Principal Component Analysis & Binary Classification To Determine Office Room Occupancy

Gursimarjit Singh Saini
Student Id: 40220457
Github Link: https://github.com/GursimarSaini/INSE6220Project/blob/master/OccupancyDetectionPCA.ipynb

***Abstract*— Principal Component Analysis (PCA) is a fast and flexible unsupervised dimensionality reduction method that transforms a high dimensional data with correlated features to low dimensional data with uncorrelated features. This report illustrates the use of PCA when applied to the office room occupancy data set attributes and then classifying orthogonal data to check if the room is occupied. Determination of occupancy detection in a room can lead to considerable energy savings in modern smart home/buildings. Four classifiers: Decision Trees Classifier, Extra Trees Classifier, Random Forest Classifier, and Light Gradient Boosting Machines were picked with the help of PyCaret to use on data. The evaluation of models is done on basis of MCC score. Extra Trees classifier performed the best on the original data set without tuning and LightGBM with tuning. At last, ML classification model's prediction is explained with Shapley value.**

Keywords—Principal Component Analysis (PCA), Binary Classification, Decision Trees Classifier, Random Forest Classifier, Extra Trees Classifier, Light Gradient Boosting Machines, Explainable AI, Shapley Value.

I. INTRODUCTION

With the decreasing price of sensors and the availability of reasonable computational power for automation systems, determining occupancy is a very promising way to lowering energy usage in buildings through appropriate control of HVAC and lighting systems. Threat of climate adversity has made it important for the production of most energy efficient products [1]. The precise detection of occupancy in buildings has been projected to save energy in the range of 30 to 42 percent. When occupancy data was employed as an input for HVAC control algorithms, it resulted in energy savings of 37 percent without sacrificing indoor climate and between 29 and 80 percent in another [2]. When privacy matters are considered, it makes much more sense to use sensors for getting accurate occupant numbers than to use cameras. Determining building inhabitants behavior and Security are another two applications for occupancy detection.

The research [2] used data from light, temperature, humidity, and CO2 sensors to detect occupancy, as well as a digital camera to determine ground occupancy for data labelling. This data set created for occupancy detection is used for this study.

Working with a huge dataset as what used in this study is usually perplexing and laborious. To make the research easier, the approach must incorporate dimension reduction, while preserving the majority of the data variability. PCA is generally used for such tasks [3], which is described and implemented in Section 3, after giving a brief Exploratory Data Analysis in Section 2. Section 4, throws light on applicable Classifiers with brief explanations and discussion of the most promising model that helps in detection for this process in Section 5. Section 6 gives a brief introduction about Shapley value and how it's implemented. Section 7 closes with a summary of the findings.

II. EXPLORATORY DATA ANALYSIS

Exploratory Data Analysis refers to the crucial process of conducting preliminary investigations on data in order to uncover patterns, spot anomalies, test hypotheses, and validate assumptions using summary statistics and graphical representations. Here it's done in three parts, first by giving a brief introduction for the raw data set, then discussing the cleaning process and description for used data set. At last, checking distribution and outliers with Box Plots and correlation of points with Correlation Matrix.

A. Raw Data Set Description

As mentioned in the study [2], the following variables were observed in an office space with approximate dimensions of 5.85m, 3.50m, 3.53m (W D H): timestamp, temperature, humidity, light, and CO2 levels. The study collects the data using a microcontroller. It was linked to a ZigBee radio, which was used to relay the data to a recording station. A digital camera was utilized to assess whether or not the room was inhabited. Every minute, the camera time stamped an image, which was then manually examined to identify the data. The humidity ratio is another additional variable in the data model, calculated as:

$$W = 0.622 \times \frac{p_w}{p - p_w}$$

The data was collected in February in Mons, Belgium, during the winter. The room was heated by hot water radiators, which kept the temperature above 19 degrees Celsius. The models are tested for data sets with the office door open and closed in order to estimate the difference in occupancy detection accuracy provided by the models. The measurements were obtained at 14-second intervals/3-4 times every minute, and then averaged for that minute.

B. Data Cleaning

All three data sets were missing column name for their first column, which was named as "id" and then dropped in data pre-

processing. For the purpose of this study, only training data will be used.

C. *Used Data Set Description*

Data Set	Number of Observations	Data Class Distribution	
		0 (non occupied)	1 (occupied)
Training	8143 of 7 variables	0.79	0.21
Testing 1	2665 of 7 variables	0.64	0.36

Fig. 1. Data Set Description

The description for these two datasets is summarized in Figure 1. No duplicate rows or NaN values were found for both datasets. And all the values are floating point numbers, except the column "Occupancy" which is labelled with int values, 0 and 1.

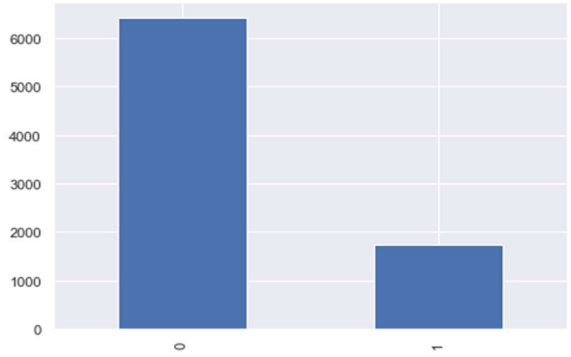

Fig. 2. Class Distribution

The distribution of class can be seen with the bar plot in Fig. 2. As said, the label 1 represents that the room was occupied and Class 0 for unoccupied rooms.

D. *Data Analysis*

	Temperature	Humidity	Light	CO2	HumidityRatio
count	8.143000e+03	8.143000e+03	8.143000e+03	8.143000e+03	8.143000e+03
mean	7.818326e-16	4.467615e-16	2.233807e-16	-1.884775e-16	1.116904e-16
std	1.000061e+00	1.000061e+00	1.000061e+00	1.000061e+00	1.000061e+00
min	-1.592248e+00	-1.624791e+00	-6.137261e-01	-6.165933e-01	-1.394355e+00
25%	-9.038502e-01	-1.000115e+00	-6.137261e-01	-5.330748e-01	-9.201475e-01
50%	-2.252867e-01	8.877312e-02	-6.137261e-01	-4.869407e-01	-7.243751e-02
75%	7.581387e-01	8.681862e-01	7.027469e-01	1.027265e-01	5.742528e-01
max	2.518470e+00	2.420232e+00	7.326619e+00	4.524170e+00	3.066492e+00

Fig. 3. Descriptive Statistics

Standardization is put into use to adjust each input variable independently by removing the mean, and dividing by the standard deviation to shift the distribution to have a mean of zero and a standard deviation of one [4]. After standardization, the Descriptive Statistics metrics can be seen in Figure 3**.** If it's not done, then covariances for larger number ranges will be much higher.

Putting this standardize matrix in a Box Plot gives us the idea about the distribution of the data, measures of central tendency and spread. In Figure 4**,** we can see that, all the data attributes are positively skewed to an extent. Data is centered around 0, because of standardization and variability is minimum for the same reason. Outliers are present for 3 of the 5 attributes and all of them are on the skewed side of whiskers.

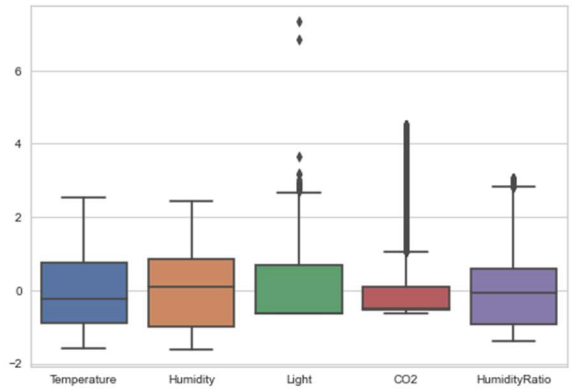

Fig. 4. Box Plot

To understand the relationship among the attributes, Correlation Matrix and Pair Plot are used. It's evident that almost all of the parameters are positively correlated with each other's.

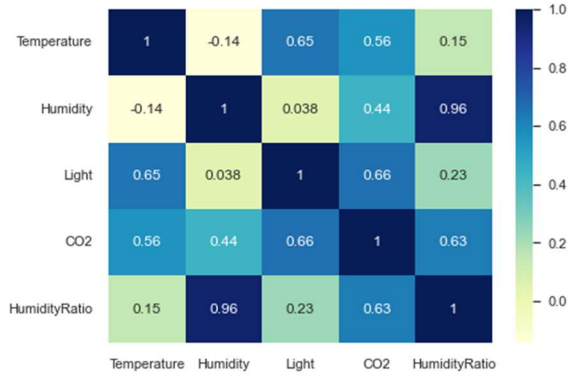

Fig. 5. Correlation Matrix

There is no presence of any variable with negative correlation with all of the others variables. CO2 has significant correlation with rest of variables over others as seen in Figure 5.

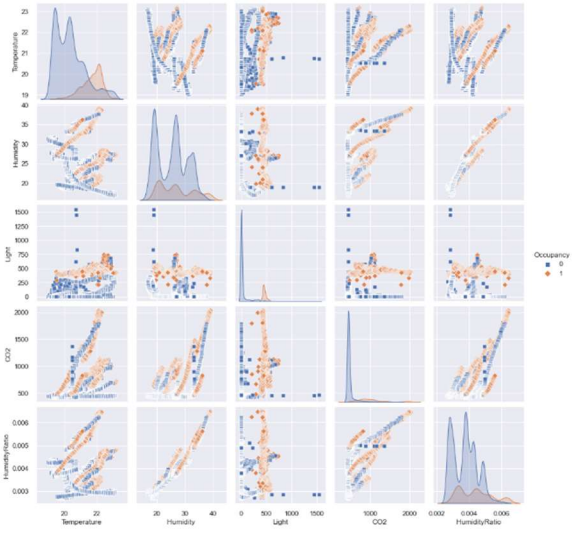

Fig. 6. Pair Plot

This can also be supported with the help of pairplot in Figure 6. The strong positive correlations are determined with increasing line. Whereas, weak correlations form clusters rather than an increasing line in pair plot.

The multiple correlations among data set parameters, is the reason why PCA is implemented to get un-correlated data.

III. Principal Component Analysis

PCA is typically used to reduce the dimensionality of data while retaining as much of the information present in the original data as feasible. It does this by examining a data table including observations characterized by numerous dependent variables that are, in general, inter-correlated. Its purpose is to extract the key information from the data table and express this information as a set of new orthogonal variables known as principal components. Simply put, PCA is important so as to:

- Extract the most relevant information from the data table,
- Compress the size of the data set by maintaining just the most significant information,
- Simplify the data set description, and
- Evaluate the structure of the observations and variables.

PCA output comprises of coefficients that specify the linear combinations used to obtain the new variables (PC loadings) as well as the new variables themselves (PCs). The first PC must have the greatest potential variance. The second component is calculated with the constraint of being orthogonal to the first component and having the greatest possible inertia. The other components are calculated in the same way. [6]

A. Implementation of PCA in steps

	Temperature	Humidity	Light	CO2	HumidityRatio
0	23.18	27.2720	426.0	721.25	0.004793
1	23.15	27.2675	429.5	714.00	0.004783
2	23.15	27.2450	426.0	713.50	0.004779
3	23.15	27.2000	426.0	708.25	0.004772
4	23.10	27.2000	426.0	704.50	0.004757

Fig. 7. Before Standardization

We need to make sure, data should be structured in a typical matrix format, with n rows of samples and p columns of variables. There should be no missing values: each variable should have a value for each sample, which can be zero [7]. The steps needed for PCA is as follows [3].

	Temperature	Humidity	Light	CO2	HumidityRatio
0	2.518470	0.278526	1.573763	0.364948	1.091757
1	2.488967	0.277713	1.591735	0.341881	1.080555
2	2.488967	0.273645	1.573763	0.340290	1.075888
3	2.488967	0.265508	1.573763	0.323587	1.066555
4	2.439796	0.265508	1.573763	0.311655	1.049523

Fig. 8. After Standardization

1. Centering the dataset: For this step we subtract the mean of a variable from all of its values, so that the data stays centered on the origin of main components, because any algorithm which is based on distance computations are affected a lot if the data used isn't normalized/centralized [8] as seen in Fig. 7 & 8.

$$Y = H \times X \tag{1}$$

For our study, standardization is preferred over centering to avoid precision error when range of variables is different.

	0	1	2	3	4
0	1.000123	-0.141777	0.650022	0.559963	0.151780
1	-0.141777	1.000123	0.037833	0.439077	0.955315
2	0.650022	0.037833	1.000123	0.664104	0.230449
3	0.559963	0.439077	0.664104	1.000123	0.626633
4	0.151780	0.955315	0.230449	0.626633	1.000123

Fig. 9. Covariance Matrix

2. Calculate Covariance Matrix: Covariance matrix of size $p \times p$ is produced to check if data set has correlated features and also so as eigen decomposition can be applied to the data [3].

$$S = \frac{1}{n-1} \times Y^T \times Y \tag{2}$$

$$A = \begin{bmatrix} 0.343856 & 0.535864 & -0.713374 & 0.225382 & -0.186850 \\ 0.395664 & -0.574111 & 0.009249 & 0.226966 & -0.679888 \\ 0.414149 & 0.444612 & 0.665424 & 0.433177 & 0.019235 \\ 0.550070 & 0.120106 & 0.110938 & -0.817265 & -0.052622 \\ 0.501117 & -0.413692 & -0.189517 & 0.205245 & 0.706895 \end{bmatrix}$$

Fig. 10. Eigenvectors

$$\lambda = \begin{bmatrix} 2.736860 \\ 1.699679 \\ 0.348872 \\ 0.214393 \\ 0.000809 \end{bmatrix}$$

Fig. 11. Eigenvalues

3. Eigen Decomposition: Eigenvectors and eigenvalues are obtained from the Covariance matrix S with eigen decomposition. Eigenvectors give direction of Principal Components with variance of PCs dented with eigenvalues and are given by [3]:

$$S = A \times \lambda \times A^T \tag{3}$$

where A is a $p \times p$ orthogonal eigenvector matrix and is a diagonal eigenvalue matrix. In our study, we have a 5×5 eigenvector matrix and a 1×5 column matrix of eigenvalues, both given in Fig 10 & 11, respectively.

4. Principal Components: The last step yields a $n \times p$ matrix Z, with its rows giving the observed values and columns representing the PCs as given in Fig. 14. Given by equation [3]:

$$Z = Y \times A \tag{4}$$

The variance of j^{th}PC is given as following [3]:

$$l_j = \frac{\lambda_j}{\Sigma_j^p \lambda_j} \times 100\%, for\ j = 1, \dots, p \tag{5}$$

where $\lambda_j$gives the variance of j^{th} PC. Both Scree/Elbow plots can be used to get an idea of how

many PCs are needed to represent the variance present in the data.

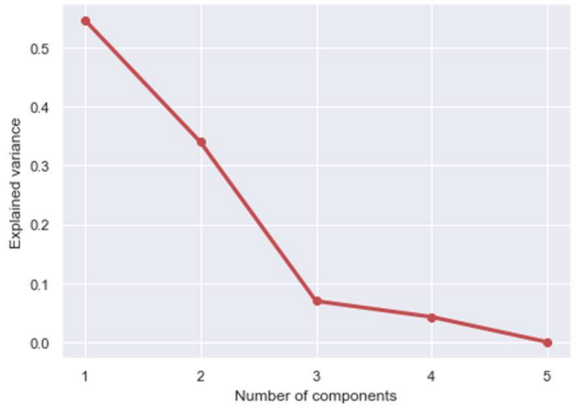

Fig. 12. Scree Plot

In our study, we found out that variance accounted for first PC is l_1= 54.7% and by 2nd PC it is l_2= 33.9% and that by 3rd PC is l_3= 6.97%. The elbow joint in the scree plot, shows a bend at PC number 3, that is also supported with Pareto Chart. So, it's safe to assume that dimensions of eigenvector or Z components can be reduced to 3.

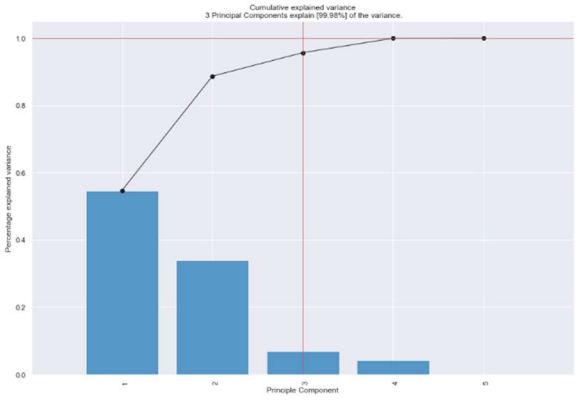

Fig. 13. Pareto Chart

Because first 3 PCs explain 99.98% of the whole variance of dataset. The first PC component Z1 is given by [3]:

$$Z_1 = 0.55007 \times X_4 + 0.501116 \times X_5 + 0.414149 \times X_3 + 0.395664 \times X_2 + 0.343856 \times X_1$$

X_4 (CO_2), X_5 (Humidity Ratio), X_3 (Light), X_2 (Humidity), and X_1 (Temperature), contribute most to the 1st PC, respectively. For Z_2, we got

$$Z_2 = 0.574111 \times X_2 + 0.535864 \times X_1 + 0.444612 \times X_3 - 0.413692 \times X_5 + 0.120106 \times X_4$$

For both first two principal components we don't have any attribute contributing negligibly to them. But in case of, third principal component, X_2 doesn't affect it that much, so effective PC will be:

$$Z_3 = -0.713374 \times X_1 + 0.665424 \times X_3 - 0.189517 \times X_5 + 0.110938 \times X_4$$

This same could be verified with the help of PC coefficient Plot as in Fig. 14.

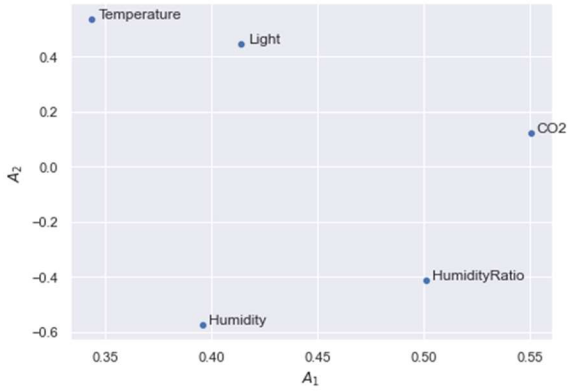

Fig. 14. PC Coefficient Plot

Temperature and Humidity lies in the lower range for the first PC, along with CO_2 and Humidity Ratio being the most important factors for consideration, with light being somewhere in the middle of first two and latter two. Whereas all the bottom 3 contributors of 1st PC got to be at the top for 2nd PC and CO_2 being the least.

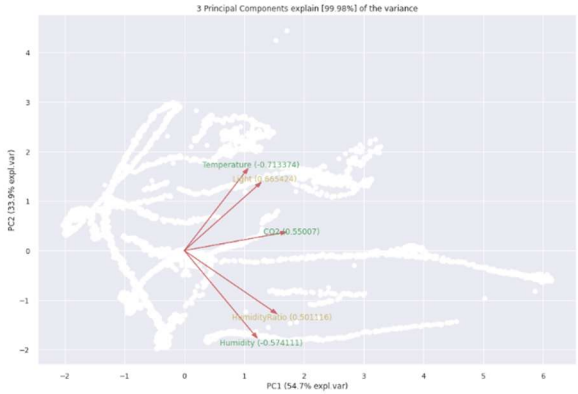

Fig. 15. 2D Biplot

Biplot gives the same information as of Fig. 14. The angles between the vectors (rows of eigenvector matrix) and axes (representing the first two PCs) gives the contribution of variables to PCs [3], i.e., the vector with smallest angle with the axe contributes most to that axe/PC. Also, each observation is scattered as a point in the plot.

This same could be represented for 3 PCs with help of 3d Biplot which is not added to this report as it was looking incomprehensible but is a part of Colab notebook .

IV. CLASSIFICATION ALGORITHMS

Classification assigns a label value to a given class and then determines if a specific type is of one kind or another [9]. It can be used with both unstructured and structured data. It's a supervised learning concept and basically categorizes data into classes [10]. So, depending upon the data, it can either be binary classification or a multi-class classification. Classification algorithms works in a lot of different ways, primarily Logic

based, Perceptron based, Statistic techniques, Support Vector Machines, and others. Making a choice among these, is inconclusive as it depends to a great extent on usage and attributes/properties of data [11]. So, to make the process easier for this study, the choice is made with help of PyCaret. This could be seen in Figure 16.

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
lr	Logistic Regression	0.9853	0.9931	0.9873	0.9546	0.9683	0.9588	0.9610	0.0330
gbc	Gradient Boosting Classifier	0.9709	0.9913	0.9272	0.9451	0.9313	0.9131	0.9168	0.3530
ridge	Ridge Classifier	0.9691	0.0000	0.9988	0.9097	0.9445	0.9249	0.9331	0.0160
lda	Linear Discriminant Analysis	0.9651	0.9935	0.9988	0.9035	0.9398	0.9179	0.9274	0.0180
rf	Random Forest Classifier	0.9622	0.9878	0.9370	0.9114	0.9184	0.8942	0.8988	0.3100
nb	Naive Bayes	0.9609	0.9937	0.9873	0.8913	0.9276	0.9028	0.9126	0.0160
knn	K Neighbors Classifier	0.9608	0.9749	0.9029	0.9193	0.9043	0.8800	0.8851	0.0610
ada	Ada Boost Classifier	0.9541	0.9785	0.9249	0.9130	0.9015	0.8733	0.8865	0.1370
qda	Quadratic Discriminant Analysis	0.9527	0.9932	0.9647	0.8886	0.9119	0.8823	0.8944	0.0160
lightgbm	Light Gradient Boosting Machine	0.9457	0.9769	0.9006	0.9007	0.8905	0.8563	0.8635	0.0720
dt	Decision Tree Classifier	0.9438	0.9094	0.8497	0.9091	0.8590	0.8254	0.8390	0.0210
svm	SVM - Linear Kernel	0.9417	0.0000	0.9583	0.8719	0.8956	0.8596	0.8754	0.0320
et	Extra Trees Classifier	0.9352	0.9647	0.8665	0.8765	0.8603	0.8194	0.8273	0.2090
dummy	Dummy Classifier	0.7877	0.5000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0150

Fig. 16. Comparison of Models with PyCaret

Generally, The Receiver Operating Characteristics (ROC) curve, which illustrates the connection between the true positive rate and the false positive rate, is used for visual comparison of classification models. The area under the ROC curve represents the model's accuracy [10].

For testing purposes, the training data was split into 70-30 ratio. From Figure 16, it's obvious that the best model is Extra Trees Classifier on original data without tuning. Three other models were chosen, on basis of their evaluation scores being close to the best model, those are: Decision Trees Classifier, Random Forest Classifier, and Light Gradient Boosting Machine Classifier. Tuning was done with the help of PyCaret too, so no hyperparameter selection for best model performance was explicitly done in this report [12].

A. Decision Trees Classifier

Decision Tree is a supervised learning approach that may be used to solve both classification and regression problems, however it is most commonly employed to solve classification issues. Internal nodes contain dataset attributes, branches represent decision rules, and each leaf node provides the conclusion in this tree-structured classifier.

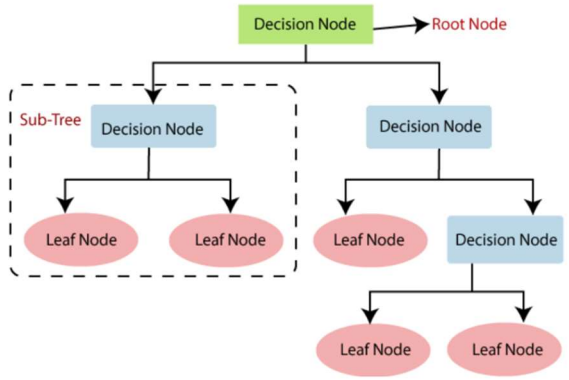

Fig. 17. Decision Trees [13]

The Decision Node and the Leaf Node are the two nodes of a Decision tree. Leaf nodes are the result of those decisions and do not include any more branches, whereas Choice nodes are used to make any decision and have several branches. The judgments or tests are made based on the characteristics of the provided dataset. It's a graphical depiction for obtaining all feasible answers to a problem/decision depending on certain parameters. It's termed a decision tree because, like a tree, it starts with the root node and grows into a tree-like structure with additional branches. A decision tree simply asks a question and divides the tree into subtrees based on the answer (Yes/No) [13].

The procedure for determining the class of a given dataset in a decision tree starts at the root node of the tree. This algorithm checks the values of the root property with the values of the record (actual dataset) attribute and then follows the branch and jumps to the next node depending on the comparison. The algorithm checks the attribute value with the other sub-nodes and moves on to the next node. It repeats the process until it reaches the tree's leaf node [13]:

The trained and tuned model resulted in the ROC curve shown in Fig 18 with an AUC = 99.42% which was 98.93% without tuning of the model.

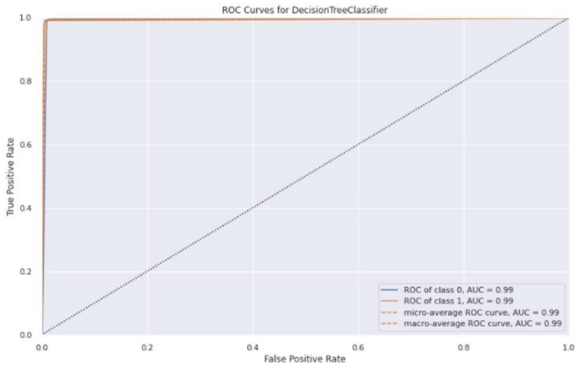

Fig. 18. AUC Curve for Decision Trees Classifier

B. Random Forest Classifier

Random forests, also known as random choice forests, are an ensemble learning approach for classification, regression, and other problems that works by generating a large number of decision trees during training. The decision forest is often trained using the "bagging" approach. The bagging approach is based on the premise that combining learning models improves the final output [14]. For classification problems, the random forest output is the class chosen by the majority of trees. [15] [16].

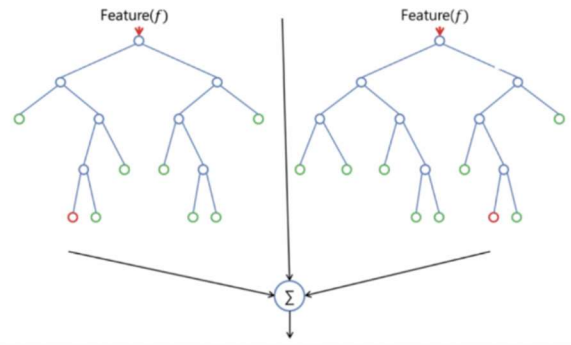

Fig. 19. Random Forests Sample [14]

Deep-grown trees tend to acquire very irregular patterns: they overfit their training sets, resulting in low bias but very high variance. Random forests are a method of averaging numerous deep decision trees that have been trained on various regions of the same training set to reduce variance [17]. This results in a minor increase in bias and some loss of interpretability, but it considerably enhances the final model's performance.

The hyperparameters of random forest are nearly equivalent to those of decision trees and bagging classifiers. Random Forest adds more randomness to the model as it grows the trees. When dividing a node, the best feature from a random collection of qualities is chosen rather than the most important trait. As a result, there is a large range of variability, resulting in a better model overall. As a result, the random forest strategy for dividing a node examines just a random subset of the features [14].

The trained and tuned model resulted in the ROC curve shown in Fig 20 with an AUC = 99.88% which was 99.93% without tuning of the model.

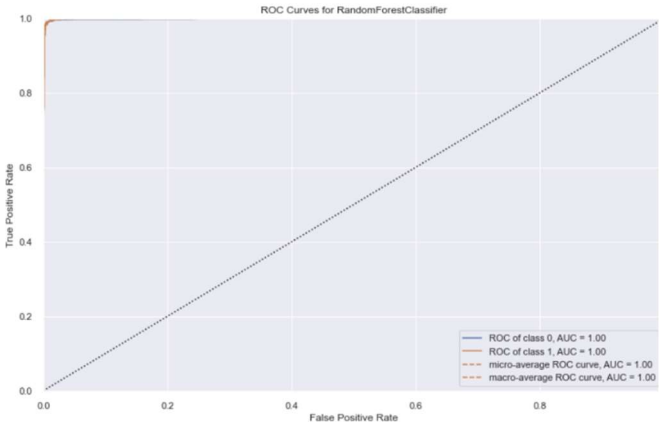

Fig. 20. AUC Curve for Random Trees Classifier

C. *Extra Trees Classifier*

To get its classification result, Extra Trees Classifier (Extremely Randomized Trees Classifier) is a form of ensemble learning technique that aggregates the outcomes of several de-correlated decision trees collected in a "forest." It is conceptually identical to a Random Forest Classifier, with the exception of how the decision trees in the forest are constructed [18].

The Extra Trees Forest's Decision Trees are all made from the original training sample. Then, at each test node, each tree is given a random sample of k features from the feature set, from which it must choose the best feature to split the data according to certain mathematical criteria (typically the Gini Index). Multiple de-correlated decision trees are created from this random sample of features [18].

The Extra Trees Classifier is less computationally costly than a Random Forest since splits are picked at random for each feature. However, Decision Trees have a high variance, Random Forests have a medium variance, and Extra Trees have a low variance [19].

During the construction of the forest, the normalized total reduction in the mathematical criteria used in the decision of feature of split (Gini Index if the Gini Index is used in the construction of the forest) is computed for each feature to perform feature selection using the above forest structure. This value is called the Gini Importance of the feature. To execute feature selection, each feature is ranked in descending order by Gini Importance, and the user selects the top k features based on his or her preferences [18].

The trained and tuned model resulted in the ROC curve shown in Fig 21 with an AUC = 99.50% which was 99.94% without tuning of the model. Making it third best at AUC when tuned, whereas it was showing the best AUC without it.

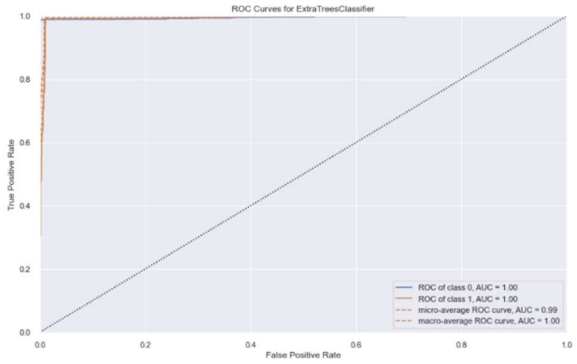

Fig. 21. AUC Curve for Extra Trees Classifier

D. *Light Gradient Boosting Machines Classifier*

Gradient boosting is a type of machine learning method that may be used to solve classification or regression predictive modelling challenges. Decision tree models are used to create ensembles. To repair the prediction mistakes caused by past models, trees are introduced to the ensemble one at a time and fitted. The boosting model is a sort of ensemble machine learning model. Models are fitted using a gradient descent optimization approach and any arbitrary differentiable loss function. Gradient boosting gets its name from the fact that the loss gradient is reduced when the model is fitted, much like a neural network. [20].

LightGBM improves on the gradient boosting technique by using a sort of autonomous feature selection and focuses on boosting cases with greater gradients. This can result in a significant increase in training speed and enhanced prediction performance. The implementation presents two major concepts: EFB and GOSS [20].

1. Gradient-based One-Side Sampling, or GOSS for short, is a gradient boosting technique modification that concentrates emphasis on training samples that result in a bigger gradient, hence speeding up learning and decreasing the approach's computing cost.
2. Exclusive Feature Bundling, or EFB, is a method for grouping mutually incompatible characteristics that are sparse (mainly zero), such as categorical variable inputs that have been one-hot encoded. As such, it falls under the category of automated feature selection.

It splits the tree leaf-wise with the greatest fit since it is based on decision tree algorithms, unlike other boosting methods split the tree depth-wise or level-wise rather than leaf-wise. So, while growing on the same leaf in Light GBM, the leaf-wise approach may minimize more loss than the level-wise technique, resulting

in considerably superior accuracy than any of the existing boosting algorithms can seldom accomplish. It is also astonishingly quick, thus the name 'Light' [21].

The Fig. 22. and Fig. 23. shows this difference.

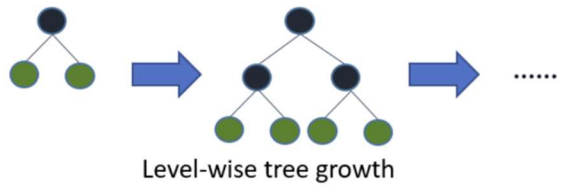

Fig. 22. Level-wise tree growth with xbgoost

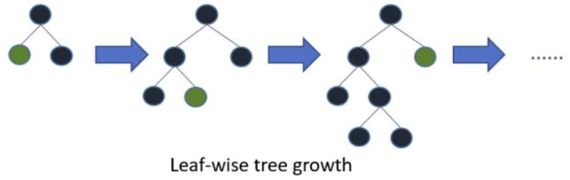

Fig. 23. Leaf-wise tree growth with LightGBM

The trained and tuned model resulted in the ROC curve shown in Fig 24 with an AUC = 99.96% which was 99.91% without tuning of the model. Giving it the best AUC score when tuned.

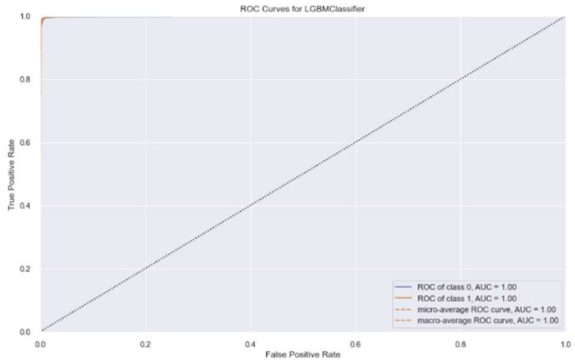

Fig. 24. AUC Curve for LGBM

V. CLASSIFICATION V/S CLASSIFICATION+PCA

This section discusses the effect of PCA on classification of the dataset for the four models that performed the best at the original dataset without PCA. For each algorithm, F1 score and MCC score have been noted to evaluate the performance of each model. Evaluation was done with the help of PyCaret's "evaluate_model" function [22]. PyCaret provides a few performance metrics by itself, namely: Recall, Precision, Accuracy, AUC (Area Under Curve), F1, Kappa, and MCC. The first five of these can be derived from Confusion Matrix alone. For explaining it, first we need to define [23]:

1. A "True Positive" (TP), is when the model correctly classifies the positive class.
2. A "False Positive" (FP) is when the model incorrectly classifies the positive class.
3. A "False Negative" (FN) is when the model incorrectly classifies the negative class.
4. A "True Negative" (TN) is when the model correctly classifies the negative class.

In confusion matrix, the first row and first column give the TP value of model, second column from first row gives the FP. Subsequently, the first column from second row represents FN, while the rightmost bottom gives TN [23].

For our dataset, the confusion matrix is given in Fig 25.

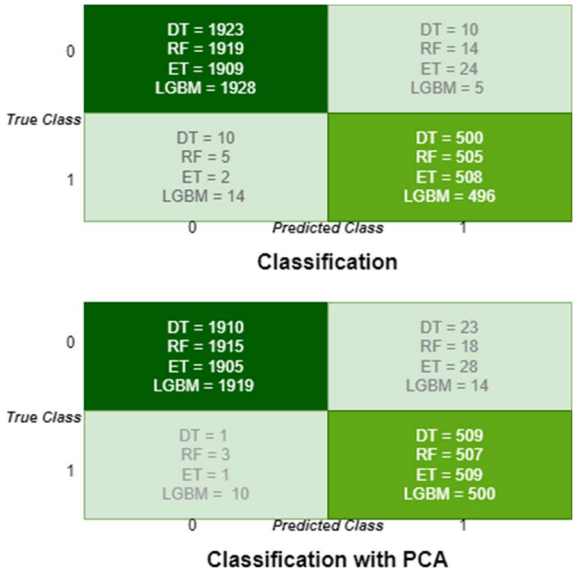

Fig. 25. Confusion Matrices

The first confusion matrix is for the classification models applied on original data directly and second for classification on PCA'ed data.

Now, Accuracy is given as the number of correct predictions over the output size, i.e.,

$$Accuracy = TP + TN / TP + TN + FP + FN,$$

which is best for LGBM model for both pre-PCA and PCA'ed tuned models. Precision measures the rate of false positives and is given by,

$$Precision = TP / TP + FP,$$

Recall measures false negatives against true positives.

$$Recall = TP / TP + FN,$$

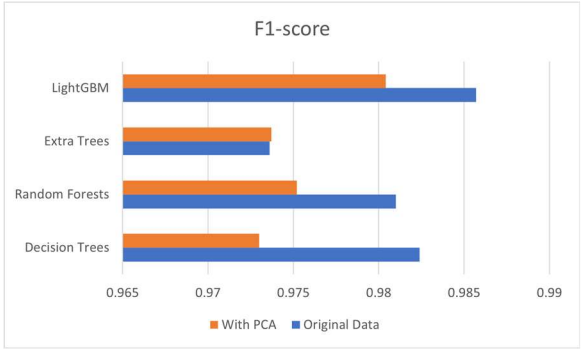

Fig. 26. F1 scores

Extra Trees gives the best performance for both Precision and Recall criterion. F1-score is the harmonic mean of precision and recall accomplishing both high precision and high recall.

$$F1 = 2 \times \frac{precision \times recall}{precision + rec}$$

the model's performance is given in Fig 26.

F1 score ignores the count of True Negatives. In contrast, MCC kindly extends its care to all four entries of the confusion matrix [24]. MCC is given as,

$$MCC = \frac{TN \times TP - FP \times FN}{\sqrt{(TN + FN)(FP + TP)(TN + FP)(FN + TP)}}$$

F1 score is highly influenced by which class is labeled as positive. The F1 score differs so much from MCC in magnitude as the minority class is labeled as negative [24]. So, Matthews Correlation Coefficient score should be preferred over F1, in cases when the weight of both TP and TN is supposed to be same.

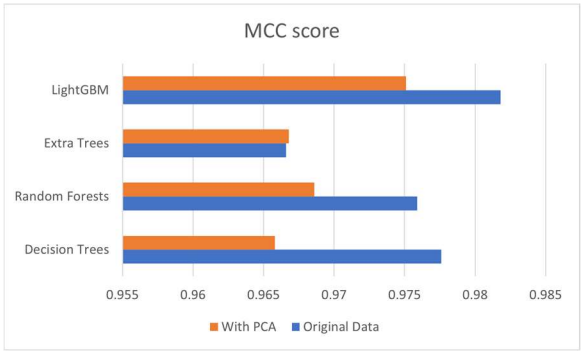

Fig. 27. MCC scores

As may be seen from the Fig. 27 Light Gradient Boosting Machine was the most outperforming model as it displayed the highest MCC, F1, Accuracy, Recall, and AUC of all the models, for both without PCA and with PCA, except precision score, for which it's score reduces when applied with PCA.

VI. Explainable AI with Shapley Values

SHAP (SHapley Additive exPlanations) is a strategy for explaining individual predictions that is based on theoretically ideal Shapley values in the coalition game. The purpose of SHAP is to compute the contribution of each feature to the prediction of an instance x. A data instance's feature values operate as coalition members. Shapley values inform us how to distribute the "payout" (the prediction) among the characteristics in a fair manner. A player can be a single feature value, as in tabular data. A player may alternatively be defined as a set of feature values. To compute Shapley values, we simulate that only some feature values ("present") are active and others ("absent") [25].

SHAP describes the three desired traits listed below:

1. Local Accuracy: The model's local accuracy indicates the uncertainty of a position in relation to other ones nearby.
2. Missingness: Missingness states that a missing feature receives a zero attribution.
3. Consistency: The consistency property states that if a model is changed in such a way that the marginal contribution of a feature value rises or remains constant (independent of other features), the Shapley value similarly grows or remains constant [25].

The summary plot combines the relevance of features with the impacts of features. Each point on the summary plot represents a Shapley value for a feature or instance. The feature determines the location on the y-axis, while the Shapley value determines the position on the x-axis. The color denotes the feature's value, which ranges from low to high. Overlapping points are jittered in the y-axis direction to provide insight into the distribution of Shapley values per feature. The features are arranged in descending order of significance. [25].

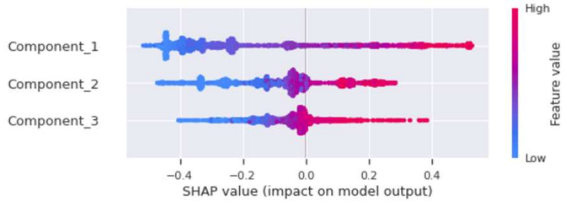

Fig. 28. Shapley Summary

The summary Fig. 28, shows the first signs of a link between the value of a feature/Principal Component and its influence on prediction [25].

SHAP clustering works by grouping each instance's Shapley values. This means that you group occurrences based on their explanatory similarity. All SHAP values have the same unit — the prediction space unit. The purpose of clustering is to locate groups of examples that are similar. Clustering is often based on characteristics. Feature scales are often used.[25].

The plot is made up of several force plots, each of which explains a forecast for a specific event. The force charts are rotated vertically and placed side by side based on their clustering similarity [25]. As a result, just one plot is displayed, but the remainder may be seen by running the notebook.

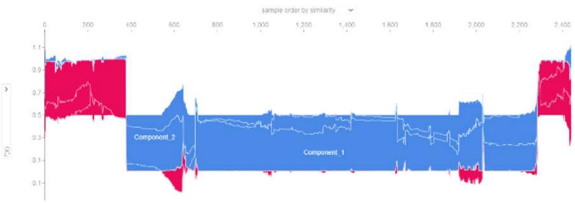

Fig. 29. Shapley Reasoning

VII. Conclusion

The work was done on the same dataset after it had been pre-processed to eliminate extraneous data. PCA was applied, and it was revealed that 95.57 percent of the variance was in the top three PCs, therefore the dimension was decreased to three, and the data was analyzed using them. In Section 6, we found out tree classifiers got to be performing best at used dataset. But

performance for these was reduced by using PCA in general except for Extra Trees Classifier, for which it was mostly equivalent to results of direct classification. Besides that, the LightGBM model proved to be the most successful in recognizing occupancy, with an accuracy of 99.39 percent for data without PCA and 99.16 percent for data with it. Shap was utilized to determine if the prediction is spread equitably across the feature values.

In summary, the dataset was successfully reduced to a lower dimension and converted into uncorrelated data, the chosen ML classifier model displayed great reliability in recognizing the correct target, and the model's prediction is explained using Shap. As a result, the project's goals were met.

REFERENCES

[1] B. Boardman, "New directions for household energy efficiency: evidence from the UK", Energy Policy, vol. 32, no. 17, pp. 1921-1933, 2004. Available: 10.1016/j.enpol.2004.03.021.

[2] L. Candanedo and V. Feldheim, "Accurate occupancy detection of an office room from light, temperature, humidity and CO 2 measurements using statistical learning models", Energy and Buildings, vol. 112, pp. 28-39, 2016. Available: 10.1016/j.enbuild.2015.11.071.

[3] A. Ben Hamza, Advanced Statistical Approaches to Quality, unpublished.

[4] "sklearn.preprocessing.StandardScaler", scikit-learn, 2022. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html .

[5] "The why of principal component analysis – (i) Standardization & Covariance – GodzillaButNicer", Godzillabutnicer.com, 2022. [Online]. Available: https://godzillabutnicer.com/the-why-of-principal-component-analysis-standardization-covariance/.

[6] H. Abdi and L. Williams, "Principal component analysis", Wiley Interdisciplinary Reviews: Computational Statistics, vol. 2, no. 4, pp. 433-459, 2010. Available: 10.1002/wics.101.

[7] "Data Analysis in the Geosciences", Strata.uga.edu, 2022. [Online]. Available: https://strata.uga.edu/8370/lecturenotes/principalComponents.html.

[8] "Data Standardization or Normalization", RP's Blog on Data Science, 2022. [Online]. Available: https://datafai.com/2017/10/27/data-standardization-or-normalization/.

[9] "A Complete guide to Understand Classification in Machine Learning", Analytics Vidhya, 2022. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/09/a-complete-guide-to-understand-classification-in-machine-learning.

[10] M. Waseem, "Classification In Machine Learning | Classification Algorithms | Edureka", Edureka, 2022. [Online]. Available: https://www.edureka.co/blog/classification-in-machine-learning/#:~:text=In%20machine%20learning%2C%20classification%20is,recognition%2C%20document%20classification%2C%20etc.

[11] S. Kotsiantis, I. Zaharakis and P. Pintelas, "Machine learning: a review of classification and combining techniques", Artificial Intelligence Review, vol. 26, no. 3, pp. 159-190, 2006. Available: 10.1007/s10462-007-9052-3.

[12] "Release Notes - PyCaret Official", Pycaret.gitbook.io, 2022. [Online]. Available: https://pycaret.gitbook.io/docs/get-started/release-notes.

[13] "Machine Learning Decision Tree Classification Algorithm - Javatpoint", www.javatpoint.com, 2022. [Online]. Available: https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm.

[14] "Random Forest Algorithm: A Complete Guide", Built In, 2022. [Online]. Available: https://builtin.com/data-science/random-forest-algorithm.

[15] Tin Kam Ho, "Random decision forests," Proceedings of 3rd International Conference on Document Analysis and Recognition, 1995, pp. 278-282 vol.1, Available: 10.1109/ICDAR.1995.598994.

[16] Tin Kam Ho, "The random subspace method for constructing decision forests", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 20, no. 8, pp. 832-844, 1998. Available: 10.1109/34.709601.

[17] T. Hastie, R. Tibshirani and J. Friedman, The Elements of Statistical Learning, 2nd ed. New York, NY: Springer New York, 2009.

[18] "ML | Extra Tree Classifier for Feature Selection - GeeksforGeeks", GeeksforGeeks, 2022. [Online]. Available: https://www.geeksforgeeks.org/ml-extra-tree-classifier-for-feature-selection/.

[19] "An Intuitive Explanation of Random Forest and Extra Trees Classifiers", Medium, 2022. [Online]. Available: https://towardsdatascience.com/an-intuitive-explanation-of-random-forest-and-extra-trees-classifiers-8507ac21d54b.

[20] J. Brownlee, "How to Develop a Light Gradient Boosted Machine (LightGBM) Ensemble", Machine Learning Mastery, 2022. [Online]. Available: https://machinelearningmastery.com/light-gradient-boosted-machine-lightgbm-ensemble/.

[21] "Light GBM vs XGBOOST: Which algorithm takes the crown", Analytics Vidhya, 2022. [Online]. Available: https://www.analyticsvidhya.com/blog/2017/06/which-algorithm-takes-the-crown-light-gbm-vs-xgboost/.

[22] "Classification — pycaret 2.3.5 documentation", Pycaret.readthedocs.io, 2022. [Online]. Available: https://pycaret.readthedocs.io/en/latest/api/classification.html.

[23] "A Pirate's Guide to Accuracy, Precision, Recall, and Other Scores", FloydHub Blog, 2022. [Online]. Available: https://blog.floydhub.com/a-pirates-guide-to-accuracy-precision-recall-and-other-scores/.

[24] "Matthews Correlation Coefficient: when to use it and when to avoid it", Medium, 2022. [Online]. Available: https://towardsdatascience.com/matthews-correlation-coefficient-when-to-use-it-and-when-to-avoid-it-310b3c923f7e#:~:text=MCC%20ranges%20from%20%2D1%20to,especially%20the%20negative%20class%20samples.

[25] C. Molnar, "9.6 SHAP (SHapley Additive exPlanations) | Interpretable Machine Learning", Christophm.github.io, 2022. [Online]. Available: https://christophm.github.io/interpretable-ml-book/shap.html.