 indicates that the convergence criterion is high- the model tries harder to reduce the loss. Something to keep in mind is the larger the number of neurons in the hidden layers, and the higher the number of iterations, the more likely that overfitting can occur. However, in this case, we see that this did not occur, as the test and train accuracy are nearly identical.

Scaled ANN round 2: testing multiple configurations

	layers	iterations	tol	train_accuracy	test_accuracy
0	(5, 5)	200	0.00010	0.465196	0.478649
1	(10, 10)	500	0.00010	0.563181	0.570588
2	(50, 50)	500	0.00010	0.782843	0.637037
3	(100, 50, 25)	500	0.00010	0.893301	0.627233
4	(100, 50, 25)	1000	0.00001	0.866503	0.611111
5	(100, 50, 25)	1000	0.00100	0.894826	0.616558



Interpretation: Configuration #2 proved to have the best-combined train (**0.7828**) and test (**0.637**) accuracy, with a **(50,50) architecture, 500 iterations and a tolerance of 0.0001**. In this case, only two hidden layers with 50 neurons each produced the best test accuracy; while there were other configurations with better training accuracies, the test accuracy was lower for each one, an indication that overfitting was more of an issue in those models. The

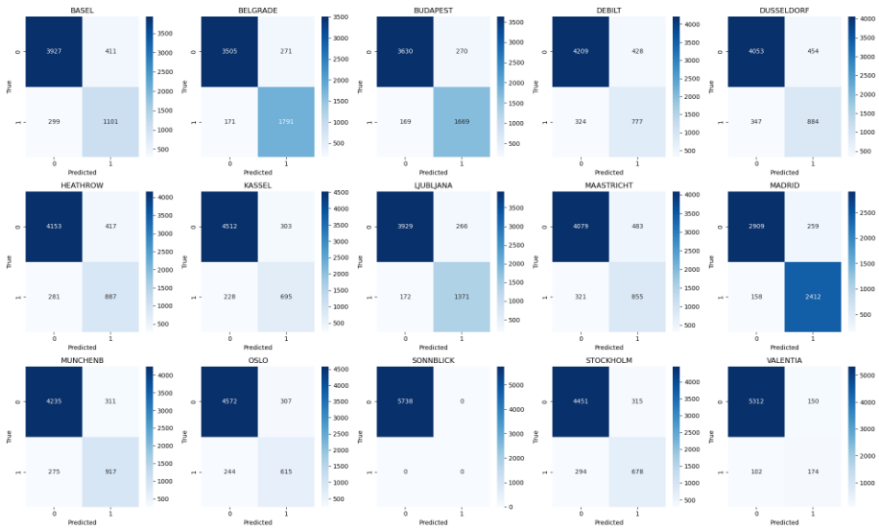
confusion matrix tells us that the model performs well on some stations (**Basel, Belgrade, Munchenb**), but other stations have a lot of misclassifications (**Budapest, Ljubljana**). Overall, this scaled configuration outperformed the best unscaled configuration, both in terms of test and training accuracy.

Across KNN, Decision tree, and ANN- Which model proved to be the best?

While none of the three models have proved to be perfect (as I'm sure is what happens in real life), with the different iterations I ran for the three different models, I believe the **decision tree model (second iteration, after pruning)**, proved to be the best-performing model. I say this because it yielded a test and train accuracy of **0.93** for each. While other performance metrics, like **weighted average and F1-scores**, were not as good (reflecting some class imbalance), on the whole, the model did well. No weather station was fully accurate in any iteration, however, **Sonnblick** performed consistently well, presenting with few misclassifications relative to the other stations. Perhaps this is due to distinct weather patterns or less inherent noise. Features that could have improved the model's performance include the presence of stations with balanced sample sizes, like **Basel and Belgrade**. Conversely, class imbalances in other stations, like **Valentia**, played a role in reducing accuracy.

Decision Tree Model- Revision

Class weighting round 1

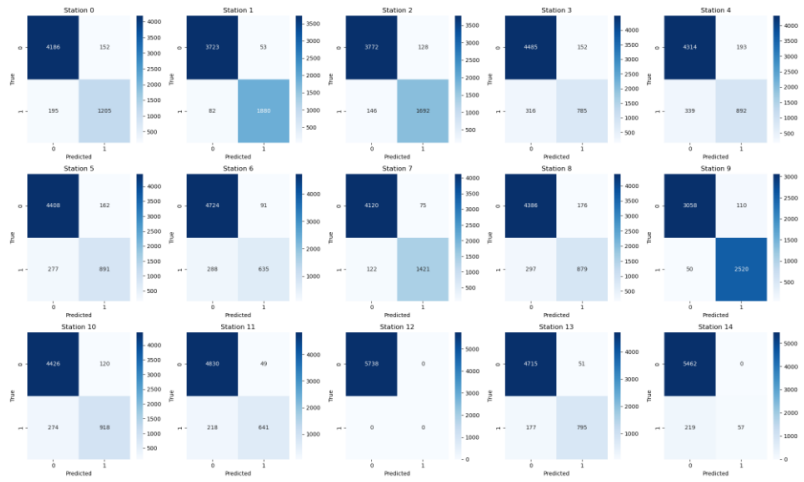


Classification Report with Class Weighting:

	precision	recall	f1-score	support
0	0.73	0.79	0.76	1400
1	0.87	0.91	0.89	1962
2	0.86	0.91	0.88	1838
3	0.64	0.71	0.67	1101
4	0.66	0.72	0.69	1231
5	0.68	0.76	0.72	1168
6	0.70	0.75	0.72	923
7	0.84	0.89	0.86	1543
8	0.64	0.73	0.68	1176
9	0.90	0.94	0.92	2570
10	0.75	0.77	0.76	1192
11	0.67	0.72	0.69	859
12	0.00	0.00	0.00	0
13	0.68	0.70	0.69	972
14	0.54	0.63	0.58	276
micro avg	0.76	0.81	0.79	18211
macro avg	0.68	0.73	0.70	18211
weighted avg	0.76	0.81	0.79	18211
samples avg	0.47	0.48	0.45	18211

Interpretation: The new model performs well for major stations like Basel, Sonnblick, Belgrade, and Madrid. However, it struggles with smaller stations like Valentia and Budapest. With a weighted average of **0.79**, we know the model is decent, but not at the level of the post-pruning model I ran previously.

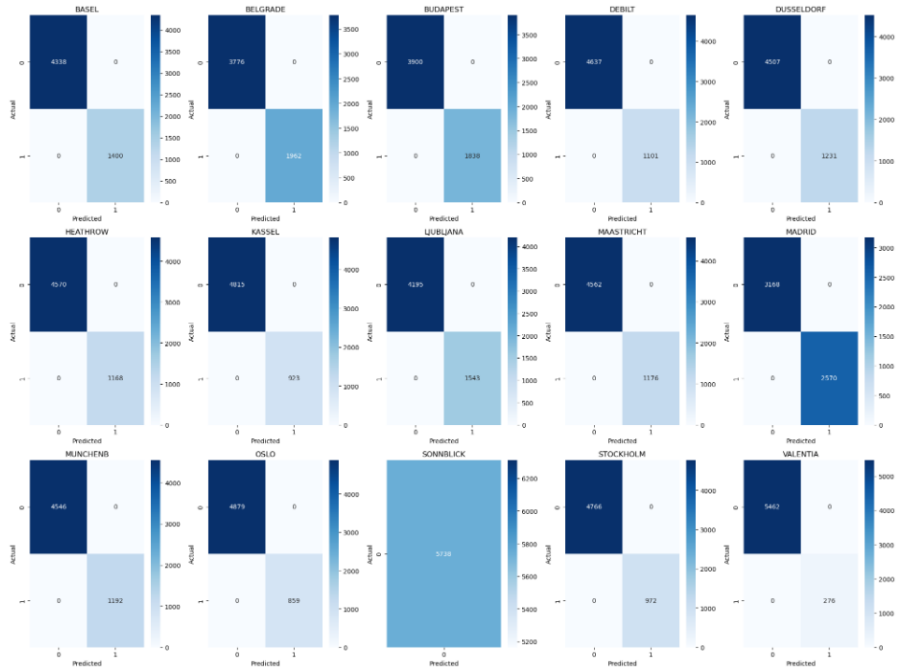
Class weighting round 2



```
Classification Report (with Class Balancing - Decision Tree):
              precision    recall  f1-score   support

           0       0.89      0.86      0.87      1400
           1       0.97      0.96      0.97      1962
           2       0.93      0.92      0.93      1838
           3       0.84      0.71      0.77      1101
           4       0.82      0.72      0.77      1231
           5       0.85      0.76      0.80      1168
           6       0.87      0.69      0.77       923
           7       0.95      0.92      0.94      1543
           8       0.83      0.75      0.79      1176
           9       0.96      0.98      0.97      2570
          10       0.88      0.77      0.82      1192
          11       0.93      0.75      0.83       859
          12       0.00      0.00      0.00         0
          13       0.94      0.82      0.87       972
          14       1.00      0.21      0.34       276

   micro avg       0.91      0.84      0.87     18211
   macro avg       0.84      0.72      0.76     18211
weighted avg       0.91      0.84      0.87     18211
 samples avg       0.53      0.47      0.49     18211
```

Interpretation: This model performs better than the previous iteration, however, it is still not as good as the pre-pruning version. The pre-pruning version had much higher test accuracy, meaning it generalized better to unseen data. This model attempted to handle class balance better, but it came at the cost of overall accuracy.

Class weighting round 3



	precision	recall	f1-score	support
0	0.73	0.79	0.76	1400
1	0.88	0.91	0.89	1962
2	0.87	0.90	0.88	1838
3	0.63	0.71	0.67	1101
4	0.66	0.72	0.69	1231
5	0.68	0.76	0.72	1168
6	0.68	0.76	0.72	923
7	0.84	0.88	0.86	1543
8	0.65	0.72	0.68	1176
9	0.90	0.93	0.92	2570
10	0.74	0.76	0.75	1192
11	0.66	0.70	0.68	859
12	0.00	0.00	0.00	0
13	0.68	0.69	0.68	972
14	0.57	0.63	0.59	276
micro avg	0.76	0.81	0.78	18211
macro avg	0.68	0.72	0.70	18211
weighted avg	0.76	0.81	0.79	18211
samples avg	0.47	0.48	0.45	18211

Interpretation: The weighted averages for precision, recall and F1-score are good, hovering between 0.76 and 0.81 and an overall test accuracy of **0.76**. Still though, larger classes like Belgrade and Budapest had better scores than the smaller ones, like Kassel and Stockholm. Overall, the pruning iteration from before had a more balanced performance and is the better-performing model.

ANN- Revision

	layers	iterations	tol	train_accuracy	test_accuracy
0	(50, 50)	500	0.0001	0.445155	0.442140
1	(50, 50)	500	0.0010	0.445155	0.442140
2	(50, 50)	1000	0.0001	0.445155	0.442140
3	(50, 50)	1000	0.0010	0.445155	0.442140
4	(100, 50)	500	0.0001	0.340344	0.347159
5	(100, 50)	500	0.0010	0.340344	0.347159
6	(100, 50)	1000	0.0001	0.340344	0.347159
7	(100, 50)	1000	0.0010	0.340344	0.347159
8	(100, 50, 25)	500	0.0001	0.413374	0.412513
9	(100, 50, 25)	500	0.0010	0.413374	0.412513
10	(100, 50, 25)	1000	0.0001	0.413374	0.412513
11	(100, 50, 25)	1000	0.0010	0.413374	0.412513
12	(150, 100, 50)	500	0.0001	0.457123	0.454862
13	(150, 100, 50)	500	0.0010	0.457123	0.454862
14	(150, 100, 50)	1000	0.0001	0.457123	0.454862
15	(150, 100, 50)	1000	0.0010	0.457123	0.454862

	layers	train_accuracy	test_accuracy
0	(50, 50)	0.775331	0.617463
1	(100, 50)	0.817279	0.594284
2	(50, 50, 25)	0.763886	0.629313

	layers	train_accuracy	test_accuracy
0	(200, 150, 100, 50)	0.091738	0.099861
1	(250, 200, 150, 100)	0.450267	0.445974
2	(300, 250, 200, 150)	0.491924	0.457128
3	(400, 300, 200, 100)	0.629793	0.487975

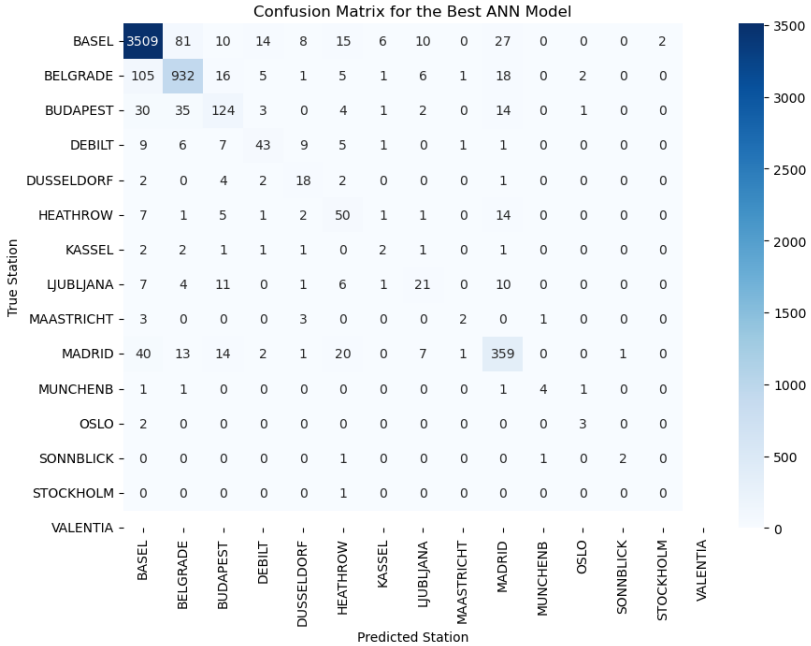
	alpha	train_accuracy	test_accuracy
0	0.0001	0.629793	0.487975
1	0.0010	0.610214	0.493726
2	0.0100	0.448292	0.439526
3	0.1000	0.432605	0.438132

	tol	learning_rate	train_accuracy	test_accuracy
0	0.00001	0.0010	0.768475	0.634890
1	0.00001	0.0005	0.764757	0.618508
2	0.00010	0.0010	0.768475	0.634890
3	0.00010	0.0005	0.764757	0.618508
4	0.00100	0.0010	0.768475	0.634890
5	0.00100	0.0005	0.764757	0.618508

Interpretation: As you can see above, I tried several approaches to fine-tune the ANN, including layer adjustments, learning rate tuning, tolerance adjustments, and alpha regularization. Most of these yielded poorer results than the best iteration I found before the revisions, which yielded a test accuracy of **0.64**. However, the one approach with the closest performance was using a less-complex layering, at 50,50 (telling us that a more complex architecture leads to diminishing returns); a low tolerance of 0.00001, and a learning rate of 0.001. This yielded a test accuracy of **0.635**. Still, this did not improve on the previous best iteration.

```
Test Accuracy for the Best ANN Model: 0.6124085047054723
Classification Report for the Best ANN Model:
```

	precision	recall	f1-score	support
0	0.92	0.92	0.92	1400
1	0.93	0.93	0.93	1962
2	0.90	0.95	0.93	1838
3	0.87	0.85	0.86	1101
4	0.90	0.91	0.90	1231
5	0.79	0.92	0.85	1168
6	0.87	0.89	0.88	923
7	0.85	0.91	0.88	1543
8	0.93	0.86	0.89	1176
9	0.96	0.96	0.96	2570
10	0.83	0.97	0.90	1192
11	0.85	0.87	0.86	859
12	0.00	0.00	0.00	0
13	0.90	0.84	0.87	972
14	0.74	0.71	0.73	276
micro avg	0.89	0.91	0.90	18211
macro avg	0.82	0.83	0.82	18211
weighted avg	0.89	0.91	0.90	18211
samples avg	0.52	0.53	0.51	18211



Last remarks: The classification report tells us that, once again, the model performs well on stations that are well-represented, and poorly on underrepresented stations (like Sonnblick and Valentia); this tells us that the perennial issue of class imbalance is still playing a major part. This is reflected in the confusion matrix, where stations like Basel and Belgrade perform well. Also, with test accuracy being a step lower than training accuracy in most cases, this tells us that **overfitting** is still playing a part in the model performance as well. Overall, the **pre-pruning decision tree model** still reigns supreme.