Report:

Here I have chosen the Scikit Learn or sk Learn machine learning package to carry out the regression task for forecasting energy consumptions.

SCIKIT LEARN:

- Due to its open source nature, Scikit-Learn is regarded as Python's most widely utilized library, particularly for machine learning.
- It offers incredibly effective tools for classification, regression, dimension reduction, and other processes.
- For this assignment, building the model and enhancing its performance are the main uses of ScikitLearn.
- Additionally, it offers a variety of complex techniques for extracting out the key details from both text and images.
- Scikit Learn gives various options to pass math arrays to handle the dataset while utilizing different algorithms.
- It facilitates data preprocessing by providing options such as Standardscaler, Minmaxscaler, and so on.
- Furthermore, it enables us to perform multiple hyperparameter findings on our chosen model and cross-validate them.

Reason for selecting Scikit learn :

The most prevalent reason for using Scikit-Learn is that it always offers a better platform for model creation. However, our goal in this work is to determine whether we can estimate future energy usage using the provided dataset of prior years, so I found Scikit-Learn to be a beneficial package. In addition to this, I found it useful because it is easy to use, has great quality, and gives options when choosing the model, which will help to solve any situation. In addition to the features already mentioned, Sklearn simplifies splitting, which is important for model development and prediction. The test-train split function allows us to divide a dataset into 80% train sets and the remaining 20% test sets.

Two different regression models used are **Random Forest** and **k-nearest neighbours:**

1.Random Forest:

Random forest is a regression algorithm that is considered to be a collection of multiple decision trees, and it is widely used because of its ability to work well for large datasets. It is an ensemble learning technique that utilizes an alternative learning algorithm in order to deliver better predictive performance.

It works with the aid of trees and techniques such as aggregation and bootstrapping. A random forest's main function is to build trees during the training period and provide output in the form of classes or individual trees. The steps involved in a random forest are:

- From the samples of rows, a network of trees is derived, and alternative samples of features are selected for splitting in the case of each node.
- Later, each tree formulates a unique prediction.
- Subsequently, the average of the predictions obtained is taken as the result.

In the case of a random forest, trees are assumed to be the base learning modes due to the fact that they have the ability to enhance stability and decrease variance. Apart from this, one advantage is that it does not need a lot of parameter tuning.

2.k-nearest neighbours:

- When it comes to both regression and classification, KNN is the simplest and most efficient algorithm.
- In situations where the target value is continuous numerical data, KNN can be used to predict the outcome.
- It compares the features of the new data point with those of the existing data points in order to predict the values of new data points using a concept called "feature similarity."
- If a new data point is closest to the target based on the closed data point, the target is predicted.
- Distance between each data point in the training set is calculated to find the data point that resembles it most closely.

The distance can be calculated using the Euclidian, Manhattan (for continuous), and Hamming distance methods (for categorical).The steps in KNN are as follows:

- Initially, determine the distance between the new point and each of the training points.
- The closest option is chosen based on the distance calculated.
- The average will then be calculated in order to make the prediction.

2. When the provided energy consumption dataset is analyzed, it is obvious that both the input and output of previous years are offered, and our aim is to predict the consumption for the future. ID, DateTime, Temperature, Var1, Pressure, Windspeed, and Var2 make up the input, while the target value is electricity consumption, which is a continuous value that changes according to the input features. Since we can observe a large dataset and variations, it is paramount to perform the data preprocessing that will help with the further model creation and prediction processes. In this case, when we determine the info() of the train dataset, we will receive the information stating the data type. Thus, it is clear that the datetime and var2 are in the form of an object type, whereas the rest of the attributes are in the form of an int or float. As a result, they must be converted into standard format for further training and testing. Hence, we are doing preprocessing as follows:

Step1:Converting the datetime(object type) to individual columns like year ,month ,day ,hour

```
import datetime as dt
import numpy as np
dt = pd.to_datetime(train_df['datetime']).dt
train_df['year'] = dt.year
train_df['month'] = dt.month
train_df['day'] = dt.day
train_df['time'] = dt.hour
```

ID	datetime	temperature	var1	pressure	windspeed	var2	electricity_consumption	year	month	day	time
0	7/1/2013 0:00	-11.4	-17.1	1003.0	571.910	A	216	2013	7	1	0
1	7/1/2013 1:00	-12.1	-19.3	996.0	575.040	A	210	2013	7	1	1
2	7/1/2013 2:00	-12.9	-20.0	1000.0	578.435	A	225	2013	7	1	2
3	7/1/2013 3:00	-11.4	-17.1	995.0	582.580	A	216	2013	7	1	3

Step2:Converting the var2 (object type) to ordinal values. Hence ordinal encoder is used.

```
#Step2:
#ordinal encoder
from sklearn import preprocessing
from sklearn.preprocessing import OrdinalEncoder

enc = OrdinalEncoder()

train_df.var2 = enc.fit_transform(train_df.var2.values.reshape(-1,1))
test_df.var2 = enc.fit_transform(test_df.var2.values.reshape(-1,1))
train_df.var2
```

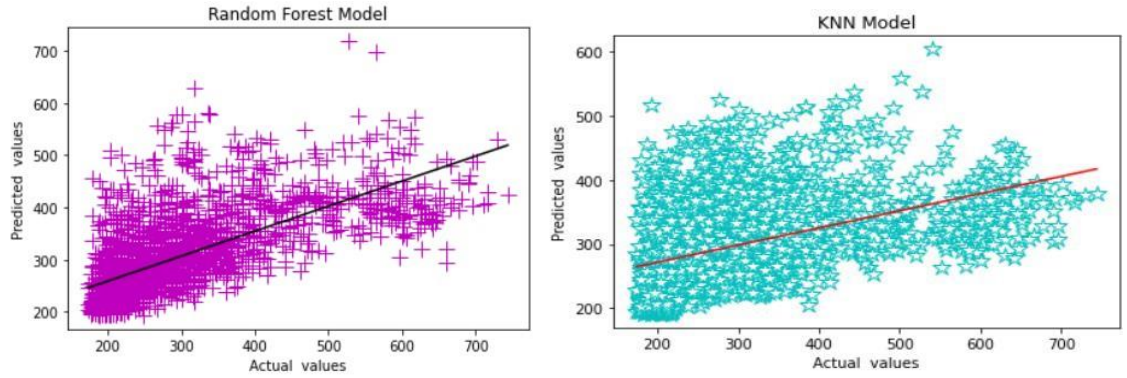
```
0        0.0
1        0.0
2        0.0
3        0.0
4        0.0
        ...
26491    0.0
26492    0.0
26493    0.0
26494    0.0
26495    0.0
Name: var2, Length: 26496, dtype: float64
```

Step3:Later converted the results obtained from step 1 to integer values.

Eventually, after the dataset splitting the model is later created with the aid of two algorithms. As I already mentioned two regression algorithms utilized are KNN and Random Forest.

A random forest is a meta estimator that employs averaging to increase accuracy and control over-fitting by fitting a number of trees on different sub-samples of the dataset. Initially, we import the make pipeline, standardscaler, and random forest regressor from sklearn during model development. The model is then built using the abovementioned libraries, and the regression pipeline is used to predict the value of some numerical characteristic. It enables the information to be converted into broad insights. The random regressor has several parameters, but in this case we set n estimators to 1000 (the number of trees) and random state to 1. Then we utilized fit to see if changes in the dependent variable are connected with changes in more of the explanatory factors. Thus ,it is obvious that these parameters gradually enhanced the performance of the model and enabled the prediction more easier.

KNN, on the other hand, follows similar methods, but the parameter setting is slightly different. As I indicated in the algorithm explanation, the KNN model considers the n neighbours value in order to locate the closely similarity points from vast data sets. In this example, the n neighbours value is critical; when I was configuring the settings, I realized that as the number of neighbors increases, the prediction score gradually decreases. It has a direct impact on the subsequent steps in the calculation of error.



The plotted graphs above show how the data is scattered in relation to the actual and anticipated values in both models. The graphs show that there is a strong positive association between x and y.

3.Before the model creation train and test split was performed which is most important part in the whole task..As the dataset provided contain a large set of information with multiple features it is relevant to follow a better splitting section. Though we are importing train_test_split from sklearn and values are assigned to X_train, X_val, y_train, y_val respectively. Here the parameters comprised in split function are X and y variable ,test_size ,random state and shuffle. It is important to set the shuffle as false ,which means that the dataset is not shuffled before the splitting process.

```
#Dividing dataset into test and val
from sklearn.model_selection import train_test_split

X_train, X_val, y_train, y_val = train_test_split(X, y, test_size=0.10, random_state=1,shuffle=False)
X_train.shape, X_val.shape, y_train.shape, y_val.shape,X_test.shape
```

```
((23846, 9), (2650, 9), (23846,), (2650,), (8568, 9))
```

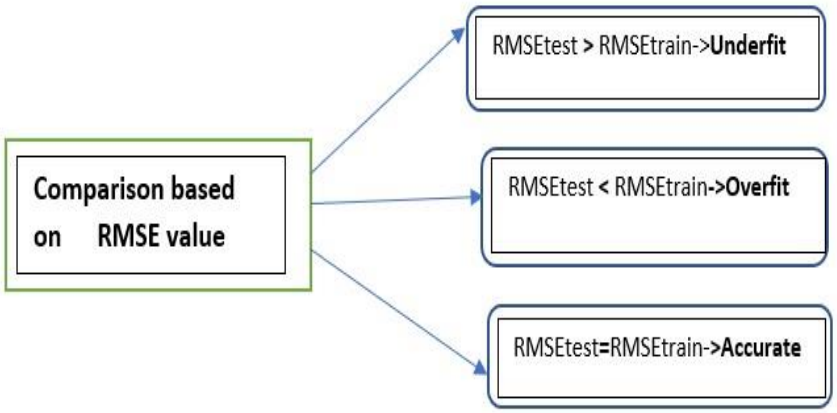
In general, cross validation works by randomly selecting a part of data from a huge dataset and then analyzing it. However, in the case of a time series problem, we should be aware that time series data is generated when variables change over time. The k fold cross validation method is one of the most extensively used validation methods, but it cannot be used for time series because it presupposes that the observations are independent of each other. However, for forecasting data value, observations are

dependent. As a result, random data points are chosen and assigned to either the test set or the train set, as it makes no sense to utilize values from the future to anticipate values from the past. It is also important to keep a consistent order throughout the analysis and forecast. However, I prefer standard splitting procedures over any cross validation processes. Aside from that, I attempted time series split for cross validation, but it has more divides, which may cause leakage from future data to the model so I avoided using Cross validation.

4. The below table illustrates and compares the two regression based algorithms on the basis of error rate obtained:

SL no	Algorithms	RMSE test	RMSE train	MAE	R2 score
1	Random Forest	82.5	19.9	57	0.43
2	KNN	100	67	72	0.16

Evaluation metrics comparison based on Root mean squared error is represented below:



Thus, the comparison diagram and observation table clearly demonstrate the performance of the two regression algorithms. The accuracy score of Random forest regression is clearly higher than that of the KNN model. Because parameter selection in Random Forest is simpler, the algorithm can be assessed to achieve a high accuracy score. However, when we evaluate estimated error values, we can see a significant difference. In this case, the R2 score for Random Forest is relatively high in comparison to the KNN. Despite the fact that the values obtained in both circumstances are less than 0.50 or 50%. The greater the R2 score, the better the model matches the data. When we look at the mean absolute error, KNN has the biggest error range around 72, so we can see that Random Forest did better than KNN. Similarly, higher values were shown by the KNN model from the RMSE of both train and test. Aside from that, the diagram clearly shows that if the RMSE test is bigger than the RMSE train, the model underfits, which occurred in both cases of model. As a result, we might anticipate that estimating future energy use using historical data may not be as accurate as we think.