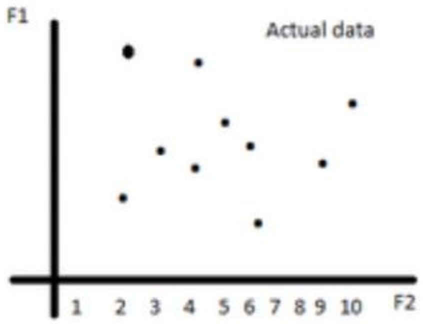
Room Occupancy Estimation

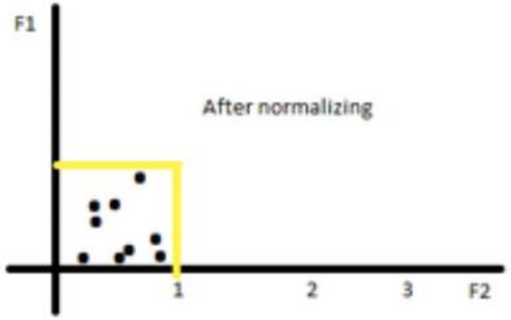
Crosby Pineda

Dataset

- Dataset for estimating the number of people in a room using multiple sensors for temperature, light, sound, CO2 and digital passive infrared(PIR). Using this to predict the number of people in a room over a span of 4 days ranging from 0 to 3. A little data prep is to normalize it as well as dealing with time

S4_Light	Feature	Integer
S1_Sound	Feature	Continuous
S2_Sound	Feature	Continuous
S3_Sound	Feature	Continuous
S4_Sound	Feature	Continuous
S5_CO2	Feature	Integer
S5_CO2_Slope	Feature	Continuous
S6_PIR	Feature	Binary
S7_PIR	Feature	Integer





Clustering

I wanted to do clustering to see how the data looked. For simplification I used Light and passive infrared(PIR) as that should be a strong correlation.

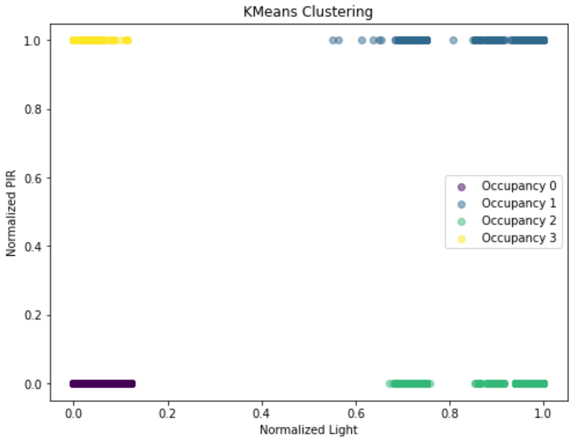
Silhouette Scores:

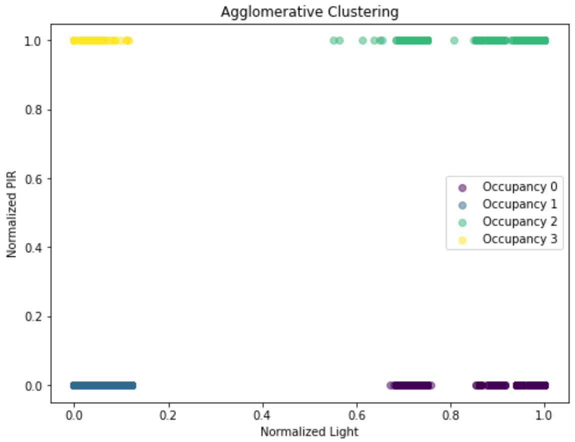
K-Means: 0.95

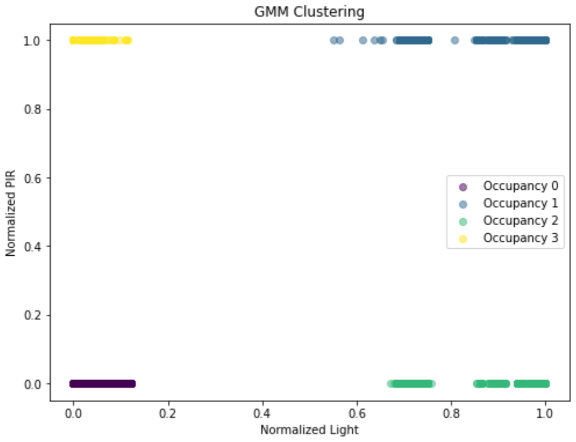
Agglomerative Clustering : 0.95

Gaussian Mixture Model : 0.95

Very high scores closer to the ideal 1. As it can range from -1 to 1.

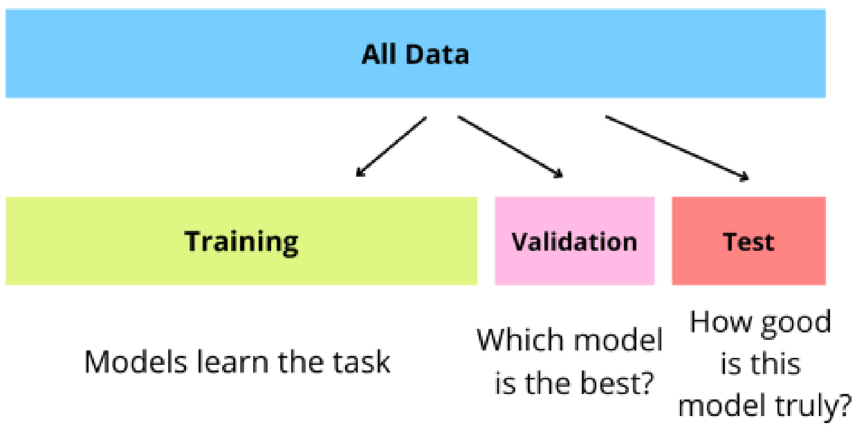






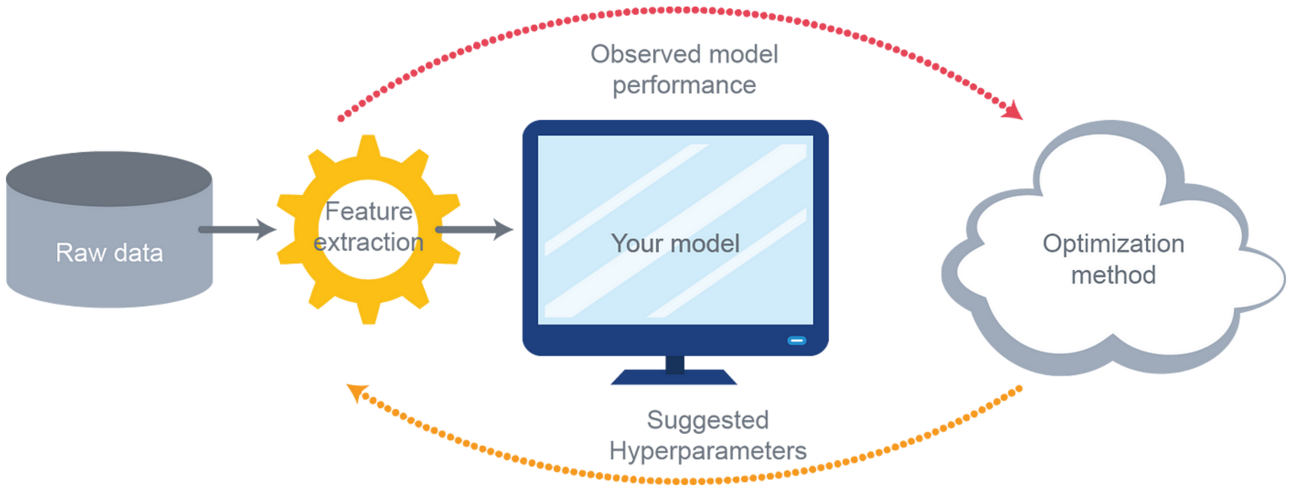
Classification

Naturally, this is a classification problem, where there are 17 features correlating with 0-3 people in a room.
I used Random Forest, Logistic Regression, Support Vector Machine, and Gradient Boosting. Making a 60% training, 20% validation, and 20% test split.



Hyper-Parameter Optimization

To have the models perform at their best I want to optimize the parameter. I used Grid Search Cross-Validation in order to parallelize the range of parameter values to optimize the model's performance. Using CPU cores to increase the search process . The best performing parameters are then selected for each classifier based on the highest accuracy achieved during cross-validation.



Hyper-Parameter Optimization

The HPO results show optimal settings for each classifier's performance.

For Random Forest, it only needed 50 trees and minimal splitting of 5, meaning that a less complex model avoids over fitting.

Logistic Regression regularization strength of C = 100, showing that stronger regularization helped.

SVM achieved the best results with a high penalty parameter C = 10 as well and a linear kernel, meaning that a linear decision boundary is all that was needed.

Gradient Boosting favored a good learning rate and depth with 100 trees, to help balance learning efficiently and avoiding overfitting.

Dask

I used Dask in data processing, data transformations with Dask Array, Parallel Hyper parameter tuning, parallel model training and prediction, and normalizing.

Strong Scaling Test:

Workers: 1, Time: 0.6483 seconds

Workers: 2, Time: 0.4116 seconds

Workers: 4, Time: 0.6338 seconds

Workers: 8, Time: 0.8900 seconds

Weak Scaling Test:

Workers: 1, Problem Size: 1000x1000, Time: 0.3240 seconds

Workers: 2, Problem Size: 2000x2000, Time: 0.4266 seconds

Workers: 4, Problem Size: 4000x4000, Time: 0.8184 seconds

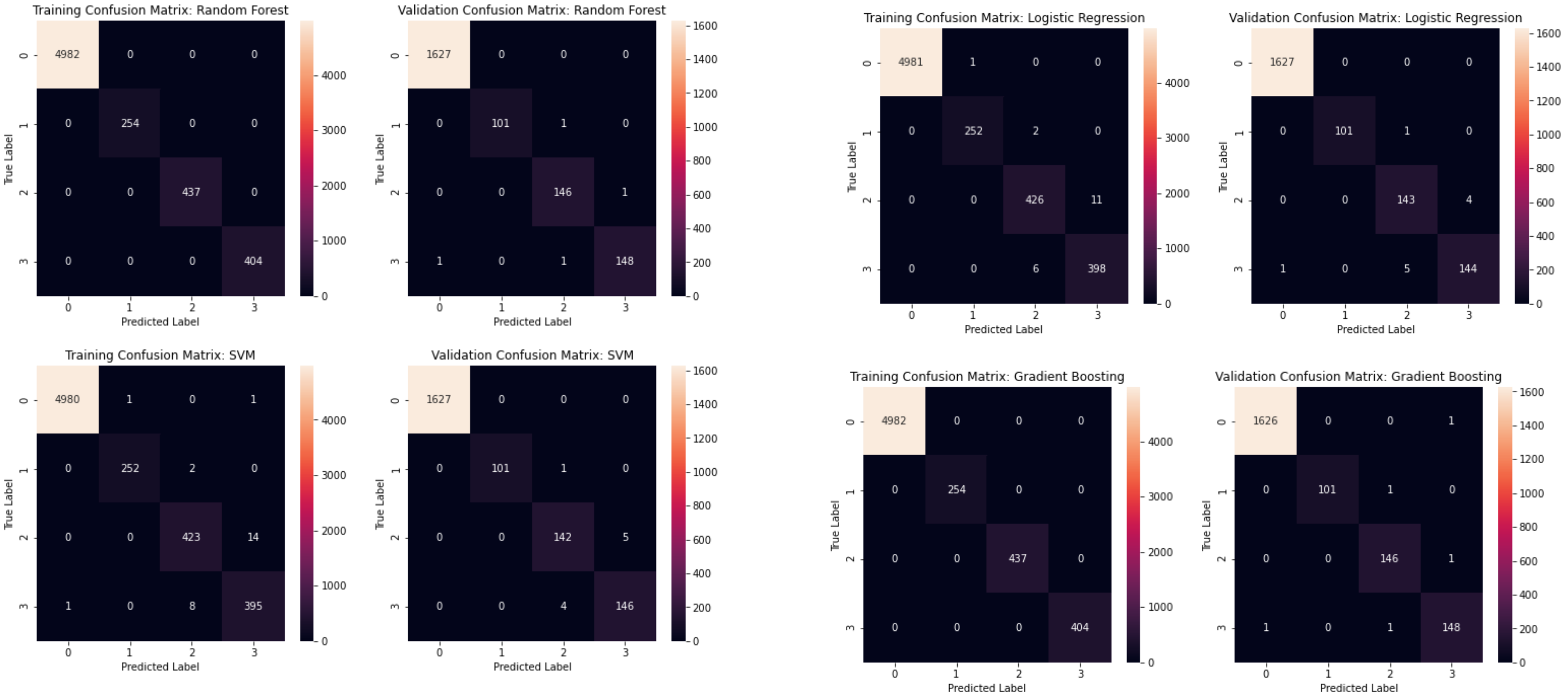
Workers: 8, Problem Size: 8000x8000, Time: 2.5435 seconds

Joblib & Multiprocessing

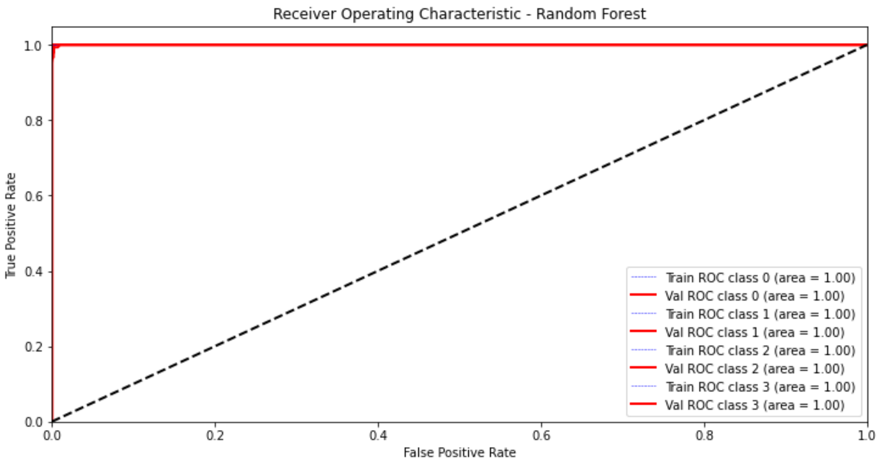
Hyper parameter optimization was taking too long , so I used joblib library is utilized to parallelize the process of training, tuning, and evaluating all the classifiers.

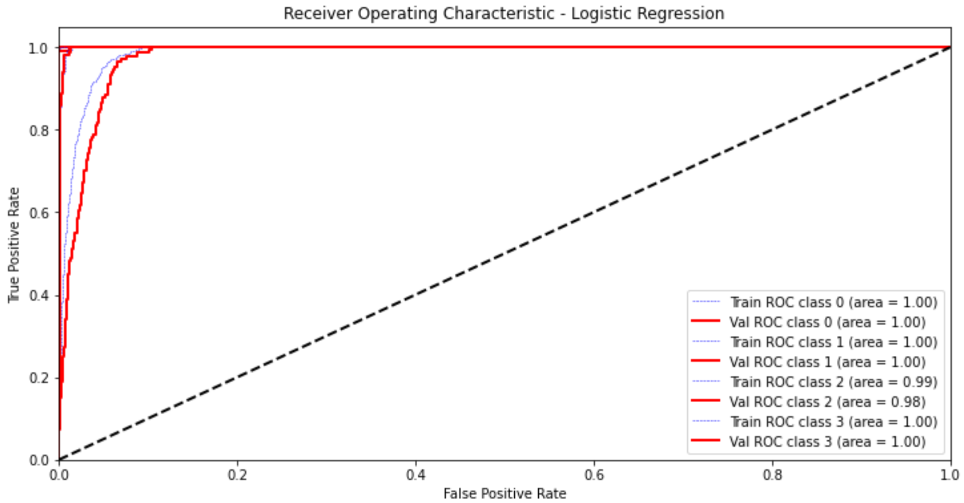
A process pool is created with the multiprocessing library to use all available CPU cores for parallel execution. Each classifier is subjected to a hyperparameter grid search, efficiently used across multiple processes using the pool's map function. After the grid searches are complete, the multiprocessing pool is closed

I am Confusion???

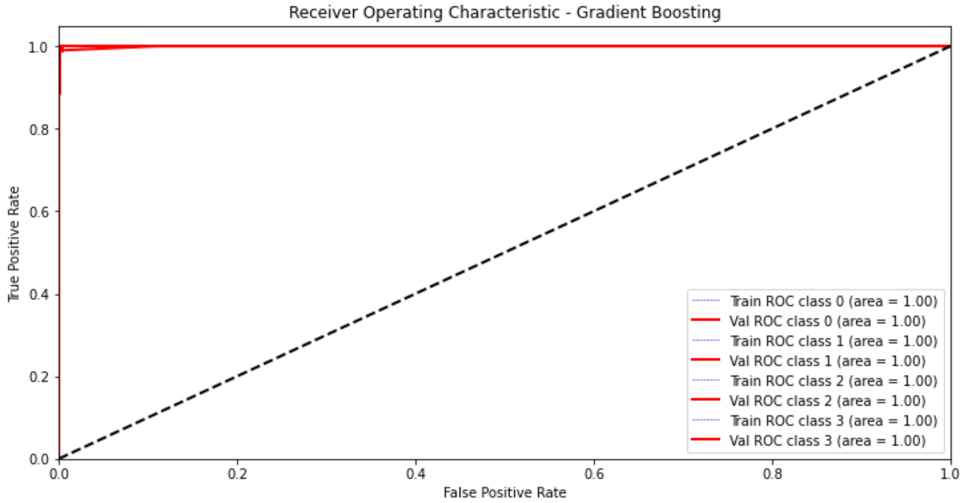


ROC Curve


Receiver Operating Characteristic - Random Forest
True Positive Rate
False Positive Rate
Train ROC class 0 (area = 1.00)
Val ROC class 0 (area = 1.00)
Train ROC class 1 (area = 1.00)
Val ROC class 1 (area = 1.00)
Train ROC class 2 (area = 1.00)
Val ROC class 2 (area = 1.00)
Train ROC class 3 (area = 1.00)
Val ROC class 3 (area = 1.00)


Receiver Operating Characteristic - Logistic Regression
True Positive Rate
False Positive Rate
Train ROC class 0 (area = 1.00)
Val ROC class 0 (area = 1.00)
Train ROC class 1 (area = 1.00)
Val ROC class 1 (area = 1.00)
Train ROC class 2 (area = 0.99)
Val ROC class 2 (area = 0.98)
Train ROC class 3 (area = 1.00)
Val ROC class 3 (area = 1.00)


Receiver Operating Characteristic - SVM
True Positive Rate
False Positive Rate
Train ROC class 0 (area = 1.00)
Val ROC class 0 (area = 1.00)
Train ROC class 1 (area = 1.00)
Val ROC class 1 (area = 1.00)
Train ROC class 2 (area = 0.99)
Val ROC class 2 (area = 0.98)
Train ROC class 3 (area = 1.00)
Val ROC class 3 (area = 1.00)


Receiver Operating Characteristic - Gradient Boosting
True Positive Rate
False Positive Rate
Train ROC class 0 (area = 1.00)
Val ROC class 0 (area = 1.00)
Train ROC class 1 (area = 1.00)
Val ROC class 1 (area = 1.00)
Train ROC class 2 (area = 1.00)
Val ROC class 2 (area = 1.00)
Train ROC class 3 (area = 1.00)
Val ROC class 3 (area = 1.00)

Results

Classifier	AUC
Gradient Boosting	1
Random Forest	1
SVM	.99
Logistic Regression	.99

Parallelize	Time
Dask	3 minutes and 48 seconds
Joblib	5 Minutes
Non-Parallelization	14 minutes and 14 seconds
Multiprocessing	5 hours

Credit

Singh,Adarsh Pal and Chaudhari,Sachin. (2023). Room Occupancy Estimation. UCI Machine Learning Repository. https://doi.org/10.24432/C5P605