Occupancy Detection Dataset Analysis

Description:

The "Occupancy Detection" dataset records the presence of occupants in a room, which is determined from a set of time stamped pictures that were taken every minute from 14:19:00 on 02-02-2015 to 09:17:59 on 02-18-2015. For each instance, it also stores the attributes such as temperature, humidity, light and CO2 level in that room when the picture was clicked and whether there was anyone present in the room. The instances were already divided into test and training sets, we merged them into a single dataset and reassigned instances to create our own training, testing and validation sets.

The chosen dataset has ~20600 instances which helped us to make a good split of data for our train/test/validation sets. The database had a total of only 7 usable attributes, due to which we were able to inspect our model to determine errors in classification. All the attributes in this dataset have real valued data. The attributes we used for our model are given below:

- Time: Stores the time in HH:MM:SS format.
    - Data Type: time
- Temperature: The temperature in Celsius at that given date and time.
    - Data Type: Numeric
- Relative humidity: Percentage of humidity at the given date and time.
    - Data Type: Numeric
- Light: Light (in Lux) at the given date and time.
    - Data Type: Numeric
- CO2 : The amount of CO 2 (in ppm) at the given date and time.
    - Data Type: Numeric
- Humidity ratio: A ratio of temperature and humidity at the given date and time.
    - Data Type: Numeric
- Occupancy: The room occupancy is represented by either 1 or 0, where 1 indicates that the room is occupied and 0 indicates the room is not occupied. This is our classifier attribute.
    - Data Type: Nominal

The original dataset contained an attribute which stored a sequence number for each instance, which was discarded as part of our cleaning process. We removed this attribute because neither classification nor clustering could be performed on sequence numbers. Our classifier attribute is "Occupancy" which records binary values (0- room not occupied, 1- room occupied). The URL to the Occupancy Detection Data Set is as follows:
https://archive.ics.uci.edu/ml/datasets/Occupancy+Detection+#

Activity and Findings:

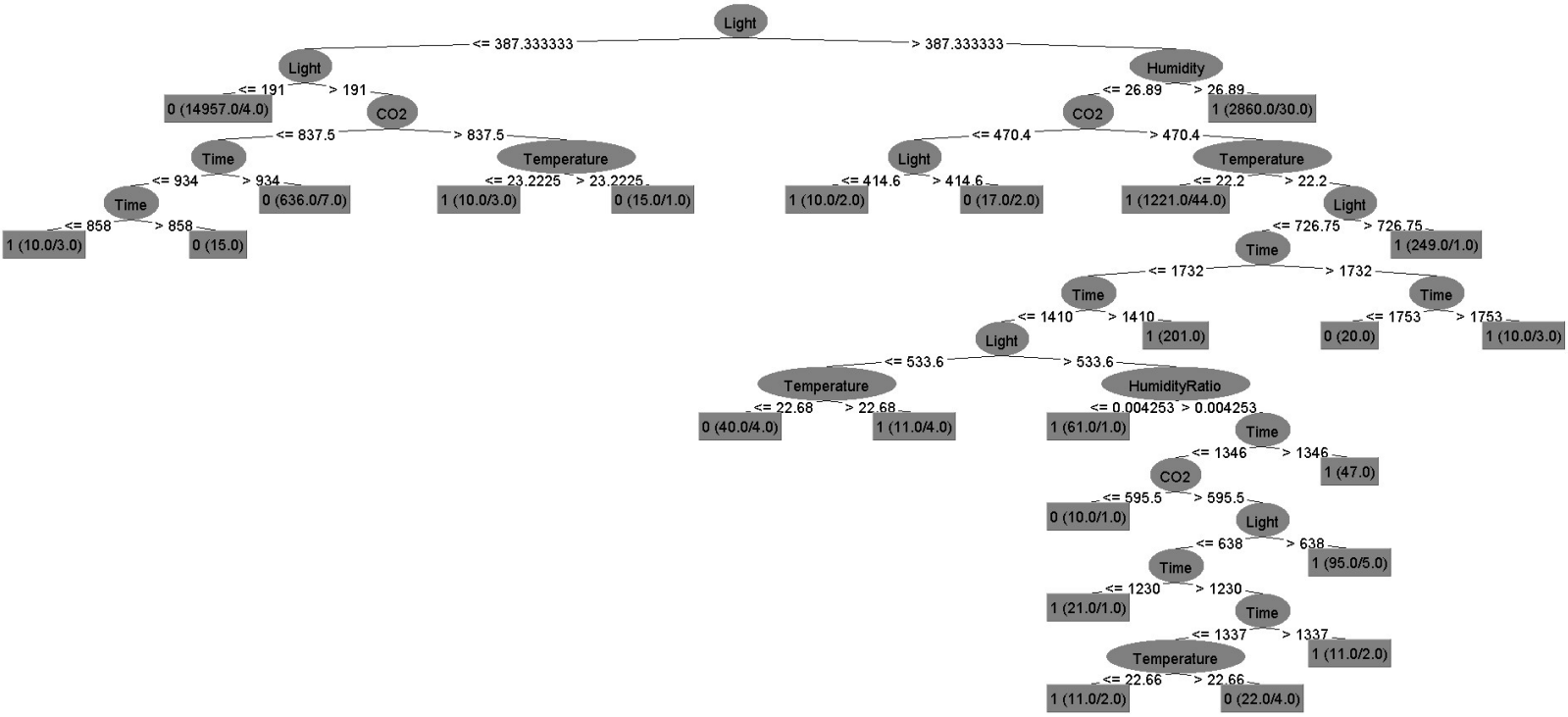
1. J48 Decision tree

For this analysis, we got rid of the attribute “Seq ID” since it was irrelevant to the analysis, and changed the format of “Date Time” attribute from yyyy/mm/dd hh:mm to just time hhmm since the date was irrelevant.

The first split that we used had 90% training and 10% testing split for which we got 98.6% accuracy. We noticed that each node of the final decision tree has only two branches because the data in our set is numeric and continuous. We also observed that since the attribute “Light” provided most information to the classifier, i.e., it had the highest information gain among all the attributes, it was chosen as the root node of the decision tree. Similarly, its child nodes were chosen in a decreasing order of information gain.

```
=== Confusion Matrix ===

   a    b   <-- classified as
1577   13 |    a = 0
   8  458 |    b = 1
```



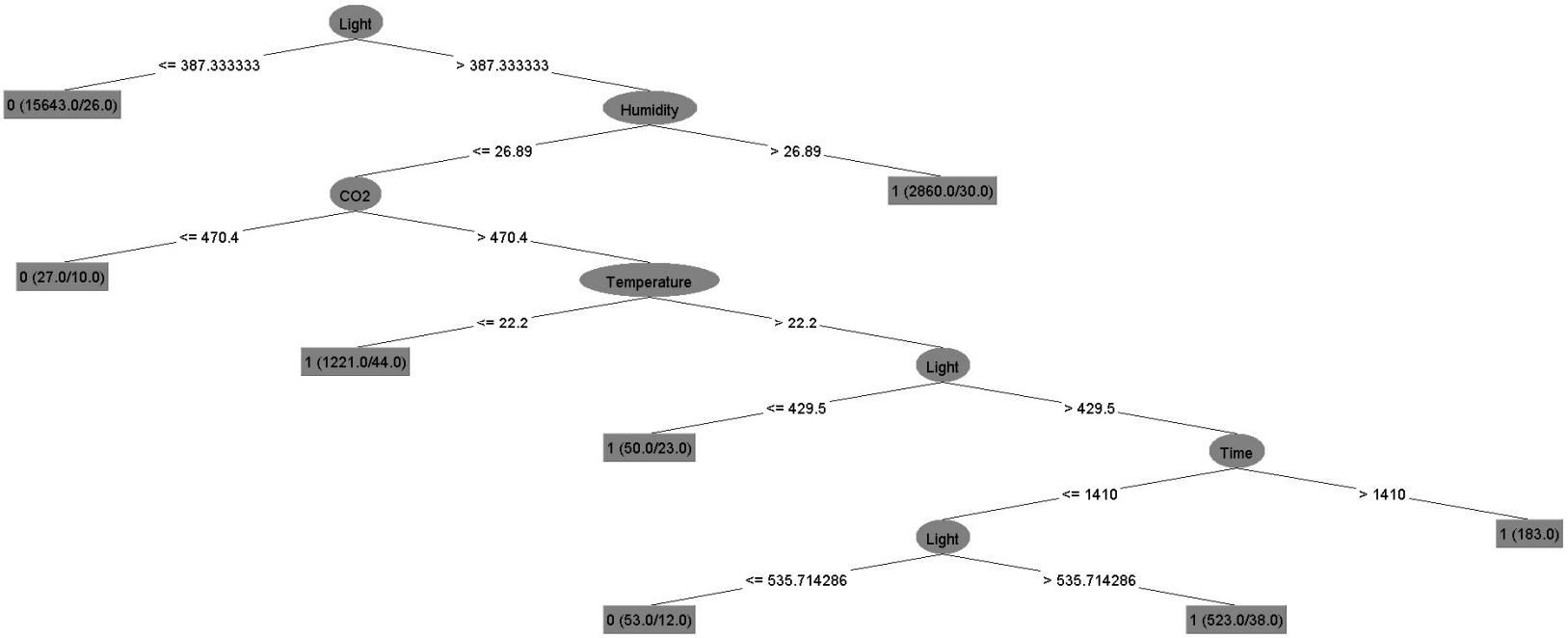
The confusion matrix states the classification result of the 2056 testing instances. Here, ‘a’ represents occupancy = 0, and ‘b’ represents occupancy = 1. Out of the 1590 instances classified as ‘a’, 1572 instances were correctly classified and out of the 464 instances classified as ‘b’, 458 instances were correctly classified.

As we can see, the model was almost perfect, however, the size of the resulting tree was huge, having 47 nodes and 24 leaves. So we increased the value of the parameter ‘minNumObj’

(minimum number of instances required for classification) from 10 to 50 to get a more compact tree.

```
=== Confusion Matrix ===

    a    b   <-- classified as
 1572   18 |    a = 0
    8  458 |    b = 1
```

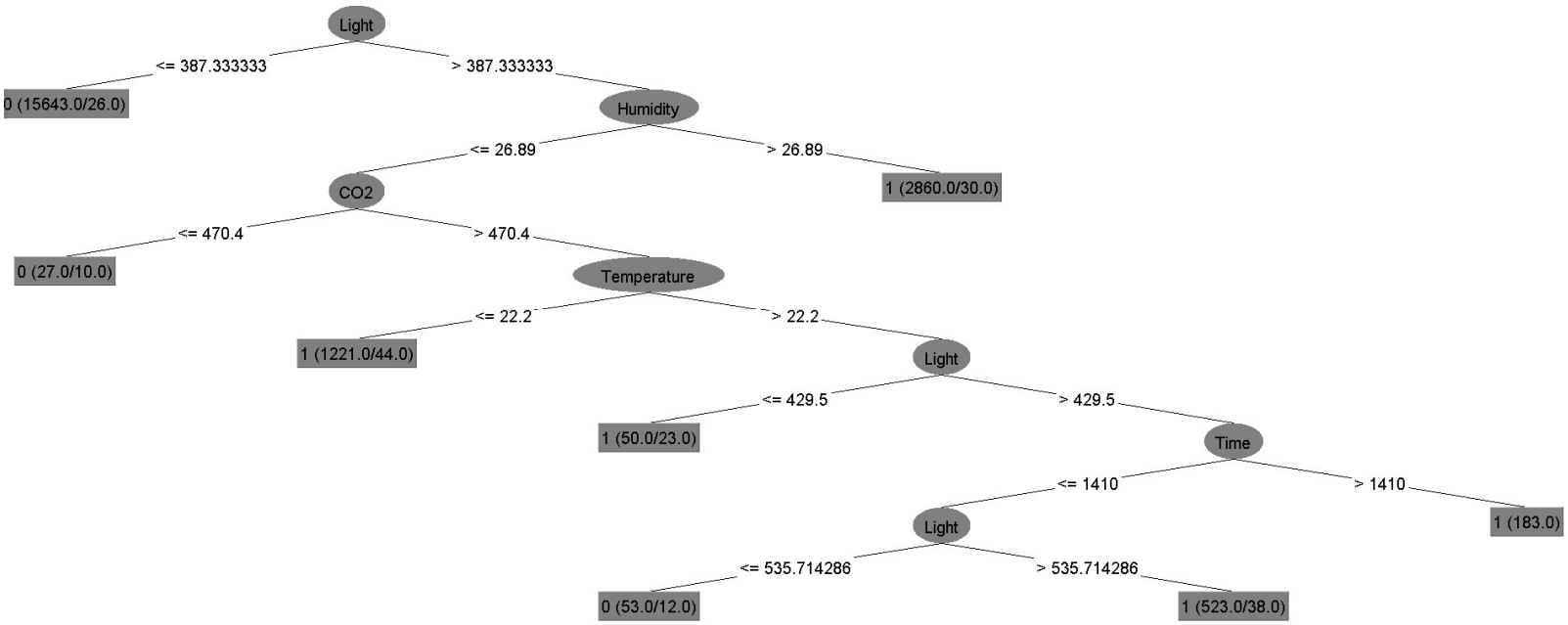


Even though this tree was more readable, having 15 nodes and 8 number of leaves, we still got the same accuracy of 98.7%. Even here, the attribute “Light” has the highest information gain and thus is chosen as the root of the tree.

The second split that we used had 70% training and 30% testing split for which we got 98.8%. (with ‘minNumObj’ set to 50)

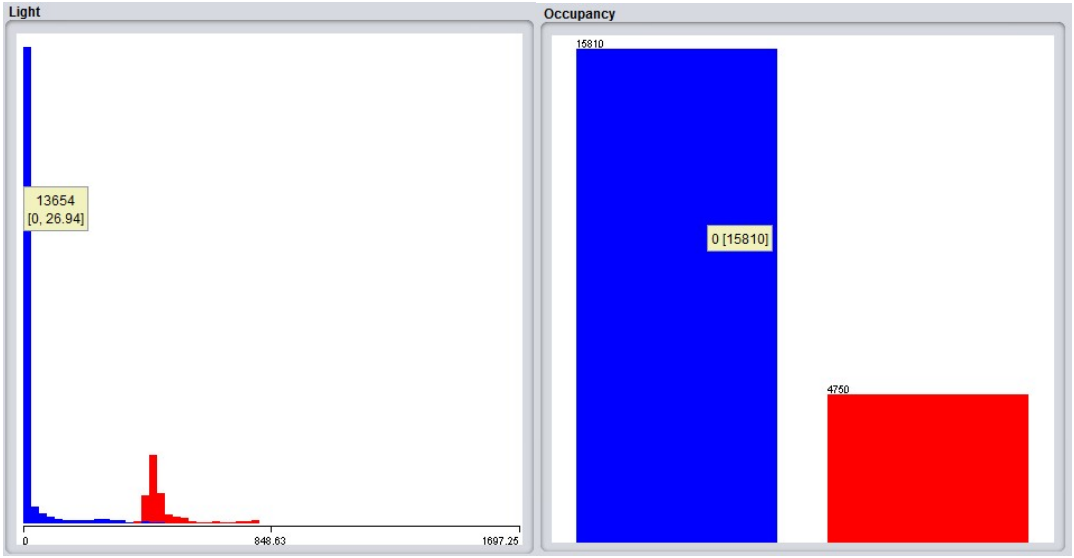
```
=== Confusion Matrix ===

    a    b   <-- classified as
 4688   62 |    a = 0
    6 1412 |    b = 1
```



The confusion matrix states the classification result of the 6168 testing instances. Out of the 4750 instances classified as ‘a’, 4688 instances were correctly classified and out of the 1418 instances classified as ‘b’, 1412 instances were correctly classified.

We observed that the reason for this consistency in the high accuracy given different splits of data was because 13654 out of 15810 of the instances had Occupancy = 0 in the dataset had “Light” set to <=26.94. Thus, all the trees we constructed so far classify over 75% of the instances at level 2 itself using just one attribute i.e. “Light”.

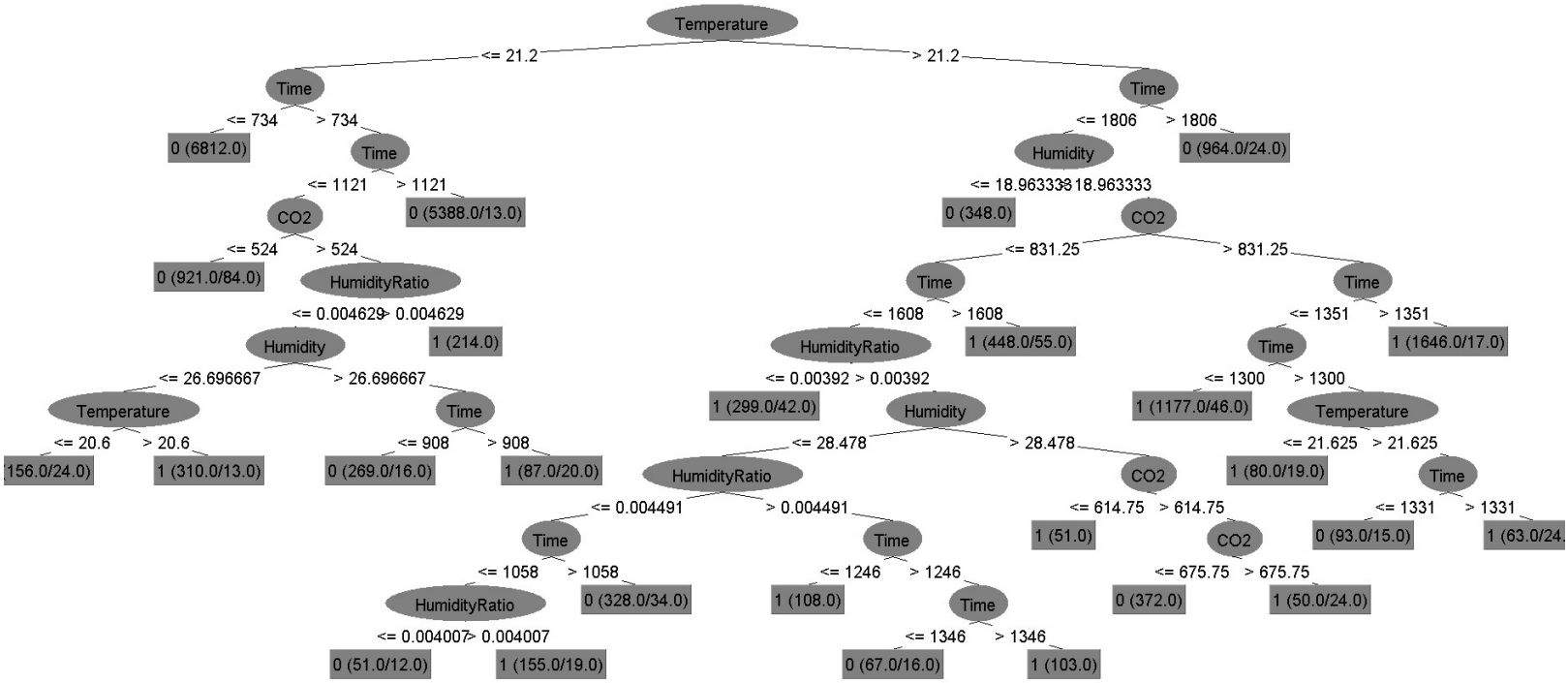


Though this resulted in models which are very accurate (over 98%), we were not able to experiment and learn much. Thus, we decided to build decision trees henceforth ignoring the "Light" attribute of the dataset.

We built a decision tree again (ignoring "Light") using a split that had 90% training and 10% testing split for which we got 96.6% accuracy.

```
=== Confusion Matrix ===

   a    b   <-- classified as
1556   34 |    a = 0
  36  430 |    b = 1
```



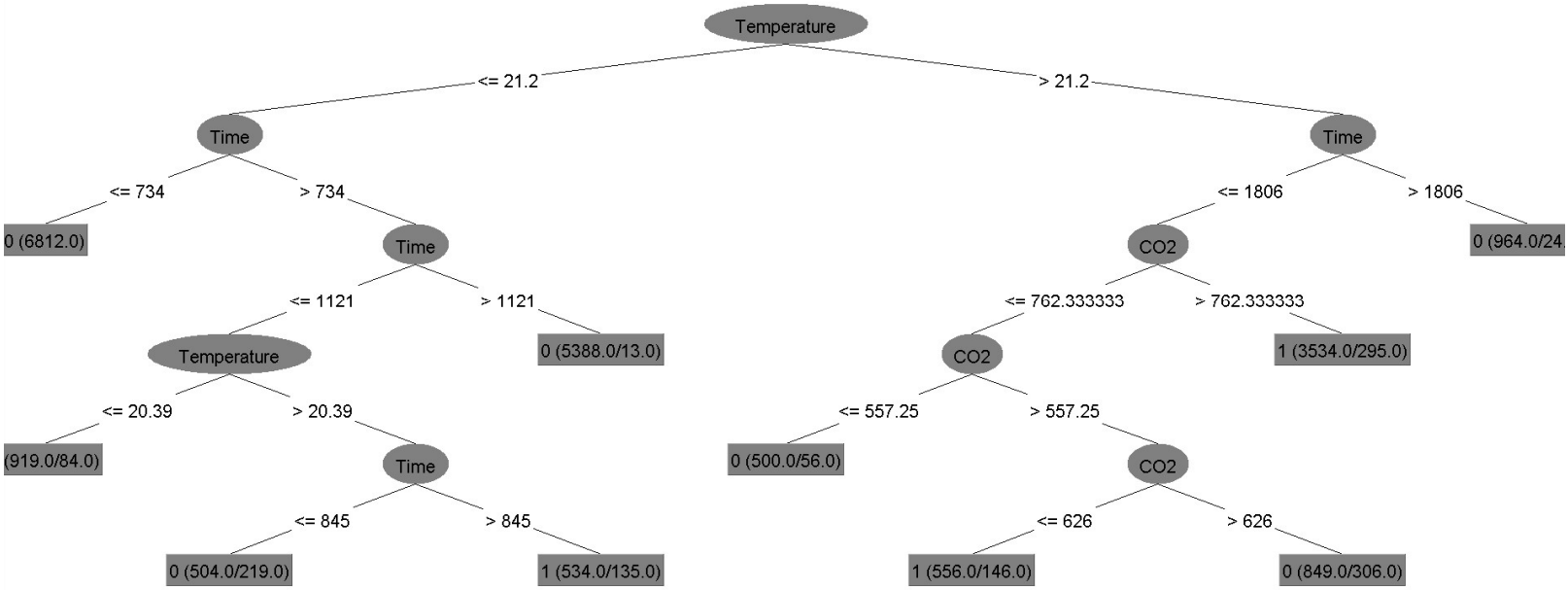
As we can see, after ignoring the attribute "Light", the tree was more balanced than all the previous trees because the instances were more evenly classified by different attributes and at different levels of the tree. But, it resulted in a huge tree of size, having 51 nodes and 26 leaves. In this model, the attribute "Temperature" has the highest information gain and thus is chosen as the root of the tree. The attribute "Time" has the 2nd highest information gain and is used to classify at level 2 of the tree. Further "Humidity", "CO2" and "Time" are chosen based on highest information gains, and the model similarly keeps choosing attributes for the further branches.

To get a more readable model, we increased the parameter 'minNumObj' from 50 to 500. The resulting decision tree ("Light" ignored and with 90-10 Train-Test split) has a decreased accuracy of 93.1%. The reason for the decreased accuracy is the change of the 'minNumObj' parameter. The 'minNumObj' parameter now limits the minimum number of instances that should

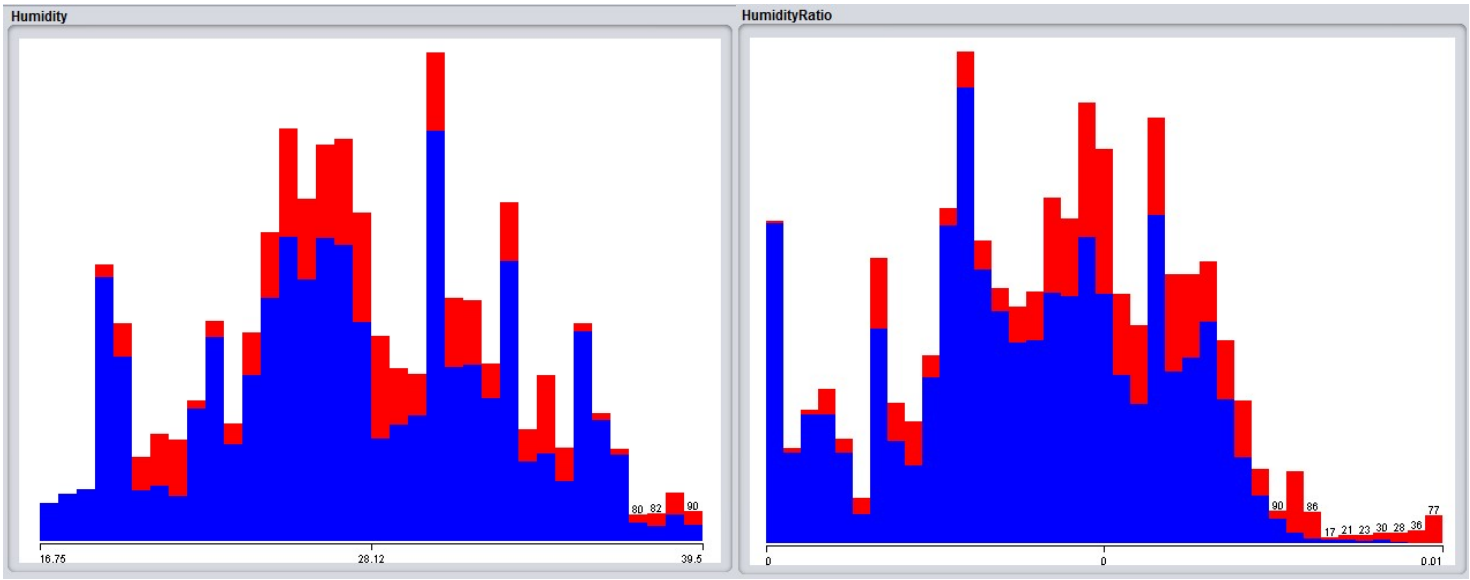
be considered for classification for each branch to 500. Thus, the overall size of the tree is reduced to just 19 nodes and 10 leaves.

```
=== Confusion Matrix ===

    a    b   <-- classified as
 1496   94 |    a = 0
   44  422 |    b = 1
```



In this model, the attribute "Temperature" has the highest information gain and thus is chosen as the root of the tree. The attribute "Time" has the 2nd highest information gain and is used to classify at level 2 of the tree. Further "CO2", "Time" and "Temperature" are chosen based on highest information gains, and the model similarly keeps choosing attributes for the further branches. This model doesn't consider the attributes "Humidity" and "HumidityRatio" as they are more evenly distributed in the training set and their information gain is lower than the other attributes being considered.



We decided to further test the above model using cross-validation using 10 folds (1 fold for testing and 9 folds for training). The cross-validation resulted in an accuracy of 93.1% and the confusion matrix generated was as follows:

```
=== Confusion Matrix ===

     a     b   <-- classified as
 14917   893 |     a = 0
   520  4230 |     b = 1
```

As we can see from the confusion matrix above, 20560 instances were classified as either 'a' (occupancy = 0) or 'b' (occupancy = 1). Out of the 15810 instances classified as 'a', 14917 (94.3%) instances were classified correctly, and out of the 4750 instances classified as 'b', 4230 (89%) instances were classified correctly.

Lessons Learned:

References:

- https://archive.ics.uci.edu/ml/datasets/Occupancy+Detection+#

Classification:

- We changed the type of attribute Occupancy from Numeric to Nominal
-

Analysis:

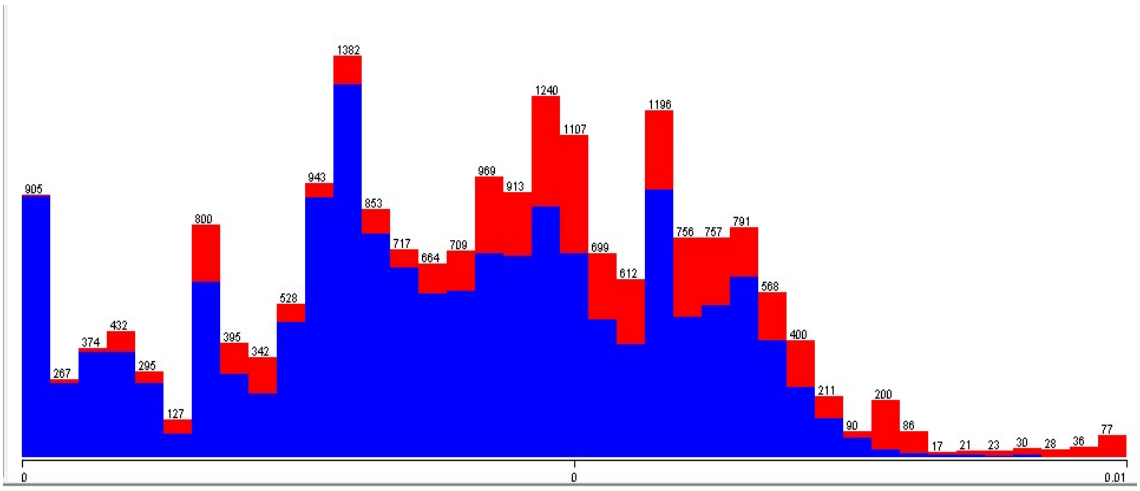
Understanding data:

Occupancy = 0 for 15810/20560 instances (76%)
Occupancy = 1 for 4750/20560 instances (24%)

Observation: Approximately 76% of instances have occupancy = 0 and only 24% of data has occupancy value set to 1. Below are some observations from the histograms and visualization plots produced by weka for each attribute when we initially load the data. Please note that all the below observations are performed on the complete dataset before splitting it into training/testing/validation sets so that we can make more informed decision on how to split the dataset for building and testing our model.

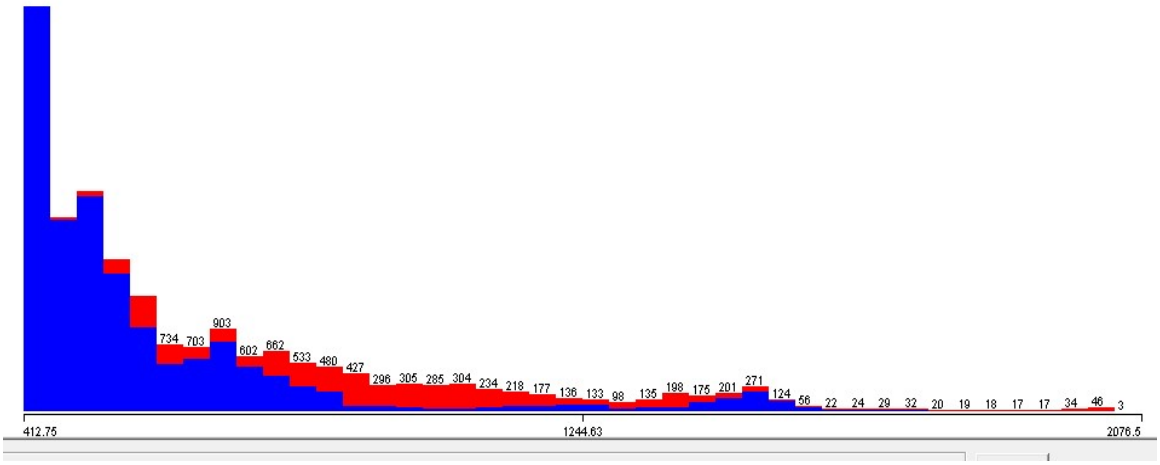
HumidityRatio values [0 – 0.006]

Large number of instances with occupancy = 0 are present in the range [0 – 0.005]. Occupancy = 1 instances are comparatively less in this range; however, they are more in the range [0.005 – 0.006].



CO2 values [412.75 – 2076.5]

The mean value is 690, so that tells us that it must be having many values on lower side of range. We observe that majority number of instances with occupancy 0 are in the range [412 - 610]



Light values [0-1697]:

Although this attribute has wide range of values, almost 99% of all instances in our dataset have values in range [0 - 800] and around 70% of these instances (~14000/20600) have value between [0-26] and occupancy = 0. The remaining 30% of these instances are in the range [404-484] with occupancy = 1. An important observation we can make here is our data is separated out very well into the two classes. So, we can visualize that our decision tree should have this attribute at a higher level as we can easily categorize all instances with light value <= 26 as occupancy 0 and >= 26 as occupancy 1.

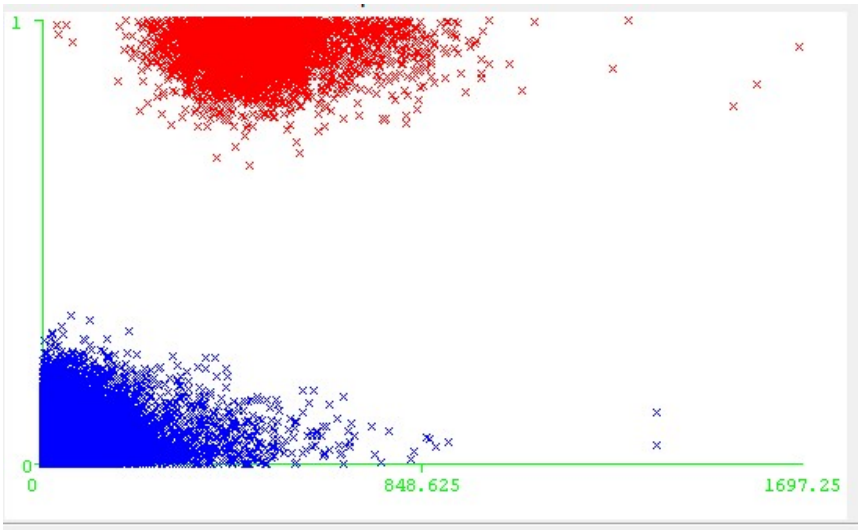
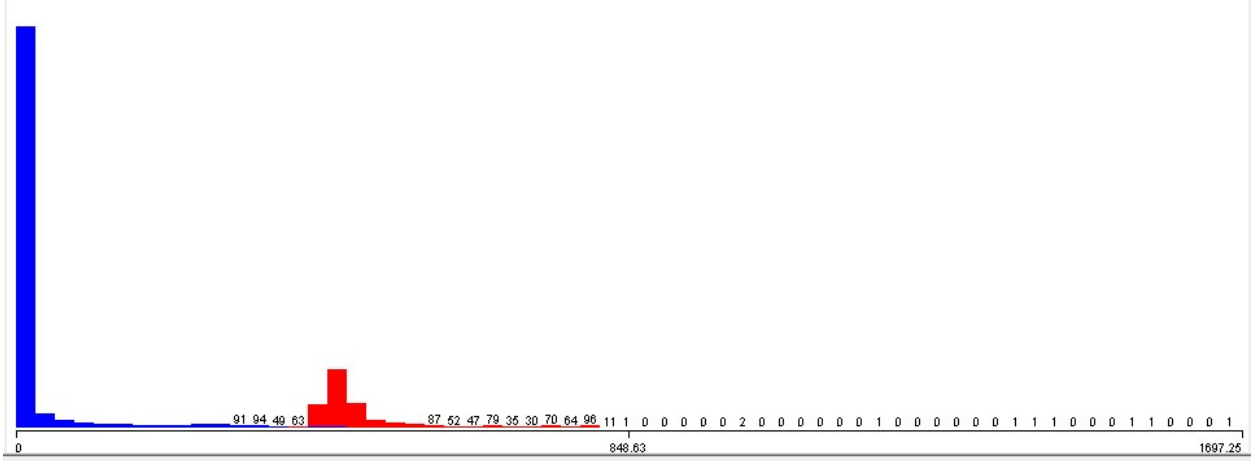
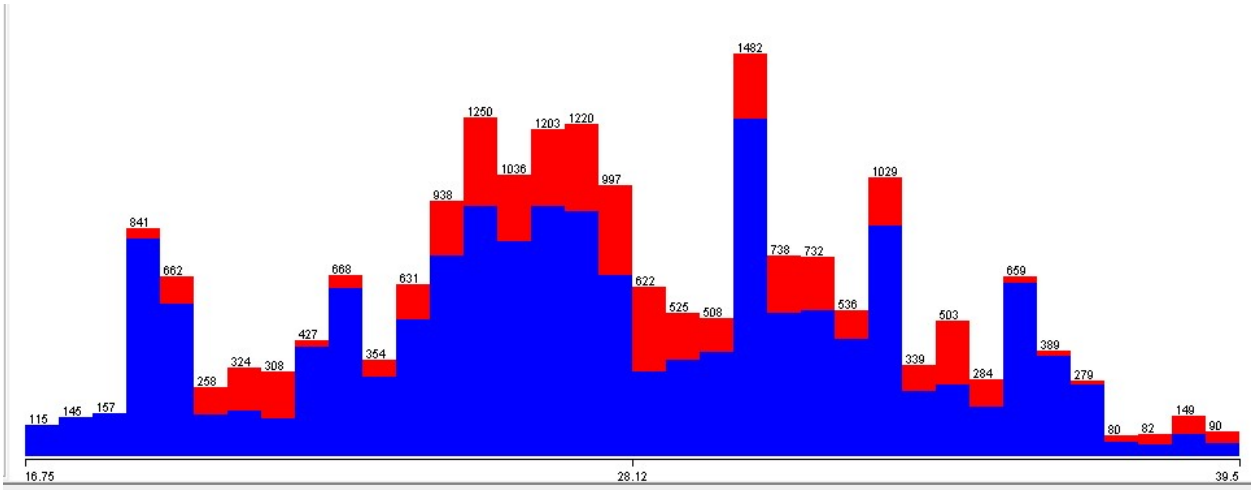


Fig. Plot of Occupancy(Y axis) against Light(X axis)



Humidity Values [16 - 40]:

This attribute has more uniform distribution of instances across the complete range as compared to other attributes, so Humidity should always be at lower levels in decision tree for final classification into leaf nodes.



Temperature values [19-25]:

Most of the instances with occupancy 0 are within range [19-20] and with occupancy 1 are spread in the range [21-25]. Mean for this attribute is 20.9, which is closer to the lower limit of range and we know this is because of the distribution of data in our dataset (75% - Occupancy 0 and 25% - Occupancy 1).

Time [0 – 23:59]:

We learned from the histogram that the room is occupied (occupancy = 1) mostly during the working hours of day from [7:30 – 19:00]. This makes time a good candidate for better classification based on time slots.

CLUSTERING

Preprocessing:

Simple K-means works by measuring the distance between data instances. Thus, having all attributes of numeric data type in our dataset makes it suitable as-is for K-means clustering. We did not perform any conversion of types except for the classifier attribute (Occupancy), which we converted to nominal type with two values (0,1). Occupancy being our classifier attribute, we ignore this attribute for all runs of K-means algorithm.

Also, there were no missing values in our dataset, so all the values used for clustering were from the original dataset.

1st run:

- Clusters: 2
- No. of iterations: 20
- Seed: 10
- Ignore attribute: Occupancy

- Cluster Mode: Use Training set
- Results:

  Clustered Instances
  0 8786 (43%)
  1 11774 (57%)

We observe cluster 1 has large chunk of data and as per the characteristics of our dataset (75% Occupancy = 0, 25% Occupancy = 1), we can relate the assigned clusters formed in the result of this run to our original classification. For this run of clustering, we carefully analyze the relation of between classes and clusters by reading the plots of individual attributes:

1. Time:

Time attribute has values in the range [00:00 – 23:59] and we observed in our initial analysis of data that occupancy is 1 only from [7:30 – 19:00]. Below is a plot for time attribute against the occupancy attribute in our original dataset and after clustering.

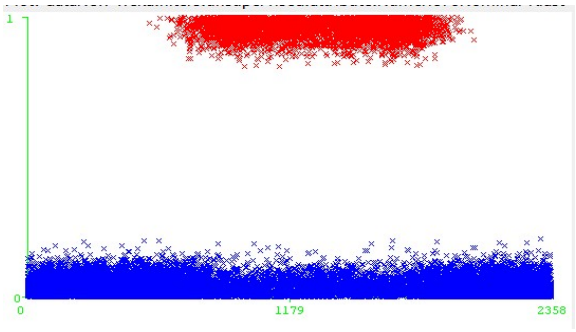


Fig. Time (X axis) vs Occupancy (Y axis) (original dataset)

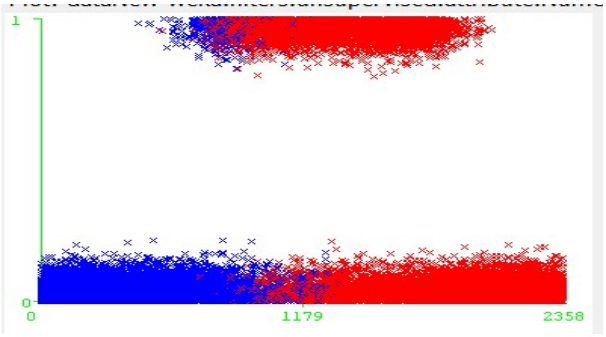


Fig. Time (X axis) vs Occupancy (Y axis) (Results of K-means)

From the above plot of clustered data, we clearly observe the partition of clusters around 11:00, this is because of the initial k seed points (2 in our case) chosen by simple k-means.
Initial cluster centroids: 2113, 1539
Final cluster centroids: 472.8, 1697

The algorithm was set to choose 2 random points to begin clustering and it chose 2313 and 1539. After multiple iterations, it obtained two clusters with mean values/centroids of clusters as 472.8 and 1697, which we can see in the plot (one cluster with centroid around 4:00AM morning, and another cluster with centroid 4:00PM evening).

2. Light:

While understanding the original dataset, we learned that many instances with occupancy = 0 had values between [0-25] and instances with occupancy 1 spanning the range [404-484]. The initial centroids chosen by k-means in this run had value : 0 and 469 and after all the iterations the final centroids were 38 (cluster 0) and 139 (cluster 1), which is evident from our resulting clusters.

As the centroid of cluster 1 moves towards 0 every iteration, many points which were part of cluster 0 are shifted to cluster 1. The situation would have been better if the cluster centroids were chosen around 300, 700.

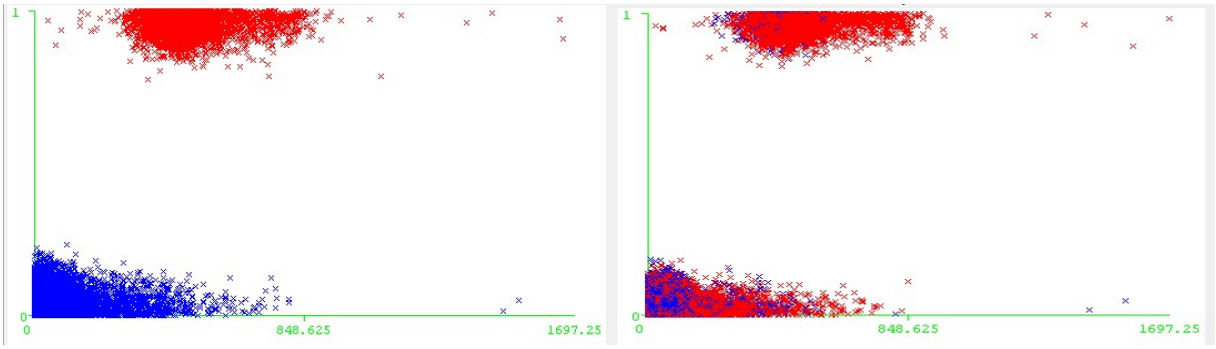


Fig. Plot of Light (X-axis) vs Occupancy (Y-axis) on original dataset values (left) and clustered data values using simple K-Means clustering (right).

Other observations:

- Clustering was decent on all other attributes as they had a better distribution than the above two attributes.
- The results of K-means clustering are much dependent on the initial cluster centroids.
- For our dataset, which has more than 2-Dimensions, K-means calculates the Sum of squared differences to measure the distance and between data points.
- The final cluster centroids of full data are appropriate for most of the attributes.

2nd run:

- Clusters: 2
- No. of iterations: 12
- Seed: 31
- Ignore attribute: Occupancy
- Cluster Mode: Use Training set
- Results:

Clustered Instances	
0	11774 (57%)
1	8786 (43%)

In this run we used a different seed value so that K-means would select a different initial seed points for this run. Below is a table which summarizes the change in centroids:

Centroids	Time	Temperature	Humidity	Light	CO2	HumidityRatio
1st Run Initial centroids cluster 0	2313	20.7	22.29	0	455	0.00336
1st Run Initial centroids cluster 1	1539	23.2	28.65	469	1124.25	0.005043
2nd Run Initial centroids cluster 0	2035	21.2	19.39	0	472	0.003012
2nd Run Initial centroids cluster 1	531	20.5	23.745	0	585	0.003536

We analyzed this run for individual attributes and compared it with the previous run. We used the additional attribute which stores the cluster assignments on completion of K-means for our analysis.

Observations:

Although we changed the seed value and K-means started its clustering from different initial centroids, the results of k-means matched the results of previous run after all its iterations. This similarity can be seen from the table of final cluster centroids in Weka in which all the final centroids match very closely to final centroids of the previous run. We analyzed what caused this, our observation follows:

- Most of the instances were assigned in a similar fashion as before, however, we observed that the clusters formed for values of light attribute differed largely.
- The initial cluster choices made was 0 for both clusters and we know that K-means optimizes locally and not globally. So, it forms very close clusters which do not match our original classification.
- Below is a plot of original data and clustered data, we can observe most points assigned to the two clusters are mixed and we cannot conclude anything from this clustering.
- We get a good match of final centroids between several runs because of the characteristics of our data. On observing the results ignoring the light attribute, we see a little deviation in the results as the cluster assignments remained the same for other attributes.
- We also noticed that with this set of centroids K-means only made 12 iterations to come to a similar conclusion as the previous run which performed 22 iterations.

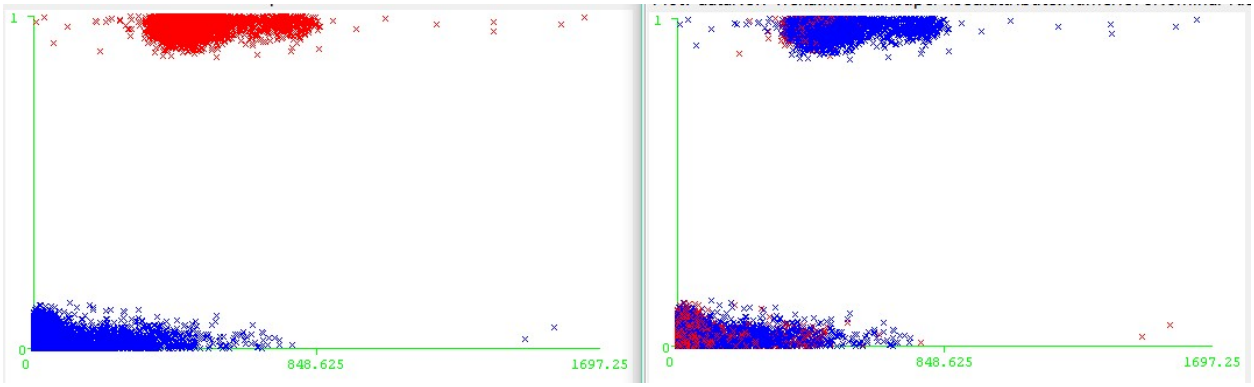


Fig. Plot of Light (X-axis) vs Occupancy (Y-axis) on original dataset values (left) and clustered data values using simple K-Means clustering (right).

3rd run:

- Clusters: 2
- No. of iterations: 11
- Seed: 37
- Ignore attribute: Occupancy
- Cluster Mode: Classes to Cluster evaluation
- Initial centroids:

	Time	Temperature	Humidity	Light	CO2	HumidityRatio
Cluster0:	836	20.865	19.7675	433	605.5	0.003008
Cluster1:	1549	21.39	18.89	33.5	436.75	0.002969

- Results:

  Clustered Instances
  0 8786 (43%)
  1 11774 (57%)

- Classes to Cluster Evaluation matrix:

```
    0    1  <-- assigned to cluster
 8188 7622 | 0
  598 4152 | 1
```

The instances with occupancy=1 are correctly placed in a cluster with 87% accuracy, however, we have a poor clustering for the instances with occupancy = 0 (~51% accuracy). The plots generated with classes to clusters evaluation methods gives a better understanding of the wrongly clustered instances. The correct instances are plotted as before (shown as x) and all incorrectly classified instances are shown as rectangles. Below is a plot of Humidity Ratio and occupancy in original dataset and same plot after classes to clusters evaluation:

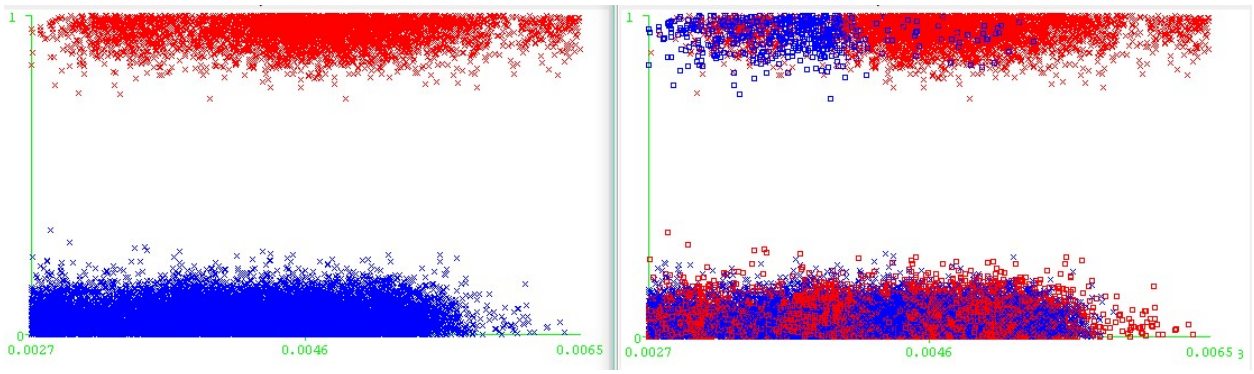


Fig. Plot of Humidity Ratio (X-axis) vs Occupancy (Y-axis) on original dataset values (left) and clustered data values using simple K-Means clustering along with classes to clusters evaluation (right).

Here, we observe several instances with occupancy 0 clustered incorrectly, which is evident from the evaluation matrix (~57% accuracy for Occupancy 0). We observed the plots for all the attributes and found similar results of clustering, based on which we could conclude the following:

The clustering performed by K-means is completely based on Euclidian distance and it has no information about the occupancy. So, it tries to cluster together instances which have similar characteristic features. In our dataset, we know that our data is biased (3:1) as we have 75% of instances in a class and 25% in the other. On closely looking at the data and analyzing these plots, we understood that those 25% instances have very similar characteristics. For example, the room is occupied mostly when the light value is high, the temperature is favorable, and in a particular time frame every day. K-means clusters these instances, with similar characteristics very well and helps us understand the co-relation between the instances in our dataset.