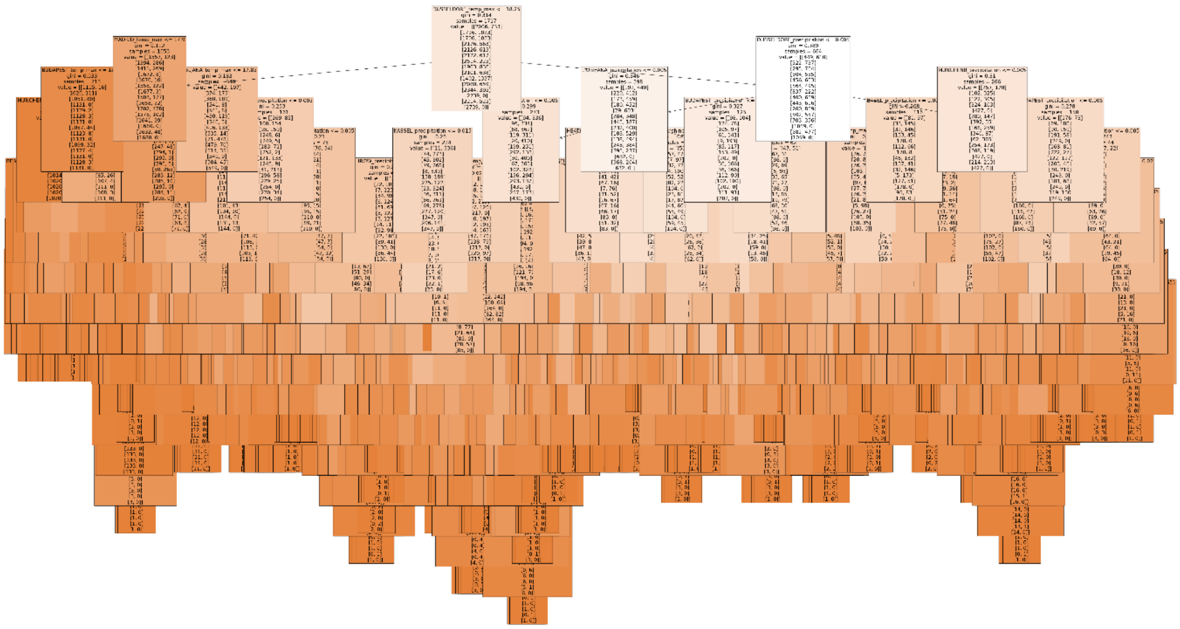
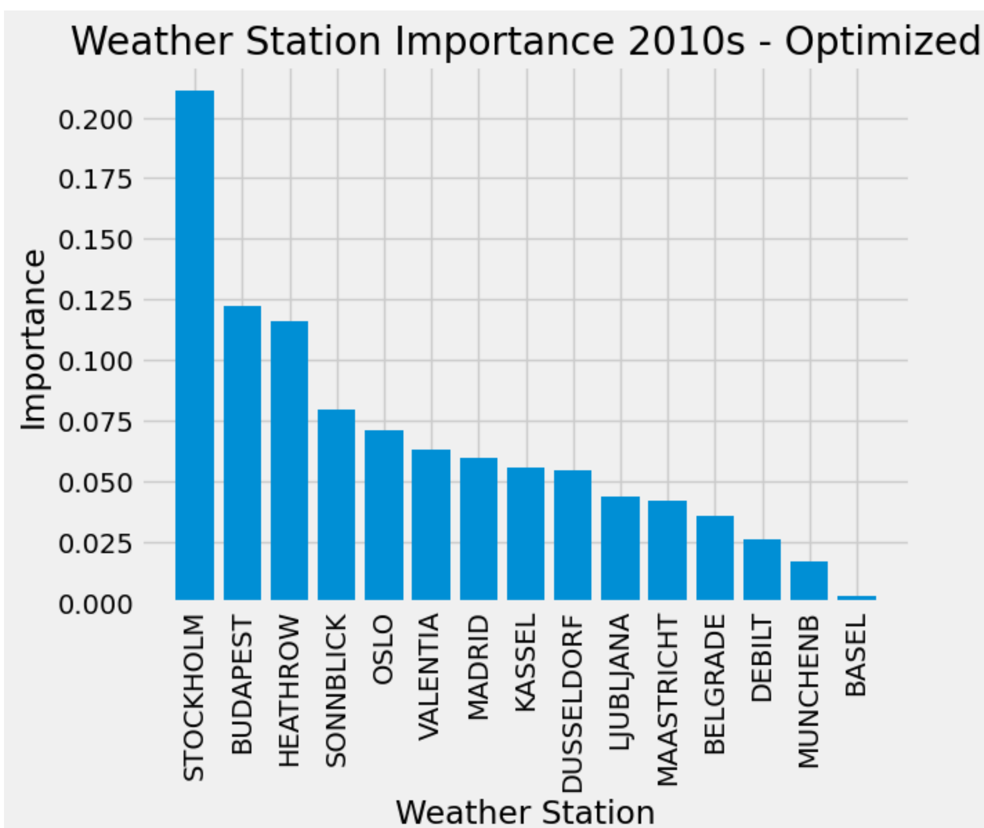
Part 1: Grid Search and Random Search Optimization for Random Forest

Tree Figure and Feature Importance for Optimized Random Forest (2010s)





Optimized Hyperparameters

Max depth: 60

Max features: 69

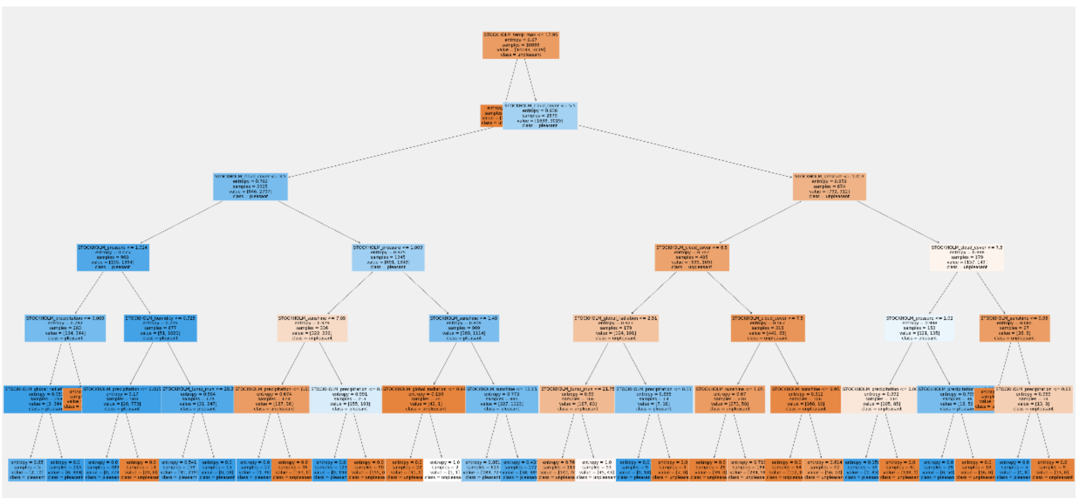
Min samples leaf: 1

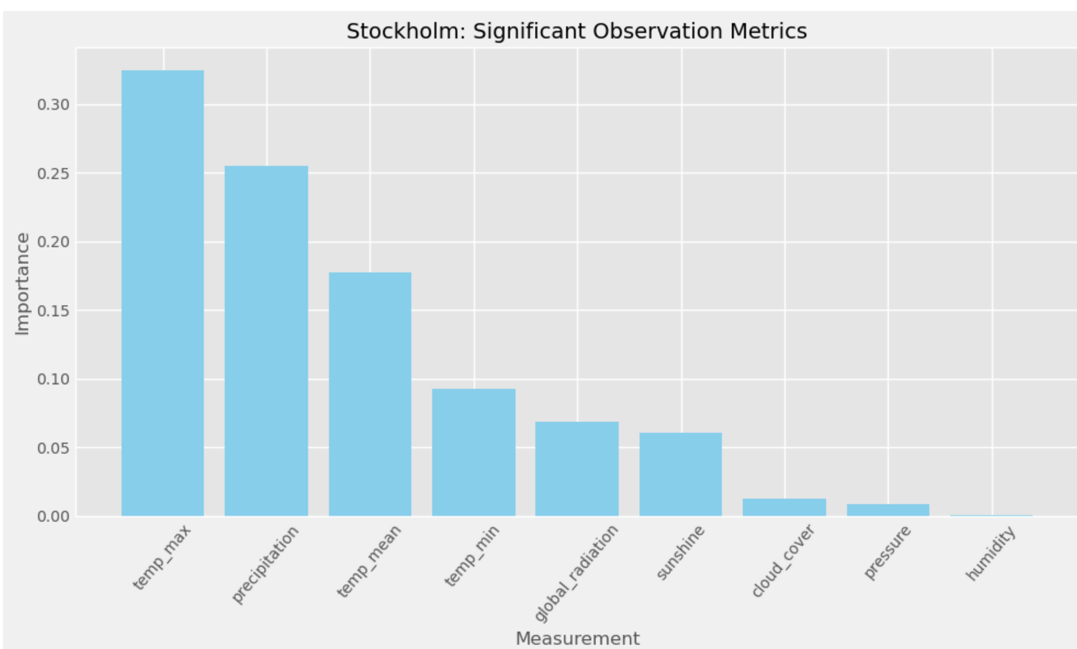
Min samples split: 2

# of estimators: 175

Criterion: gini

Tree Figure and Feature Importance for Optimized Random Forest (Stockholm)





Optimized Hyperparameters

Max depth: 6

Max features: 3

Min samples leaf: 2

Min samples split: 8

# of estimators: 150

Criterion: entropy

Comparison to previous findings: After running both a grid search and random search for the hyperparameters, the most important stations (features) for a **decade of data** were different compared to the previous iteration. This time, **Stockholm, Budapest and Heathrow** were the top 3, instead of **Maastricht, Kassel and Ljubljana**. On top of that, the overall model accuracy improved from **0.597** previously to **0.665** this time around, reflecting that the new hyperparameters have indeed improved the model.

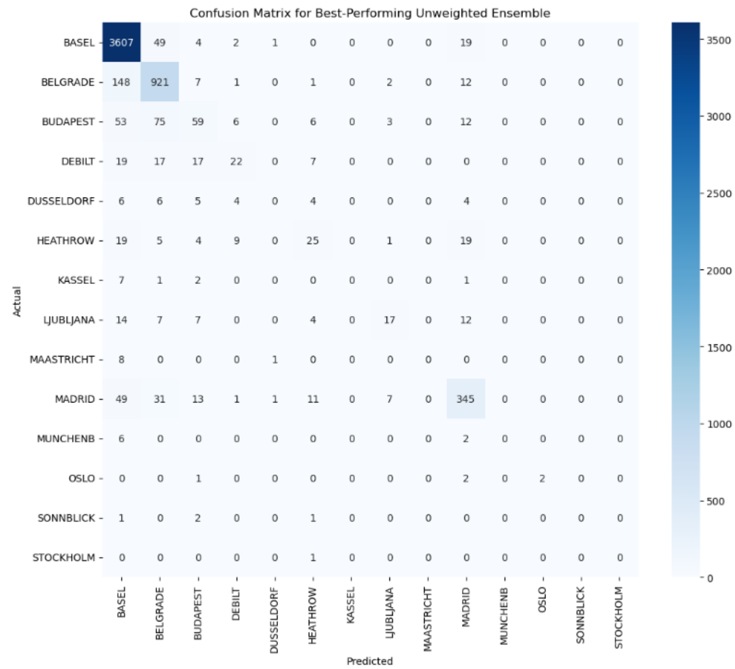
For the random forest of **one station** across all years (Stockholm), the top 3 features did not change compared to pre-optimization, however their order did change. Instead of precipitation being the top variable, this new optimized model put **max temperature at the top, followed by precipitation and mean temperature**. With that being the case, I would say that this is not a significant change from the previous iteration, instead it is a slight update in priorities. Concerningly, the model accuracy was once again **1.0**, telling us that there could be more overfitting taking place. As such, we need to keep in mind the same information I shared in the previous assignment- taking measures to validate the model in other ways prior to trusting it completely.

Part 2: Bayesian Optimization for Deep Learning Models

```
Classification Report:
              precision    recall  f1-score   support

       BASEL       0.92      0.98      0.95      3682
    BELGRADE       0.83      0.84      0.84      1092
    BUDAPEST       0.49      0.28      0.35       214
      DEBILT       0.49      0.27      0.35        82
  DUSSELDORF       0.00      0.00      0.00        29
    HEATHROW       0.42      0.30      0.35        82
      KASSEL       0.00      0.00      0.00        11
   LJUBLJANA       0.57      0.28      0.37        61
  MAASTRICHT       0.00      0.00      0.00         9
      MADRID       0.81      0.75      0.78       458
    MUNCHENB       0.00      0.00      0.00         8
        OSLO       1.00      0.40      0.57         5
   SONNBLICK       0.00      0.00      0.00         4
   STOCKHOLM       0.00      0.00      0.00         1

    accuracy                           0.87      5738
   macro avg       0.39      0.29      0.33      5738
weighted avg       0.85      0.87      0.86      5738
```



Analysis: Once again, the model does well with major classes like Basel and Belgrade, which resulted in high precision and recall. Smaller classes like Dusseldorf, Kassel, Stockholm, and Sonnblick struggle, and therefore have low precision and recall. This is also reflected in the confusion matrix, which shows a lot of correct classifications for the major stations and many misclassifications for the smaller ones. Compared to the previous best-performing model, this shows that optimization did indeed help with model performance- the overall accuracy is higher, but also the macro and weighted averages are better as well.

Subsequent attempts: Instead of stopping at this model, I tried a few more iterations to try to address the class imbalance, first trying a class weight approach, then random oversampling, SMOTE, an ensemble of models, and further hyperparameter tuning. This did not improve upon overall model performance. The problem of class imbalance seems to be a difficult one to address.

Part 3: Iteration

Component testing: Given the nature of the climate dataset, with 15 different weather stations collecting various variables over multiple decades, there appears to be a few ways of breaking down the set: temporally, looking at data from individual years or decades; also spatially, analyzing individual station performance over time. This is what I did with the random forest models- analyzing one decade first, the 2010s, then looking at individual stations after that. This was not done with the deep learning models, but it could be useful to do so, given that the random forests for individual stations routinely gave a training accuracy of 1.0, which is highly suggestive of overfitting.

Variables to pay attention to: Since the same three features were prominent regardless of location, I think it's safe to say that max temperature, mean temperature and precipitation are the top three variables to pay attention to when deciding whether it's safe to fly. Precipitation is an obvious one- if it looks like it'll rain or snow, then flying conditions may not be ideal. For the temperature variables, I think seeing a lower temperature, under freezing, in combo with the presence of precipitation clouds, can tell us that the likelihood of snowfall is high.