POWER SAVING TASK.

Final Report, Section (3)

JANUARY 18, 2024

OSAMA ZUMARI
Fall Semester

Introduction:

Describe the problem you are addressing and why is it important.

There is a room that contains 4 office desks and a big window in the wall. Also, there are sensors placed in different parts in the room, each desk has a sensor on it, and a sensor in the middle of the room, and a sensor above the door and another one towards it directly. The task is to detect how many people are in the room to enhance power saving by using the data that sensors collected. The task is important to minimize the high cost of electricity and be more environmental-friendly. In addition to extend the lifetime cycle for machines that use electricity by turning them off when no one use them. we will use Artificial Intelligence techniques to detect the occupancy of the room with the help of multiple sensors that are mentioned that produced helpful data to perform the task.

About Data:

Describe the dataset's source, collection method, attributes, size, and domain.

Data was published publicly to anyone who wants to use it and test it out, the main website of it is: Room Occupancy Estimation - UCI Machine Learning Repository. Collection method was sensors to record the stats of the room, it was placed in different areas where the four sensors that records temperature, light and sound were placed in each desk office in the room, and the CO2 sensor was placed in the middle of the room and two PIR sensors one of them was above the entrance door of the room and the other one was towards it in the other side.. Each sensor is considered as a feature. The features of the data are:

1. Date: The date of the recorded instance.
2. Time: The time of the recorded instance.
3. S1_Temp, S1_Light, S1_Sound: A sensor that records temperature, light, and sound of the room. The values of these three features are numerical (float or integer).
4. S2_Temp, S2_Light, S2_Sound: A sensor that records temperature, light, and sound of the room. The values of these three features are numerical (float or integer).
5. S3_Temp, S3_Light, S3_Sound: A sensor that records temperature, light, and sound of the room. The values of these three features are numerical (float or integer).
6. S4_Temp, S4_Light, S4_Sound: A sensor that records temperature, light, and sound of the room. The values of these three features are numerical (float or integer).

7. S5_CO2: A sensor that records CO2 of the room. The values of this feature are numerical (integer).
8. S5_CO2_Slope: it is the slope of the recorded CO2 values. It was calculated using linear regression in a window that contained 25 points at each instance and in addition to calculating the slope of the line. The values of this feature are numerical (float).
9. S6_PIR: A sensor that records any motion that happens in the room. The values of this feature are binary (True or False).
10. S7_PIR: A sensor that records any motion that happens in the room. The values of this feature are binary (True or False).
11. Room_Occupancy_Count: feature that has the number of occupancies (persons) in the room, it is classified to four classes; 0 class which indicates that there are no one in the room, 1 class, 2 class and 3 class which indicates the number of people inside the room.

In a nutshell, the data has 10129 records and 18 features and the target feature is Room_Occupancy_Count that will be determined using the other features. The domain of this field is power saving to reduce costs and electricity usage which lead to save environment[1] and extend the lifetime of machines such as air conditioning and more.

Learning Problem:

Describe the learning problem you are trying to solve.

To conclude, the learning problem that we will solve is a supervised, classification problem. We will make pre-processing techniques for the data and machine learning models to get the greatest performance we can get to detect the number of people in a room to help us minimize electricity cost and more.

Notes:

- Supervised learning means that the data is labelled where you give the model the results of instances to train on them and understand the patterns behind them (learning by examples).
- Classification means that the labeled data is classified into categories that have specific meaning.
- In our problem, we will be using supervised-classification learning where the class indicates to the number of people in the room.

Data Exploring, Preparing and Preprocessing Phase:

How did you prepare training and test data before implementing machine learning models.

First step was to have a good look at the data to understand what we are dealing with. We looked at the correlation between features and realized that almost all features have linear correlations between all other features. Then, we checked the missing values and fortunately there were no missing values in the data. Then, we checked the datatypes of each feature and we concluded that all the features are numerical except for Date and Time features. After that, we checked if there were duplicated instances, fortunately there was no duplication in the data. Then, we printed the description of the data to look at some statistical information of each feature.

Second step was initial preparing, we divided the date and time features to extract useful information to use them in exploring the data deeper and may be useful in the modeling phase. In the Date feature, we concluded that the date of these instances was in winter semester and seven days partially separated, the days were from December/22 to December/26 and January/10 to January/11. So, we created a new feature that has the new representation for the date feature which is days counter from 1 to 365 (a year), but since we have only 7 days only. For this task only, we just renamed the features in that way. The new feature became as the following:

Date:	New Representation:
2017/December/22	Day 1
2017/December /23	Day 2
2017/December/24	Day 3
2017/December/25	Day 4
2017/December/26	Day 5
2018/January/10	Day 6
2018/January/11	Day 7

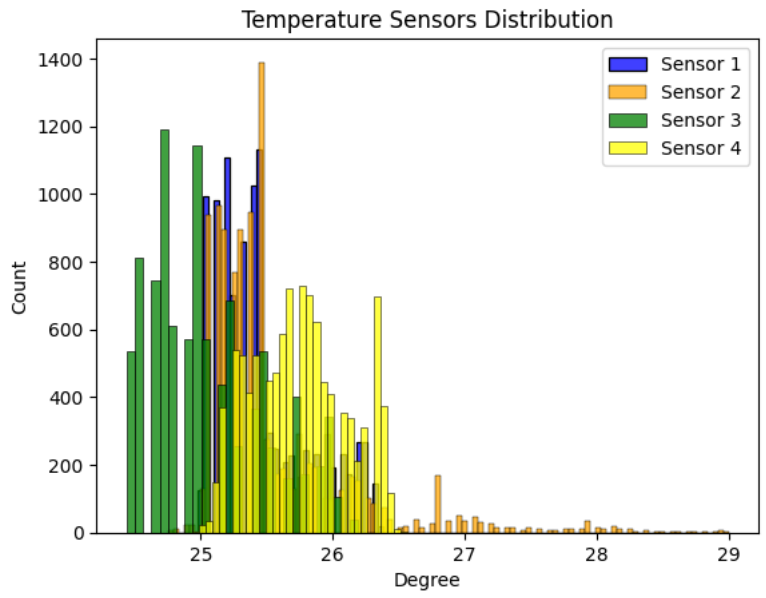
Which by then, we can remove date feature since we extracted the useful information of it. And for Time feature, we had created a new feature as well that has the hour only from the time feature since we personally saw that it is better to divide the time into hourly information so we extract the useful part of the feature. the new feature has values between 0 to 24 (24 hours based). In addition, to make it clearer, we had created a new feature that has values of days intervals which is as the following:

Hour Interval:	Period Name:
0am – 5am	Night
6am – 11am	Morning
12pm – 17pm	Afternoon
18pm – 23pm	Evening

In that case, we have created two more useful features, one has the hour of the record, and the other has the interval in which that hour is represented on. And due to that, we removed the time feature that contained the hour, minute and the second of that record. In this way, our model will be useful in hourly predictions which is reasonable to avoid changes on technology machines such as turning the Air Conditioner on and off frequently.

Exploratory Data Analysis:

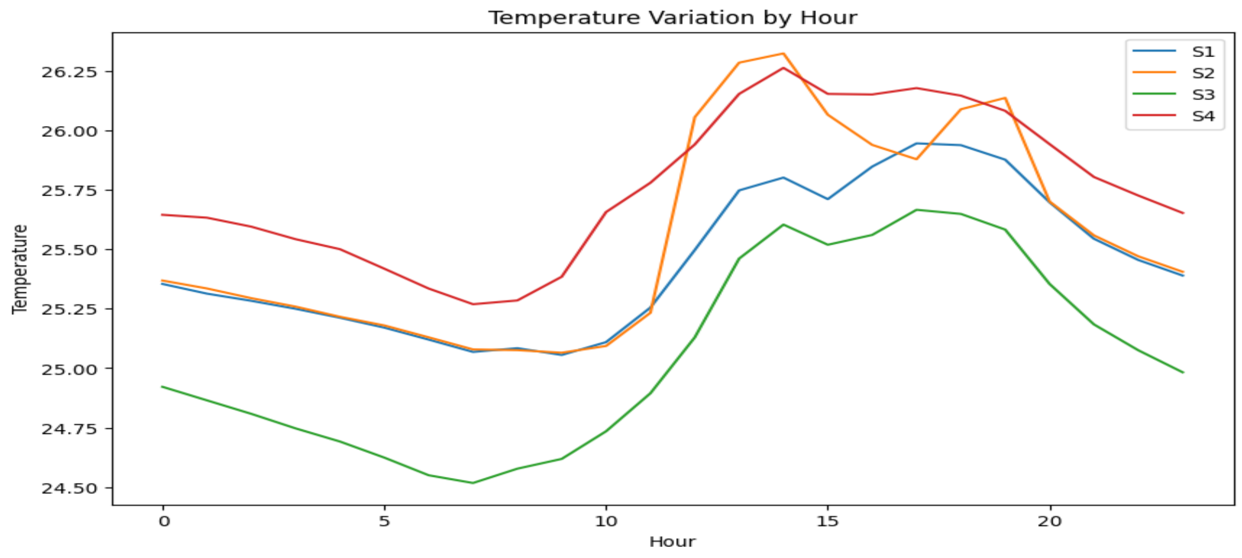
In this step, we used seaborn library to start visualizing the features and have a deeper understanding of them. The following graphs provided us with many useful information that helped us to create a better models:



In this graph, we visualized the four sensors of temperature reads, and with the help of statistical operations such as minimum and maximum operation, we concluded as the following:

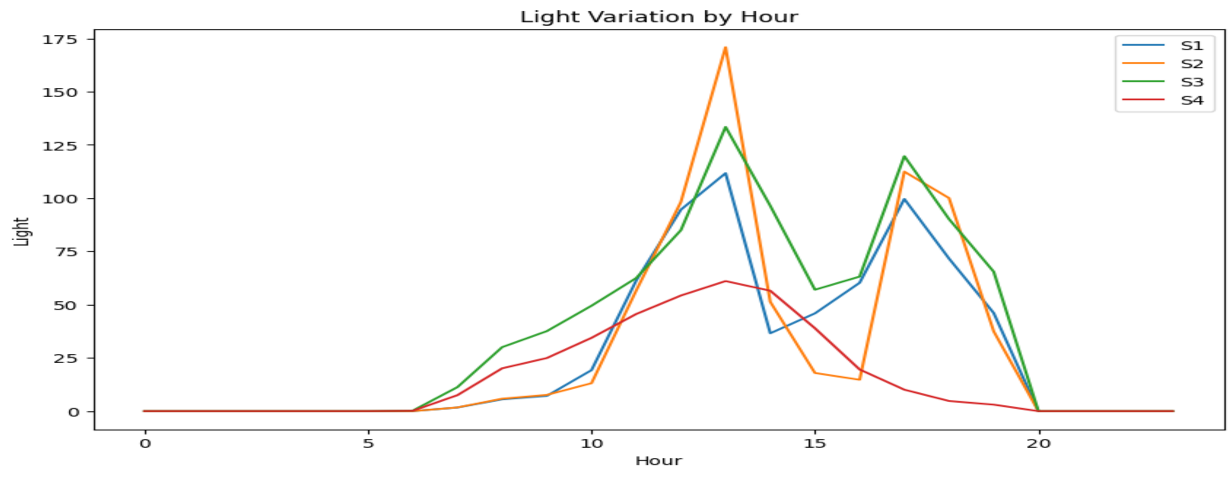
Sensor:	Minimum Value	Maximum Value	Mean Value	Max-Min Diff
S1 Sensor	24.94	26.38	25.45	1.44
S2 Sensor	24.75	29	25.54	4.25
S3 Sensor	24.44	26.19	25.05	1.75
S4 Sensor	24.94	26.56	25.75	1.62

Now, to enhance our decision, we made another graph that shows the mean of temperature of each hour through a day for each sensor:



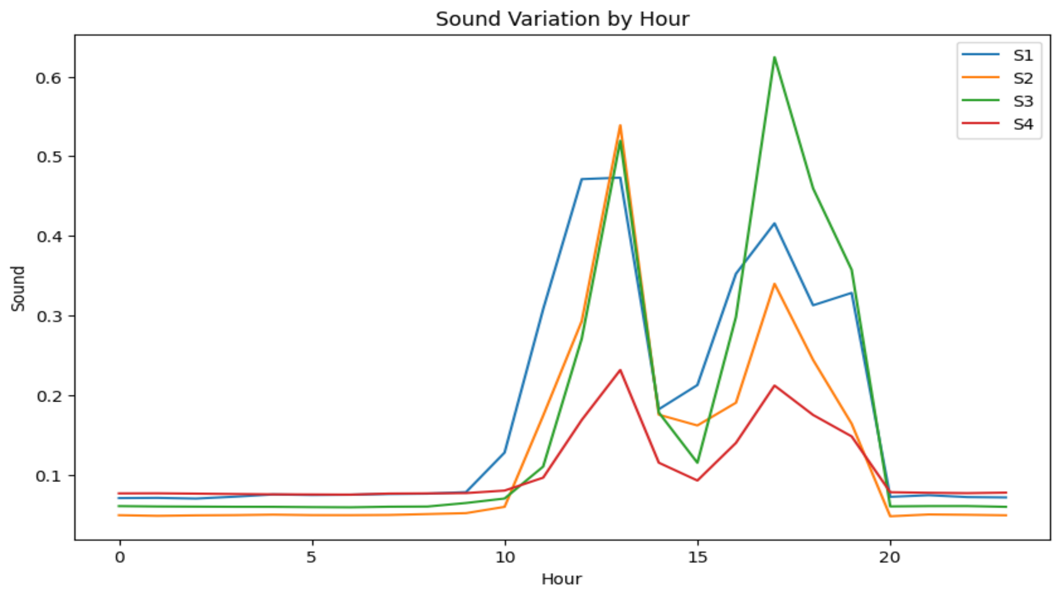
From these two graphs, we concluded that Sensor 1 and Sensor 2 are the same, but after afternoon period (after 12 o'clock) and before night period (specifically before 20 o'clock), Sensor 2 has outliers. In these hours, Sensor 2 read temperature more than all the others, and the increasing point of all sensors expect the second are reasonable and lookalike. But in Sensor 2, the rising point is clearly higher and unexpected especially that it should be as same as sensor 1 due to the similarity in the other hours. In addition, from the statistical table, we ensured that Sensor 2 also has high differentiation between the minimum value and the maximum value. Due to these results, we agreed that we should remove Sensor 2 that reads the temperature since it contains outliers. In addition to the outliers, Sensor 1 is enough to represent Sensor 2 normal reads. In another meaning, we can release Sensor 2 reads since we have Sensor 1 that almost the same reads as Sensor 2.

Next, we graphed light reads of sensors such as we did in the previous graph:



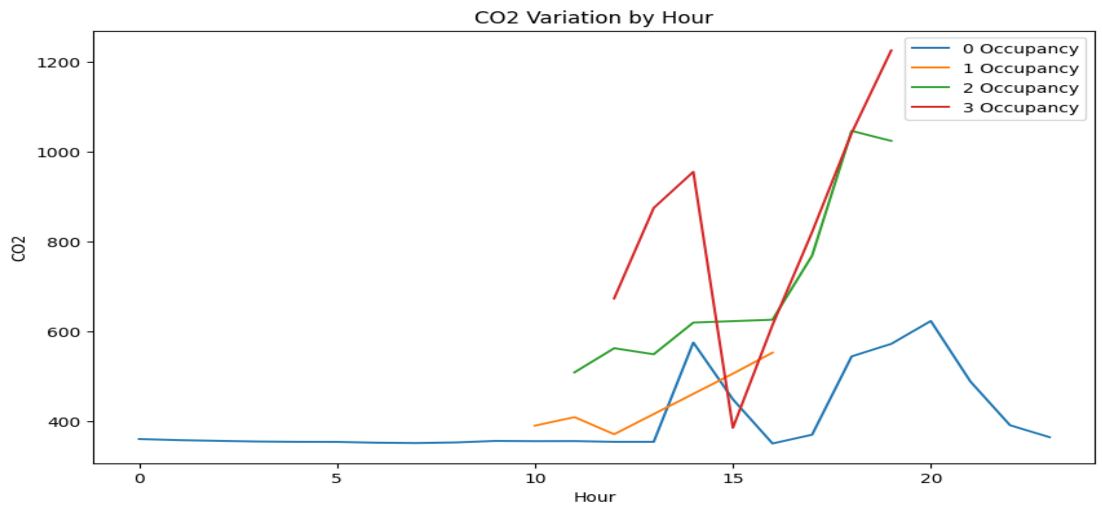
From the graph, we could enhance our theory that Sensor 2 has outliers in afternoon period and that Sensor 1 is similar to Sensor 2. As science says, when light is shed on an object, the object gets warmer[2]. So, we conducted a theory that Sensor 2 may be affected by the sun or something that produces light, and that light directly sheds on the sensor which raises the temperature and light as well. Therefore, we removed Sensor 2 light reads.

Next, we graphed the sound variation such as we did on the previous graphs:



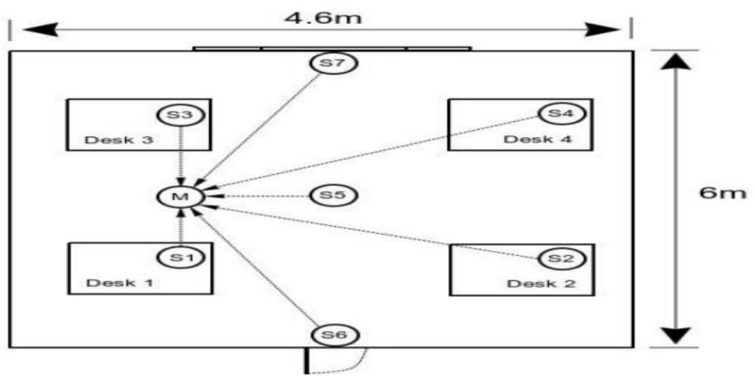
The sound is a representation of how much the volume was hearable (high) in the time, we can see that in late evening and night periods, there are almost no sound in the room which indicates that there are no one in the room in that time which is reasonable. Also, it looks like the work time in the room starts in 9 o'clock in the morning since it is the first point where it records a higher sound. We agreed that we will keep all the four sensors of sounds readings since it is useful to detect how many people are in a room and the sound from a place to place differs so we cannot judge.

Next, we graphed the CO2 variation in each hour for each number of people in the room as the following:



From the graph, we can improve our theory that the work time starts at 9 or 10 o'clock since CO2 was constant all the night period. Also, it is known that CO2 is a great indication for people count, studies show that CO2 sensors can detect the number of people in a room[3]. Therefore, we agreed that we will keep the feature for the modelling phase.

Next, we used the picture provided for the room where the experiment was placed in:

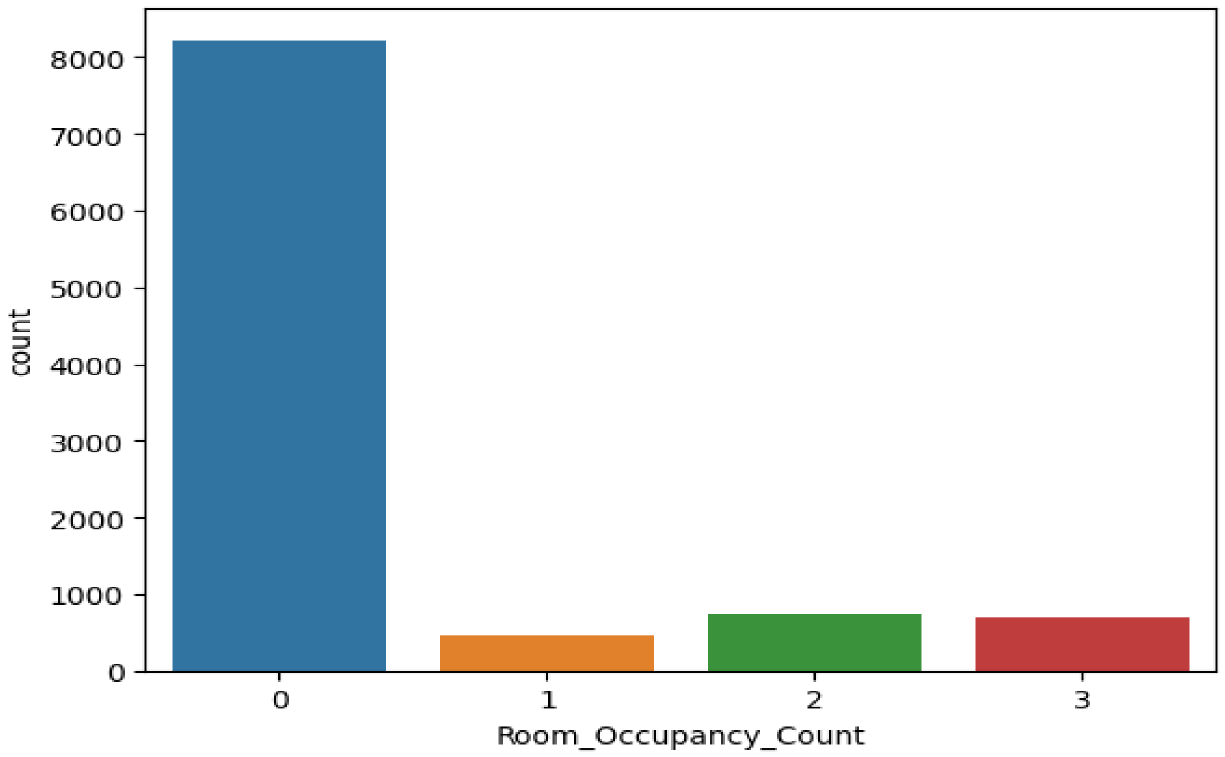


We had a closer look at the PIR sensors (Sensor number 6 and 7), and we decided that we will remove sensor 7 for multiple reasons, and this table will explain why we removed it:

Possible Scenario	Sensor 6 Read	Sensor 7 Read	Conclusion
If a person moved from desk (4 or 3) to desk (3 or 4):	False	True	The person is still in the room.
If a person moved from desk (3 or 4) to desk (2 or 1):	True	True	The person is still in the room
If a person moved from desk (3 or 4) to exit from the room:	True	True	The person is out from the room.
If a person moved from desk (1 or 2) to desk (3 or 4)	True	True	The person is still in the room.
If a person moved from desk (2 or 1) to desk (1 or 2):	True	False	The person is still in the room.
If a person moved from desk 1 or 2 to exit from the room	True	False	The person is out from the room.

This table contains all the possibility of the motion detection, since we don't have an indication where a person has moved, we considered that sensor 7 would be not useful since it is far away from the door, and each trues will indicate that a person moved in that area. However, sensor 6 has a possibility that a motion is detected because someone is leaving or entering the room, unlike sensor 7 that does not indicate that. Therefore, we guarantee that sensor 7 has no indication for someone leaving or entering the room, and sensor 6 has a better indication for that because it would read True. In a nutshell, Sensor 6 detections have a possibility of someone leaving or entering the room, but Sensor 7 has no indication of someone leaving or entering the room.

Next, we graphed the target feature:



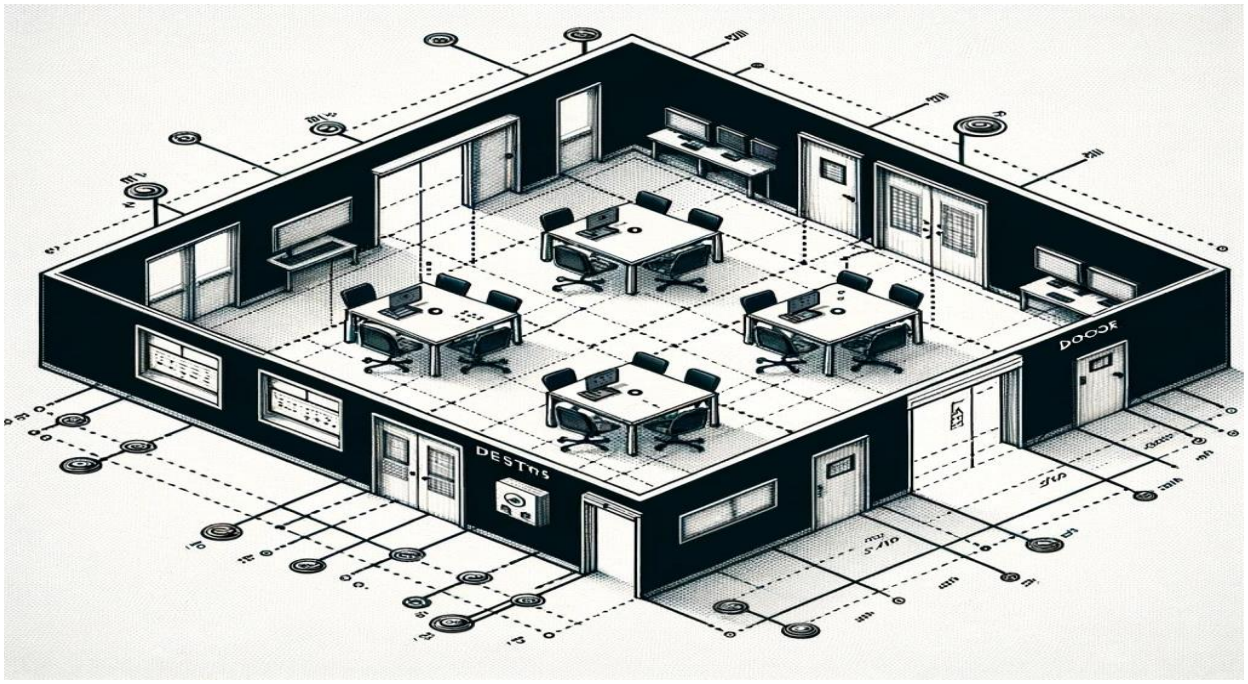
In this graph, we saw that the data is imbalanced where class 0 is the majority and has more records which affects machine learning models since that could cause biased predictions. However, with all of these exploration and digging deep to understand the data, we might solve it in a good way.

With all of information obtained and analyzing of the data, we agreed on taking the mean of the temperature sensors reads for each instance for the following reasons:

1. Values are near each other which means there is no huge variation in temperatures degrees expect sensor 2 which we removed it previously:

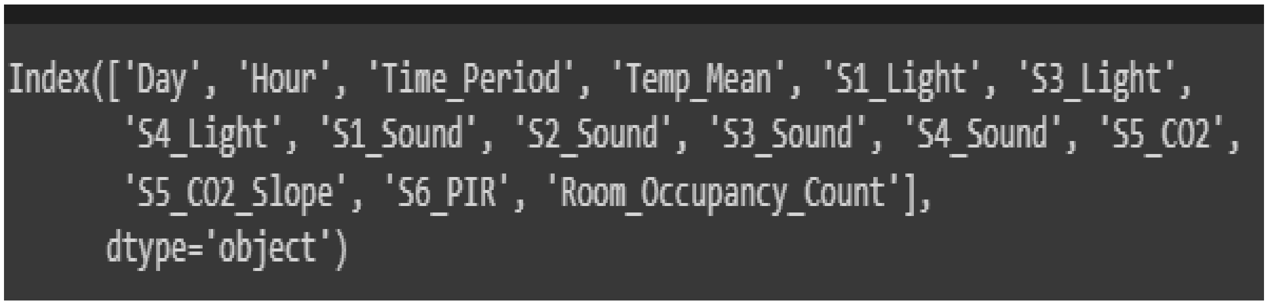
Sensor:	Minimum Value	Maximum Value	Max-Min Diff
S1 Sensor	24.94	26.38	1.44
S2 Sensor (Deleted)	24.75	29	4.25
S3 Sensor	24.44	26.19	1.75
S4 Sensor	24.94	26.56	1.62

2. Sensors are reading the temperature of the same room which means all sensors should give approximately the same degree especially if the room is small.



(ChatGPT-4 Generation for a 4.6m x 6m room)

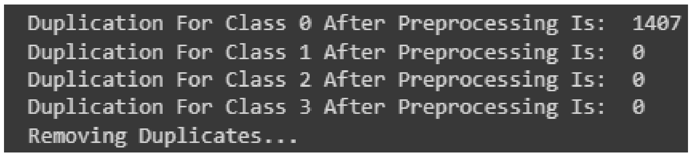
3. There are many factors that may affect one sensor only such as a person smoking in his desk which slightly raises temperature sensor reads[10][11]. Therefore, it indicates the necessity of taking an average value of temperature in the room. Keep in mind that the mean value will give the most accurate temperature since sensors are recording the same room temperature. For example, if two sensors recorded a degree of 25c, and another sensor 27c, we can consider that majority vote is the best case which is 25c, but we would take into consideration the other sensor as well, so the value would be 25.6c, which is close to the majority vote value but with the respect of the remaining sensor as well.



```
Index(['Day', 'Hour', 'Time_Period', 'Temp_Mean', 'S1_Light', 'S3_Light',
       'S4_Light', 'S1_Sound', 'S2_Sound', 'S3_Sound', 'S4_Sound', 'S5_CO2',
       'S5_CO2_Slope', 'S6_PIR', 'Room_Occupancy_Count'],
      dtype='object')
```

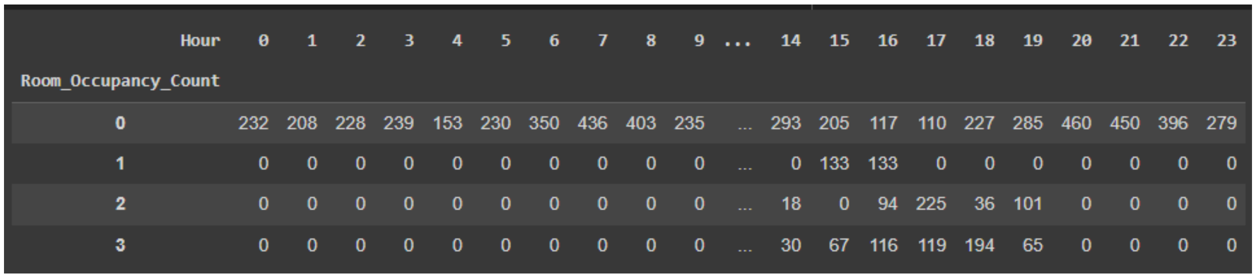
Note: We did not take the mean for light and sound sensors readings because they differ from each other.

After these preprocessing steps, we found out that there are new duplicated data, which is reasonable after specifying the day and time, also after deleting unwanted features. The main reason why it did show is because we made the time feature as a period of hourly, which means from 10:00 to 10:59 is considered as 10. Thus, our model will be predicting hourly which is reasonable for power saving enhancements because it will save the machines from damaging and prevent costing more power to be turned on again. In another meaning, if the model predicted results shorter than an hour, it may cause the AC condition for example to be turned on and off multiple times in short period of time, which may damage to the AC condition[12][13]. In addition to raising the cost of reworking again[14]. Also, it would be more professional to change the statues of technology machines such as AC each hour according to data.



```
Duplication For Class 0 After Preprocessing Is:  1407
Duplication For Class 1 After Preprocessing Is:  0
Duplication For Class 2 After Preprocessing Is:  0
Duplication For Class 3 After Preprocessing Is:  0
Removing Duplicates...
```

We found that all duplications are within class 0 (the majority class), which made us think that there must be more hidden duplications, for example, we know that in night period (0 to 5 o'clock), there is no work. Thus, there is no one in the room. Thus, to proof our theory, we created a contingency table that counts the frequency of two categories shows together as shown in the following graph:

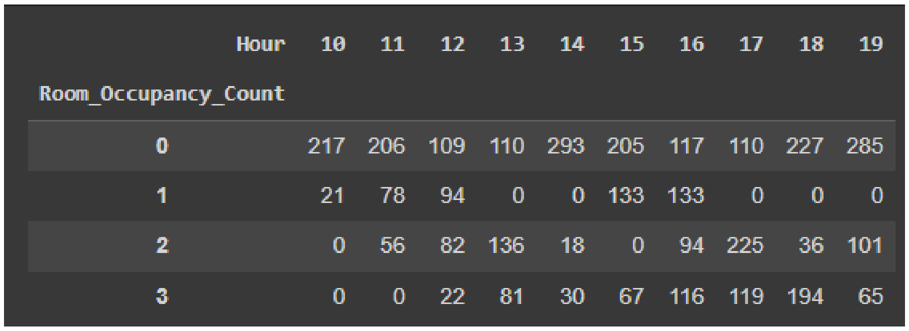


Hour	0	1	2	3	4	5	6	7	8	9	...	14	15	16	17	18	19	20	21	22	23
Room_Occupancy_Count																					
0	232	208	228	239	153	230	350	436	403	235	...	293	205	117	110	227	285	460	450	396	279
1	0	0	0	0	0	0	0	0	0	0	...	0	133	133	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	...	18	0	94	225	36	101	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	...	30	67	116	119	194	65	0	0	0	0

This contingency table proofs our theory that there is no one in the room from 0 to 9 o'clock. Therefore, we decided that we will do manual data balancing instead of relying on models to do so which might be not accurate because models will make over sampling by making the minority near to the majority by duplicating the instances of the minority classes, or under sampling which makes the majority class near to the minority class by removing some instances. However, we found a logical way to make the data partially balanced without relying on models.

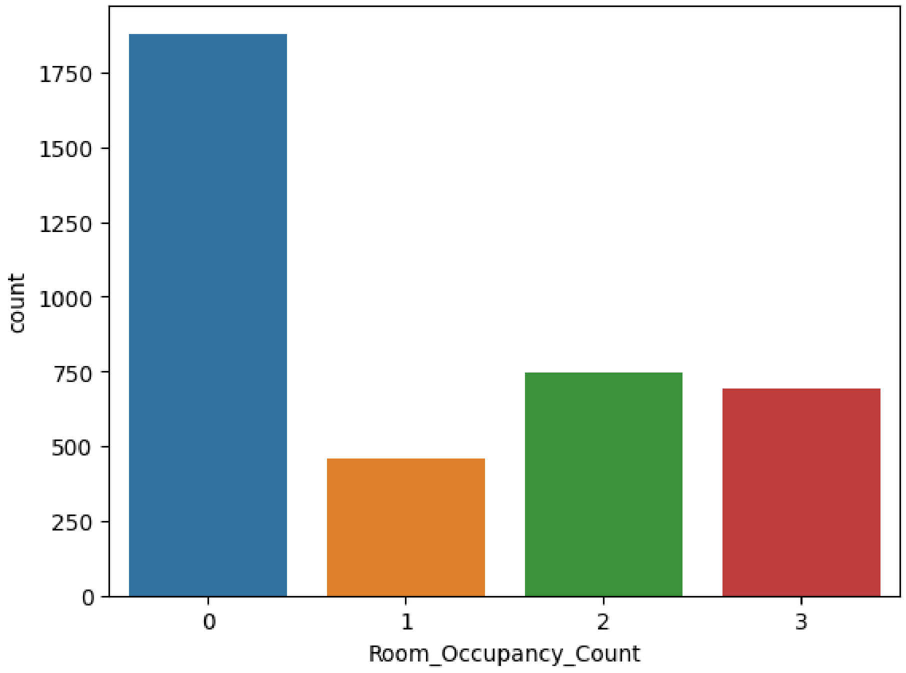
In addition to a very important note, we are predicting the status of a room within hours work to optimize power savings through optimizing the performance of technology machines such as AC and ventilation. Furthermore, when everyone is leaving the room, the machines will be turned off completely. ("No one is leaving a house with lights turned on"). Therefore, night, and late evening data is removed since our focuses are on in work hours.

The contingency table after deletion:



Hour	10	11	12	13	14	15	16	17	18	19
Room_Occupancy_Count										
0	217	206	109	110	293	205	117	110	227	285
1	21	78	94	0	0	133	133	0	0	0
2	0	56	82	136	18	0	94	225	36	101
3	0	0	22	81	30	67	116	119	194	65

And the classes became as the following:



Data size dropped from 10129 to 3780!. However, worry not, all data that is removed is not important for our task. However, data is still imbalanced even after all what we have done. Therefore, we will apply oversampling technique, which is SMOT, which means Synthetic Minority Oversampling Technique which creates similar data points for the minority class to increase its size and make it equal to the highest class[15]. We will be using SMOT after we split the data into training and testing that we will mention later.

Machine Learning Algorithms:

Explain why the provided models are appropriate to solve this problem.

Explain in detail the machine learning algorithms you are using to address this problem.

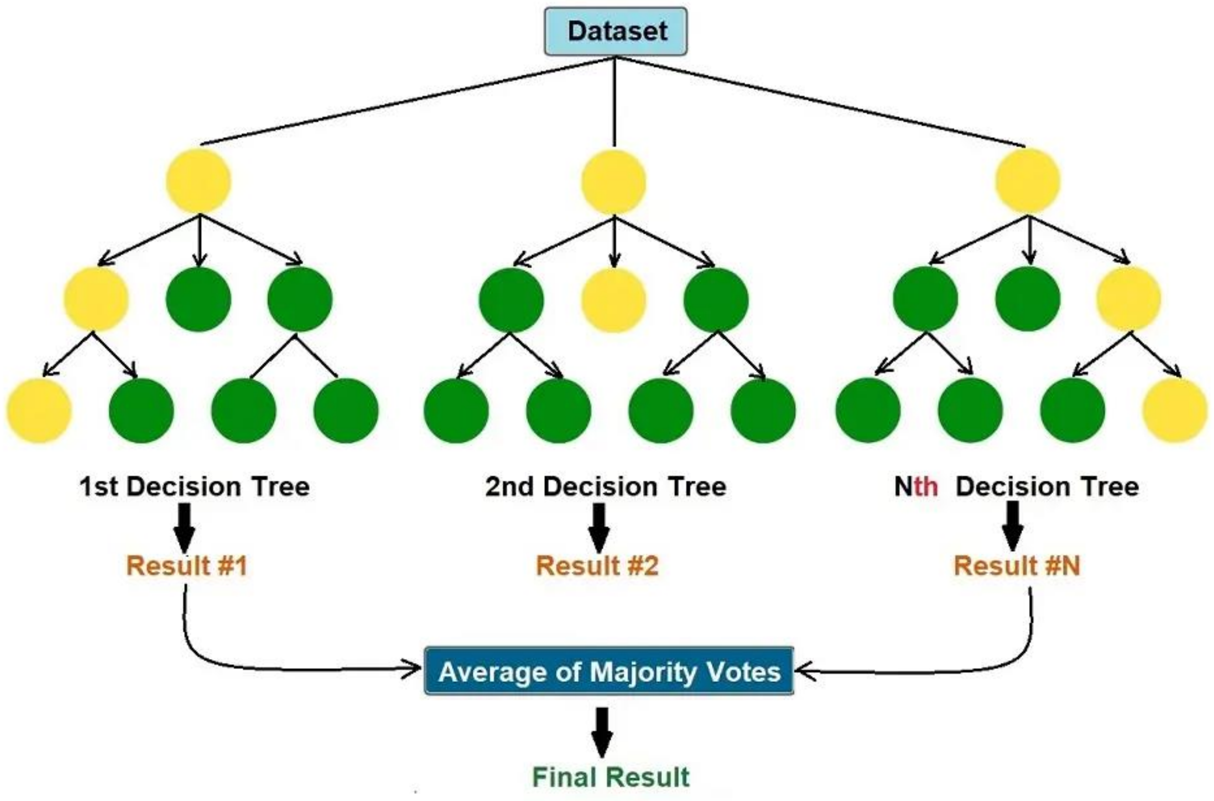
Machine learning algorithms are made for predictions tasks, it is useful to understand and predict what is happening in the future to prevent risks or losses and gain benefits. One of the best benefits we see is enhancing decision-making, by providing the necessary information, experts responsibilities is to take the best decisions from the results that machine learning algorithms provide. For our task, we will be using four models which are Random Forest, Support Vector Machine, Gradient Boost and XGBoost. In the following paragraphs, it will discuss about each one of them in details and why they are appropriate for this specific task.

Random Forest:

Definition: An ensemble tree based algorithm, it can be considered as the enhanced version of decision tree algorithm that works by conditions to give predictions.

Approach: It creates subsets of the data, then it creates boundaries according to the subset it is working on by creating a tree based that has conditions to determine what is the final result, and the final result will be according to the majority vote of all decision trees have been decided.

Advantages: It is proper for our task because random forest is almost immune against noise data. In addition to its strength against complexity which means that no matter how many trees were added to the random forest model, it will be almost impossible to get affected by overfitting that happens because the model is too complex[3]. In addition, Random Forest algorithm does not get affected by multicollinearity features[4], and almost all of our features have high correlations with other independent features.



As we can see from this graph, random forest is a multiple decision trees that takes subsets from the dataset. Each decision tree takes features and data points and gives a vote; this vote is the result. Lastly, all decision trees will give a vote, and the majority vote will be the result.

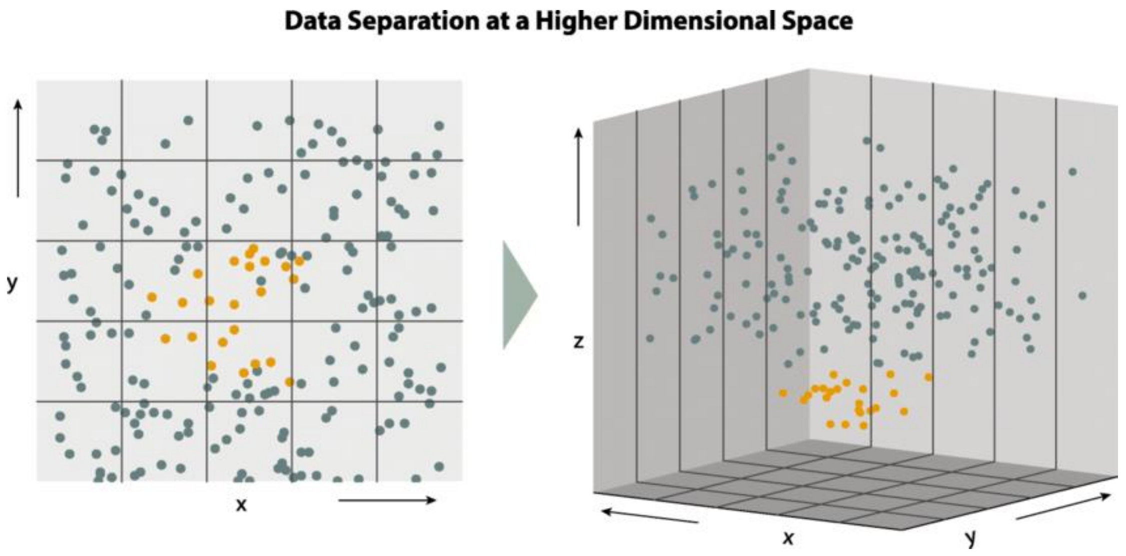
Applications: Banking: fraud detection, and in healthcare for diabetes predictions.[20]

Support Vector Machine (SVM):

Definition: A model designed to find the best separable line (boundary) which is called a hyperplane which classifies data points in a proper way. The hyperplane is considered as a decision boundary in the high-dimensional space of features which assigns each point to its class based on its position relative to the boundary.

Approach: The model uses mathematical transformations to present the data into a higher-dimensional space which allows for the creation of a more complex boundary that is not limited to a simple straight line[5]. Which by that helps us to classify all points to its classes.

Advantages: The model is known for its capability to handle low amounts of data[6], which is important for our task since we have a small dataset with only 3780 instances after the preprocessing was made. In addition to its capabilities on handling high dimensional datasets.



As we can see from the graph, it puts the data points in a proper way to perform the hyperplane and decide the boundaries.

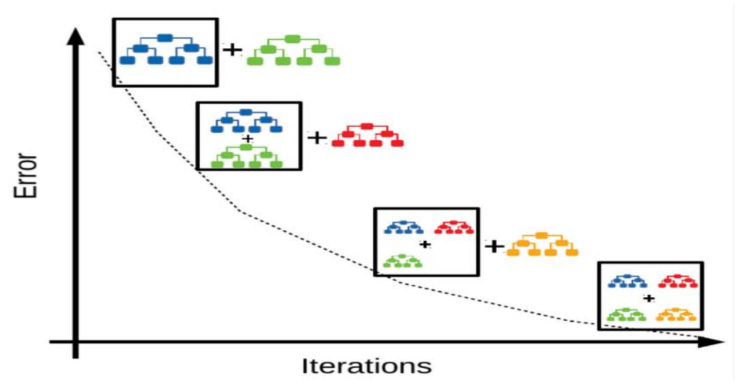
Applications: Image and text classification. [22]

Gradient Boost:

Definition: A tree based model that combines weak learners of decision trees, it performs and gives better results because it adjusts weights and learns from its errors from the previous iteration which make it adaptable to learn complex patterns in the data[7].

Approach: It is a sequential model which means that the previous training sessions are being adapted in the next iteration by learning from the errors that trees have made.

Advantages: The model is fast learner which is effective for our situation since we need the model to train on small amount of data. Furthermore, this model is effective for overfitting and especially underfitting problems since our data is small and it may lead to underfitting[8].



As we can see in the graph, each time, one decision tree is performed, and each iteration learns from the previous errors to give better and accurate decisions.

Applications: Recommendation systems in entertainment and e-commerce. Additionally, in Kaggle competitions, gradient boosting is great for customer churn and asteroids.[22]

XGBoost:

Definition: A model that is used for data that is structured (tabular data) that is well organized such as our data after preprocessing. It is an enhanced version of Gradient Boost which is the reason why it's called extreme gradient boost. This model is specialized on complicated datasets that have complex patterns.

Approach: Same as gradient decent in the approach.

Advantages: the model has regularizations techniques which means putting penalties on features to prevent overfitting from happening and thus give better results in training and testing phases. In addition, it is a good model when the features are numerical which fits our data[9].

Uniqueness: Tree Pruning and Parallelization are characteristics of XGBoost, tree pruning means that the model is removing trees that are not helpful for the model. As a result of that, the model becomes less complex and has no negative impact trees. Parallelization means the model can do multiple tasks at a time, which means multiple trees can perform at the same time unlike gradient boosting which requires one at a time. And due to that, this model fits larger problems and datasets because of the speed.

Applications: Used in finance for fraud detections and healthcare. [11]

Modelling Phase:

Demonstrate how you will test the machine learning application using a range of test data and explain each stage of this activity (Apply k-fold cross-validation).

First step is to prepare the data to fit it inside the models, since the data is numerical and we already did the preprocessing, we had two steps left which are label encoding which transform categories into numerical data for the machine to understand it, and standard scaler to put the data under specific range since there are variation between numbers in the data. For label encoding, we used Label Encoder from sklearn library, and apply that into Time_Period feature that has these values: ("Morning, "Afternoon", "Evening", "Night"). Then, we initialized (x) which contains the data of independent features and gave it every feature expect the target which is Room_Occupancy_Count. And (y) is initialized with the target feature data. Lastly, we used Standard Scaler to put the data within the same range which by that it has mean of 0 and standard deviation of 1. Now, we split the data into training and testing, the testing size is 20%% of the total data. Now, we will apply SMOT into the training set only to increase the training size and balance all classes for the model to train on them. Now, we have balanced classes training set, so we are ready to fit the models, but since we have many hyperparameters for each model, we decided that we will use Grid Search Cross Validation technique. First, cross validation is an evaluation measure that is applied on the training set, it's task is to split the data into subsets, and each subset considered as a fold. Then, all folds will be trained on by the model expect one fold that will be considered as a testing set and redo this step to every folds. In another meaning, all folds are training and testing sets. Cross validation helps to detect whether the model is facing overfitting by showing whether the model is well trained but fails to discover new patterns and fails in predictions or not.

Notes:

Overfitting: A model is trained on the data well and is performing well in training but fails in predictions and capturing patterns. The causes are the dataset is small and the model is very complex.

Underfitting: A model is not trained well on the data and is performing poorly in training and fails in predictions and capturing patterns. The causes are the dataset is small and the model is very simple.

Now, back to Grid Search Cross validation, it takes the model estimator and the pre-defined hyperparameters and the number of folds for cross validation. In this way, the model will be trained on each element of hyperparameter with other hyperparameters to determine what the best hyperparameters are to use them for the model in this specific task. Then, we will pass it the training set by using fit function that means giving the model the dataset to train and learn from it. Now, after training session, we will use predict function that takes the x testing set without the target data, this function is used for predictions.

Evaluation:

What performance measures did you use to evaluate the effectiveness of your models.

Why did you use these metrics?

To decide whether a model is great and reliable or not, we need to have evaluation measures to measure how well our models are. For this task, we used multiple classification evaluation measures which are:

F1-score: It is a measure that computes the harmonic mean of precision and recall, it is better than accuracy since it's give overall performance on predictions for all classes unlike the accuracy score that only focuses on the correct predictions without caring about the classes. So, it is proper for multi-class cases such as our case by giving accurate measures of the models performance.

Precision: It is a measure of the true positive ratio of all true positives in the data, it computes the accuracy of positive predictions. In our multi-class problem case, the precision is calculated for each class by considering the class as Positive and the others as Negative. It shows the reliability of a model in predicting the positive cases which would be suitable for our case to give accurate number of occupancy in a room.

Recall: It is a measure that computes the ability of models to predict accurately for each class exists. Recall gives the percentage of correct positive predictions of all positive points. In another meaning, it cares about every class in a dataset, which is helpful for our situation because recall focuses on each class independently, and by that it would avoid unneeded power consumption.

Validation Score: It is a measure of cross validation techniques, where it calculates the accuracy of each fold, then takes the average of these folds, which gives an overall performance of the model. In addition, the measure would make predictions on unseen data by training and testing on each fold. Which by that give an accurate average of models performances.

Model Enhancements:

Evaluate how, based on the performance measures, you were able to enhance the model.

How did we enhance the models? Well, about 20 percent of performance is based on hyperparameters of the best model for a case, and 80 percent of performance is based on data engineering which is the preprocessing steps. We already enhanced our model by making the best preprocessing we could come with from data reduction, dimensionality reduction, standard scaler, encoding and applying SMOT and so on…, and it was clear that our preprocessing steps saved us much time and performed very well. And about the hyperparameters, since we used grid search, we already defined some important hyperparameters for each model and made the function to look up for the best hyperparameters:

For random forest, estimators: 100, 150, 200 and max depth 4, 6, 8. These two hyperparameters are widely used for random forest, estimators enhance the model performance and the depths show how well the model in dealing with high complexity trees.

For support vector machine, kernel: linear, poly, rbf, sigmoid. All of them are transformation functions which is used to find the best way to transform the data.

For gradient boosting, estimators: 100, 150, 200 and learning rate: 0.1, 0.01, 0.001, 0.0001. estimators are as random forest, which enhances the model performance. In addition, learning rate hyperparameter helps in preventing overfitting if the value was low, which helps the model to perform correctly unlike when the value is large because it would learn fast which by that it causes overfitting.

For xgboost, estimators: 100, 150, 200 and learning rate: 0.1, 0.01, 0.001, 0.0001 and lambda: 1.0. Same as gradient boosting, but additional to Lambda, which is regularization hyperparameter to put penalties into features.

And all of them were on 4 folds of cross validation.

Results And Discussion:

Discuss the reliability of your results and whether they are balanced, overfitting, or underfitting.

Analyze the result of the applications to determine the effectiveness of the algorithms.

Draw conclusions regarding the strengths and weaknesses of the different algorithms.

We will be comparing these models with our preprocessing techniques and without it to determine whether what we have done is great or not.

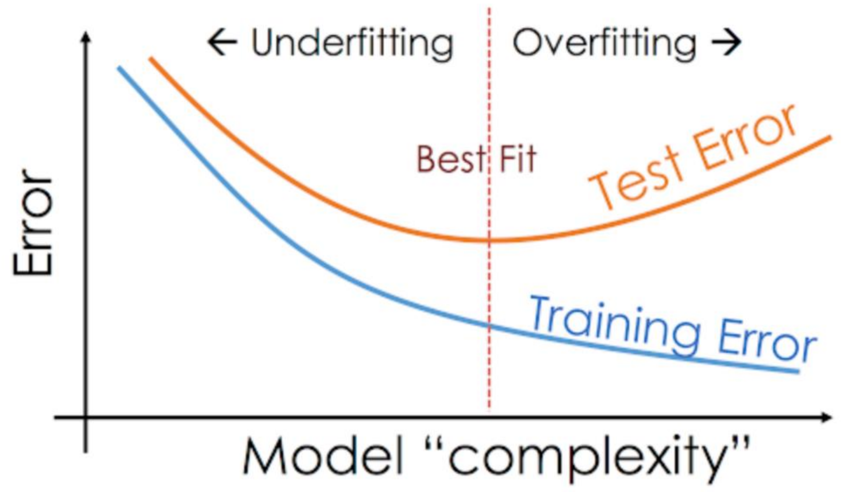
Random Forest	F1-Score	Precision	Recall	Validation Score	Time Complexity
With-our Preprocessing	0.99	0.99	0.99	0.994	34s
Without-our Preprocessing	0.98	0.98	0.98	0.997	2m.14s
Support Vector Machine	F1-Score	Precision	Recall	Validation Score	Time Complexity
With-our Preprocessing	0.99	0.99	0.99	0.986	6s
Without-our Preprocessing	0.98	0.98	0.98	0.996	1m.7s

Gradient Boost	F1-Score	Precision	Recall	Validation Score	Time Complexity
With-our Preprocessing	0.99	0.99	0.99	0.994	8m.34s
Without-our Preprocessing	0.99	0.99	0.99	0.998	39m.51s
XGBoost	F1-Score	Precision	Recall	Validation Score	Time Complexity
With-our Preprocessing	0.99	0.99	0.99	0.996	46s

Without-our Preprocessing	0.99	0.99	0.99	0.998	1m.52s

All models gave high results, with our preprocessing and without it. However, the trade between performance and time is crucial. In machine learning science, it is recommended to sacrifice some degrees of performance for saving time. The good part is that our preprocessing gave very similar results and performed better in some models, and a complete domination of saving time for all models. In real world, time is the most important thing to take into consideration, our preprocessing techniques made the four models perform fantastic, F1-score approximately 99% which means the models are performing well in general, and approximately 99% in recall and precision which indicates that the models are performing well for all classes in the data. Lastly, high validation score is approximately 99% which proofs that the models are balanced and not overfitted or underfitted.

As we mentioned before, overfitting is when a model performs well in training phase but poorly in testing phase. That happens when the data is small or the model is too complex. We can detect that by cross validation which we defined it previously. From our validation score, we can ensure that our models are not overfitted, which means they are not complex. In the other hand, underfitting is when a model performs poorly in training phase and poorly in testing phase. That happens when the data is small or the model is too simple. It will be obvious that the model is poor by the evaluation measures. From our results, we can ensure that our models are not underfitted, which means they are not very simple.



In a nutshell, our models are well balanced between complexity and simplicity, which by that, they could perform well in training and testing phase by giving high results.

As we said, all models performed well, however, time factor is the most important between all results; random forest took 34 seconds to accomplish which is recommended to use since it gave high results in short amount of time, support vector machine took 6 seconds and high results which would be the best recommendation to use, gradient boost took 8 minutes and 34 seconds to give high results, but not recommended since it took 8 and half minutes to finish and xgboost took 46 seconds and gave high results which is recommended as well. In conclusion, the best model between all these models is support vector machine in time matter, and all only 1 percent far from xgboost that, which is fair trade off where we trade one percent of performance exchange of saving more time.

Very important note: all models with our preprocessing techniques and without were working on the same parameters and hyperparameters and testing size. So, the competition was fair enough.

General Weaknesses In These Models:

For Random Forest: The way to enhance performance is by adding more trees which as a result for that, it will increase the time taken to perform decisions. [16]

For Support Vector Machine: The model is not suitable for large datasets. In addition to the weakness of the model at handling noise data and outliers which requires good preprocessing techniques to handle noise data. [17]

For Gradient Boosting: It is computationally expensive which means that the model takes much time to perform decisions. [18]

For XGBoost: The model performs poorly for unstructured datasets. Additionally, the model is sensitive to outliers since the model learns from its previous errors. [19]

Further Enhancements:

Identify further enhancements which can be done in the future? Discuss any limitations and future improvements of your project.

For future enhancements, we would like to ask for more data, since it is the most important thing to obtain, with data, anything could be possible using artificial intelligence. In addition, we would recommend putting the sensors far from human's domain, because we saw that sensor 2 is affected by something external. Additionally, we can also recommend using computer vision by placing a camera to detect how many objects are in the room and immediately determine how many people are in the room. We also would like to have more data in different seasons not only in winter, by that, the results may differ between each season. Lastly, we used only few hyperparameters for each model, which is fine since we got high results. However, we need to increase the number of hyperparameters that is fit the problem that we are trying to solve, specifically support vector machine model, since we only used one hyperparameter. However, we might need stronger computational devices to reduce the time taken for each training for the models. Thus, we would have given each model it's rights in the number of hyperparameters without taking much time if stronger devices were used.

References:

1. US EPA, O. (2015). *About the U.S. Electricity System and its Impact on the Environment*. [online] US EPA.
2. Student Materials. (2017). *Thermal Energy from Light*.
3. Chaudhary, A., Kolhe, S. and Kamal, R. (2016). An improved random forest classifier for multi-class classification. *Information Processing in Agriculture*, 3(4), pp.215–222.
4. https://homework.study.com/explanation/does-multicollinearity-matter-for-the-random-forest-model.html#:~:text=Multicollinearity%20does%20not%20affect%20the,different%20set%20of%20data%20points.
5. Tabsharani, F. (2023). *What is support vector machine (SVM)? - Definition from WhatIs.com*.
6. Saini, A. (2021). *Guide on Support Vector Machine (SVM) Algorithm*. [online] Analytics Vidhya.
7. Saini, A. (2021). *Gradient Boosting Algorithm: A Complete Guide for Beginners*. [online] Analytics Vidhya.
8. Analytics Vidhya (2019). *Guide to Hyperparameter Tuning in Gradient Boosting (GBM) in Python*.
9. Khandelwal, N. (2020). *A Brief Introduction to XGBoost*. [online] Medium.
10. Cokeley, M. (2008). *Does Smoking Contribute to Global Warming?* [online] Popular Science.
11. ChatGPT
12. Quora. (n.d.). *Does turning an AC on/off damage it?*
13. Quora. (n.d.). *Does turning an AC on/off damage it?*
14. Quora. (n.d.). *Does turning the AC on and off cost more?*

15. Brownlee, J. (2020). *SMOTE for Imbalanced Classification with Python*. [online] Machine Learning Mastery.
16. Donges, N. (2021). *A Complete Guide to the Random Forest Algorithm*. [online] Built in.
17. Dhiraj, K. (2020). *Top 4 advantages and disadvantages of Support Vector Machine or SVM*.
18. Wizards, D.S. (2023). *Understanding the Gradient Boosting Algorithm*. [online] Medium.
19. Hachcham, A. (2023). *XGBoost: Everything You Need to Know*.
20. Meena, M. (2020). *Applications of Random Forest*.
21. Articles, M. (2023). *SVMs in Practice: Applications and Use Cases for Machine Learning's Most Effective Model*.
22. https://www.datacamp.com/tutorial/guide-to-the-gradient-boosting-algorithm