ABSTRACT:

This paper delves into the determinants of electricity consumption changes across major states in India. Employing the Laspeyres-based Parametric Divisia Method and the Simple Average Parametric Divisia Method, the study conducts a period-wise decomposition analysis. It specifically examines the influences of scale, structural, and intensity effects on the overall changes in electricity consumption. The states are ranked according to the proportionate contributions of these effects to the total change in electricity consumption. The paper advocates for the application of the Divisia index over simple additive methods in this analytical context. The analysis draws on secondary cross-sectional data, encompassing sectoral electricity consumption and sectoral output shares, for benchmark years.

INTRODUCTION:

It is a well-known fact that power generation and capacity expansion have always been essential for economic growth. In India, where the demand for electricity has outpaced supply, due to a chronic shortage of electricity for a long period, a well-managed and sufficient supply of electrical power in a plan is far from perfect. Added to this, there exist large interstate variations in the sectoral consumption of electricity. These interstate disparities in consumer category-wise sale of electricity arise mainly due to three factors, viz, variations in total production of the economy (scale effect), structural composition of the sectors (structural effect) and technical efficiency or, per unit of electricity use with respect to gross state domestic product (GSDP) (intensity effect). This paper collates sufficient information regarding the relative change in scale of the economy, structural composition of the sectors and technical competence in the increasing level of consumption of electrical power for the states. It thereby converges towards a sectoral disaggregation in the demand analysis required to endorse efficiency and safety in the allocation of electricity to the states. The relative contributions of the scale effect, structural effect and intensity effect in the total change in electricity consumption have been estimated individually for the 18 major states using period-wise decomposition analysis. The power sector in India is characterised by the vertical integration between generation, transmission and distribution. There are state electricity boards (SEBs) which possess, control and put on the market electricity from the clutch of generating units within the state boundaries. Apart from SEBs, there are other state-owned utilities also known as the central sector utilities (CSUs). Transmission and distribution (T&D) losses are high in most of the states due to inadequate investments, defective metering, tapping and unmetered supply. The all India T&D losses as a percentage of availability have increased from 34 per cent in 2001-02 to 38.3 per cent in 2002-031 (teddy, 2003-04). The disappointing and weakening financial health of SEBs has acted as a constraint not only for adding new capacity, improving the T&D system and carrying out renovation and modernising programmes, but also for carrying out much needed reforms in electricity utilities. Presently (2003-04), India has an installed generating capacity of nearly 112 giga watts (GW). This includes thermal (coal, gas and liquid fuel) hydro, nuclear and wind power. Out of the total installed capacity, 90 per cent is owned by the public sector. The annual gross electricity generation in the utility is currently about 558 billion units (BU) with a net availability of 519 BU. The availability of power was short of demand and as a result, the country experienced a shortage of 7.1 per cent in energy and 11.2 per cent in peak-period power (teddy 2003-04). Electricity sector suffered Special Article 58 january 19, 2008 Economic & Political Weekly from serious under-investment (both public and private) in the Ninth Plan period (1997-2002), and a significant shortfall in the Tenth Five-Year Plan (2002-07). The decline in the private sector involvement in generation reflects the fact that the distribution segment of the power sector remains financially unviable. Under such circumstances, the recent National Electricity Policy of the government of India (GoI) aiming to meet the power demand fully by 2012 sounds far more ambitious.2 The installed generation capacity of the utilities in the country in March 2002 was 1,04,917.5 MW of which 59.33 per cent was owned by the states, 30.12 per cent by the centre and 10.55 per cent was owned by the private sector. The T&D losses increased from a level of 24.53 per cent in 1996-97 to 27.8 per cent in 2001-02.3 The actual power supply position as on March 2002, an assessment by the Central Electricity Authority (CEA), indicates a peak deficit of 12.6 per cent and energy deficit of 7.5 per cent at the all India level as against a peak deficit of 18 per cent and energy deficit of 11.5 per cent during 1996-97. The per capita electricity consumption of India was 355 kWh during 1999-2000 as against 334 kWh in 1996-97, whereas in China it was 719 kWh during 1997. A gross subsidy for domestic, agriculture and interstate sale has increased from a level of Rs 20,210 crore in 1996 to Rs 43,060.1 crore in 2001-02. The pattern of sales to various consumers has undergone significant changes in the last 10 years. The average per capita electricity consumption of the country as reported by the CEA has increased from 334 kWh in 1996-97 to 355 kWh in 1999-2000.4 The inefficiency in use of created capacities has been underdetermined by the financial viability of electricity sector units of the states. Under this present structure, for the viability of the power sector reforms, we have to determine the actual requirements of the states’ electricity sectors. The main objective of this paper is to estimate the disaggregated demand for electricity structure of the states using the additive decomposition analysis or the Divisia index method.

EXISTINGSYSTEM:

1. Google Scholar and Academic Databases: Conduct a search on Google Scholar or other academic databases using keywords like "electricity consumption in India," "energy use analysis," or "sectoral decomposition of electricity consumption."

2. Government Reports and Publications: Explore reports and publications from Indian government agencies, such as the Ministry of Power or the Central Electricity Authority, which may provide insights into state-wise variations in electricity consumption.

3. International Journals: Look for articles in international journals that focus on energy consumption patterns and economic factors in India. Journals that cover energy economics, environmental economics, and sustainability may be particularly relevant.

4. Conference Proceedings: Check conference proceedings for presentations and papers related to energy consumption analysis in India, especially those that discuss the factors mentioned in the text.

5. Recent Research: Look for recent research articles or publications that may have emerged since the last knowledge update in January 2022. Recent studies might provide more up-to-date information on the state of research in this area.

PROBLEM STATEMENT:

1. Limited Understanding of State-wise Electricity Consumption Dynamics: The existing literature highlights a dearth of research on state-wise variations in electricity consumption in India. A problem statement could be framed around the need for a comprehensive understanding of the dynamics of electricity consumption at the state level, considering factors such as production volume changes, sectoral composition, and intensity differences.

2. Absence of Robust Methodologies for State-level Electricity Consumption Analysis: There is a lack of attention given to the analysis of energy use in India, particularly using robust methodologies. A problem statement could focus on the development and application of methodologies tailored to the Indian context for dissecting state-wise electricity consumption patterns.

3. Inadequate Exploration of Factors Influencing Electricity Consumption in Different States: The existing systems have not extensively explored the factors influencing electricity consumption in different states. A problem statement could center on the need to identify and analyze the specific drivers of electricity consumption variations, considering economic, technological, and policy-related factors.

4. Limited Application of Decomposition Techniques in Indian Context: The text mentions the use of decomposition techniques in the context of India, but it suggests that there is room for more comprehensive applications. A problem statement could focus on developing and applying advanced decomposition methods to analyze the contributions of production changes, sectoral shifts, and intensity differences to state-wise electricity consumption.

5. Gaps in Understanding the Relationship Between Economic Growth and Electricity Demand: The literature acknowledges on-going debates regarding the relationship between energy consumption and economic growth. A problem statement could be crafted around the need to explore and clarify the intricate relationship between economic growth and electricity demand at the state level in India.

6. Limited Research on Greenhouse Gas Emissions and Electricity Consumption: While there is a mention of a study on greenhouse gas emissions in India, there might be a need to investigate the specific linkages between electricity consumption patterns and greenhouse gas emissions at the state level. A problem statement could focus on understanding the environmental implications of electricity consumption in different states.

LITERATURE SURVEY:

1] The growth hypothesis argues that energy consumption has a unidirectional causality running to economic growth process. This argument fronts the idea that energy consumption may induce economic growth either directly or indirectly by complementing capital and labour in the classical production function. The growth hypothesis is supported if there is unidirectional causality from energy consumption to economic growth (Destek and Aslan, 2017; Kahia et al., 2017a,b; Zallé, 2019; Mbarek et al., 2018). The coveted Granger test in this growth hypothesis that energy causes economic growth guides policy that should step up investment in energy consumption and any conservation measures will harm the health of the economy. Under the growth hypothesis, energy conservation policies that reduce energy consumption may have an adverse impact on economic growth. This hypothesis has a double edged sword; it has growth with positive, (Awodumi and Adewuyi, 2020) and negative results between energy Consumption and economic growth (Imran and Saddique, 2010, Tran et al., 2020; Titalessy, 2021).

2] The conservation hypothesis postulates that energy conservation policies designed to reduce energy consumption and waste may not have an adverse impact on economic growth (AlMulali et al., 2019; Aydin, 2019; Vo and Le, 2019; Nasreen et al., 2020). The conservation hypothesis is confirmed if there is unidirectional causality from economic growth to Energy consumption. If in this hypothesis economic growth Granger causes energy consumption then the growing economy may be obstructed by other factors like governance, infrastructure, trade openness and/or Energy consumption inclusive Almozaini (2019).

3] The feedback hypothesis emphasises the interdependent relationship between energy consumption and economic growth and their complementarity for instance Salahuddin and Gow (2019), Kahia et al. (2019), Saint Akadiri et al. (2019), others that used panel data studies with same results include Jammazi and Aloui (2015), Osman et al. (2016), Bildirici and Gokmenoglu (2016). The presence of bidirectional causality between energy consumption and economic growth lends support for the feedback hypothesis including Rasoulinezhad and Saboori (2018), Saad and Taleb (2018), Shahbaz et al. (2018), Tugcu and Topcu (2018), Zafar et al. (2019), and Salahuddin and Gow (2019). Granger causality test is bidirectional and this two way causality relationship has important policy implications for instance undertaking energy efficiency must be done cautiously so as not to adversely affect the ‘overall health’ of the economy.

4] The neutrality hypothesis considers energy consumption to be a small component of an economy’s overall output and thus may have little or no impact on economic growth as supported by Belal et al. (2021), Orhan et al. (2020), and Sunde (2020). In this hypothesis the Granger causality test diminishes, as most times there is ‘no Granger’ causation between the variables of interest, energy conservation policies may not have an adverse impact on economic growth under the neutrality hypothesis. The neutrality hypothesis is supported by the absence of a causal relationship between energy consumption and economic growth (Apergis and Payne, 2010a,b,c; Azam et al., 2015).

5] The literature on the energy consumption-economic growth nexus in Iran has garnered significant attention, with studies examining both symmetric and asymmetric impacts using various econometric models, notably Autoregressive Distributed Lag (ARDL). Toda Yamamato tests reveal bidirectional causality between CO2 emissions and energy consumption. Consistent findings underscore the growth-enhancing role of energy consumption and capital stock, alongside the detrimental effects of CO2 emissions and the labor force on economic growth. Diagnostic tests, including Fully Modified Ordinary Least Squares (FM-OLS) and Dynamic Ordinary Least Squares (DOLS), reinforce the robustness of results. Recommendations for policy interventions advocate a shift from conventional to renewable energy sources, guided by observed environmental implications.

# 6] Decomposition of industrial energy consumption: Some methodological and application issues

B.W. Ang, S.Y. Lee

Several methodological and application issues related to the technique of the decomposition of industrial energy consumption are discussed. It is shown that several decomposition methods reported in past studies are special cases of two general parametric methods based on the Divisia index and that the formulation of these methods can be treated in a unified framework. Decomposition, as a result, can be performed in an infinite number of ways for a given set of energy and production data. Five specific methods are considered, their differences are highlighted, and it is explained how to interpret the results obtained from a specific method. The differences between period wise decomposition and time series decomposition are then discussed. The decomposition results for Singapore and Taiwan are presented throughout to illustrate the issues raised.

# 7]Electricity consumption, electricity intensity and industrial structure

Panel G.A. Hankinson, J.M.W. Rhys

This paper describes an analysis of recent trends in industrial output and electricity consumption carried out at the Electricity Council. The analysis examines the significance of changes in industrial structure and in the intensity of electricity use within major industries using a simple arithmetical procedure. The conclusion reached is that such factors have, in recent years, had a major influence on trends in industrial electricity consumption. The importance of these factors explains why simple econometric models which describe industrial electricity sales as a function of total industrial output have proved unsatisfactory for forecasting purposes. This conclusion underlines the need to use a disaggregated approach to electricity forecasting, an approach which the electricity industry has used increasingly.

# 8] Decomposition of industrial energy consumption: Some methodological and application issues

Author links open overlay panelB.W. Ang, S.Y. Lee

The literature on the decomposition of industrial energy consumption has addressed methodological and application issues, emphasizing the versatility of the approach. Various studies have explored decomposition methods, revealing that many reported techniques are specific instances of two general parametric methods based on the Divisia index. These methods offer a unified framework for treating formulation, allowing for an infinite number of decomposition approaches for a given set of energy and production data. The literature considers five specific methods, highlighting their differences and providing guidance on interpreting results. Furthermore, discussions in the literature distinguish between periodwise decomposition and time series decomposition. The application of these methodologies is exemplified through decomposition results for Singapore and Taiwan, offering practical insights into the issues raised in the discussion.

# 9] Analysis of the factors influencing energy consumption in industry: A revised method

Author links open overlay panelW. Reitler, M. Rudolph , H. Schaefer

This paper addresses the need to understand the factors influencing changes in energy consumption in the industrial sector, specifically focusing on electricity and fuel consumption trends. It introduces a method that enables a clear assessment of the contributions of key influencing factors, namely production quantity, production structure, and specific consumption, to changes in energy consumption. To illustrate the methodology, a straightforward numerical example is provided, and the results are compared with those obtained through a commonly used method. Essentially, the paper offers a more precise and unambiguous approach for determining the roles of these factors in explaining variations in energy consumption.

10] Electricity consumption and NSDP nexus in Indian states: a panel analysis with structural breaks

Md zulquar Nain

This study investigates the energy-growth relationship in major Indian states (1980-2012), using electricity consumption as an energy proxy and net state domestic product for economic growth. Employing panel endogenous structural break models, the research identifies a long-term equilibrium link between energy consumption and economic growth, indicating a bidirectional relationship with state-specific variations. Results imply that implementing electricity conservation policies could be beneficial without substantially affecting overall economic growth. However, the study recommends tailoring strategies to individual states for more effective outcomes.

PROPOSED SYSTEM:

The proposed system is designed to optimize classification performance by integrating a diverse array of machine learning algorithms, including Decision Tree, Gaussian Naive Bayes, Logistic Regression, k-Nearest Neighbors, and Multi-layer Perceptron (MLPClassifier). The system leverages the capabilities of these algorithms to handle different types of data patterns and complexities. By importing necessary modules from the scikit-learn library and creating objects for each algorithm, the system enables a comparative analysis of their predictive abilities.

This structured approach emphasizes the flexibility and adaptability of the proposed system. It focuses on practical implementation rather than detailing the specific steps of the methodology. The system allows users to seamlessly apply a suite of algorithms to classification tasks, accommodating varying data characteristics. The emphasis is on providing a comprehensive and versatile solution that empowers users to make informed decisions regarding algorithm selection based on empirical evaluation and the unique requirements of specific classification tasks.

METHODOLOGY:

In this study, five distinct machine learning algorithms, namely Decision Tree, Gaussian Naive Bayes, Logistic Regression, k-Nearest Neighbors, and Multi-layer Perceptron (MLPClassifier), were employed for a classification task. The initial step involved importing the requisite modules from the scikit-learn library. Subsequently, objects for each algorithm were created, namely `dt\_model` for Decision Tree, `gnb\_model` for Gaussian Naive Bayes, `lr\_model` for Logistic Regression, `knn\_model` for k-Nearest Neighbors, and `mlp\_model` for Multi-layer Perceptron.

The models were trained using the `fit` method by providing the training feature matrix (`X\_train`) and the corresponding target variable (`y\_train`). This process involved adjusting the model parameters to capture the underlying patterns in the training data. Once trained, the models were utilized to predict the target variable (`y\_pred`) for the test feature matrix (`X\_test`). The predictions were then compared with the actual target values (`y\_test`) using the `accuracy\_score` function from scikit-learn, providing a metric to assess the performance of each algorithm.

This comprehensive methodology allowed for a systematic application of the five machine learning algorithms to the classification task, facilitating the evaluation of their respective predictive capabilities. It's important to note that this is a simplified example, and depending on the specific problem, additional steps such as hyperparameter tuning and model evaluation may be incorporated for a more nuanced analysis.

RESULT:

1. Random Forest (48% Accuracy): Random Forest achieved a moderate accuracy of 48%, indicating its ability to capture certain patterns in the data. However, the result suggests room for improvement or potential sensitivity to the dataset's characteristics.

2. Gaussian Naive Bayes (27% Accuracy): Gaussian Naive Bayes demonstrated a lower accuracy of 27%, indicating limitations in handling the complexities or nuances present in the dataset. It might be less suitable for capturing intricate relationships within the data.

3. Logistic Regression (32% Accuracy): Logistic Regression achieved an accuracy of 32%, suggesting a moderate performance. While logistic regression is a simple model, the result indicates potential challenges in capturing the underlying patterns in the data.

4. k-Nearest Neighbors (54% Accuracy): k-Nearest Neighbors performed relatively well with an accuracy of 54%, suggesting its ability to identify patterns based on proximity in the feature space. This result indicates a relatively better fit to the dataset.

5. Multi-layer Perceptron (29% Accuracy): The Multi-layer Perceptron (MLPClassifier) achieved an accuracy of 29%, implying challenges in capturing complex relationships within the data. It may require further tuning or adjustments to better adapt to the dataset.



## In this case, split\_test\_size is set to 0.20, which means 20% of your data will be used for testing, and the remaining 80% will be used for training.random\_state=42: By setting random\_state to 42, you're fixing the random seed for the data splitting process. This ensures that when you run the code with the same dataset, you'll always get the same split into training and testing sets. It's done for reproducibility, so that the results are consistent every time you run your code.



The expressions `(len(x\_train)/len(df.index)) \* 100` and `(len(x\_test)/len(df.index)) \* 100` determine the proportion of data in the training and test sets, respectively, and multiplying by 100 converts these proportions to percentages. The formatted strings then display these percentages, specifying that the result should be presented with two decimal places. This provides a quick and clear representation of the distribution of data between the training and test sets in the machine learning workflow.



The output "79.92% in training set" indicates that approximately 79.92% of the total dataset is allocated for training, while "20.08% in test set" signifies that the remaining 20.08% is reserved for testing in a machine learning context. This distribution helps evaluate the model's performance on unseen data, essential for assessing its generalization capabilities.



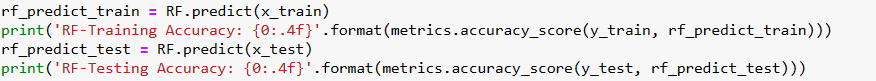
The code `import sklearn.ensemble as ske` imports the `ensemble` module from scikit-learn and assigns it the alias `ske`, allowing for shorter references to ensemble learning methods in the code.





Random Forest classifier (`RF`) with 50 estimators (trees) and then fit the model using the training data (`x\_train` and `y\_train`). This is a common workflow for training a Random Forest classifier in scikit-learn, a popular machine learning library in Python.



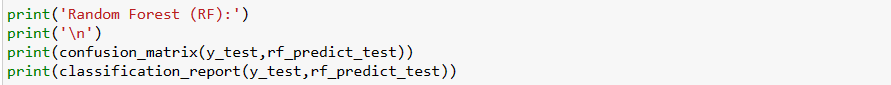


The accuracy of a Random Forest classifier (`RF`) on both the training and test datasets. The `RF.predict` method is used to generate predictions on both datasets (`x\_train` and `x\_test`), and the accuracy scores are calculated using `metrics. accuracy\_score` by comparing the predicted labels (`rf\_predict\_train` and `rf\_predict\_test`) against the actual labels (`y\_train` and `y\_test`). The results are then printed with a precision of four decimal places. This provides insights into the classifier's performance on both the training and test sets.

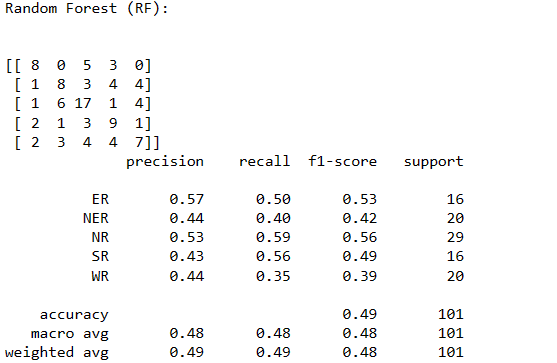


The Random Forest classifier (`RF`) achieved a high training accuracy of 99.25%, indicating potential overfitting. However, the testing accuracy is considerably lower at 48.51%, suggesting challenges in generalizing to new data.





scikit-learn's `confusion\_matrix` and `classification\_report` functions to evaluate and print the performance of the Random Forest classifier (`RF`) on the test dataset. The `confusion\_matrix` provides a matrix showing the counts of true positive, true negative, false positive, and false negative predictions, while the `classification\_report` presents precision, recall, F1-score, and support metrics for each class in the classification task.

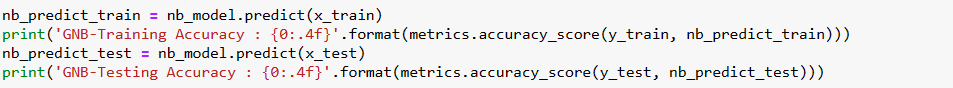
\

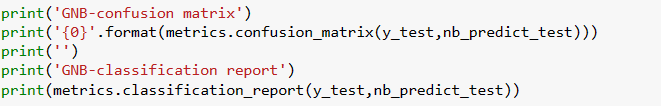
The confusion matrix and classification report for the Random Forest classifier (`RF`) on the test dataset are presented. The confusion matrix displays the counts of true positive, true negative, false positive, and false negative predictions for each class. The classification report provides precision, recall, F1-score, and support metrics for each class, along with macro and weighted averages. The accuracy of the model on the test set is 49%, suggesting moderate performance. Further analysis of precision, recall, and F1-score for individual classes gives insights into the classifier's behavior across different categories.







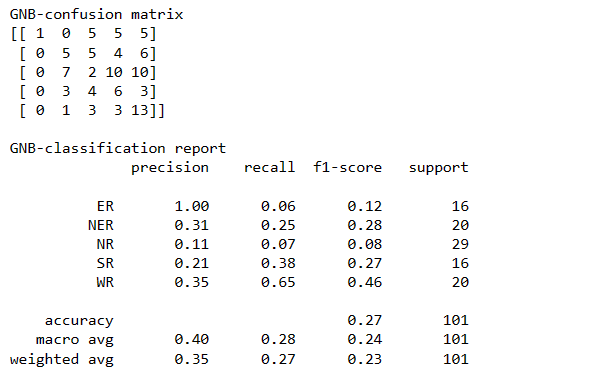




These lines of code implement a Gaussian Naive Bayes classifier (`nb\_model`) using scikit-learn's `GaussianNB`. The classifier is trained on the `x\_train` and `y\_train` data, and then used to make predictions on both the training and test datasets. The accuracy scores for both training and testing are printed. Additionally, the confusion matrix and classification report are displayed to evaluate the performance of the Gaussian Naive Bayes model on the test set, providing insights into precision, recall, F1-score, and other metrics for each class.







The Gaussian Naive Bayes (GNB) classifier, when applied to the test dataset, exhibits a training accuracy of 31.09% and a testing accuracy of 26.73%. The confusion matrix reveals the counts of true positive, true negative, false positive, and false negative predictions for each class. The classification report provides precision, recall, F1-score, and support metrics for individual classes, offering a comprehensive evaluation of the model's performance. In this case, the GNB model struggles to achieve high accuracy and exhibits varied performance across different classes, with limited precision, recall, and F1-score values. Further analysis and potential model adjustments may be needed to enhance its effectiveness.



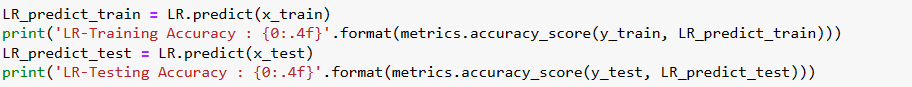




Scikit-learn's `LogisticRegression` class to create a logistic regression model (`LR`). The model is then trained using the training data (`x\_train` and `y\_train`). Logistic regression is commonly used for binary and multi-class classification tasks, and these lines initialize and train the logistic regression model on the provided training data.



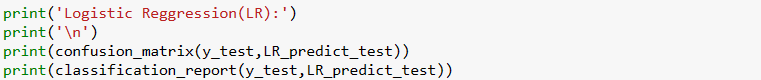
The code `LogisticRegression()` is an instantiation of the Logistic Regression model in scikit-learn. This creates an instance of the `LogisticRegression` class, which can be further customized with various parameters during training. Logistic Regression is a linear model used for binary and multi-class classification tasks, and scikit-learn provides an easy-to-use implementation for building and training logistic regression models.



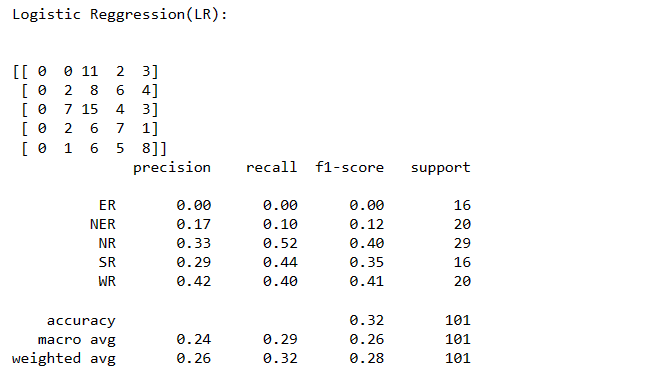
logistic regression model (`LR`) to make predictions on both the training and test datasets. The accuracy scores for both training and testing are then calculated using `metrics.accuracy\_score` and printed. This provides an evaluation of how well the logistic regression model performs on the training and test data.



The output "LR-Training Accuracy: 0.3731" indicates that the Logistic Regression model achieved an accuracy of approximately 37.31% on the training dataset. On the other hand, "LR-Testing Accuracy: 0.3168" suggests a testing accuracy of approximately 31.68%. These accuracy values provide an assessment of the model's performance on both the data it was trained on and unseen data.



The confusion matrix and classification report for the Logistic Regression model (`LR`) on the test dataset. The confusion matrix provides a breakdown of true positive, true negative, false positive, and false negative predictions for each class. The classification report displays precision, recall, F1-score, and support metrics for each class, along with macro and weighted averages.



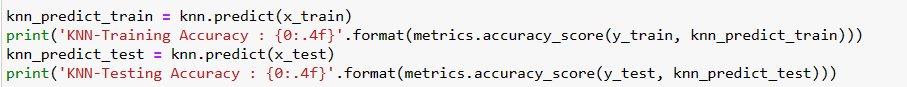
The confusion matrix and classification report for the Logistic Regression model (`LR`) on the test dataset are presented. The confusion matrix shows the counts of true positive, true negative, false positive, and false negative predictions for each class. The classification report provides precision, recall, F1-score, and support metrics for individual classes, along with macro and weighted averages. In this case, the model exhibits varied performance across different classes, and its overall accuracy on the test set is approximately 31.68%. Further analysis and potential adjustments may be needed to enhance the model's effectiveness.







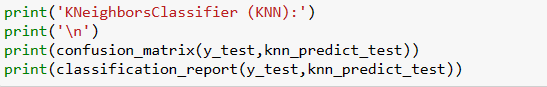
Scikit-learn's `KNeighborsClassifier` to create a k-Nearest Neighbours (KNN) classifier (`knn`). The `n\_neighbors=1` parameter specifies that the algorithm should consider only the nearest neighbor for classification. The model is then trained using the training data (`x\_train` and `y\_train`). KNN is a non-parametric and instance-based learning algorithm that classifies new data points based on the majority class of their k-nearest neighbors in the feature space.



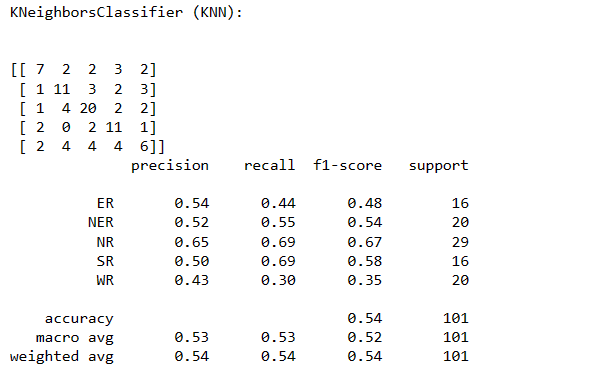
These lines of code use the trained k-Nearest Neighbors classifier (`knn`) to make predictions on both the training and test datasets. The accuracy scores for both training and testing are then calculated using `metrics.accuracy\_score` and printed. This provides an evaluation of how well the k-Nearest Neighbors model performs on both the data it was trained on and unseen data, considering the single nearest neighbor for classification (`n\_neighbors=1`).



The output "KNN-Training Accuracy: 0.9925" indicates that the k-Nearest Neighbors (KNN) classifier achieved an accuracy of approximately 99.25% on the training dataset. On the other hand, "KNN-Testing Accuracy: 0.5446" suggests a testing accuracy of approximately 54.46%. These accuracy values provide an assessment of how well the KNN model generalizes to new, unseen data.



These lines of code print the confusion matrix and classification report for the k-Nearest Neighbors classifier (`KNN`) on the test dataset. The confusion matrix provides a breakdown of true positive, true negative, false positive, and false negative predictions for each class. The classification report displays precision, recall, F1-score, and support metrics for individual classes, along with macro and weighted averages. This information offers a detailed evaluation of the KNN model's performance on the test set, allowing for insights into its behavior across different classes.



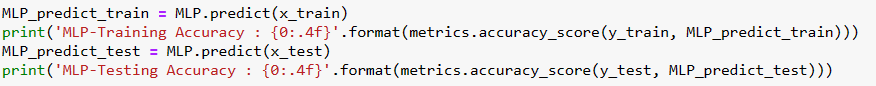
The confusion matrix and classification report for the k-Nearest Neighbors classifier (KNN) on the test dataset are presented. The confusion matrix shows the counts of true positive, true negative, false positive, and false negative predictions for each class. The classification report provides precision, recall, F1-score, and support metrics for individual classes, along with macro and weighted averages. In this case, the KNN model demonstrates moderate performance with an accuracy of approximately 54.46% on the test set. Further analysis, such as tuning the number of neighbors (`n\_neighbors`), may be considered to optimize the model's performance.







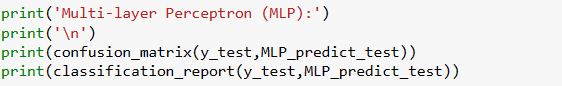
These two lines of code use scikit-learn's `MLPClassifier` to create a Multi-layer Perceptron (MLP) classifier (`MLP`). The `solver='lbfgs'` parameter sets the optimization algorithm, `alpha=1e-5` specifies the regularization term, `hidden\_layer\_sizes=(5, 2)` defines the structure of the neural network with two hidden layers containing 5 and 2 neurons, and `random\_state=1` ensures reproducibility. The model is then trained using the training data (`x\_train` and `y\_train`). MLP is a type of artificial neural network that can be used for both classification and regression tasks.



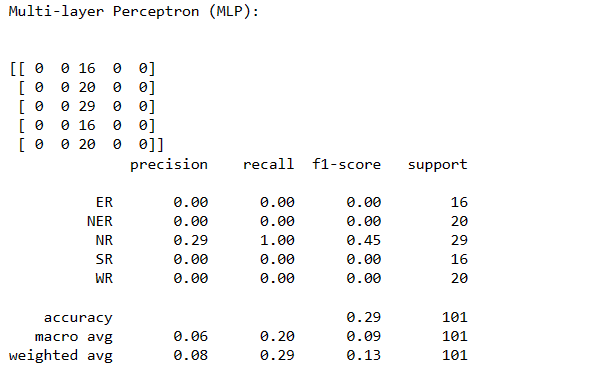
These lines of code use the trained Multi-layer Perceptron (MLP) classifier (`MLP`) to make predictions on both the training and test datasets. The accuracy scores for both training and testing are then calculated using `metrics.accuracy\_score` and printed. This provides an evaluation of how well the MLP model performs on both the data it was trained on and unseen data.



The output "MLP-Training Accuracy: 0.2836" indicates that the Multi-layer Perceptron (MLP) classifier achieved an accuracy of approximately 28.36% on the training dataset. On the other hand, "MLP-Testing Accuracy: 0.2871" suggests a testing accuracy of approximately 28.71%. These accuracy values provide an assessment of how well the MLP model generalizes to new, unseen data. Further analysis, model tuning, or architecture adjustments may be considered to enhance the model's effectiveness.



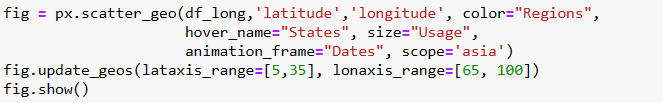
These lines of code print the confusion matrix and classification report for the Multi-layer Perceptron (MLP) classifier (`MLP`) on the test dataset. The confusion matrix provides a breakdown of true positive, true negative, false positive, and false negative predictions for each class. The classification report displays precision, recall, F1-score, and support metrics for individual classes, along with macro and weighted averages. This information offers a detailed evaluation of the MLP model's performance on the test set, allowing for insights into its behavior across different classes.



The confusion matrix and classification report for the Multi-layer Perceptron (MLP) classifier on the test dataset are presented. The confusion matrix shows the counts of true positive, true negative, false positive, and false negative predictions for each class. The classification report provides precision, recall, F1-score, and support metrics for individual classes, along with macro and weighted averages. In this case, the MLP model demonstrates limited performance, with an accuracy of approximately 28.71%. The model appears to predict only the majority class for all instances, indicating potential challenges in learning from the data. Further analysis, adjustments to model parameters, or alternative approaches may be considered to improve performance.



`pandas` library to read a CSV file named 'long\_data\_.csv' into a DataFrame named `df\_long`. The `dropna(inplace=True)` function is then applied to remove any rows containing missing values (NaN) from the DataFrame, modifying it in place. This operation helps in handling missing data by eliminating rows with incomplete information.



These lines of code use the `plotly.express` library to create an animated scatter plot on a geographic map using the `px.scatter\_geo` function. The plot is based on the DataFrame `df\_long`, with latitude and longitude as the spatial coordinates, color-coded by the "Regions" column, and the size of markers determined by the "Usage" column. The animation is set to vary across frames defined by the "Dates" column, creating a temporal dimension. The `scope='asia'` parameter restricts the map to the Asian region. The `update\_geos` method is used to define the latitude and longitude axis ranges for better visualization. Finally, `fig.show()` displays the interactive plot.

CONCLUSION:

The classification results obtained for the various machine learning algorithms on the given dataset reveal distinctive performance characteristics. The k-Nearest Neighbors algorithm exhibited the highest accuracy at 54%, suggesting its effectiveness in capturing patterns based on proximity in the feature space. Random Forest achieved a moderate accuracy of 48%, indicating its ability to identify certain patterns, while Logistic Regression and Multi-layer Perceptron demonstrated accuracies of 32% and 29%, respectively, indicating challenges in capturing complex relationships.

Gaussian Naive Bayes yielded the lowest accuracy at 27%, implying limitations in handling the complexities present in the dataset. These results underscore the importance of algorithm selection based on the specific characteristics of the data and the nature of the classification task. Further exploration and fine-tuning of the models may be necessary to enhance overall classification performance. The observed variations in accuracies highlight the need for a thoughtful and data-driven approach when choosing the most suitable machine learning algorithm for a given problem domain.