University of Victoria
Faculty of Engineering
Spring 2023 Final Report

USING MACHINE LEARNING TO PREDICT ENERGY CONSUMPTION IN RESIDENTIAL BUILDINGS

University of Victoria
Faculty of Engineering
Victoria, British Columbia

Caleb Uzodinma
V00900721
ENGR 446
Electrical and computer engineering
Calebuzodinma@gmail.com

April 3, 2023

In partial fulfillment of the requirements of the B.Eng. Degree

Ash Senini
Engr 446 Instructor
Faculty of Engineering
University of Victoria
3800 Finnerty Road
Victoria, BC, V8P 5C2
Canada.

April 3rd, 2023.

Re: “Using Machine learning to show energy consumption forecasts in residential buildings”.

Dear Ash,

As a fourth year electrical engineering student, I am pleased to submit the accompanying final report titled as “Using machine learning to show energy consumption forecasts in residential buildings”. This report is written for the completion of the ENGR 446 in spring 2023. Since sustainability has become a global issue, it can be determined that conservation of energy is a reliable way to sustain energy. Moreover, to conserve energy we must first look at how we consume energy. Hence, the ability to compare the projected energy consumption and actual energy consumption to conserve energy would be essential to saving energy.
Following are brief bullets that capture the tasks were completed throughout this term

- Extensively researched machine learning algorithms; various scholarly articles provided valuable information when implementing machine learning algorithms such as: the type of data needed, programs and frameworks used to implement the models.
- Analyzed datasets provided for energy consumption in residential buildings.
- Implemented machine learning algorithms - “Linear regression, support vector Regression and polynomial regression”.
- Validated each regression model and carried out a performance test.

The major part of the project was analyzing and developing different machine learning algorithms. That involved a lot of research and planning to come up with an innovative idea to combine different ideologies. I believe that the knowledge gained from the class will help me in my future endeavors.

Sincerely,

Caleb Uzodinma.

Table of Contents

List of figures 4

Glossary 5

1 – Abstract 6

2 – Introduction 7

3 – Literature review 8

- **3.1 – Method 1: Linear Regression** 8
- **3.2 – Method 2: Support Vector Regression** 8
- **3.3 – Method 3: Polynomial Regression** 9
- **3.4 – Model Validation** 10
  - **3.4.1 Repeated Random test-train splits** 10
  - **3.4.2 Performance test** 11

4 – Implementation 11

- **4.1–Dataset collection and Data Preprocessing** 11
- **4.2– Linear Regression model** 15
  - **4.2.1– Model Evaluation** 15
- **4.3– Support Vector Regression model** 16
  - **4.3.1– Model Evaluation** 17
- **4.4–Polynomial Regression model** 18
  - **4.4.1– Model Evaluation** 18

5– Conclusion and Recommendations 19

6 – References 20

List of figures

Figure 1: Support vector regression underlying optimization model. 9
Figure 2: The difference between linear and polynomial regression. 10
Figure 3: Imported dataset. 11
Figure 4: Data dropped. 12
Figure 5: Energy type and location filtered. 12
Figure 6: The dependent and independent variables selected. 12
Figure 7: More independent variables are selected. 13
Figure 8: Independent variables joined. 13
Figure 9: The dependent variable. 14
Figure 10: Prediction model for linear regression. 15
Figure 11: The evaluated model for linear regression. 15
Figure 12: Scaled values for support vector regression model. 16
Figure 13: Prediction model for support vector regression model. 17
Figure 14: The evaluated model for support vector regression. 17
Figure 15: The predicted response for polynomial regression. 18
Figure 16: The evaluated model for polynomial regression. 19

Glossary

IEA International Energy Agency.

Pandas It is a software library written for the python programming language for data manipulation and analysis.

Scikit-learn library It is a machine learning library in python used for various classification and regression techniques.

MAE Mean absolute error, this is the magnitude of the difference between the prediction of an observation and the true value of that observation.

MSE Mean squared error, this is the difference between the model prediction and the ground truth, square it and average it across the whole dataset.

R Score It is a statistical measure in a regression model that determines the proportion of variance in the dependent variable that can be explained by the independent variables.

Numpy Is a library for the python programming language, adding support for large, multi-dimensional arrays and matrices.

1 – Abstract

This report discusses the challenges of predicting energy consumption in residential buildings in order to promote greater energy efficiency and conservation. For most parts, sustainability has become the way to go in the energy sector. There have been many sustainable initiatives such as electrifying public transportation and building electric vehicle infrastructure. For this report, predicting energy consumption in a bid to conserve energy and bring greater energy efficiency will be challenges that will be discussed.

This study used a dataset from the government of Canada, which contained energy consumption data of households in various provinces of Canada. The dataset for this project used electricity as its main energy source and residential buildings as its case study for infrastructure. Other energy sources, such as oil, gas, and solar, were not used as adding them would be too broad to quantify and would need a considerable amount of time to analyze. This was done to narrow down the datasets and to avoid any sort of complexity. Linear regression, Support vector regression, and polynomial regression were the machine learning methods used to predict the energy consumption forecast.

The methodology of the project followed a sequence of steps: data collection, data preprocessing, algorithm selection, and model evaluation. Model validation techniques, such as train-test split and performance evaluation, were employed to ensure that the model can perform well with new data. The limitations of the study include the absence of weather data and the use of deep learning to reduce bias. Overall, at least two models were found to be predictive and helped conserve energy.

2 – Introduction

Nowadays, the climate of the environment has become a significant concern to a large population. Finding ways to be sustainable is one of the biggest issues that plague our society today. Energy conservation is a step in the right direction towards addressing the climate crisis. The quest for greater energy efficiency has united people around the globe over the past year, driven by the concurrent challenges of the energy and climate crises [1]. There is now a sense of eagerness and urgency to save electricity.

The International Energy Agency (IEA) notes that since 2020, around $1 trillion has been mobilized to boost energy efficiency, covering initiatives as varied as making homes and commercial buildings more energy efficient, electrifying public transportation and building electric vehicle infrastructure [1]. One of the modern initiatives that can help save energy is using machine learning algorithms to forecast the energy consumption in residential buildings. Using machine learning algorithms is a modern approach that helps show the difference between the projected energy use and the actual energy consumption.

This report provides an analysis on the energy consumption in Residential areas. Most homes tend to waste energy, this can lead to pollution if not addressed. Therefore, homeowners would need to resolve the waste of energy. An energy consumption forecast will be able to resolve this issue. Homeowners can use the differences between the projected consumption and actual energy to identify areas where they can conserve energy in the building.

This report will show how three machine learning algorithms will predict energy consumption in residential buildings. Each algorithm will be explained to give an understanding to how each one works. Afterwards, each of these algorithms will be implemented to show how to get the results .For the implementation, python will be used to implement each algorithm. Then pandas and sklearn will be the frameworks needed to analyze the datasets provided. Finally, a model validation will be done by splitting test and training data to show which algorithm has the least deviation and error [2].

3 – Literature review

Using machine learning techniques in energy consumption is an innovative way to conserve energy for residential areas. Therefore, this literature review would mainly concern studies and results from scholar articles and open-source software platforms with datasets from residential buildings and the implementation scenario. During the research, three machine learning methods were found to be able to predict an energy consumption forecast. Next, each model will be implemented and validated to assess its effectiveness in predicting energy consumption.

3.1 – Method 1: Linear Regression

For predicting the energy forecast, Regression models are a popular choice for predicting energy forecasts due to their simplicity and reasonably accurate implementation when compared to other statistical methods.Regression is a convenient method for developing both energy and statistical models, particularly when past data is available.[2]. Based on the formula below, linear regression makes use of single independent variables to predict the value of a dependent variable. So, the first step would be establishing variables for the regression models . A common way of establishing a correlation between variables is determining a “best fit” regression equation to go through the dataset consisting of multiple independent variables and a dependent variable [2]. The Formula below is a simple regression formula. Y is the dependent variable, β0 is the intercept and x are the independent variables.

$y = \beta 0 + \beta 1x1$ [3].

Then, the random error is calculated using the least square function.

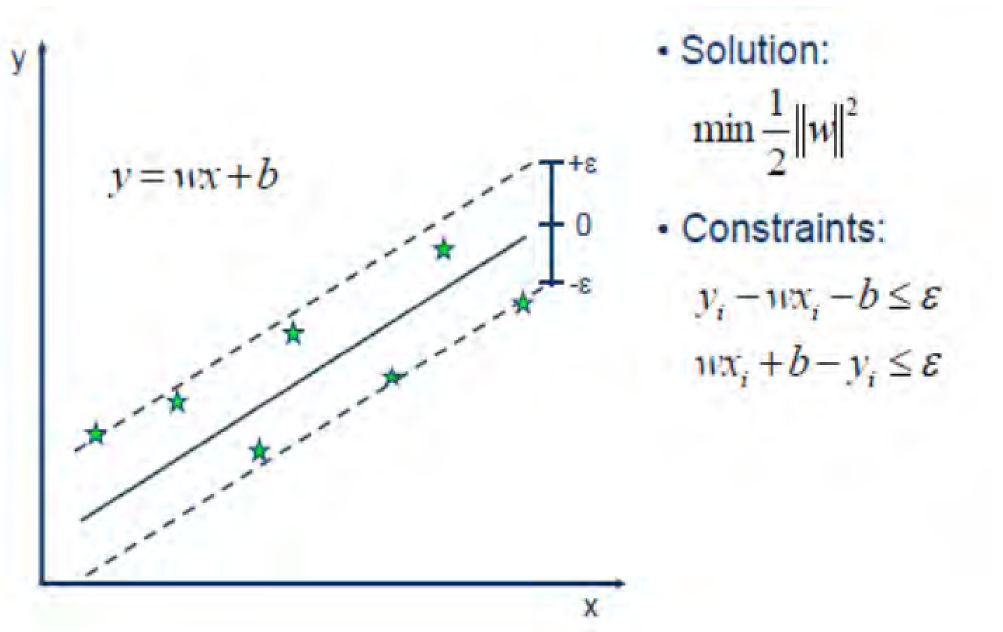
$$L = \sum_{i=1}^{n} (yi - \beta_0 - \sum_{j=1}^{k} \beta_j x_{ij})^2$$

[2].

Where L is the least square function. The equation in the squared brackets represents the distance of the fitted line from point i.

3.2 – Method 2: Support Vector Regression

Support vector regression is a type of machine learning regression that works on the principle of structural risk minimization and least maximum error threshold. It aims to keep the maximum error from a data point within a particular range or desired value [2]. The support vector regression will be used for energy models for large institutions such as Manufacturing firms. Below is the illustration for a support vector regression [2].



y
x
$y = wx + b$
+ε
0
-ε
• Solution:
$\min \frac{1}{2}\|w\|^2$
• Constraints:
$y_i - wx_i - b \leq \varepsilon$
$wx_i + b - y_i \leq \varepsilon$

Fig 1: Support vector regression underlying optimization model [2]

Figure 1 illustrates how support vector regression finds a hyperplane that separates two classes and classifies new points based on whether they lie on the positive or negative side of the hyperplane [6]. The decision boundaries are +ε and -ε , while y represents the hyperplane. The equations for the hyperplane is given as

$$y = wx + b \ [2].$$

Therefore, any hyperplane should satisfy this equation below.

$$-\varepsilon < y - wx + b < +\varepsilon \ [6].$$

3.3 – Method 3: Polynomial Regression

Polynomial Regression is a regression algorithm that models dependent variable (y) and independent variable (x) as nth degree polynomials [3]. The equation for the Polynomial Regression is given below:

$$y = \beta0 + \beta1x1 + \beta2x2 + \ldots.. + \beta nxn \ [2].$$

It is also called the special case of Multiple Linear Regression in ML. Because we add some polynomial terms to the Multiple Linear regression equation to convert it into Polynomial Regression [3]. The multiple linear regression uses the same formula as the linear regression, but it is raised to the nth power. When applying this is linear while the polynomial regression is a curved graph. The figure below shows how distinct they are.

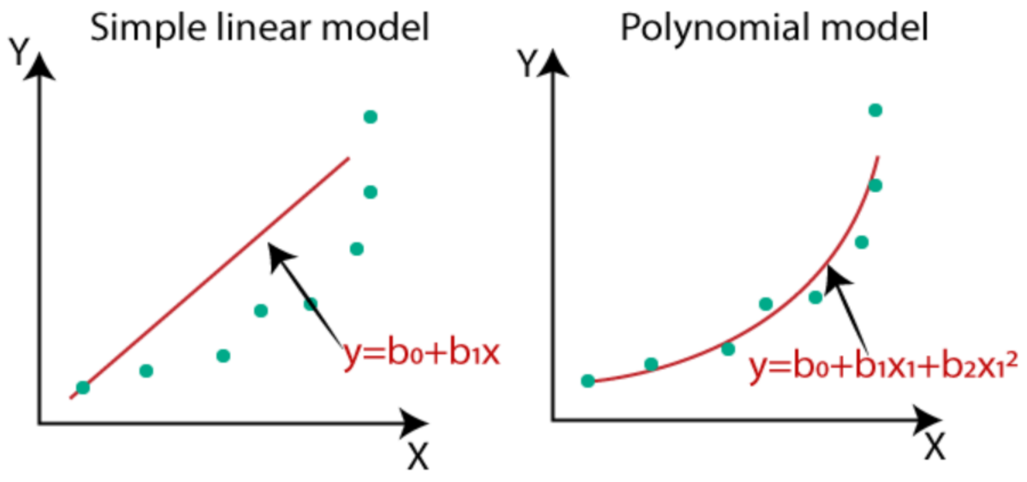


Fig 2: The difference between linear and polynomial regression [3]

3.4 – Model Validation

Model validation helps ensure that the model can perform well when given new data. If model validation is not performed then it is uncertain that the training model will work well when given new data [4]. Overall, the model validation is done to ensure that the models can predict data under given conditions. The various techniques used for validation are:

1. Hold out validation.
2. K-fold Cross-Validation.
3. Stratified K-fold Cross-Validation.
4. Leave One Out Cross-Validation.
5. Repeated Random Test-Train Splits.

All these validation techniques are recommended from data scientists. For this report , Regression models will be validated. The validation model that will be used is Repeated Random test-train splits.

3.4.1 Repeated Random test-train splits

The train-test split technique is used for regression models to test the machine learning algorithms [5]. It is implemented by using Python and Scikit learning, and the dataset is split into two subsets.

1. Training dataset: Percentage of the data set is split here to train the algorithm and fit into the machine learning model.
2. Test dataset: Percentage of the data set is split here using input from the test dataset to make predictions.

3.4.2 Performance test

Finally, the performance of each model will be evaluated using R score, Mean Absolute Error and Mean Error Squared Root. The model with the lowest R score, MAE and MSE will be selected as the best model for this project. The energy models are going to be developed using python programming.

4 – Implementation

The system implementation schedule would be composed of four phases.

Phase 1 *(Dataset collection):* The data collected from different sources to be able to predict energy consumption in residential buildings.

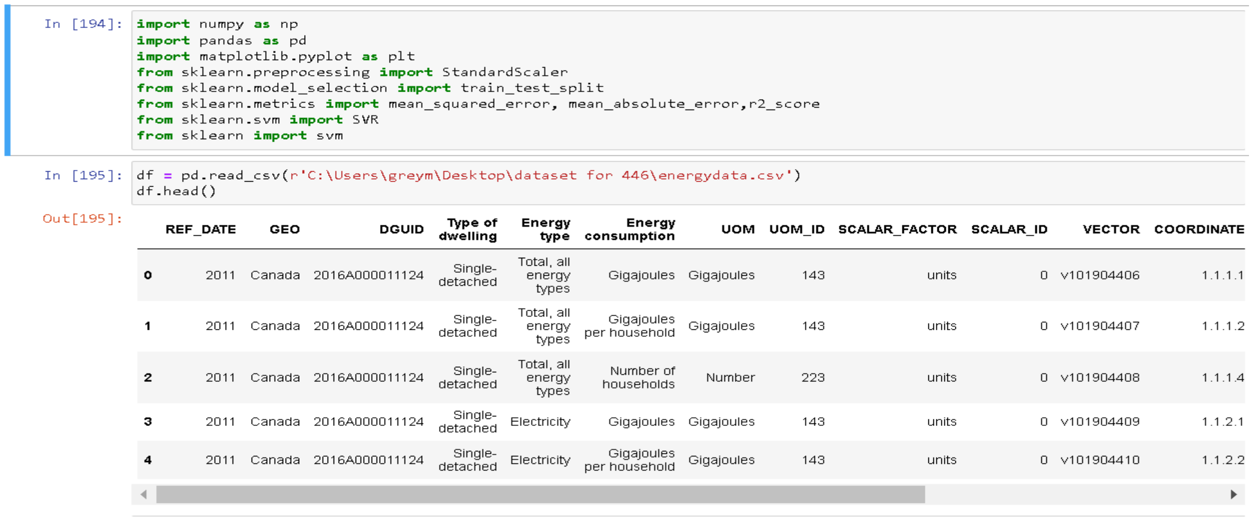
Phase 2 *(Data Preprocessing):* This phase involves data being cleaned, formatted and filtered to remove any inconsistencies, outliers and missing data.

Phase 3 *(Model Development):* Here the three machine learning algorithms will be implemented using Python programming. Pandas and Scikit-learn will be used to analyze the data and develop the model.

Phase 4 *(Model evaluation):* Here each model will undergo a test-train split to ensure that the model is predictive. Also a performance test will be done to check the best model.

4.1–Dataset collection and Data Preprocessing

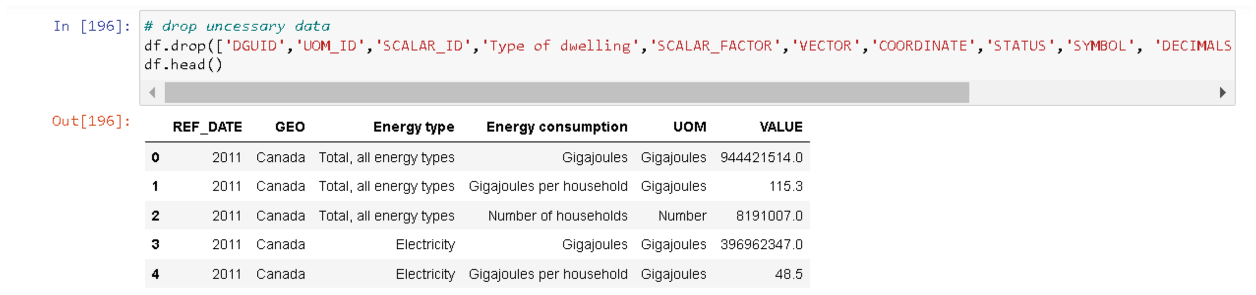
The dataset was obtained from the government of Canada. The data contains the energy consumption of households in different Provinces of Canada [5]. The code below shows data preprocessing.



```
In [194]: import numpy as np
          import pandas as pd
          import matplotlib.pyplot as plt
          from sklearn.preprocessing import StandardScaler
          from sklearn.model_selection import train_test_split
          from sklearn.metrics import mean_squared_error, mean_absolute_error,r2_score
          from sklearn.svm import SVR
          from sklearn import svm

In [195]: df = pd.read_csv(r'C:\Users\greym\Desktop\dataset for 446\energydata.csv')
          df.head()

Out[195]:
```

	REF_DATE	GEO	DGUID	Type of dwelling	Energy type	Energy consumption	UOM	UOM_ID	SCALAR_FACTOR	SCALAR_ID	VECTOR	COORDINATE
0	2011	Canada	2016A000011124	Single-detached	Total, all energy types	Gigajoules	Gigajoules	143	units	0	v101904406	1.1.1.1
1	2011	Canada	2016A000011124	Single-detached	Total, all energy types	Gigajoules per household	Gigajoules	143	units	0	v101904407	1.1.1.2
2	2011	Canada	2016A000011124	Single-detached	Total, all energy types	Number of households	Number	223	units	0	v101904408	1.1.1.4
3	2011	Canada	2016A000011124	Single-detached	Electricity	Gigajoules	Gigajoules	143	units	0	v101904409	1.1.2.1
4	2011	Canada	2016A000011124	Single-detached	Electricity	Gigajoules per household	Gigajoules	143	units	0	v101904410	1.1.2.2

Fig 3: Imported dataset

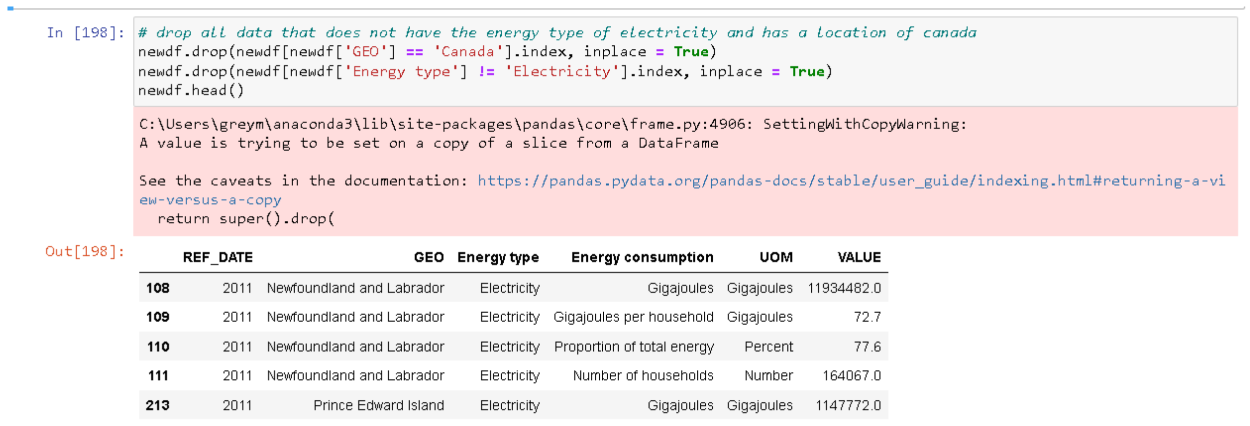


```
In [196]: # drop uncessary data
          df.drop(['DGUID','UOM_ID','SCALAR_ID','Type of dwelling','SCALAR_FACTOR','VECTOR','COORDINATE','STATUS','SYMBOL', 'DECIMALS
          df.head()
```

Out[196]:

	REF_DATE	GEO	Energy type	Energy consumption	UOM	VALUE
0	2011	Canada	Total, all energy types	Gigajoules	Gigajoules	944421514.0
1	2011	Canada	Total, all energy types	Gigajoules per household	Gigajoules	115.3
2	2011	Canada	Total, all energy types	Number of households	Number	8191007.0
3	2011	Canada	Electricity	Gigajoules	Gigajoules	396962347.0
4	2011	Canada	Electricity	Gigajoules per household	Gigajoules	48.5

Fig 4: Data dropped

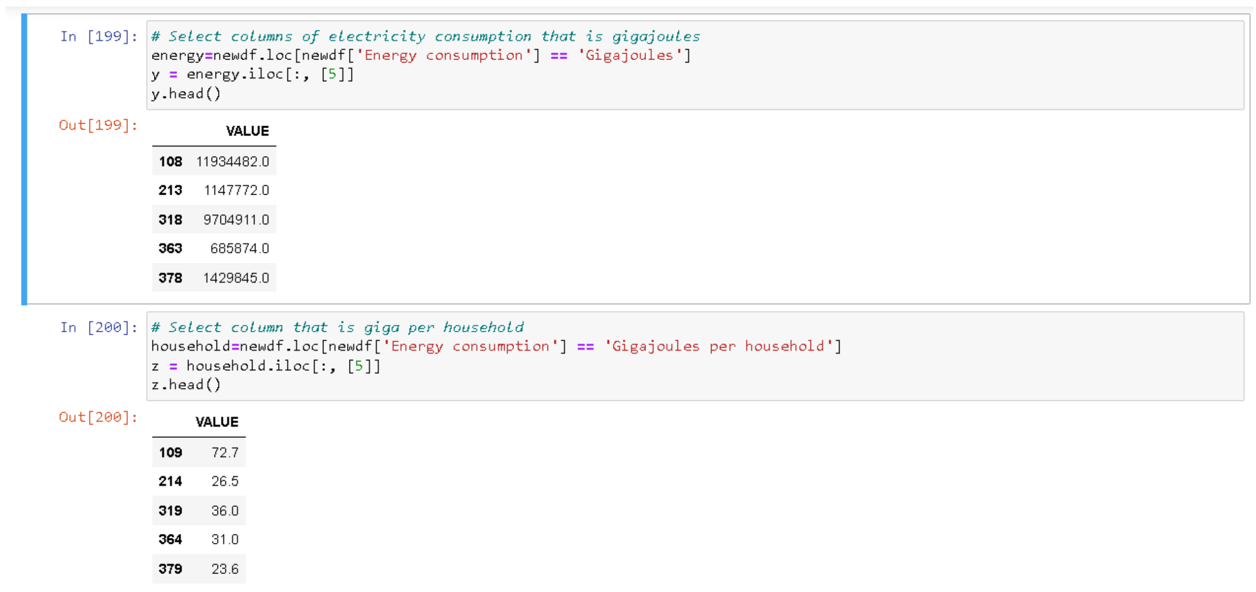


```
In [198]: # drop all data that does not have the energy type of electricity and has a location of canada
          newdf.drop(newdf[newdf['GEO'] == 'Canada'].index, inplace = True)
          newdf.drop(newdf[newdf['Energy type'] != 'Electricity'].index, inplace = True)
          newdf.head()

          C:\Users\greym\anaconda3\lib\site-packages\pandas\core\frame.py:4906: SettingWithCopyWarning:
          A value is trying to be set on a copy of a slice from a DataFrame

          See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-vi
          ew-versus-a-copy
            return super().drop(
```

Out[198]:

	REF_DATE	GEO	Energy type	Energy consumption	UOM	VALUE
108	2011	Newfoundland and Labrador	Electricity	Gigajoules	Gigajoules	11934482.0
109	2011	Newfoundland and Labrador	Electricity	Gigajoules per household	Gigajoules	72.7
110	2011	Newfoundland and Labrador	Electricity	Proportion of total energy	Percent	77.6
111	2011	Newfoundland and Labrador	Electricity	Number of households	Number	164067.0
213	2011	Prince Edward Island	Electricity	Gigajoules	Gigajoules	1147772.0

Fig 5: Energy type and location filtered



```
In [199]: # Select columns of electricity consumption that is gigajoules
          energy=newdf.loc[newdf['Energy consumption'] == 'Gigajoules']
          y = energy.iloc[:, [5]]
          y.head()
```

Out[199]:

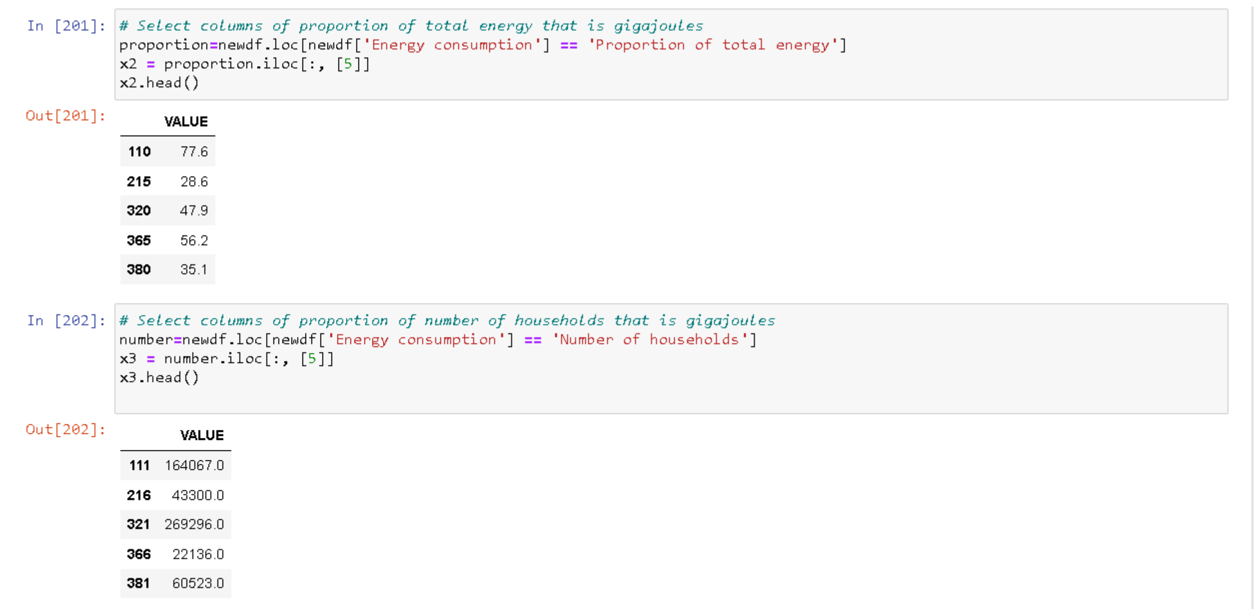
	VALUE
108	11934482.0
213	1147772.0
318	9704911.0
363	685874.0
378	1429845.0

```
In [200]: # Select column that is giga per household
          household=newdf.loc[newdf['Energy consumption'] == 'Gigajoules per household']
          z = household.iloc[:, [5]]
          z.head()
```

Out[200]:

	VALUE
109	72.7
214	26.5
319	36.0
364	31.0
379	23.6

Fig 6: The dependent and independent variables selected



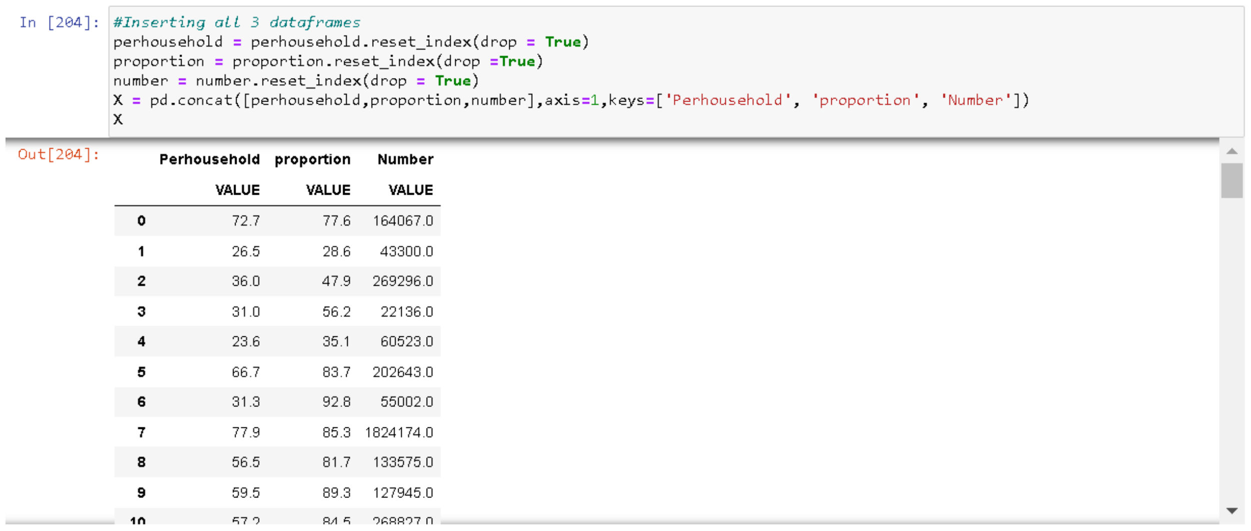
```
In [201]: # Select columns of proportion of total energy that is gigajoules
          proportion=newdf.loc[newdf['Energy consumption'] == 'Proportion of total energy']
          x2 = proportion.iloc[:, [5]]
          x2.head()
```

```
Out[201]:
           VALUE
     110    77.6
     215    28.6
     320    47.9
     365    56.2
     380    35.1
```

```
In [202]: # Select columns of proportion of number of households that is gigajoules
          number=newdf.loc[newdf['Energy consumption'] == 'Number of households']
          x3 = number.iloc[:, [5]]
          x3.head()
```

```
Out[202]:
              VALUE
     111   164067.0
     216    43300.0
     321   269296.0
     366    22136.0
     381    60523.0
```

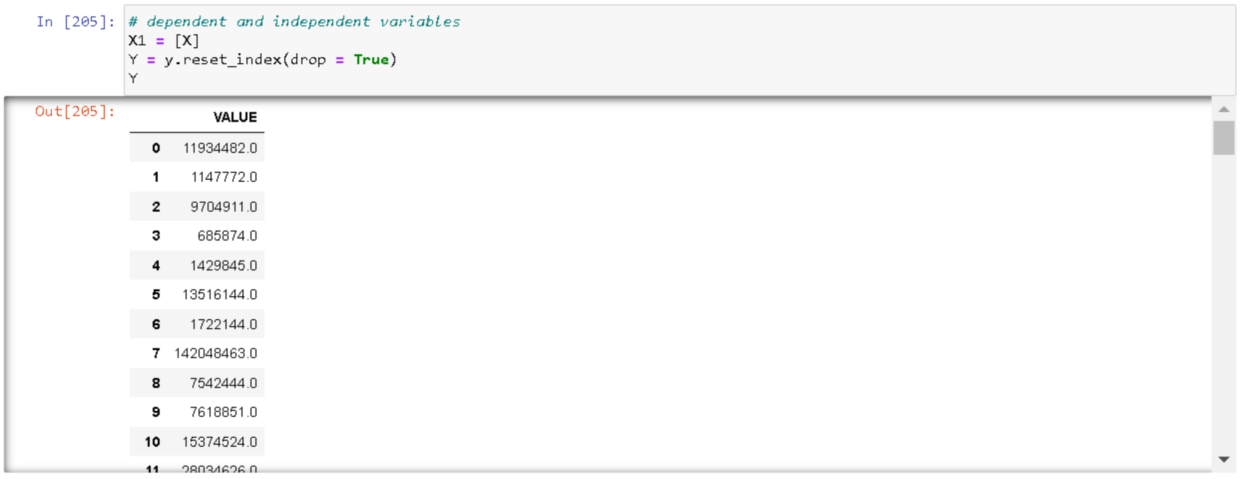
Fig 7: More independent variables are selected



```
In [204]: #Inserting all 3 dataframes
          perhousehold = perhousehold.reset_index(drop = True)
          proportion = proportion.reset_index(drop =True)
          number = number.reset_index(drop = True)
          X = pd.concat([perhousehold,proportion,number],axis=1,keys=['Perhousehold', 'proportion', 'Number'])
          X
```

```
Out[204]:
         Perhousehold  proportion     Number
                VALUE       VALUE      VALUE
     0           72.7        77.6   164067.0
     1           26.5        28.6    43300.0
     2           36.0        47.9   269296.0
     3           31.0        56.2    22136.0
     4           23.6        35.1    60523.0
     5           66.7        83.7   202643.0
     6           31.3        92.8    55002.0
     7           77.9        85.3  1824174.0
     8           56.5        81.7   133575.0
     9           59.5        89.3   127945.0
    10           57.2        84.5   268827.0
```

Fig 8: Independent variables joined


```
In [205]: # dependent and independent variables
          X1 = [X]
          Y = y.reset_index(drop = True)
          Y
```

```
Out[205]:
          VALUE
 0   11934482.0
 1    1147772.0
 2    9704911.0
 3     685874.0
 4    1429845.0
 5   13516144.0
 6    1722144.0
 7  142048463.0
 8    7542444.0
 9    7618851.0
10   15374524.0
11   28034626.0
```

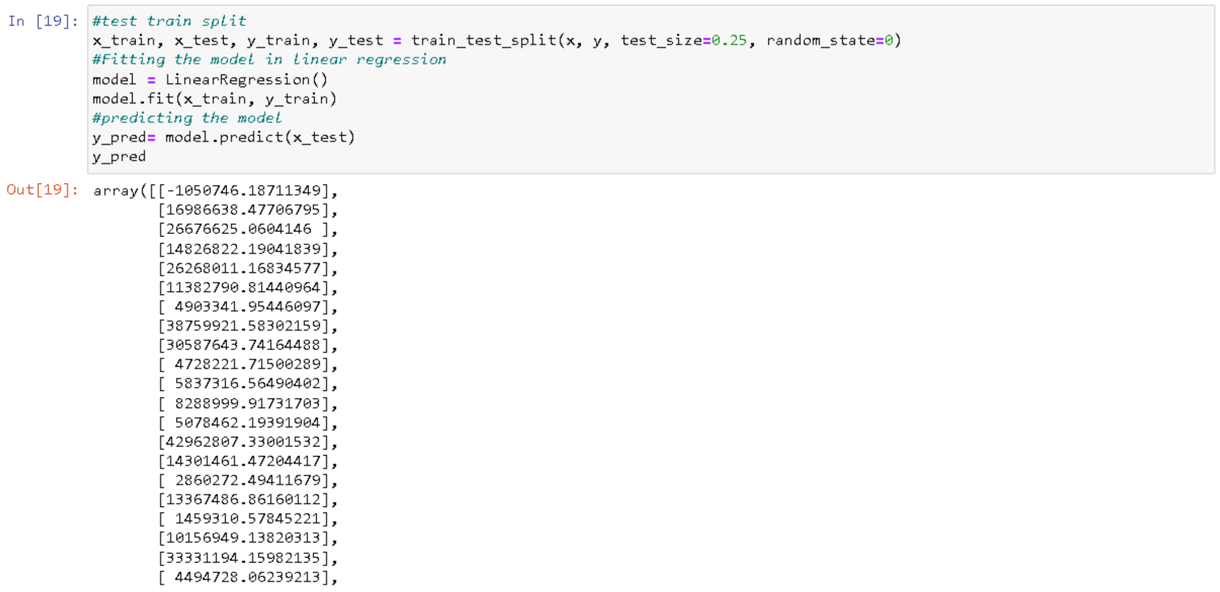
Fig 9: The dependent variable

The given figures show a data preparation process in python using the pandas library. It starts with importing necessary libraries and reading the dataset using pandas. Then, unnecessary columns in the dataset are dropped using the ‘drop’ function. After that, the data is filtered by provinces using the ‘GEO’ column, and rows with the location ‘Canada’ are dropped from the dataset. Next, the dataset is further filtered to select only the energy type ‘Electricity’ in the ‘Energy type’ column.

Furthermore, figure 6 shows that the independent and dependent variables are selected for the model, where the dependent variable is the total electricity consumed and the independent variables are the number of houses using electricity, proportion of electricity used, and the electricity used per household. The data related to each variable is selected from the energy consumption column, and the index for all the variables is dropped for efficient joining of data frames.

Lastly, the independent variables are concatenated to join their data frames into a singular data frame with the same index, and the independent and dependent variables are ready for the model development. Overall, the text provides a clear and concise overview of the data preparation process for the given dataset in python.

4.2– Linear Regression model

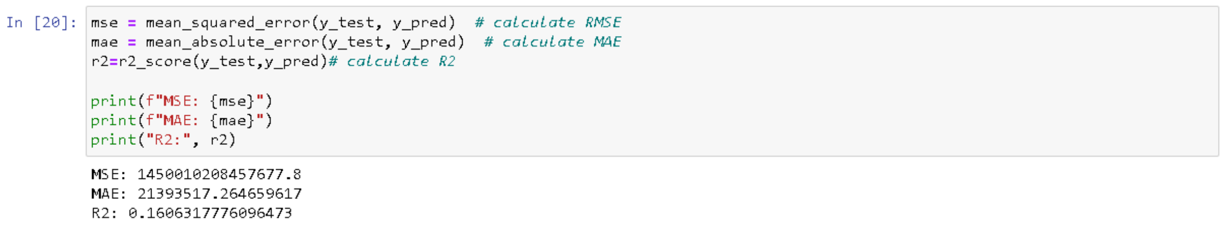


```
In [19]: #test train split
         x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.25, random_state=0)
         #Fitting the model in linear regression
         model = LinearRegression()
         model.fit(x_train, y_train)
         #predicting the model
         y_pred= model.predict(x_test)
         y_pred

Out[19]: array([[-1050746.18711349],
                [16986638.47706795],
                [26676625.0604146 ],
                [14826822.19041839],
                [26268011.16834577],
                [11382790.81440964],
                [ 4903341.95446097],
                [38759921.58302159],
                [30587643.74164488],
                [ 4728221.71500289],
                [ 5837316.56490402],
                [ 8288999.91731703],
                [ 5078462.19391904],
                [42962807.33001532],
                [14301461.47204417],
                [ 2860272.49411679],
                [13367486.86160112],
                [ 1459310.57845221],
                [10156949.13820313],
                [33331194.15982135],
                [ 4494728.06239213],
```

Fig 10: Prediction model for linear regression

The figure 10 shows a predictive model for linear regression. The model involves one independent variable (electricity consumed per household) and one dependent variable (total electricity consumed). To evaluate the performance of the model, the available data is divided into two sets using the test train split. The split used in this model is 0.25, meaning that about 31 data points are used for testing, while the remaining data points are used for training. To ensure that the test train split is consistent and reproducible, a random state of 0 is used. Once the model is trained on training data, it can be used to predict the values of the dependent variables (y) based on the values of the independent variable (x) in the testing set. The x test data is passed through the predict model, and the resulting predicted values of y are compared to actual values in the testing set to evaluate the performance of the model.

4.2.1– Model Evaluation

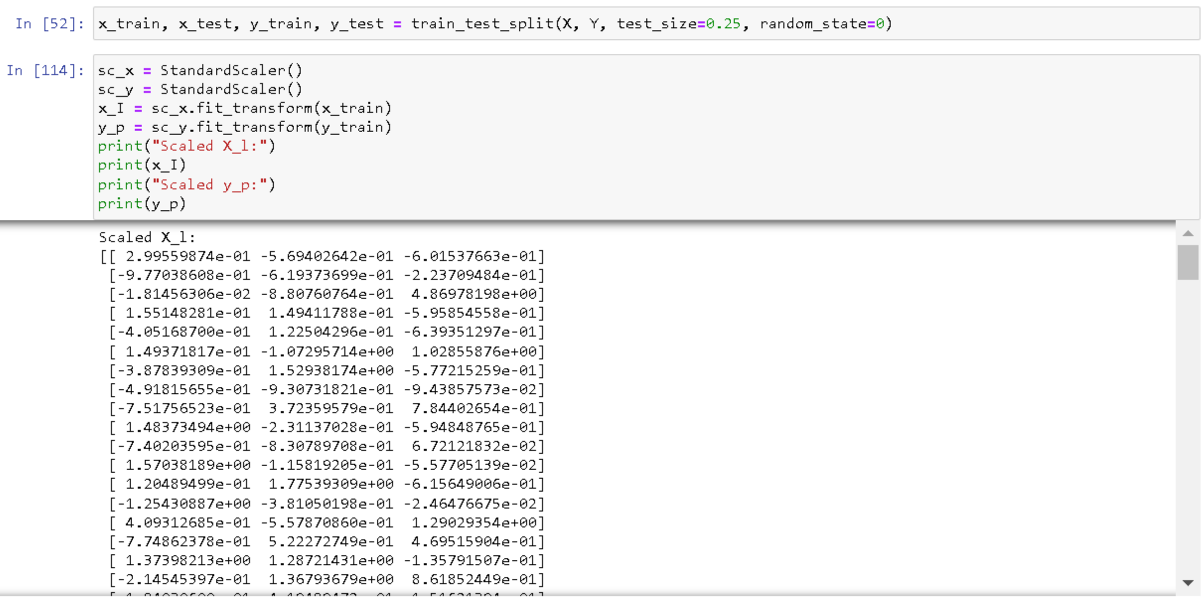


```
In [20]: mse = mean_squared_error(y_test, y_pred)  # calculate RMSE
         mae = mean_absolute_error(y_test, y_pred)  # calculate MAE
         r2=r2_score(y_test,y_pred)# calculate R2

         print(f"MSE: {mse}")
         print(f"MAE: {mae}")
         print("R2:", r2)

         MSE: 1450010208457677.8
         MAE: 21393517.264659617
         R2: 0.1606317776096473
```

Fig 11: The evaluated model for linear regression

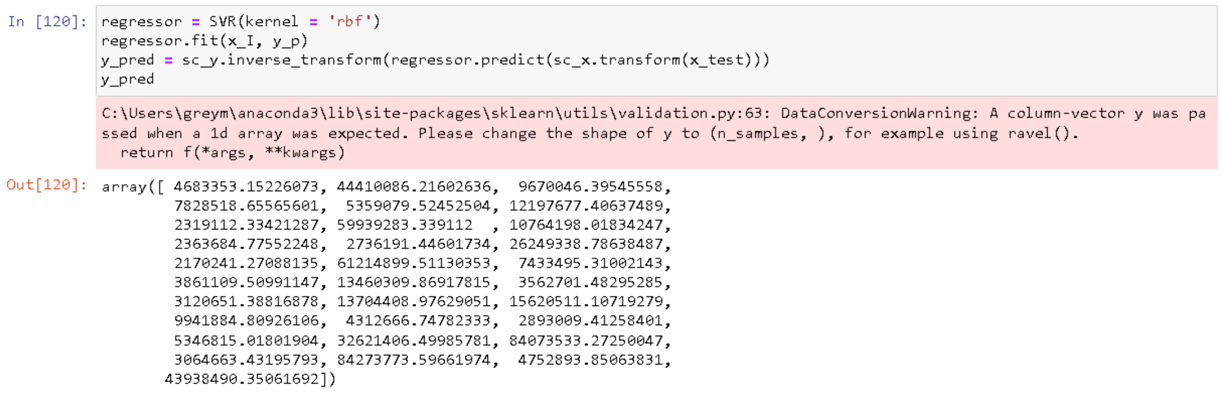
The figure 11 computes the evaluation metrics for the model. The r2_score is the R-squared coefficient. It ranges between 0 and 1, a 0 value means the model is a bad fit while a 1 value shows that the model is a good fit. The R2 score is 0.161 so the model is not a good fit. Next, the mean absolute error (MAE) measures the average absolute difference between the actual and predicted values of a target variable. The MAE in the figure above is about 21393517.3. That is a huge value, this means the model does not perform well.The mean squared error (MSE) measures the average squared difference between the actual and predicted values of the target variable. The MSE of the model is 1450010208457677.8. The number is very high therefore, the model does not perform well.
Overall, these evaluation metrics suggest that the linear regression model is not a good fit for the data and does not perform well in predicting the dependent variable based on the independent variable.

4.3– Support Vector Regression model



```
In [52]:  x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.25, random_state=0)

In [114]: sc_x = StandardScaler()
          sc_y = StandardScaler()
          x_I = sc_x.fit_transform(x_train)
          y_p = sc_y.fit_transform(y_train)
          print("Scaled X_l:")
          print(x_I)
          print("Scaled y_p:")
          print(y_p)

          Scaled X_l:
          [[ 2.99559874e-01 -5.69402642e-01 -6.01537663e-01]
           [-9.77038608e-01 -6.19373699e-01 -2.23709484e-01]
           [-1.81456306e-02 -8.80760764e-01  4.86978198e+00]
           [ 1.55148281e-01  1.49411788e-01 -5.95854558e-01]
           [-4.05168700e-01  1.22504296e-01 -6.39351297e-01]
           [ 1.49371817e-01 -1.07295714e+00  1.02855876e+00]
           [-3.87839309e-01  1.52938174e+00 -5.77215259e-01]
           [-4.91815655e-01 -9.30731821e-01 -9.43857573e-02]
           [-7.51756523e-01  3.72359579e-01  7.84402654e-01]
           [ 1.48373494e+00 -2.31137028e-01 -5.94848765e-01]
           [-7.40203595e-01 -8.30789708e-01  6.72121832e-02]
           [ 1.57038189e+00 -1.15819205e-01 -5.57705139e-02]
           [ 1.20489499e-01  1.77539309e+00 -6.15649006e-01]
           [-1.25430887e+00 -3.81050198e-01 -2.46476675e-02]
           [ 4.09312685e-01 -5.57870860e-01  1.29029354e+00]
           [-7.74862378e-01  5.22272749e-01  4.69515904e-01]
           [ 1.37398213e+00  1.28721431e+00 -1.35791507e-01]
           [-2.14545397e-01  1.36793679e+00  8.61852449e-01]
```

Fig 12: Scaled values for support vector regression model

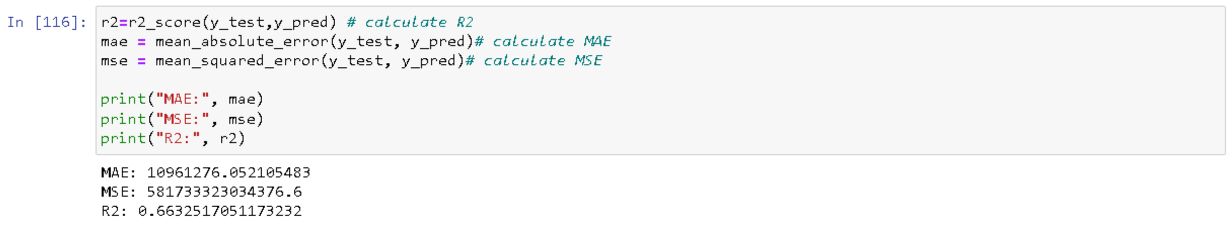


```
In [120]: regressor = SVR(kernel = 'rbf')
          regressor.fit(x_I, y_p)
          y_pred = sc_y.inverse_transform(regressor.predict(sc_x.transform(x_test)))
          y_pred

          C:\Users\greym\anaconda3\lib\site-packages\sklearn\utils\validation.py:63: DataConversionWarning: A column-vector y was pa
          ssed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
            return f(*args, **kwargs)

Out[120]: array([ 4683353.15226073, 44410086.21602636,  9670046.39545558,
                  7828518.65565601,  5359079.52452504, 12197677.40637489,
                  2319112.33421287, 59939283.339112  , 10764198.01834247,
                  2363684.77552248,  2736191.44601734, 26249338.78638487,
                  2170241.27088135, 61214899.51130353,  7433495.31002143,
                  3861109.50991147, 13460309.86917815,  3562701.48295285,
                  3120651.38816878, 13704408.97629051, 15620511.10719279,
                  9941884.80926106,  4312666.74782333,  2893009.41258401,
                  5346815.01801904, 32621406.49985781, 84073533.27250047,
                  3064663.43195793, 84273773.59661974,  4752893.85063831,
                 43938490.35061692])
```

Fig 13: Prediction model for support vector regression model

The figures above show the steps taken to train a regression model on the data. The data is first split into training and testing sets using train test split with a random state 0 and a test size of 0.25. To improve performance of the model, the variables are scaled using the standardscaler function, which helps to standardize the range of values across the features [7]. Additionally, a kernel function in the regression model captures non-linear relationships between the variables. The kernel function computes the distance between data points, which is affected by the scale of input variables. The scaled variables are then fit into the regressor model and passed through an inverse transform to give the predicted response in figure 13. The inverse transform helps obtain the actual values of the target variable that can be compared with actual values 'y_test' during model evaluation.

4.3.1– Model Evaluation

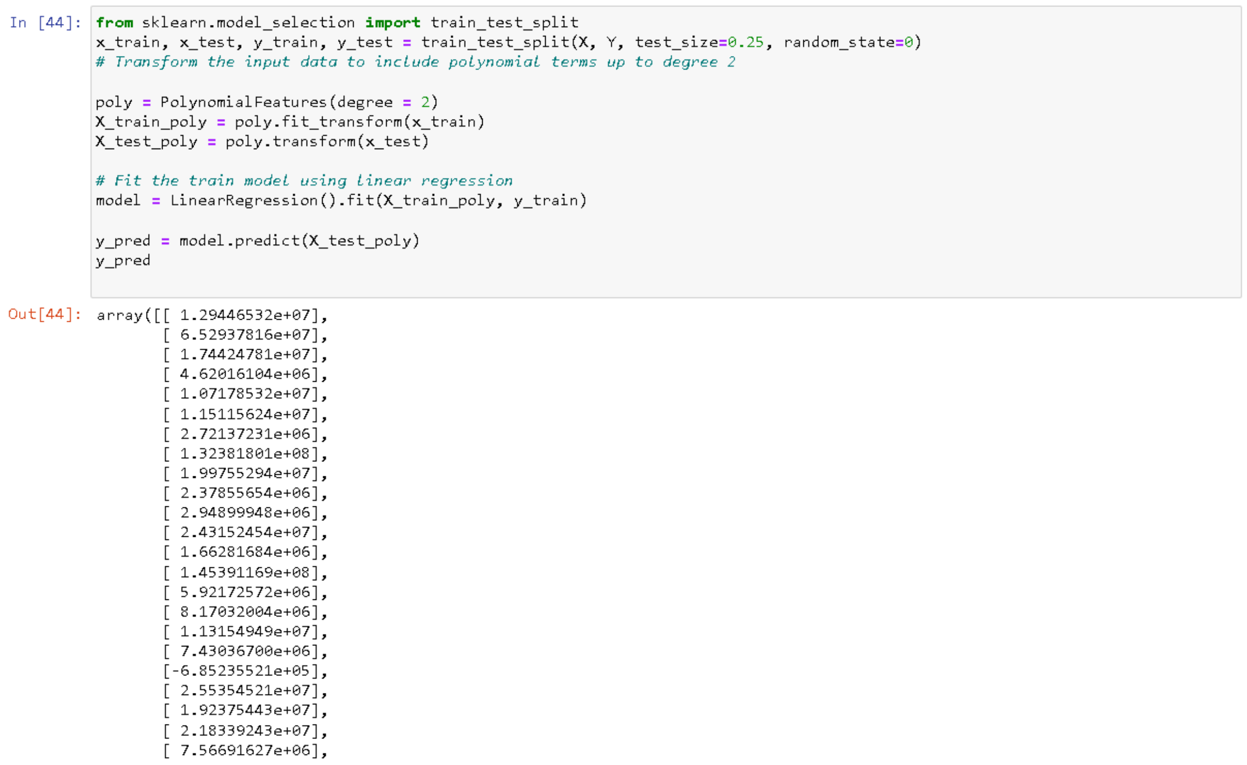


```
In [116]: r2=r2_score(y_test,y_pred) # calculate R2
          mae = mean_absolute_error(y_test, y_pred)# calculate MAE
          mse = mean_squared_error(y_test, y_pred)# calculate MSE

          print("MAE:", mae)
          print("MSE:", mse)
          print("R2:", r2)

          MAE: 10961276.052105483
          MSE: 581733323034376.6
          R2: 0.6632517051173232
```

Fig 14: The evaluated model for support vector regression

Figure 14 is the evaluation metrics for the model. The R2 score is 0.663 so the model is not a bad fit but it is also not a very good fit. The model fit is average. Next, the MAE in the figure above is about 10961276.05. That is a huge value however, due to the magnitude of the target values and test values the number shows that the model performs okay. The MSE of the model is 581733323034376.6. The number is very high but due to the test values and target values being high the model performs okay.

4.4–Polynomial Regression model

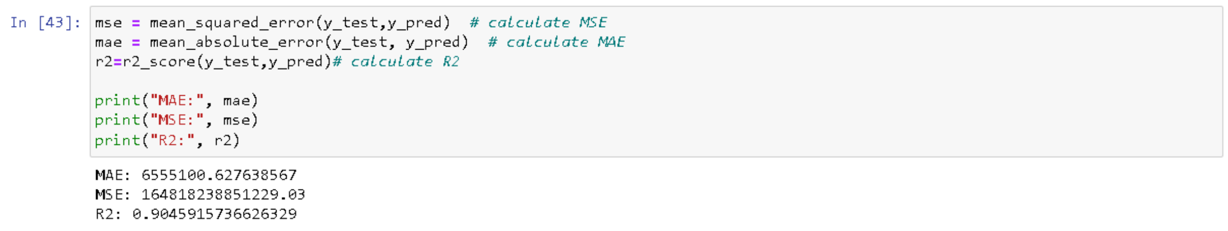


```
In [44]: from sklearn.model_selection import train_test_split
         x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.25, random_state=0)
         # Transform the input data to include polynomial terms up to degree 2

         poly = PolynomialFeatures(degree = 2)
         X_train_poly = poly.fit_transform(x_train)
         X_test_poly = poly.transform(x_test)

         # Fit the train model using linear regression
         model = LinearRegression().fit(X_train_poly, y_train)

         y_pred = model.predict(X_test_poly)
         y_pred

Out[44]: array([[ 1.29446532e+07],
                [ 6.52937816e+07],
                [ 1.74424781e+07],
                [ 4.62016104e+06],
                [ 1.07178532e+07],
                [ 1.15115624e+07],
                [ 2.72137231e+06],
                [ 1.32381801e+08],
                [ 1.99755294e+07],
                [ 2.37855654e+06],
                [ 2.94899948e+06],
                [ 2.43152454e+07],
                [ 1.66281684e+06],
                [ 1.45391169e+08],
                [ 5.92172572e+06],
                [ 8.17032004e+06],
                [ 1.13154949e+07],
                [ 7.43036700e+06],
                [-6.85235521e+05],
                [ 2.55354521e+07],
                [ 1.92375443e+07],
                [ 2.18339243e+07],
                [ 7.56691627e+06],
```

Fig 15: The predicted response for polynomial regression

For the polynomial regression model shown in figure 15, two independent variables and one dependent variable were used. The two independent variables (X) are the number of households that use electricity and the electricity per household. The dependent variable (Y) was the total electricity consumed. The polynomial degree for this model is 2, and its equation can be expressed as y = β0 + β1x1 + β2x2. The test train split is used to train the model. The split has a random state of 0 and a test size of 0.25. The train data is transformed by the polynomial regression and fit into the linear regression model. The test data is passed through the model predict to give the predicted response (y).

4.4.1– Model Evaluation



```
In [43]: mse = mean_squared_error(y_test,y_pred)  # calculate MSE
         mae = mean_absolute_error(y_test, y_pred)  # calculate MAE
         r2=r2_score(y_test,y_pred)# calculate R2

         print("MAE:", mae)
         print("MSE:", mse)
         print("R2:", r2)

         MAE: 6555100.627638567
         MSE: 164818238851229.03
         R2: 0.9045915736626329
```

Fig 16: The evaluated model for polynomial regression

Figure 16 shows the evaluation metrics for the model. The R2 score is 0.905 so the model is definitely a good fit. Next, the MAE in the figure above is about 6555100.6. That is a huge value however, due to the magnitude of the target values and test values the number shows that the model performs well. The MSE of the model is 164818238851229.0. The number is very high but due to the test values and target values being high the model performs well.

5– Conclusion and Recommendations

In conclusion, this paper has presented the development of machine learning models to predict energy consumption in residential buildings by using three algorithms: linear regression, support vector regression and polynomial regression. The dataset was collected from the government of Canada and preprocessed using python’s Pandas library to remove any inconsistencies, outliers and missing data.

The models were then developed and evaluated using evaluation metrics such as r-squared coefficient, mean absolute error, and mean squared error. The results showed that the polynomial regression model outperformed the other two models in terms of its prediction accuracy.

The findings of this study have significant implications for energy conservation and sustainability efforts in residential buildings. By accurately predicting energy consumption, building owners and operators can make informed decisions about energy usage and improve the efficiency of their buildings, ultimately reducing energy costs and carbon emissions.

Future work may involve the use of more advanced machine learning algorithms, such as deep learning, to improve the accuracy of energy consumption predictions. Additionally, the inclusion of weather data may further improve the model's accuracy as weather is known to have a significant impact on energy consumption in buildings.

Overall, this study has demonstrated the potential of machine learning models in predicting energy consumption in residential buildings, and their potential to contribute to a more sustainable and energy efficient future.

6 – References

[1]“Energy efficiency is finally turning a corner as the world tackles the energy and climate crises,” *World Economic Forum*. [Online]. Available: https://www.weforum.org/agenda/2023/01/energy-efficiency-energy-and-climate-crises/. [Accessed: 16-Feb-2023].

[2]C. R. Madhusudanan, “A machine learning framework for energy consumption ... - tigerprints,” *tigerprints.clemson.edu*, Aug-2019. [Online]. Available: https://tigerprints.clemson.edu/cgi/viewcontent.cgi?article=4191&context=all_theses. [Accessed: 16-Feb-2023].

[3]“Machine learning polynomial regression - javatpoint,” *www.javatpoint.com*. [Online]. Available: https://www.javatpoint.com/machine-learning-polynomial-regression. [Accessed: 16-Feb-2023].

[4]D. Singh, “Deepika Singh,” *Pluralsight*, 06-June-2019. [Online]. Available: https://www.pluralsight.com/guides/validating-machine-learning-models-scikit-learn. [Accessed: 16-Feb-2023].

[5]Mayuresh, “Train test split for Evaluating Machine Learning Algorithms: An important guide (2021),” *UNext*, 24-Jan-2023. [Online]. Available: https://u-next.com/blogs/artificial-intelligence/train-test-split/. [Accessed: 16-Feb-2023].

[6]A. Sethi, “Support vector regression in machine learning,” *Analytics Vidhya*, 02-Dec-2022. [Online]. Available: https://www.analyticsvidhya.com/blog/2020/03/support-vector-regression-tutorial-for-machine-learning/. [Accessed: 16-Feb-2023].

[7] B. Naibei, “Getting started with support vector regression in python,” *Section*. [Online]. Available: https://www.section.io/engineering-education/support-vector-regression-in-python/. [Accessed: 15-Mar-2023].