DATA MINING & MACHINE LEARNING ASSIGNMENT

Donal Crotty
Cloud Computing 4
A00216737

Table of Contents

Introduction

As part of the Data Mining and Machine Learning module we were given the task to select three data sets to carry out data mining techniques on. The data sets needed to be suitable for performing both regression and classification techniques on. We were required to perform three data mining techniques; Decision Trees for Classification, Linear Regression for Regression and K Nearest Neighbour (kNN) which can be used for both Classification or Regression.

When carrying out analyses on data, it is necessary to get to know the data you are dealing with prior to performing predictions etc., so you can get a feel as to whether the results you calculate are appropriate or not. In this case, it would be a requirement to explore the data; producing tables and graphs for example. It is also important to define both training and testing datasets. It is also required that you create a prediction model, apply that model to the test data and evaluate the results given.

Overview

For the Regression analysis, I have chosen to use a dataset based on Energy Efficiency. The energy analysis is performed on 12 different building shapes simulated using an energy efficiency software package. Each building differs with by the attributes glazing area, the glazing area distribution, and the orientation, amongst other attributes. Various settings are simulated as functions of the previously mentioned attributes to obtain 768 building shapes, with each instance in the dataset registered as a building shape. The aim of the dataset is to use the stored attributes to predict a quantitative estimation of energy performance of residential buildings. The two predicted attributes in this case are Cooling Load and Heating Load.

In Regression analysis, I will be performing the Linear Regression model on the dataset. Regression analysis is used to predict the value of one variable called the dependent variable based on other independent variables.

For Classification analysis, I have chosen to use a dataset based on Adult Census Income. The dataset consists of records which were extracted from census data under the following conditions: the age is greater than 16, the Annual Gross Income is less than 100,00, along with more conditions. The purpose of the data is to predict whether a person's income is greater than, equal to or less than 50,000 per year. Some of the attributes within this dataset include, marital status, sex, occupancy, work class, age, hours per week etc.

In Classification analysis, I will be performing Decision Tree modelling on the above dataset. Classification analysis is used when given a collection of records, with each record containing a set of attributes, the aim is to then assign previously unseen records to a class as accurately as possible. Decision Tree is an all-purpose classifier that does well on most problems. A decision tree is essentially a flowchart of classification and is very useful for applications where the classification mechanism must be transparent for legal reasons for example.

For kNN analysis, I will be performing the method also on the Energy Efficiency dataset. kNN is a non-parametric method. The input for the method consists of k's closest training examples in the feature space. It is the output which decides whether kNN is to be used for Classification or Regression. In Classification, the output is a class membership, whereas in Regression, the output is the property value for the object.

Data Exploration

Before performing the data mining techniques on the datasets, it is important to firstly explore the dataset we are working with. Exploring the data will allow us to understand the attributes and how important their roles are in making predictions.

Linear Regression and kNN – Energy Efficiency

Before performing the Linear Regression data mining technique on the Energy Efficiency dataset, I decided it would be beneficial to explore some of the attributes that are used to predict the Cooling Load and Heating Load attributes. As I am using the Energy Efficiency dataset for both Regression and kNN techniques, I only needed to explore the data once.

```
> cor(engy_eff)
                        RelativeCompactness   SurfaceArea   WallArea      RoofArea OverallHeight Orientation   GlazingArea
RelativeCompactness            1.000000e+00 -9.919015e-01 -0.2037817 -8.688234e-01     0.8277473  0.000000000  7.617400e-20
SurfaceArea                   -9.919015e-01  1.000000e+00  0.1955016  8.807195e-01    -0.8581477  0.000000000  4.664140e-20
WallArea                      -2.037817e-01  1.955016e-01  1.0000000 -2.923165e-01     0.2809757  0.000000000  0.000000e+00
RoofArea                      -8.688234e-01  8.807195e-01 -0.2923165  1.000000e+00    -0.9725122  0.000000000 -1.197187e-19
OverallHeight                  8.277473e-01 -8.581477e-01  0.2809757 -9.725122e-01     1.0000000  0.000000000  0.000000e+00
Orientation                    0.000000e+00  0.000000e+00  0.0000000  0.000000e+00     0.0000000  1.000000000  0.000000e+00
GlazingArea                    7.617400e-20  4.664140e-20  0.0000000 -1.197187e-19     0.0000000  0.000000000  1.000000e+00
GlazingAreaDistribution        0.000000e+00  0.000000e+00  0.0000000  0.000000e+00     0.0000000  0.000000000  2.129642e-01
HeatingLoad                    6.222722e-01 -6.581202e-01  0.4556712 -8.618283e-01     0.8894307 -0.002586534  2.698410e-01
CoolingLoad                    6.343391e-01 -6.729989e-01  0.4271170 -8.625466e-01     0.8957852  0.014289598  2.075050e-01
                        GlazingAreaDistribution  HeatingLoad CoolingLoad
RelativeCompactness                  0.00000000  0.622272179  0.63433907
SurfaceArea                          0.00000000 -0.658120227 -0.67299893
WallArea                             0.00000000  0.455671157  0.42711700
RoofArea                             0.00000000 -0.861828253 -0.86254660
OverallHeight                        0.00000000  0.889430674  0.89578517
Orientation                          0.00000000 -0.002586534  0.01428960
GlazingArea                          0.21296422  0.269840996  0.20750499
GlazingAreaDistribution              1.00000000  0.087367594  0.05052512
HeatingLoad                          0.08736759  1.000000000  0.97586181
CoolingLoad                          0.05052512  0.975861813  1.00000000
> 
```

Figure 1- Correlation of Energy Efficiency Dataset

In Figure 1, we have gotten a Correlation Coefficient of the Energy Efficiency Dataset. The cor() function allows us to explore what is the estimated rank based measure of association of the variables. The Correlation Coefficient can also be calculated by dividing the Co-variance of a variable by the sum of their individual standard deviations.

```
> var(engy_eff)
                        RelativeCompactness   SurfaceArea     WallArea      RoofArea OverallHeight Orientation   GlazingArea
RelativeCompactness            1.118887e-02 -9.242069e+00   -0.9403911 -4.150839e+00     0.1533246  0.00000000  1.073424e-21
SurfaceArea                   -9.242069e+00  7.759164e+03  751.2907432  3.503937e+03  -132.3702738  0.00000000  5.473313e-19
WallArea                      -9.403911e-01  7.512907e+02 1903.2698827 -5.759896e+02    21.4654498  0.00000000  0.000000e+00
RoofArea                      -4.150839e+00  3.503937e+03 -575.9895698  2.039963e+03   -76.9178618  0.00000000 -7.203513e-19
OverallHeight                  1.533246e-01 -1.323703e+02   21.4654498 -7.691786e+01     3.0664928  0.00000000  0.000000e+00
Orientation                    0.000000e+00  0.000000e+00    0.0000000  0.000000e+00     0.0000000  1.25162973  0.000000e+00
GlazingArea                    1.073424e-21  5.473313e-19    0.0000000 -7.203513e-19     0.0000000  0.00000000  1.774772e-02
GlazingAreaDistribution        0.000000e+00  0.000000e+00    0.0000000  0.000000e+00     0.0000000  0.00000000  4.400261e-02
HeatingLoad                    6.641607e-01 -5.849413e+02  200.5863233 -3.927638e+02    15.7156617 -0.02919817  3.627261e-01
CoolingLoad                    6.383312e-01 -5.639665e+02  177.2672425 -3.706169e+02    14.9230052  0.15208605  2.629852e-01
                        GlazingAreaDistribution  HeatingLoad CoolingLoad
RelativeCompactness                  0.00000000   0.66416070   0.6383312
SurfaceArea                          0.00000000 -584.94130650 -563.9664689
WallArea                             0.00000000  200.58632334  177.2672425
RoofArea                             0.00000000 -392.76381492 -370.6168557
OverallHeight                        0.00000000   15.71566167   14.9230052
Orientation                          0.00000000   -0.02919817    0.1520860
GlazingArea                          0.04400261    0.36272608    0.2629852
GlazingAreaDistribution              2.40547588    1.36725799    0.7454857
HeatingLoad                          1.36725799  101.81204991   93.6740637
CoolingLoad                          0.74548566   93.67406374   90.5029827
```

Figure 2- Variance of Energy Efficiency Dataset

In Figure 2, we have computed the variance of the dataset using the var() function; which is a numerical measure of how the data values are dispersed around the mean.

```
> cov(engy_eff)
                          RelativeCompactness   SurfaceArea      WallArea      RoofArea OverallHeight Orientation   GlazingArea
RelativeCompactness              1.118887e-02 -9.242069e+00    -0.9403911 -4.150839e+00     0.1533246  0.00000000  1.073424e-21
SurfaceArea                     -9.242069e+00  7.759164e+03   751.2907432  3.503937e+03  -132.3702738  0.00000000  5.473313e-19
WallArea                        -9.403911e-01  7.512907e+02  1903.2698827 -5.759896e+02    21.4654498  0.00000000  0.000000e+00
RoofArea                        -4.150839e+00  3.503937e+03  -575.9895698  2.039963e+03   -76.9178618  0.00000000 -7.203513e-19
OverallHeight                    1.533246e-01 -1.323703e+02    21.4654498 -7.691786e+01     3.0664928  0.00000000  0.000000e+00
Orientation                      0.000000e+00  0.000000e+00     0.0000000  0.000000e+00     0.0000000  1.25162973  0.000000e+00
GlazingArea                      1.073424e-21  5.473313e-19     0.0000000 -7.203513e-19     0.0000000  0.00000000  1.774772e-02
GlazingAreaDistribution          0.000000e+00  0.000000e+00     0.0000000  0.000000e+00     0.0000000  0.00000000  4.400261e-02
HeatingLoad                      6.641607e-01 -5.849413e+02   200.5863233 -3.927638e+02    15.7156617 -0.02919817  3.627261e-01
CoolingLoad                      6.383312e-01 -5.639665e+02   177.2672425 -3.706169e+02    14.9230052  0.15208605  2.629852e-01
                          GlazingAreaDistribution   HeatingLoad  CoolingLoad
RelativeCompactness                    0.00000000    0.66416070    0.6383312
SurfaceArea                            0.00000000 -584.94130650 -563.9664689
WallArea                               0.00000000  200.58632334  177.2672425
RoofArea                               0.00000000 -392.76381492 -370.6168557
OverallHeight                          0.00000000   15.71566167   14.9230052
Orientation                            0.00000000   -0.02919817    0.1520860
GlazingArea                            0.04400261    0.36272608    0.2629852
GlazingAreaDistribution                2.40547588    1.36725799    0.7454857
HeatingLoad                            1.36725799  101.81204991   93.6740637
CoolingLoad                            0.74548566   93.67406374   90.5029827
```

Figure 3- Co-variance of Energy Efficiency Dataset

Figure 3, we have computed the co-variance of the dataset using the cov() function. The co-variance measures how the variables in the dataset are linearly related. A positive value would indicate a positive linear relationship and a negative covariance would show that the variables are not linearly related.

We can also produce a matrix of scatterplots from the dataset, by using the pairs() function. This function is an easy way to quickly decipher whether you have a linear correlation between multiple variables. The scatterplot matrix shows all the pairwise scatterplots of the variables on a single plot view in a matrix format. The scatterplot matrix for this dataset can be seen in Figure 4.

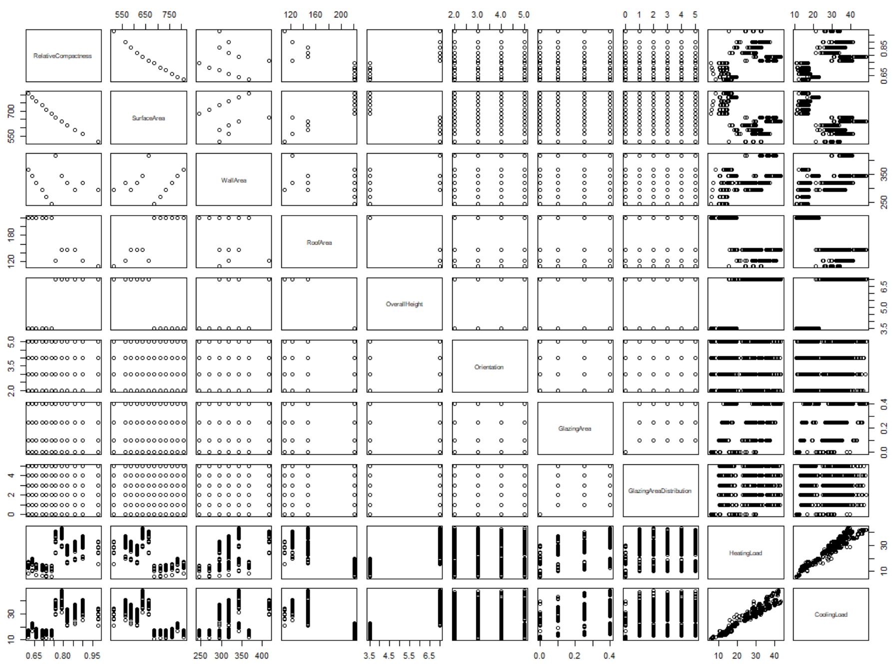


Figure 4- Scatter Plot Matrix Energy Efficiency Dataset

Classification– Adult Census Income

When considering applying a Classification technique to the Adult Income Census dataset, I felt the need to firstly explore the dataset to get a better understanding of the dataset I was working with. In figure 5, I got the head() of the dataset, this gave me the first 6 rows of the dataset. By looking at this I could then get an idea of the variations of instances within the dataset.

```
> head(adult_income)
  Age        Workclass Fnlwgt  Education Education_Num        Marital_Status              Serv  Relationship  Race
1  39        State-gov  77516  Bachelors            13         Never-married      Adm-clerical Not-in-family White
2  50 Self-emp-not-inc  83311  Bachelors            13    Married-civ-spouse   Exec-managerial       Husband White
3  38          Private 215646    HS-grad             9              Divorced Handlers-cleaners Not-in-family White
4  53          Private 234721       11th             7    Married-civ-spouse Handlers-cleaners       Husband Black
5  28          Private 338409  Bachelors            13    Married-civ-spouse    Prof-specialty          Wife Black
6  37          Private 284582    Masters            14    Married-civ-spouse   Exec-managerial          Wife White
     Sex Capital_Gain Capital_Loss Hours_Per_Week Native_Country  Wage
1   Male         2174            0             40  United-States <=50K
2   Male            0            0             13  United-States <=50K
3   Male            0            0             40  United-States <=50K
4   Male            0            0             40  United-States <=50K
5 Female            0            0             40           Cuba <=50K
6 Female            0            0             40  United-States <=50K
```

Figure 5- Head of Adult Income dataset

The next exploration of data that I performed was to get a table of the Wage and Age columns within the dataset. As seen in Figure 6, this allows us to see that for example there are 7841 people with a wage greater than 50k. It also lets us see that there are 43 people 90 years of age.

```
> table(adult_income$Wage)

        <=50K  >50K
    1   24720  7841
> table(adult_income$Age)

 17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47 
395 550 712 753 720 765 877 798 841 785 835 867 813 861 888 828 875 886 876 898 858 827 816 794 808 780 770 724 734 737 708 
 48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78 
543 577 602 595 478 464 415 419 366 358 366 355 312 300 258 230 208 178 150 151 120 108  89  72  67  64  51  45  46  29  23 
 79  80  81  82  83  84  85  86  87  88  90 
 22  22  20  12   6  10   3   1   1   3  43 
```

Figure 6- Tables of Wage and Age Columns

Next, we create a dataset of the original dataset randomly shuffled which we can then create a training and testing set of data which we will use to create our decision tree.

Analysis of Results

Linear Regression – Energy Efficiency

When performing Linear Regression on a dataset, we must first create a prediction model. In the case of the Energy Efficiency dataset; there are two predicted attributes; Heating Load and Cooling Load. This means we must create a prediction model for each predicted attribute. To do this, we add the dataset to a data frame. We then create our prediction model for Heating Load, in this case called 'regression.model'. As seen in Figure 7, we pass in our predicted variable Heating Load and describe it by the following chosen attributes: Surface Area, Roof Area and Wall Area.

```
regression.model <- lm(HeatingLoad ~ SurfaceArea + RoofArea + WallArea)
```

Figure 7- Linear Regression Model for Heating Load

We can then inject in values for the Surface Area, Roof Area and Wall Area to pass to the prediction model to allow it to make a prediction for Heating Load based on those values. In our case, Surface Area = 500, Roof Area = 150 and Wall Area = 300. We the pass the regression model and the injected data into the predict() function. As can be seen in Figure 8, the predicted Heating Load is 20.93653 Kw(Kilowatts). Also in Figure 8, we can get a summary of the prediction model as R Squared. This shows the coefficient of determination of the linear regression model. In our case, the coefficient of determination of the model is 0.7881383.

```
> predict(regression.model, newdata)
       1
20.93653
> summary(regression.model)$r.squared
[1] 0.7881383
> predict(regression.model, newdata, interval="confidence")
       fit      lwr      upr
1 20.93653 19.87002 22.00304
> predict(regression.model, newdata, interval="predict")
       fit      lwr      upr
1 20.93653 11.74533 30.12773
> plot(regression.model, which = 1)
> 
```

Figure 8- Heating Load Prediction Model Results

We can also calculate the Prediction and Confidence Intervals of the regression model. For this, we use the default 0.95 confidence level. The result is for the 95% Prediction Interval of the Heating Load, the load for a building with a Surface Area of 500, a Roof Area of 150 and Wall Area of 300, is between 11.74533 and 22.00304 Kw.

Finally, we plot the prediction linear regression model for the Heating Load in Figure 9.

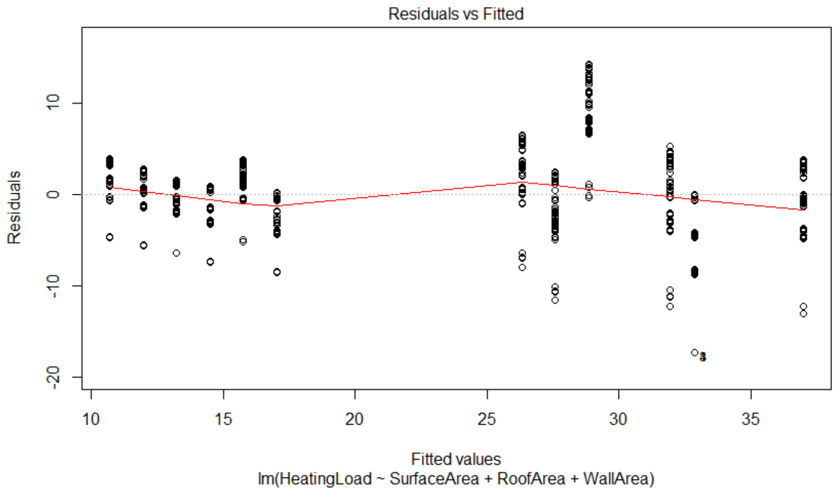


Figure 9- Heating Load Regression Model Plot

Similarly, we perform the same process for predicting the Cooling Load of the building determined by the same above injected values as for the Heating Load. We build our prediction model called 'regression.model1'.

```
regression.model1 <- lm(CoolingLoad ~ SurfaceArea + RoofArea + WallArea)
```

Figure 10- Cooling Load Regression Model

We then inject the same values in as for the Heating Load prediction, giving us the below results in Figure 11.

```
> newdata1 <- data.frame(SurfaceArea=500.0, RoofArea=150.0, WallArea=300.0)
> predict(regression.model1, newdata1)
       1
24.16372
> summary(regression.model1)$r.squared
[1] 0.7774655
> predict(regression.model1, newdata1, interval="confidence")
       fit      lwr      upr
1 24.16372 23.13317 25.19428
> predict(regression.model1, newdata1, interval="predict")
       fit      lwr      upr
1 24.16372 15.28243 33.04502
```

Figure 11- Cooling Load Prediction Model Results

As can be seen in Figure 11, the predicted Cooling Load is 24.16372 Kw(Kilowatts). Also in Figure 11, we can get a summary of the prediction model as R Squared. In our case, the coefficient of determination of the model is 0.7774655.

We once again calculate the Prediction and Confidence Intervals of the regression model. For this, we use the default 0.95 confidence level. The result is for the 95% Prediction Interval of the Cooling Load, the load for a building with a Surface Area of 500, a Roof Area of 150 and Wall Area of 300, is between 15.28243 and 33.04502 Kw.

Finally, we plot the prediction linear regression model for Cooling Load. When comparing the predicted Heating and Cooling Loads for the described building, we can then determine that a building of Surface Area 500, Roof Area 150 and Wall Area 300 will use more Kilowatts to for cooling the building (24.16372 Kw) than it will for heating the building (20.93653 Kw).

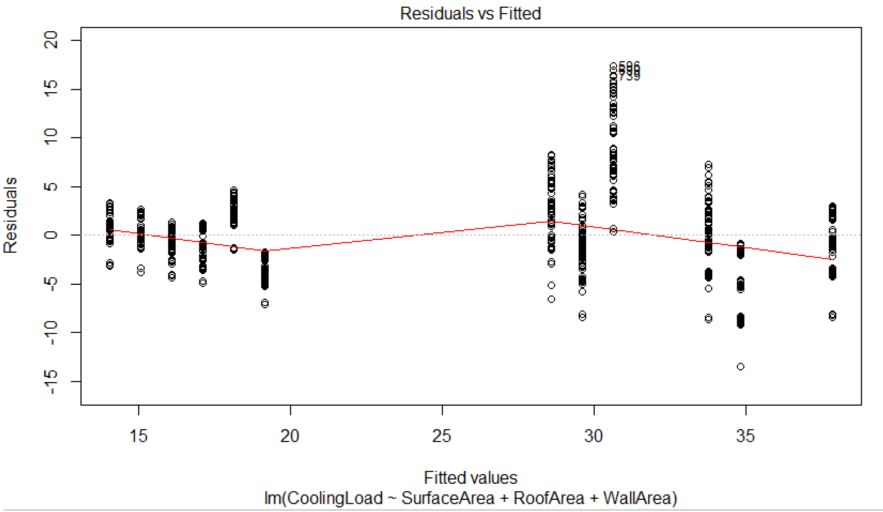


Figure 12- Cooling Load Regression Model Plot

Classification – Adult Census Income

Figure 13 represents the summary of the Adult Income Decision Tree Model. It shows that out of the 8000 instances in training data set, there were 961 wrongly classified instances. 320 instances were classified as less than or equal to 50k when they were greater than 50k. These are known as false negatives. 641 instances were classified as greater than 50k when they were less than 50k. These are known as false positives.

```
Evaluation on training data (8000 cases):

	    Decision Tree
	  ----------------
	  Size      Errors

	    64  961(12.0%)   <<


	   (a)   (b)    <-classified as
	  ----  ----
	  5778   320    (a): class <=50K
	   641  1261    (b): class >50K


	Attribute usage:

	100.00% Relationship
	100.00% Capital_Gain
	 95.78% Capital_Loss
	 54.28% Education
	 39.17% Education_Num
	 32.42% Age
	 29.38% Hours_Per_Week
	 16.70% Race
	 16.55% Serv
	  8.46% Workclass
	  2.03% Native_Country
	  0.94% Fnlwgt
	  0.77% Marital_Status
```

Figure 13 - Summary of Adult Income Decision Tree

Figure 13 also shows the estimated relevance of each attribute to making the Wage prediction. For example, it shows that Relationship and Capital Gain would be influential when making the prediction. On the other hand, it shows that Fnlwgt and Marital Status would not be influential when predicting the Wage bracket.

Figure 14 shows the Decision Tree model plot for Adult Census Income. Due to how large the dataset is, the resulting decision tree is quite messy and difficult to understand. Ideally, it would be a lot easier to read the Decision Tree by using a smaller dataset which would result in a much smaller Decision Tree.

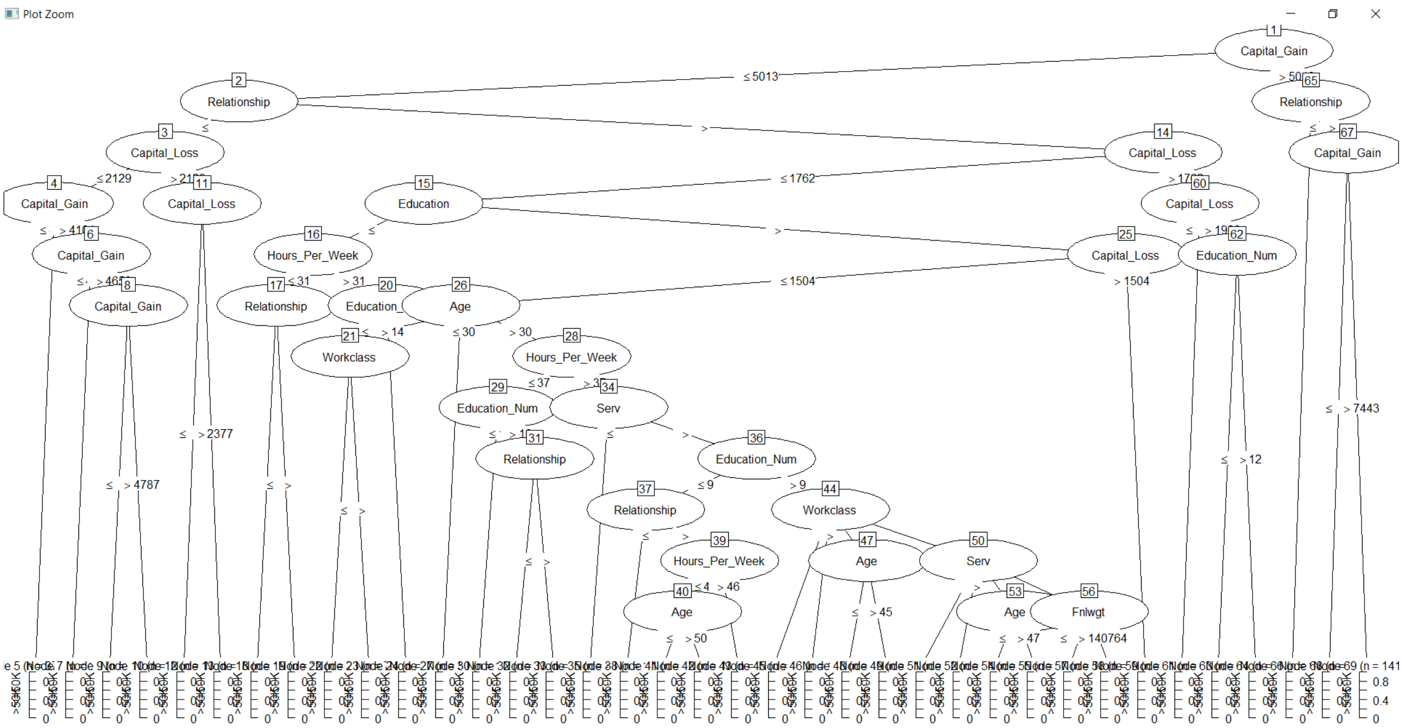
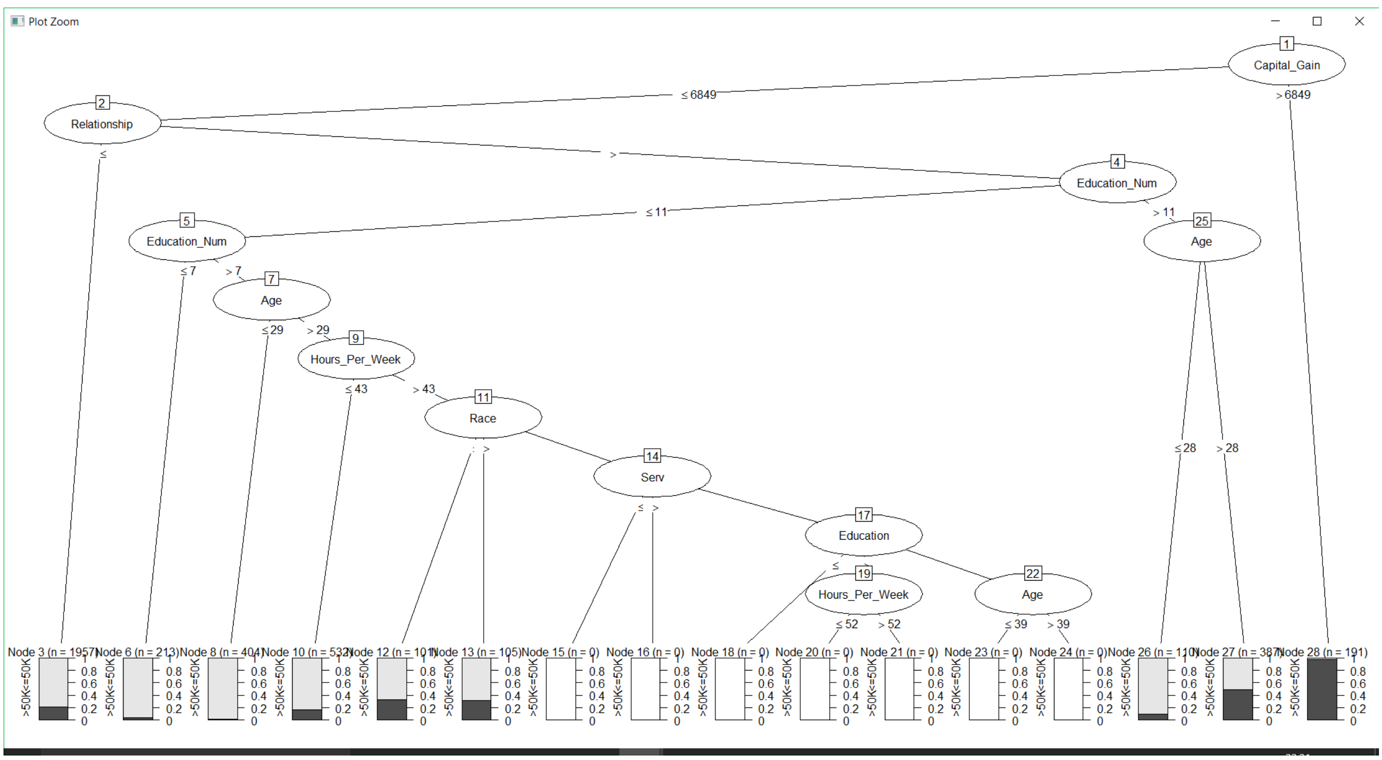


Figure 14 - Adult Census Income Decision Tree

However, by looking at the Decision Tree, we can evaluate a few attributes from it. Firstly, we can see that Capital_Gain seems to be the root of the tree with Relationship being the first branch. We can see several splitting attributes, e.g. Node 32 Education_Num splits into the Serv and Workclass nodes. There are also cases of Multi-way splits within the tree. We can also notice evidence of both Induction and Deduction between the nodes within the Decision Tree.

A way in which we could view the decision tree for this dataset in a smaller more readable way would be to reduce the size of the training set which we use to make the tree model. A smaller tree can be seen below, this has a training set of 1/10 of the original dataset.



As can be seen with the above decision tree, it is significantly easier to read the nodes on the tree, we can also view bar charts at the base of the tree that previously were too small to view. These bar charts at the base of the tree represent the value count for each node.

kNN – Energy Efficiency

When performing the k Nearest Neighbour on the Energy Efficiency dataset, we must first create a normalization function. We can then run our data through that normalization function and store it as a data frame. We then need to evaluate whether the normalization worked or not, we can do this by getting a summary of a column of the normalized dataset.

```
> summary(adult_income_n$Hours_Per_Week)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
 0.0000  0.4040  0.4040  0.4084  0.4545  1.0000 
```

Figure 15- Summary of Normalized dataset

We next create labels for the training and testing sets. When creating training and testing datasets, it is recommended that the training set should be the majority dataset with a smaller portion of data set aside for the testing set. In our case, the training set compromises of 80% of the energy efficiency dataset with 20% allocated to the testing set. We can use these in our kNN prediction algorithm as seen in Figure 16.

```
energy_efficiency_test_pred <- knn(train = energy_efficiency_train, test =
                                energy_efficiency_test, cl = energy_efficiency_train_labels, k=1)
```

Figure 16- kNN prediction model

We set the train to our training set, the test to the testing set and cl to our training labels set in our kNN prediction model. We then run the algorithm and represent it in a CrossTable which displays the cross tabulation of predicted versus actual values. We represent the values in the CrossTable to evaluate the performance of the model.

energy_efficiency_test_labels	Row Total	energy_efficiency_test_pred 0.62	0.64	0.66	0.69	0.71	0.74	0.76	0.79	0.82	0.86	0.9	0.98
0.62	24	24	0	0	0	0	0	0	0	0	0	0	0
	0.090	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
		1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
		0.090	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
0.64	24	0	24	0	0	0	0	0	0	0	0	0	0
	0.090	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
		0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
		0.000	0.090	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
0.66	24	0	0	24	0	0	0	0	0	0	0	0	0
	0.090	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
		0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
		0.000	0.000	0.090	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
0.69	24	0	0	0	24	0	0	0	0	0	0	0	0
	0.090	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
		0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
		0.000	0.000	0.000	0.090	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
0.74	24	0	0	0	0	0	24	0	0	0	0	0	0
	0.090	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000
		0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000
		0.000	0.000	0.000	0.000	0.000	0.090	0.000	0.000	0.000	0.000	0.000	0.000
0.76	24	0	0	0	0	0	0	24	0	0	0	0	0
	0.090	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000
		0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000
		0.000	0.000	0.000	0.000	0.000	0.000	0.090	0.000	0.000	0.000	0.000	0.000
0.79	20	0	0	0	0	0	0	0	20	0	0	0	0
	0.075	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000
		0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000
		0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.075	0.000	0.000	0.000	0.000
0.82	20	0	0	0	0	0	0	0	0	20	0	0	0
	0.075	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000
		0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000
		0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.075	0.000	0.000	0.000
0.86	20	0	0	0	0	0	0	0	0	0	20	0	0
	0.075	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000
		0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000
		0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.075	0.000	0.000
0.9	20	0	0	0	0	0	0	0	0	0	0	20	0
	0.075	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000
		0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000
		0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.075	0.000
0.98	20	0	0	0	0	0	0	0	0	0	0	0	20
	0.075	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000
		0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000
		0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.075
Column Total	268	24	24	24	24	24	24	24	20	20	20	20	20
		0.090	0.090	0.090	0.090	0.090	0.090	0.090	0.075	0.075	0.075	0.075	0.075

Figure 17- CrossTable of Energy Efficiency prediction model

In the case of this dataset, the predicted value relies on 8 other values for its prediction, so this will result in a larger CrossTable. As the CrossTable is quite large, it can be difficult to read, however there are some attributes we can pick out from it. Firstly, when looking at the first value, 0.62, we can see the predicted percentages for that prediction. We can see that 0.62 was predicted 24 times during the duration of the prediction model. In the last row of the CrossTable we can see the Column Totals; with 7 columns having 24 predictions and 5 columns having 20 predictions.

When working with kNN, it can be beneficial to change the value of k in the prediction model, this allows us to see the changes that the value of k will influence. I decided to change the value of k from 1 to 3 and I compared the changes. If we just compare the Column totals for the CrossTable of both values of k for easiness, we can instantly see some changes.

```
-----------------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------
|-----------|
|        20 |           0.86 |         0 |         0 |         0 |         0 |         0 |         0 |         0 |         0 |         0 |        20 |         0 |         0
|     0.075 |                |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     1.000 |     0.000 |     0.000
|           |                |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     1.000 |     0.000 |     0.000
|           |                |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.075 |     0.000 |     0.000
-----------------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------
|-----------|
|        20 |            0.9 |         0 |         0 |         0 |         0 |         0 |         0 |         0 |         0 |         0 |         0 |        20 |         0
|     0.075 |                |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     1.000 |     0.000
|           |                |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     1.000 |     0.000
|           |                |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.075 |     0.000
-----------------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------
|-----------|
|        20 |           0.98 |         0 |         0 |         0 |         0 |         0 |         0 |         0 |         0 |         0 |         0 |         0 |        20
|     0.075 |                |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     1.000
|           |                |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     1.000
|           |                |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.075
-----------------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------
|-----------|
|       268 |   Column Total |        24 |        24 |        24 |        24 |        24 |        24 |        24 |        20 |        20 |        20 |        20 |        20
|           |                |     0.090 |     0.090 |     0.090 |     0.090 |     0.090 |     0.090 |     0.090 |     0.075 |     0.075 |     0.075 |     0.075 |     0.075
-----------------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------
```

Figure 18- Column Totals for K=1

```
-----------------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------
|-----------|
|        20 |           0.86 |         0 |         0 |         0 |         0 |         0 |         0 |         0 |         0 |         0 |        20 |         0 |         0
|     0.075 |                |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     1.000 |     0.000 |     0.000
|           |                |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     1.000 |     0.000 |     0.000
|           |                |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.075 |     0.000 |     0.000
-----------------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------
|-----------|
|        20 |            0.9 |         0 |         0 |         0 |         0 |         0 |         0 |         0 |         0 |         0 |         0 |        20 |         0
|     0.075 |                |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     1.000 |     0.000
|           |                |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     1.000 |     0.000
|           |                |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.075 |     0.000
-----------------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------
|-----------|
|        20 |           0.98 |         0 |         0 |         0 |         0 |         0 |         0 |         0 |         0 |         0 |         0 |         0 |        20
|     0.075 |                |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     1.000
|           |                |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     1.000
|           |                |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.075
-----------------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------
|-----------|
|       268 |   Column Total |        27 |        25 |        21 |        24 |        16 |        31 |        24 |        20 |        20 |        20 |        20 |        20
|           |                |     0.101 |     0.093 |     0.078 |     0.090 |     0.060 |     0.116 |     0.090 |     0.075 |     0.075 |     0.075 |     0.075 |     0.075
-----------------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------
```

Figure 19- Column Totals for K=3

We can see that the number of times that an actual value was predicted has changed. For k=1, the first value in the CrossTable; 0.62 was predicted 24 times whereas in Figure 19, for k=3, we can see that 0.62 was predicted 27 times. We can also see that the accuracy as slightly increased by increasing the value of k. A small value of k means that noise will have a higher influence on the result we are given.

Conclusion

In this report, I have shown how Classification, Regression and kNN Data Mining techniques can be applied to different datasets. In Classification I have shown the benefits of a Decision Tree model, in Regression I have applied Linear Regression and shown how it can be used to predict Heating and Cooling loads on an Energy Efficiency dataset. Finally, I displayed how kNN can be used to create a CrossTable of predicted versus actual results to evaluate the model's performance.