MIDTERM REPORT

- **By Team6**

(Himaja Vadaga, Sandeep Kumar Bethi, Bryce Brako)

Introduction

Three problems were provided with specific instructions for each dataset.

Problem1: Data for Credit Card Defaults
https://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients

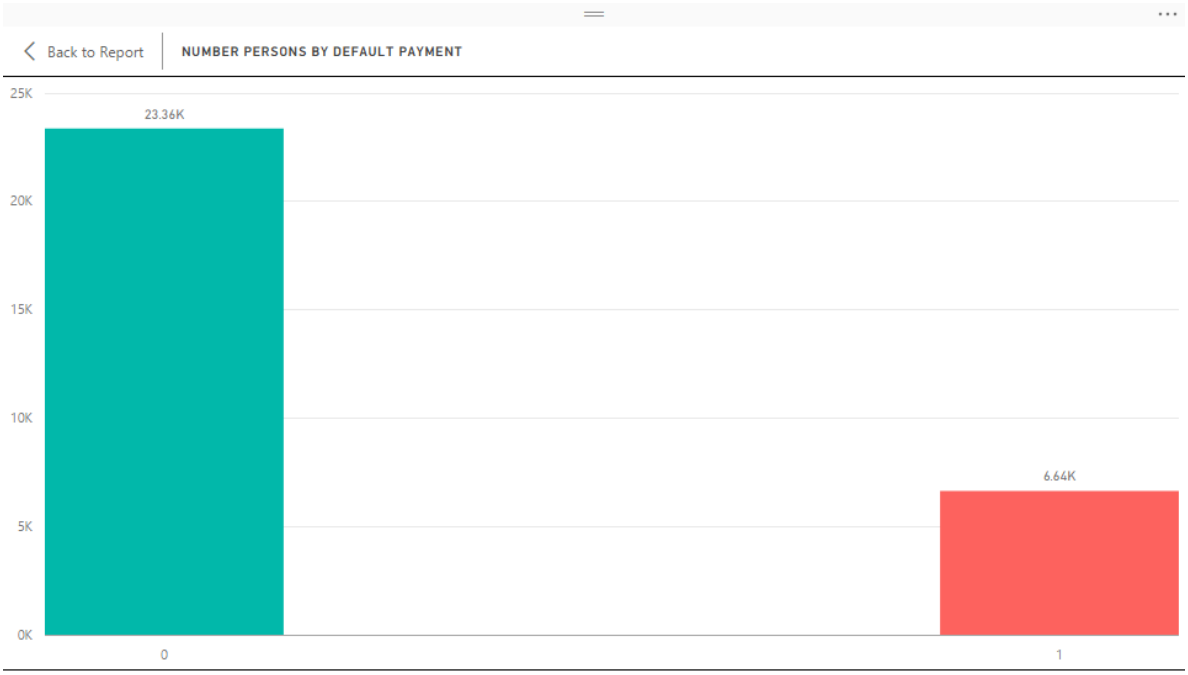
Problem2: Data for possible advertisements on Internet Pages
http://archive.ics.uci.edu/ml/machine-learning-databases/internet_ads/

Problem3: Predicting Hourly power generation up to 48 hours ahead at 7 wind farms
https://www.kaggle.com/c/GEF2012-wind-forecasting/data

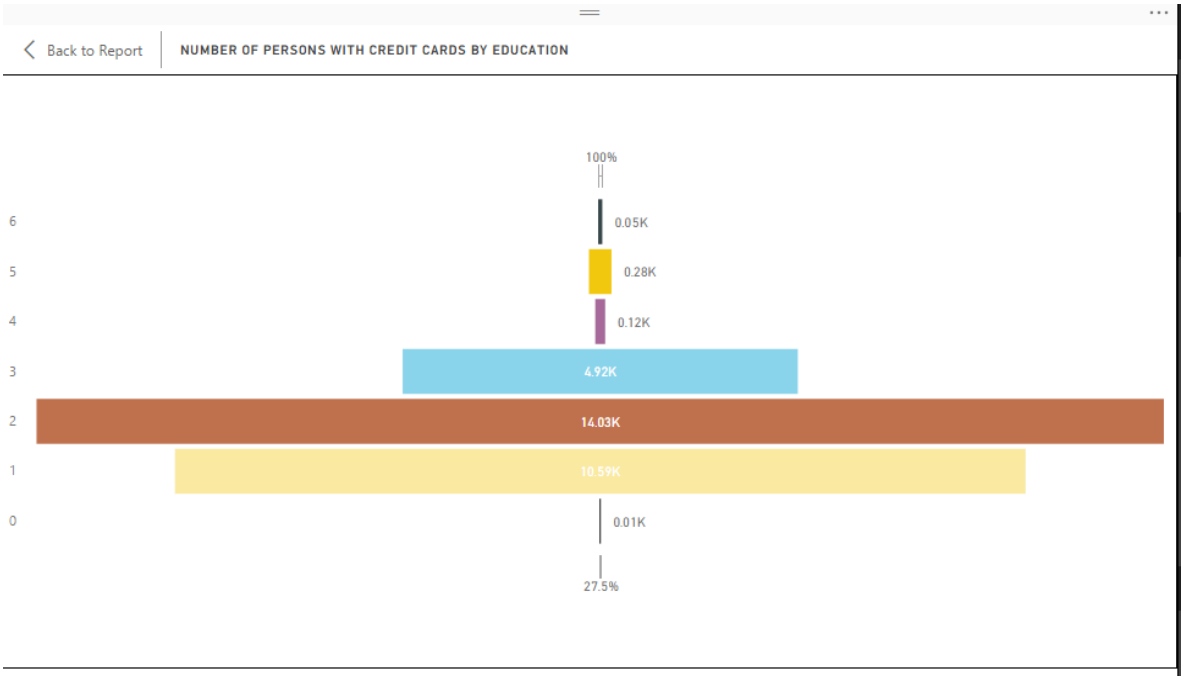
This document provides the observations we inferred out of analyzing the data and performing the instructed operations.

Problem1:

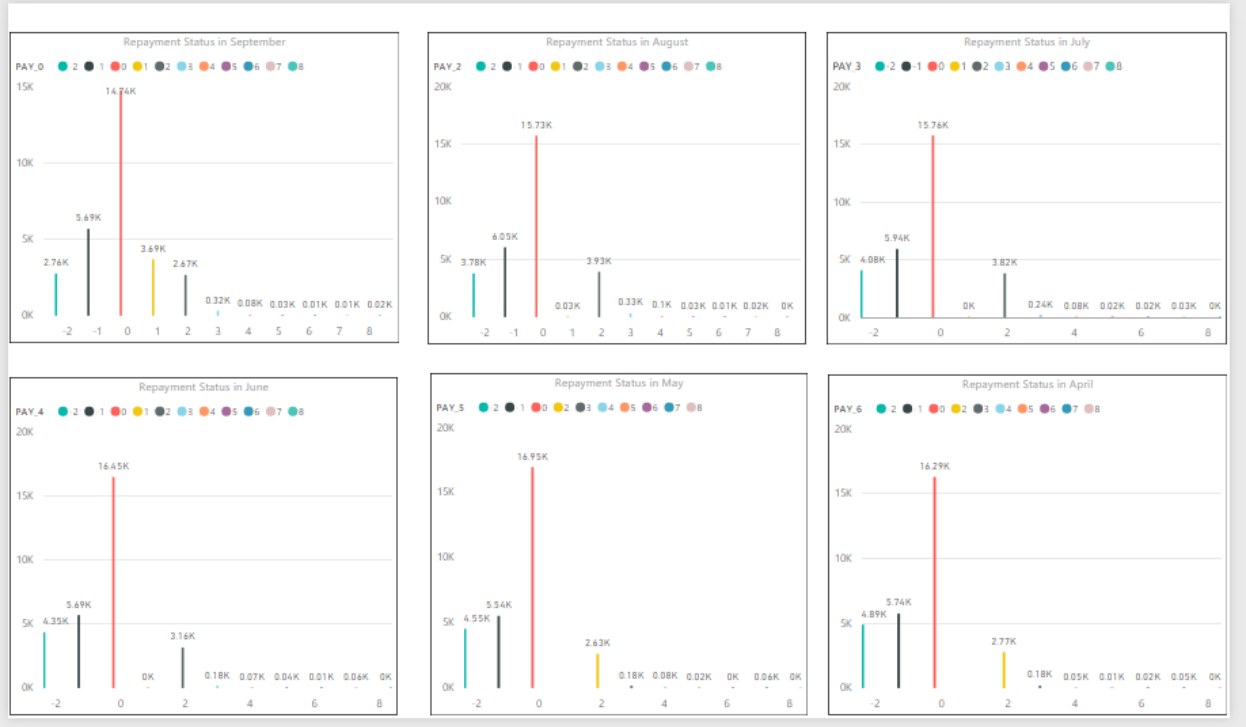
1. **POWER BI:**
   a. The data set was obtained in .xls format, had to convert it to xlsx and load it on PowerBI.
   b. Change the data types of most of the columns as it was wrongly specified as text. Changed it to whole number
2. **Graphs:**
   a. The graph below displays the number of persons that are defaulters or not.



   b. Displays the number of persons who have credit cards by education, which can be used to analyze part of a reason of why a person could be a defaulter.



c. On creating the following graph, it was easy to understand the different repayment statuses which are more than specified in the data set information. The additional statuses are -2 and 0 which do not have any description. But these status have a significance of -2 being no consumption and 0 being the use of revolving credit in the dataset and will not be removed.



3. Cleansing:

a. The file was obtained in xls format. Converted the format to .csv and then loaded it into R.

	X	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	X11	X12	X13	X14	X15	X16	X17	X18	X19	X20	X21	X22
1	ID	LIMIT_BAL	SEX	EDUCATION	MARRIAGE	AGE	PAY_0	PAY_2	PAY_3	PAY_4	PAY_5	PAY_6	BILL_AMT1	BILL_AMT2	BILL_AMT3	BILL_AMT4	BILL_AMT5	BILL_AMT6	PAY_AMT1	PAY_AMT2	PAY_AMT3	PAY_AMT4	PAY_AM
2	1	20000	2	2	1	24	2	2	-1	-1	-2	-2	3913	3102	689	0	0	0	0	689	0	0	0
3	2	120000	2	2	2	26	-1	2	0	0	0	2	2682	1725	2682	3272	3455	3261	0	1000	1000	1000	0

On loading the dataset into R, the column names were not human readable and a reference to the 1st row had to be used to understand what the value of the column is. Thus used the first row as the column names renamed the columns.
In order to achieve this used the rename function of the PLYR library.

b. There were around 795 rows which had the Bill amounts and the payment amount values for all the months as 0. So, deleting the rows to make sure an efficient model is built.

4. Classification using Logistic Regression:

a. Divided the Dataset into 75-25 % to build the training and testing dataset. It is the ideal division in order to develop a model on the training dataset and to run it on the test dataset.
b. On the training dataset, we used the glm function to build the dataset as glm is the ideal function to fit the logistic regression model or a dependent variable which has categorical values.
c. The screenshot below shows the features used in order to build this model.

```
Call:
glm(formula = Default_Payment ~ LIMIT_BAL + SEX + EDUCATION +
    MARRIAGE + AGE + repay_status_Sept + repay_status_Aug + repay_status_July +
    bill_amt_Sept + PAY_Amt_Sept + PAY_Amt_Aug, family = binomial(link = logit),
    data = train_ccd)

Deviance Residuals:
      Min         1Q     Median         3Q        Max
-3.2240041  -0.6805754  -0.5336576  -0.2860698   3.1627403

Coefficients:
                       Estimate       Std. Error  z value             Pr(>|z|)
(Intercept)       -0.6544680080780  0.1409183767477 -4.64431    0.00000341222147271 ***
LIMIT_BAL         -0.0000013434162  0.0000001844755 -7.28235    0.00000000000032805 ***
SEX               -0.0954405312374  0.0365244144262 -2.61306              0.00897352 **
EDUCATION         -0.1097534566677  0.0249940808917 -4.39118    0.00001127382290204 ***
MARRIAGE          -0.1699471585220  0.0376256309885 -4.51679    0.00000627834462239 ***
AGE                0.0056108799812  0.0021084743154  2.66111              0.00778837 **
repay_status_Sept  0.5784418018856  0.0219494340298 26.35338 < 0.000000000000000222 ***
repay_status_Aug   0.1108325523075  0.0241552223782  4.58835    0.00000446768667215 ***
repay_status_July  0.1454186895030  0.0218828643833  6.64532    0.00000000003025540 ***
bill_amt_Sept     -0.0000015258932  0.0000003073862 -4.96409    0.00000069023344037 ***
PAY_Amt_Sept      -0.0000075000977  0.0000020780338 -3.60923              0.00030711 ***
PAY_Amt_Aug       -0.0000086721775  0.0000021314459 -4.06868    0.00004727961934051 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 22749.015  on 21902  degrees of freedom
Residual deviance: 19848.158  on 21891  degrees of freedom
AIC: 19872.158

Number of Fisher Scoring iterations: 6
```

d. The following screenshot displays the confusion matrix and its statistics by which we can also calculate the overall error. 1 represents “Defaulter”, 0 represents “Not defaulter”

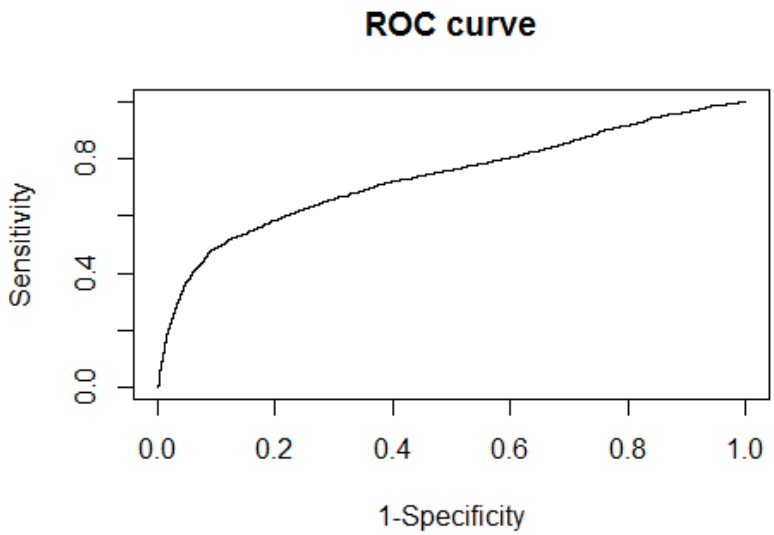
The overall Error rate is (1240+147) / (5507+147+1240+408) * 100 = 18%

Non-Defaulter Error Rate (0) = 147/5507+147 * 100 = 2%

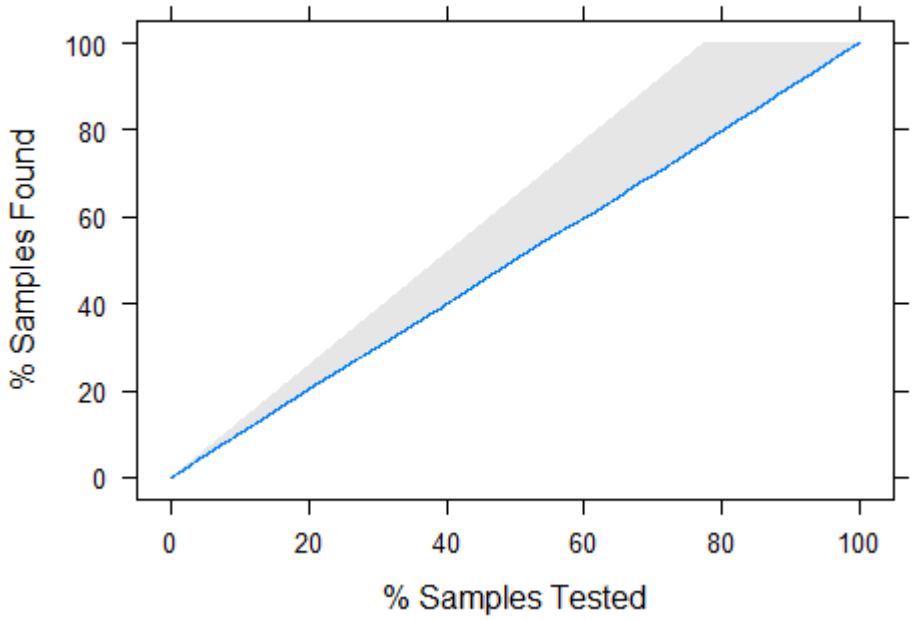
Defaulter Error Rate (1) = 1280/1280+408 * 100 = 75%

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 5507  147
         1 1240  408

               Accuracy : 0.810052
                 95% CI : (0.8008634, 0.8189925)
    No Information Rate : 0.9239934
    P-Value [Acc > NIR] : 1

                  Kappa : 0.2896223
 Mcnemar's Test P-Value : <0.0000000000000002

            Sensitivity : 0.8162146
            Specificity : 0.7351351
         Pos Pred Value : 0.9740007
         Neg Pred Value : 0.2475728
             Prevalence : 0.9239934
         Detection Rate : 0.7541769
   Detection Prevalence : 0.7743084
      Balanced Accuracy : 0.7756749

       'Positive' Class : 0
```

e. ROC curve



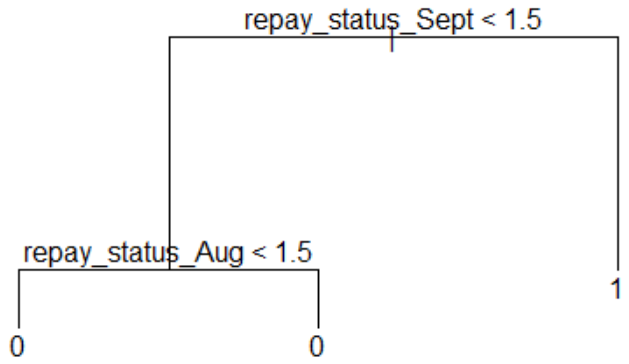
f. LIFT chart



5. Classification using Regression Trees:

a. Fitting the classification tree

```
Classification tree:
tree(formula = Default_Payment ~ repay_status_Sept + repay_status_Aug,
    data = credit_card_defaulters)
Number of terminal nodes:  3
Residual mean deviance:  0.8827926 = 25779.31 / 29202
Misclassification error rate: 0.1750728 = 5113 / 29205
```



b. Overall Error rate and Confusion Matrix

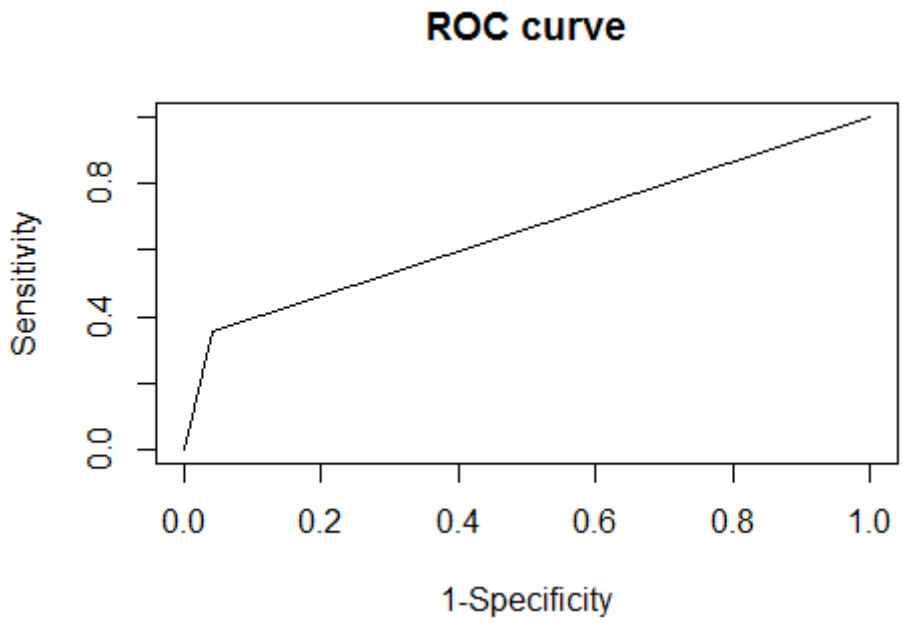
tree.pred	Default_Payment.test 0	1
0	10980	2032
1	475	1116

Overall Error Rate = 17%

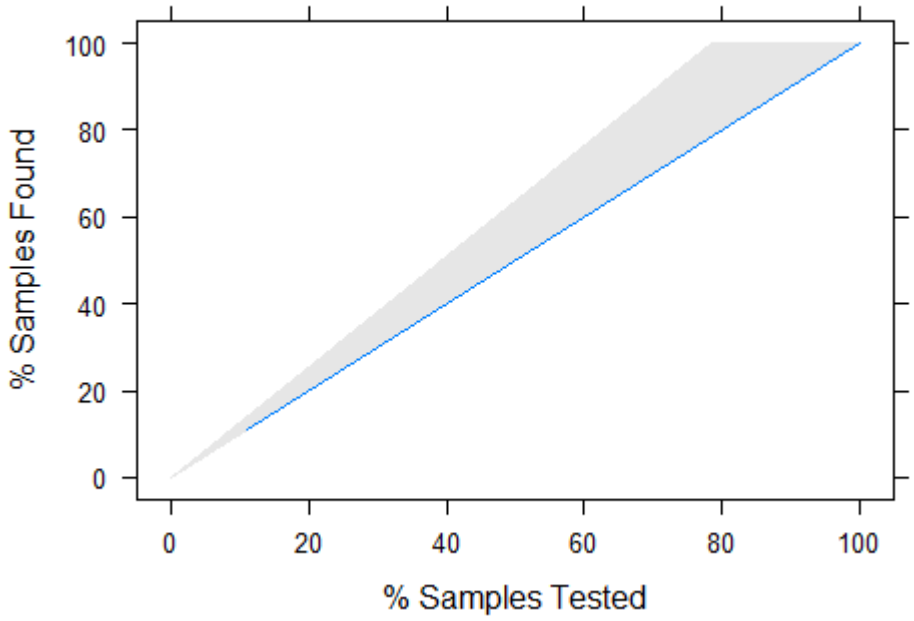
Non-Defaulter Error Rate (0) = 2032/10980+2032 * 100 = 15%

Defaulter Error Rate (1) = 475/1116+475 * 100 = 42%

c. ROC Curve



d. LIFT chart



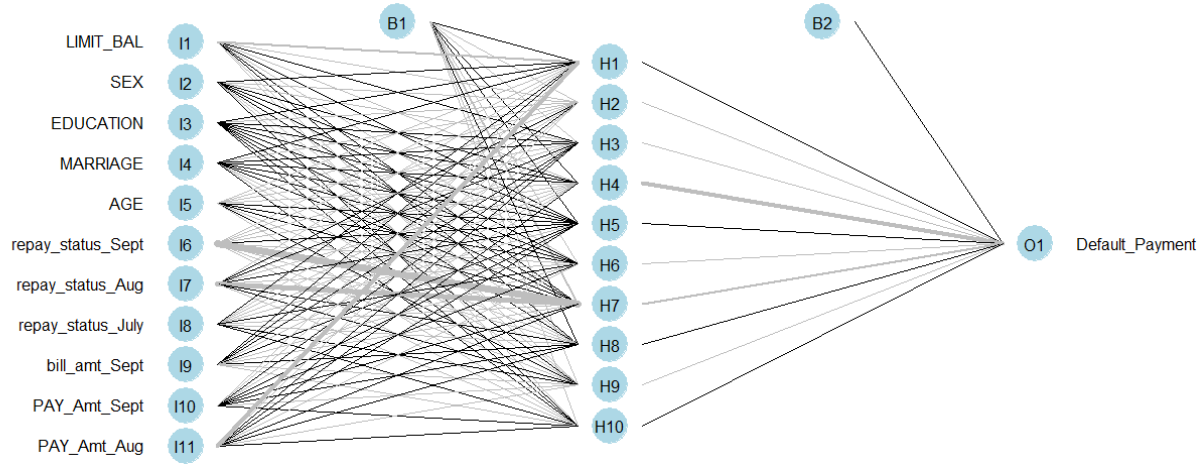
6. Classification using Artificial Neural Networks

a. Divided the dataset into training and test with 70-30 split.
b. Used the nnet function to perform the classification operations
c. Set the size of the hidden layers to 10 to improve the accuracy as anything lower than that is giving a prediction of either a 0 or a 1 which is incorrect.
d. Added additional attributes to the nnet function rang=0.03, HESS=FALSE,decay=15e-4,maxit=250 to build a good model.
e. On using the test data to test the trained model, we get the following confusion matrix

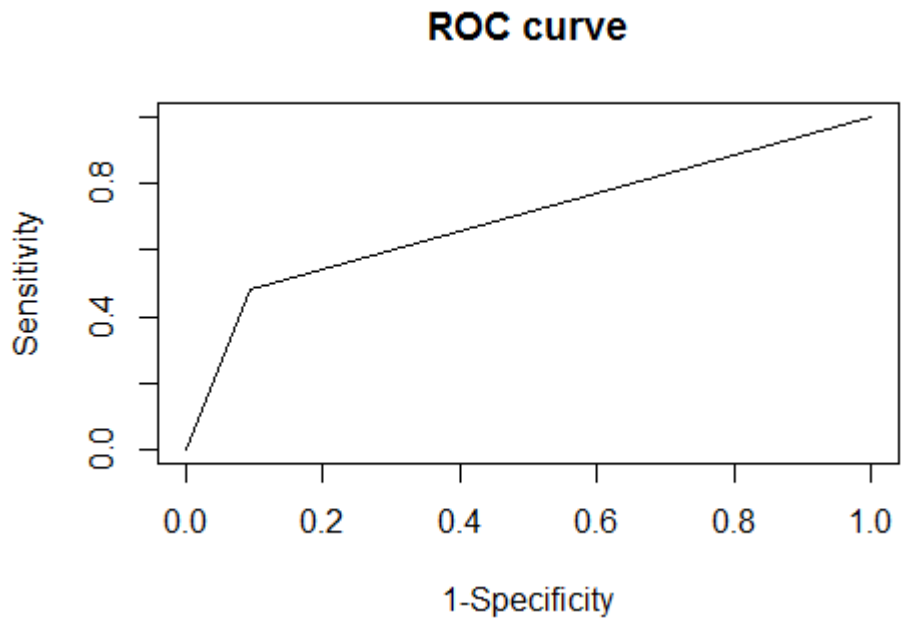
```
> table(test_ccd$Default_Payment,test.nnet)
   test.nnet
       0    1
  0 6460  371
  1 1180  751
```

Overall error Rate = (1180+371) / (6460+371+1180+751) * 100 = 17%
Non-Defaulter Error Rate (0) = (371) / (6460+371) * 100 = 5%
Defaulter Error Rate (1) = (1180) / (1180+751) * 100 = 61%

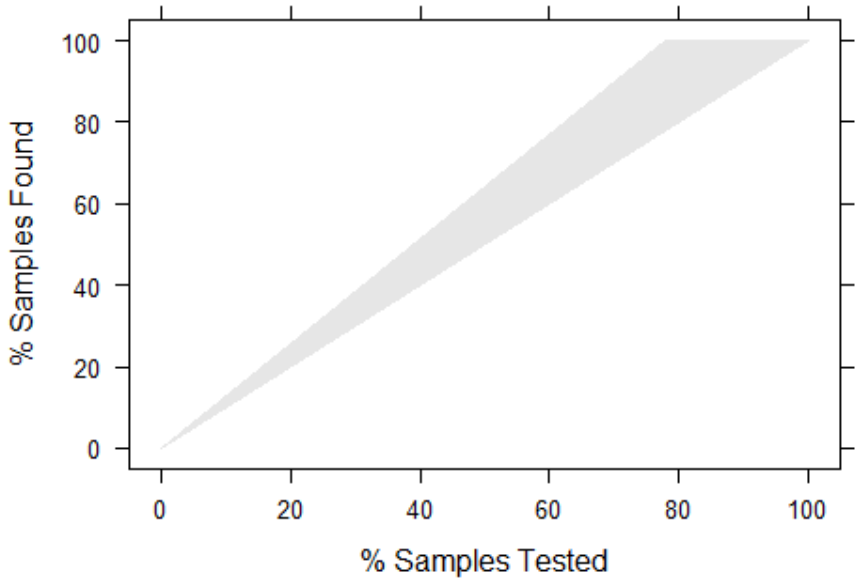
f. Plotting the neural network



g. ROC curve



h. LIFT Chart



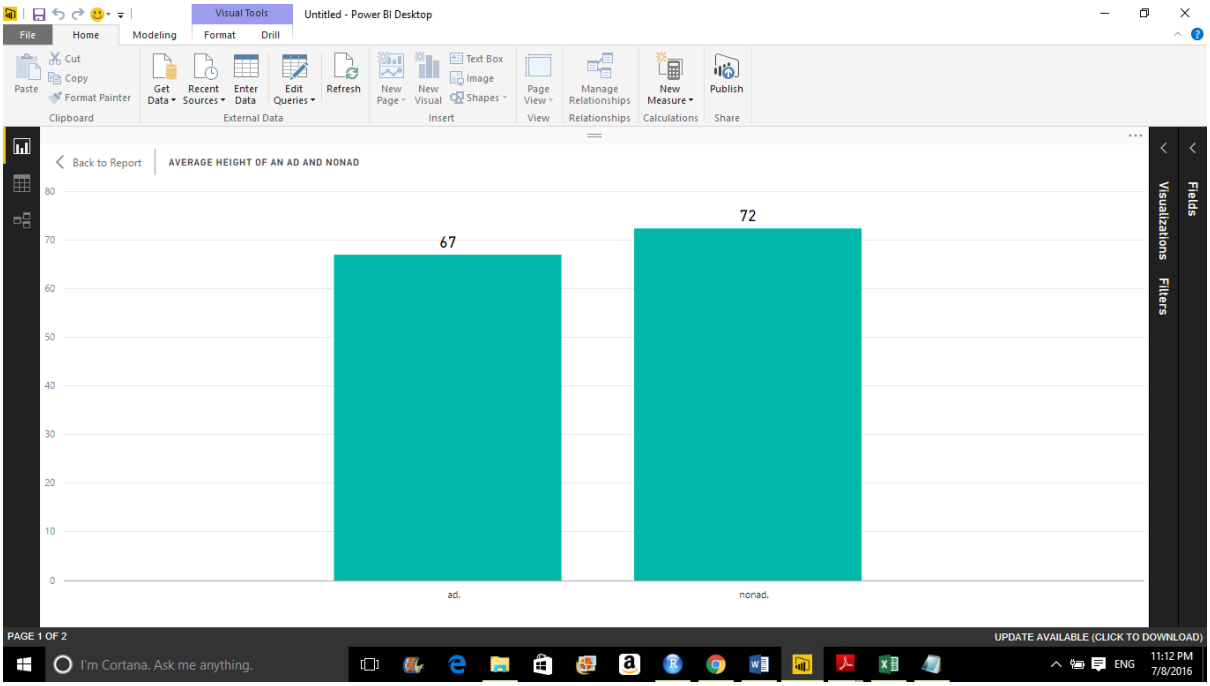
7. FINAL CONCLUSION:

On Calculating the Overall Error Rate for all the tree model, based on the classification tree having the better error rate calculations compared to the logistic regression and neural network model we would choose the Classification tree model.

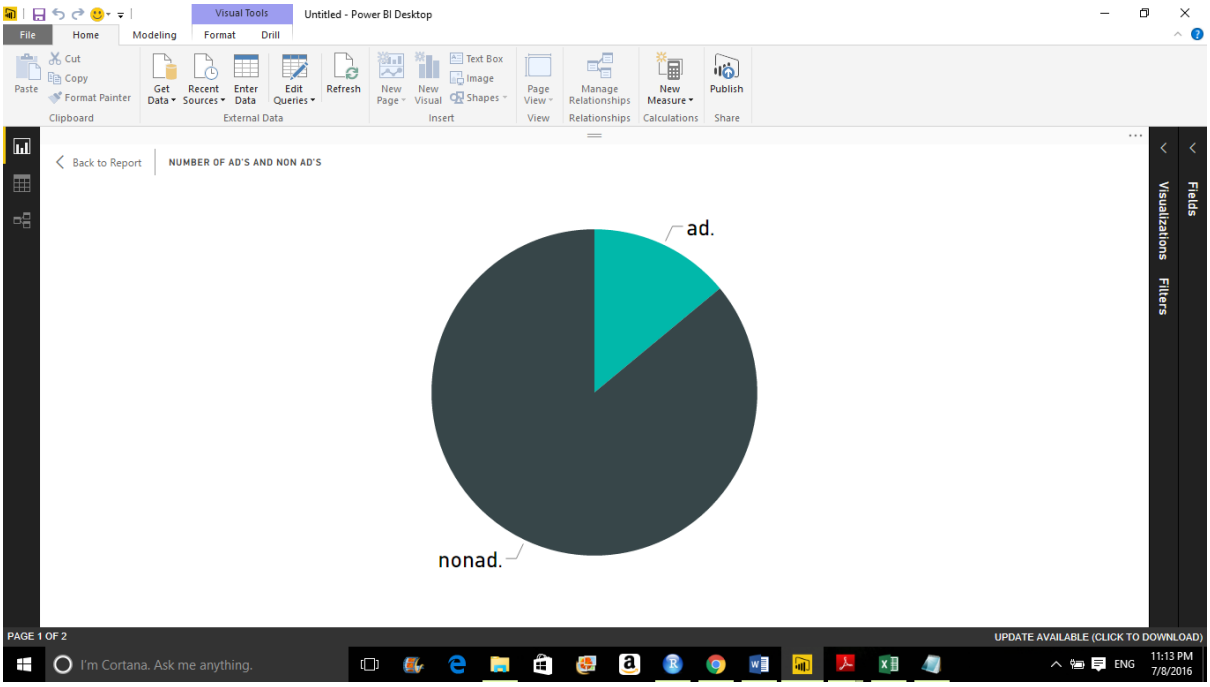
Problem2:

1. **POWER BI:**
   a. These Visualizations give information regarding the geometrical aspects of Internet Advertisements such as Height, Width, Aspect Ratio. It also gives information regarding number of Ad's deployed in local and non-local servers.
   b. **Note:** These Visualizations were done after replacing the missing values in continuos data i.e in Height, width and aspect ratio features.

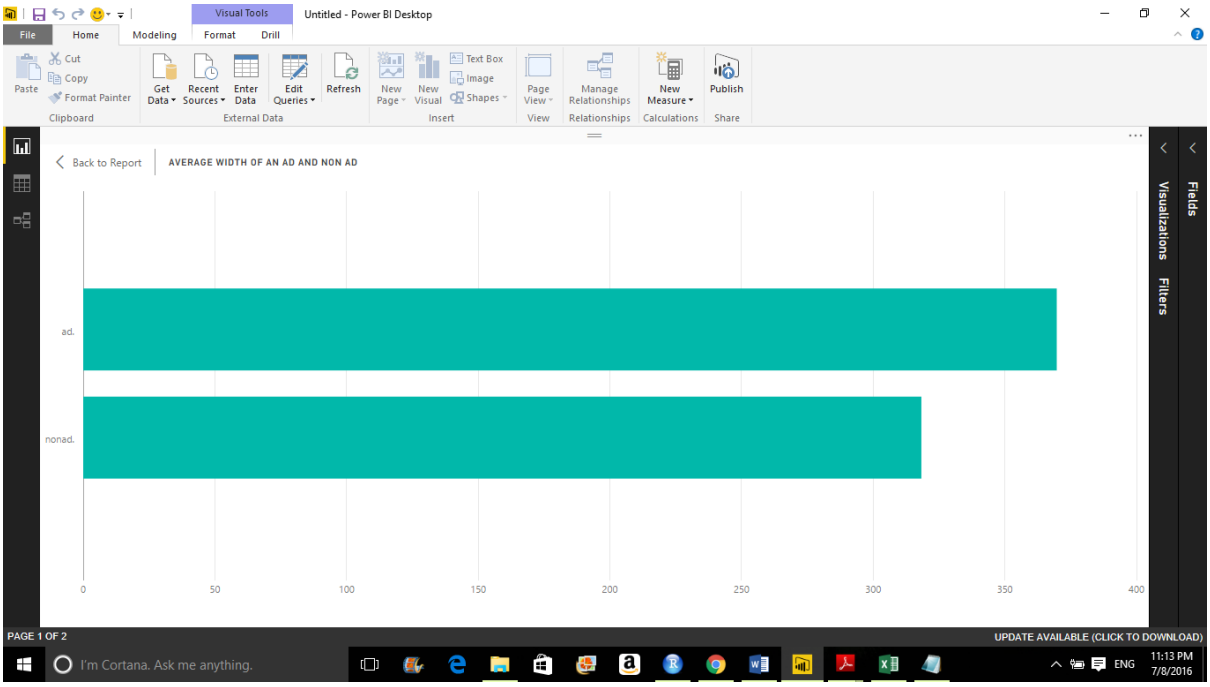
2. **Graphs:**



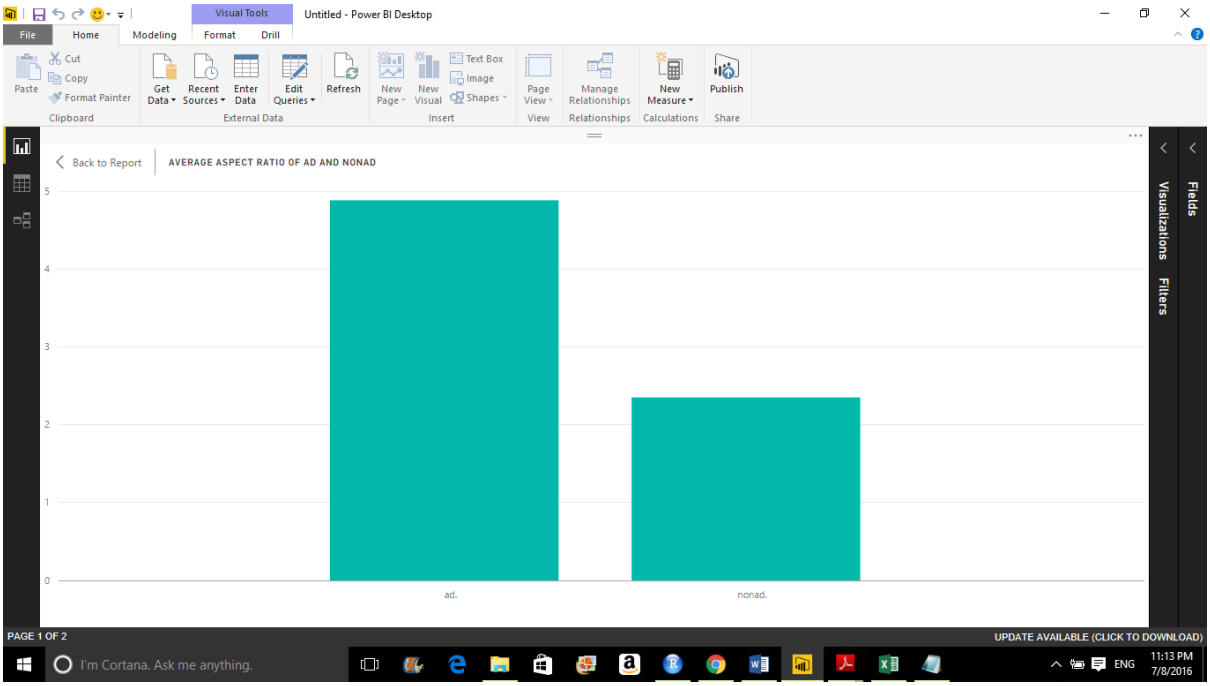
The above Visualization shows the average height of an Ad and Non Ad reported in the dataset



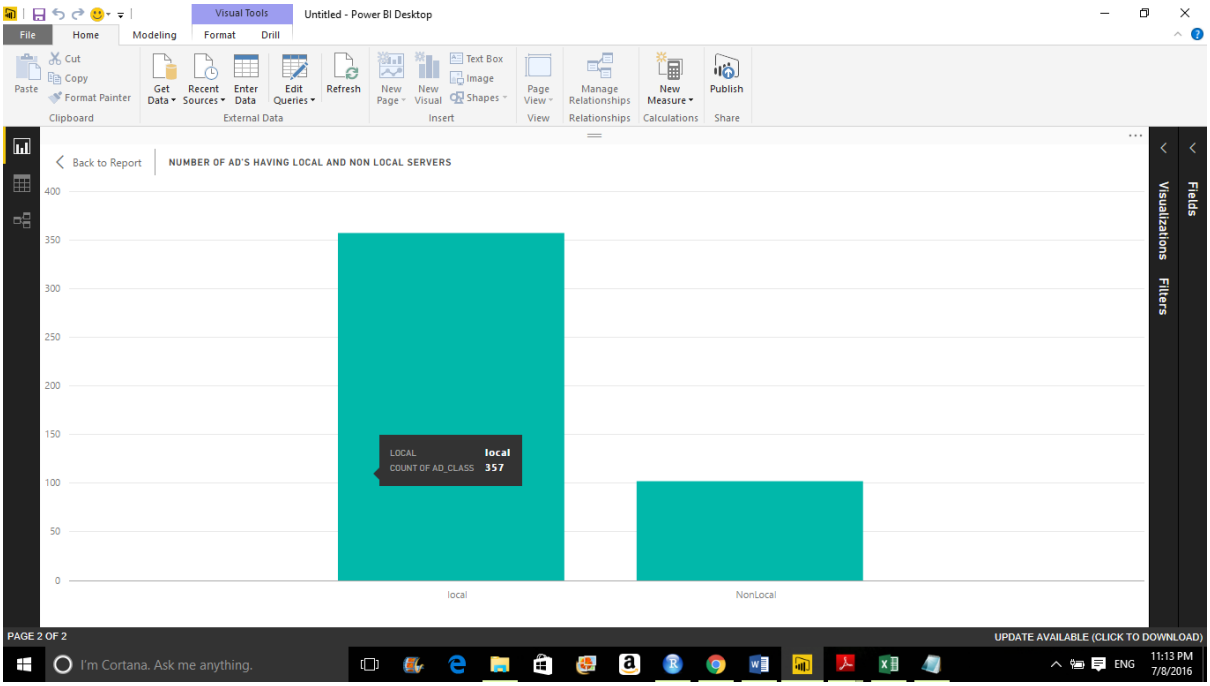
The above visualization shows the number of AD's and NonAd's



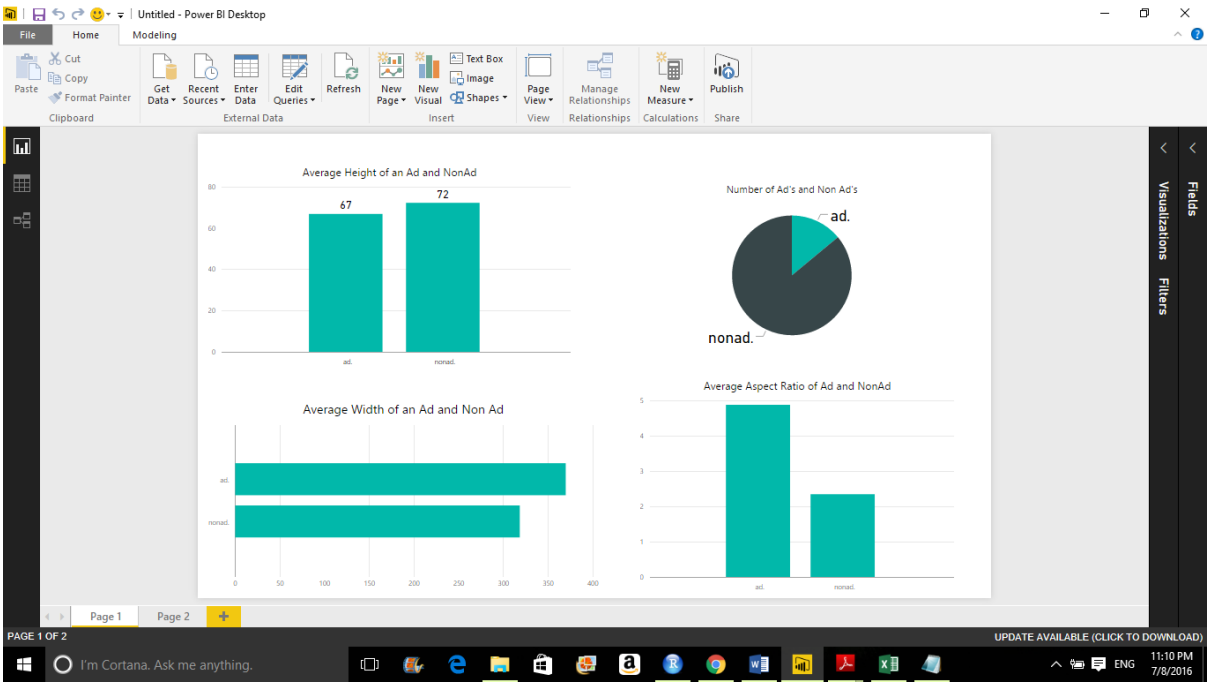
The above visualization shows the Average Width of Ad's and Non Ad's



The above visualization shows the average aspect ratio of Ad's and Non AD's



The above visualization shows the number of Ad's which are in local and non-local servers



The above picture is a dashboard of all the visualizations together

3. Cleansing:

a. Challenges and Overcoming them:

- Replacing the "?" values by NA in continuous data i.e. for the first three columns
  - ✓ The data type of the continuous variables are given in factor format
  - ✓ We converted the factor format of these continuous variables to character and later converted the character format to numeric
  - ✓ By doing this the values such as "?" are automatically converted to NA's
  - ✓ By the end of this step we had few NA's in the first three columns i.e. Height, Width and Aspect Ratio

- Estimating and Replacing the outliers for continuous data by NA's
  - ✓ To estimate the outliers of a particular data the domain knowledge of that particular data is required
  - ✓ Here we need to estimate the outliers for Height, Width and Aspect Ratio of images used in Internet Advertisements
  - ✓ For this we were able to browse and get the information regarding standard sizes of images used in Internet Advertisements
  - ✓ http://www.fileformat.info/tip/web/imagesize.html
  - ✓ The above mentioned link serves as a reference for finding out the standard sizes of images used in Internet Ads
  - ✓ From this information we were able to find out the max and min values that can exist for an image of Internet Advertisement
  - ✓ We noticed that an image can have the following max and min values for Height, Width, Aspect Ratio when used as an internet advertisement

    Height-→ Max=600, Min=30

    Width--→Max=728, Min=88

Note: Aspect Ratio = Width/Height
Hence We can find out the Max and Min Values of Aspect Ratio by above formula

Aspect Ratio→ Max=24.26, Min=0.14

- ✓ After we were able to get the above ranges for Height, Width and Aspect Ratio then we wrote an algorithm that could detect data which fall outside these ranges as outliers
- ✓ Finally these outliers that fall outside the ranges are replaced by NA's

- Replacing all the NA's by appropriate values in continuous data (we are not simply deleting the records as suggested in the pdf)
  - ✓ Now we have an accumulated set of NA values from step (a) and step (b)
  - ✓ We assumed that replacing these values of NA by Mode of the observations would be more appropriate and meaningful than replacing them with Mean of the observations.
  - ✓ We made the above assumption because each record represents different features of an advertisement and each record is independent of the other.
  - ✓ Since this is not a time series data , we thought making a mean or a median wouldn't make much of a sense
  - ✓ As this data represents the observations of a set of AD's and Non AD's we thought going for a Mode would be better option.
  - ✓ By choosing Mode the NA's in Height would be replaced by frequently occurring heights of images and similarly for NA's in width would be replaced by frequently occurring widths of images.
  - ✓ There was no inbuilt function like mean or median for calculating the mode. Hence we created a function on our own which could implement the functionality of Mode.
  - ✓ This function is built in such a way that it could handle data of numeric, character and factor format.
  - ✓ It also ignores NA's present while calculating the Mode

- Dimensionality reduction since the number of features for this dataset is huge
  - ✓ This was probably the most biggest challenge we have faced in this problem
  - ✓ The number if attributes or dimensions or features or variables for this dataset is huge. To be precise there are 1559 attributes.
  - ✓ Reducing the size of attributes is important as it would avoid overfitting and also helps in building more accurate classification models which could help in capturing meaning information
  - ✓ As these variables are highly correlated we used PCA to transform these large set of variables to a smaller set of variables

- ✓ We implement PCA by calling a function called preProcess and applying a transformation called Box-Cox in that function
- ✓ For understanding and implementing PCA we made use of the following link or reference
- ✓ http://www.r-bloggers.com/computing-and-visualizing-pca-in-r/
- ✓ We have implemented PCA for clusters of features such as URL, origurl, capture, ALT and Anchor
- ✓ After implementing the PCA in clusters of these , the total number of variables came down to **498** from **1559**. This is a significant decrease in attribute size.
- ✓ **NOTE:** The PCA can be implemented to the entire set of variables insead of applying it to clusters of variables but that wouldn't be an appropriate approach. As the variables are more correlated in clusters it is better to implement PCA in clusters of variables.
- ✓ **NOTE:** We tried implementing the PCA for the entire set of variables then the number of variables reduced to **300 from 1559.**
- ✓ But if we follow the above approach then in the later stages the performance metrics of classification models are not that good compared to the approach where we have done PCA in clusters.
- ✓ Hence we decided to choose the first approach where we implement PCA in clusters of variables such as URL Terms, origurl terms, caption etc.
- ✓ By doing this our final number of attributes in cleansed and pre-processed data came down to **498**-excluding the Ad class variable.

NOTE: To gain the domain knowledge of this particular dataset we have referred this Article called Learning to remove Internet Advertisements by Nicholas Kushmerick.
By studying this article we gained knowledge of how the instances of this dataset are encoded and recorded. We also came to know in what way each feature describes and Advertisement. For eg: the local feature describes whether the URL of anchor tag and URL of image tag are present in the same server. The link for this article is mentioned in the references section at the end.

4. Classification using Logistic Regression

a. Confusion Matrix and error rate

Below are the confusion matrix and error rate of Logisitc Regression classification model

```
       Confusion Matrix and Statistics

          Reference
Prediction ad. Nonad.
    Ad.     23     94
    nonad. 677     26
```

Overall Error rate= ((677+94)/(820))

= 94%

Error rate and accuracy in prediciting each individual class

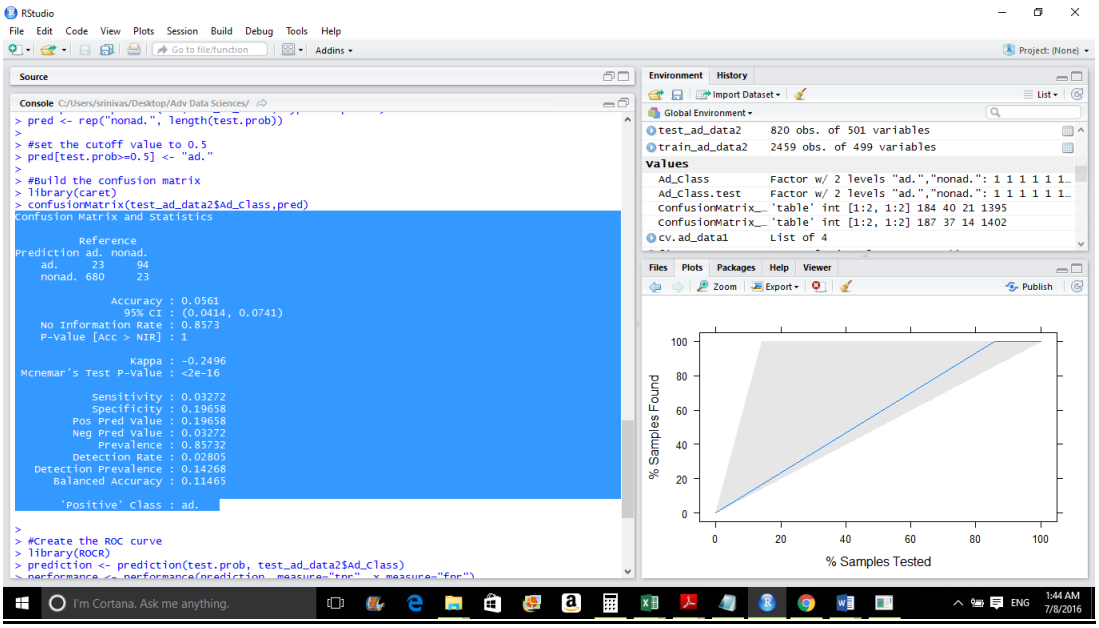
- ✓ Error rate in Predicting Ad class= 94/117= 80.3%
- ✓ Accuracy in Predicting Ad class= 23/117= 19.65%
- ✓ Error rate in Predicitng NonAD class= 680/703= 96.3%
- ✓ Accuracy in Predicitng NonAD class= 26/703= 3.6%

As we can see above the overall error rate is too high for the logistic regression model.

It predicts only 26AD cases correctly as ad and it predicts 94 AD cases wrongly as nonad

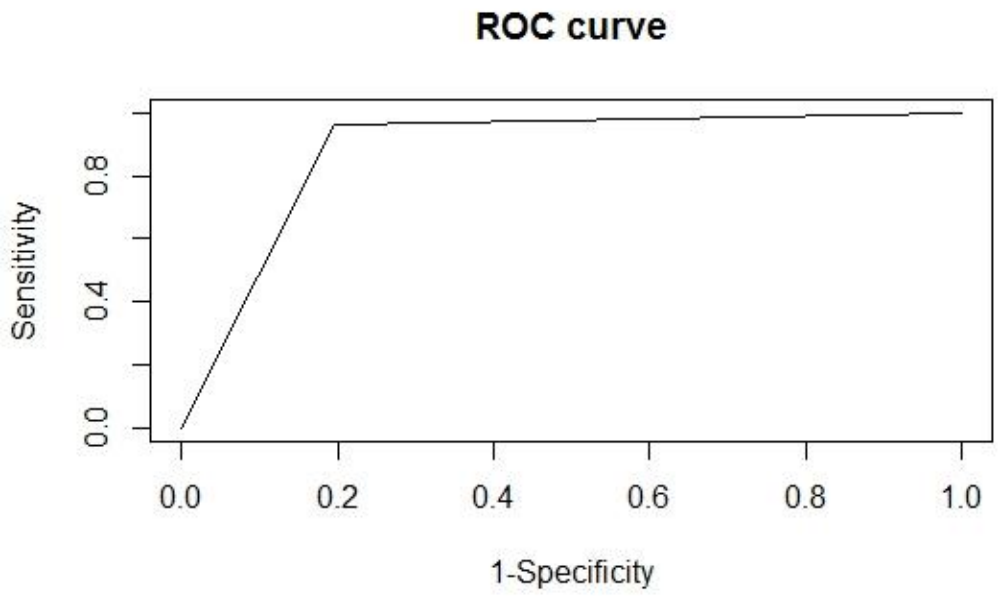
It Predicts only 26 NONAD cases correctly as nonad and it predicts 680 NONAD cases wrongly as ad

Since the accuracy rate of predicting AD class correctly is more than the accuracy of 17redicting NONAD class it is considered as a positive class for ad.



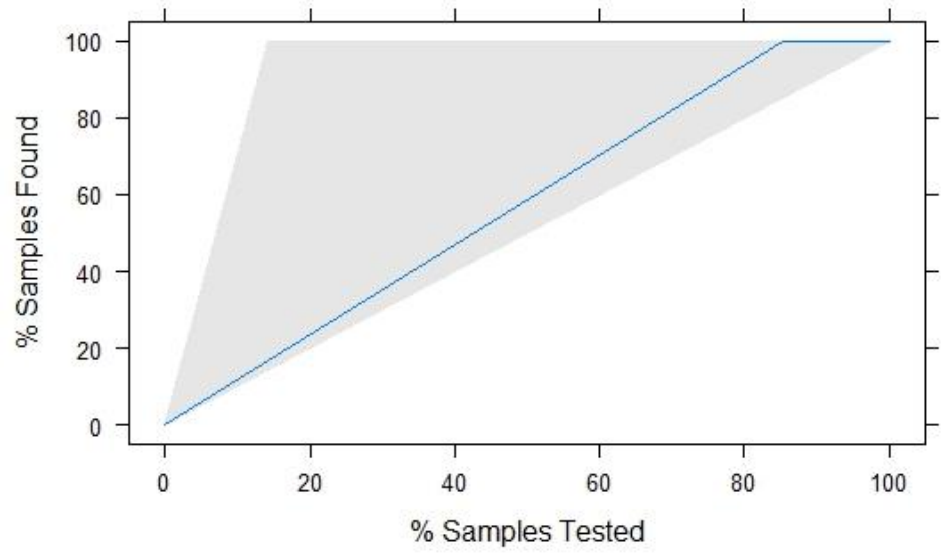
The Blue Highlighted part above shows the confusion matrix and different statistics of logistic regression.

b. ROC CURVE



The above mentioned picture shows the ROC curve for the logistic model built on Internet Advertisements dataset.

c. LIFT CHART



The above mentioned picture shows the Life Chart for the logistic regression classification model

5. Classification using Neural Networks

a. Confusion Matrix and Error Rate

While fitting in the neural network model we should be careful in giving the vakues f or size, weights, MaxNWts parameters. These values vary depending on the size of t he predictors
we are trying to fit. For this particular case where the predictors or variable size is hu ge we
are MaxNWts=5001 and size as 5. The confusion Matrix for neural network is as follo ws

```
ConfusionMatrix_Neural_Network
        test.nnet
         ad. nonad.
  ad.    123     22
  nonad.   9    830
```

Overall Error Rate= ((22+9)/(984))*100

= 3.1

Error rate and accuracy in prediciting each individual class

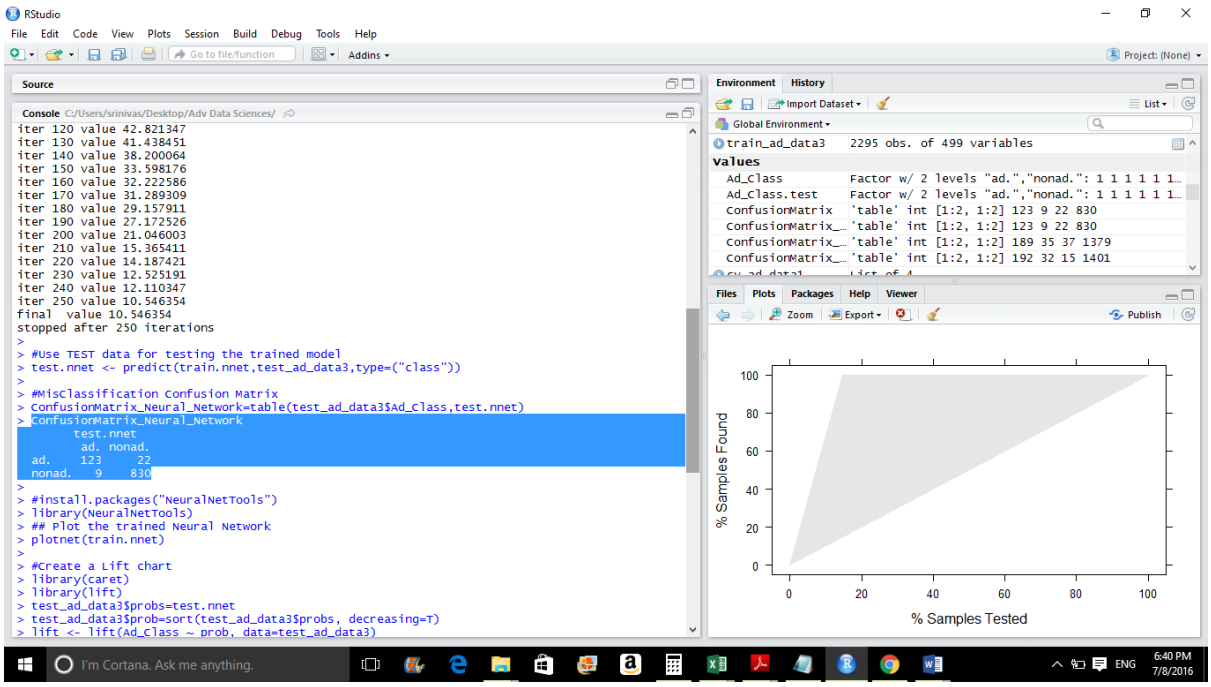
- ✓ Error rate in Predicting Ad class= 22/145= 15.1%
- ✓ Accuracy in Predicting Ad class= 123/145= 84.8%
- ✓ Error rate in Predicitng NonAD class= 9/839= 1%
- ✓ Accuracy in Predicitng NonAD class= 830/839= 98.9%

As we can see above the overall error rate is 3.1% for the neural network model.

It predicts only 123 AD cases correctly as ad and it predicts 22 AD cases wrongly as nonad

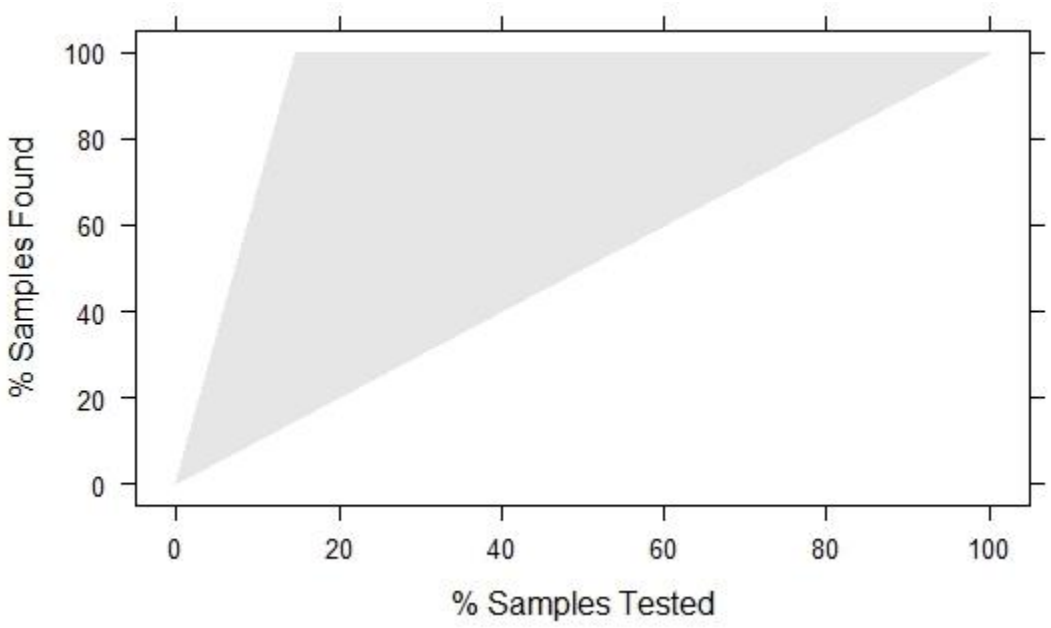
It Predicts only 830 NONAD cases correctly as nonad and it predicts 9 NONAD cases wrongly as ad

This model is more accurate in predicting the NonAD cases.

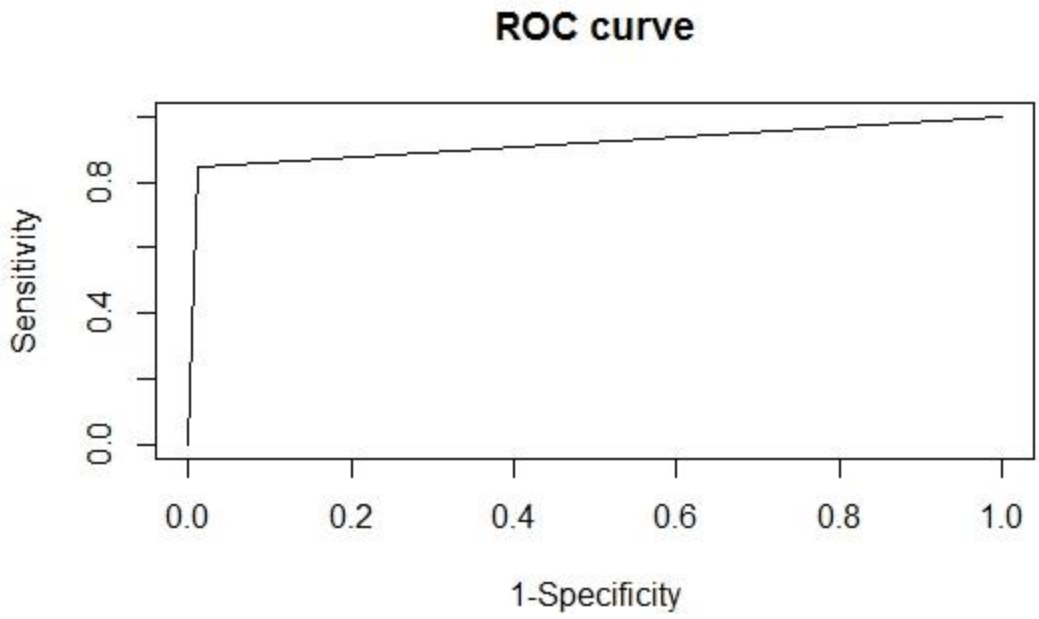


The above highlighted part shows the confusion matrix of neural network model

b. LIFT CHART NEURAL NETWORK



c. ROC CURVE FOR NEURAL NETWORK



6. Classification using Classification trees

a. Tree built without pruning

When the model is built without pruning the tree then the overall error rate is more than the overall error rate of the model which is pruned.

```
ConfusionMatrix_Tree_NotPruined
          Ad_Class.test
tree.pred  ad. nonad.
   ad.     189     37
   nonad.   35   1379
```

As you can see above in the confusion matrix, the model which is not pruined is predicting 189 AD cases correctly as “ad” and it is predicting 37 AD cases wrongly as “nonad”

Similarly it is predicting 1379 NONAD cases correctly as “nonad” and it is predicting 35 NONAD cases wrongly as “ad”

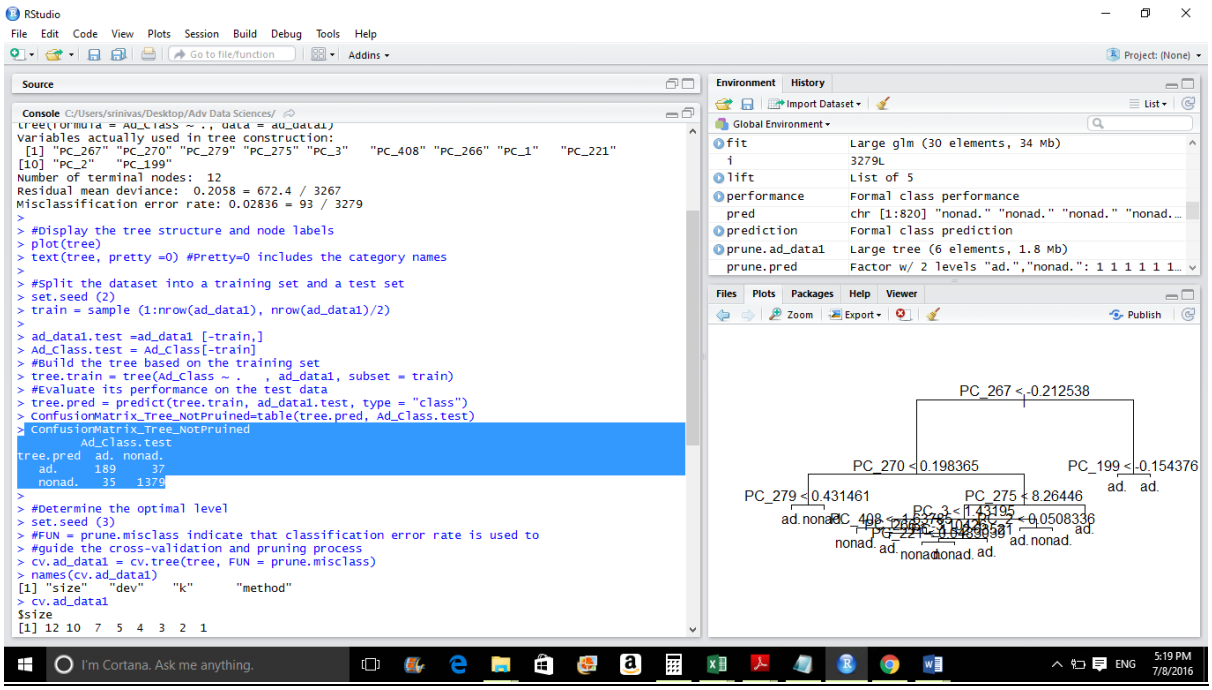
The overall error rate for the tree model which is not pruined= ((35+37)/1640)*100

(a) Overall error rate_TreeNotPruined=4.3%

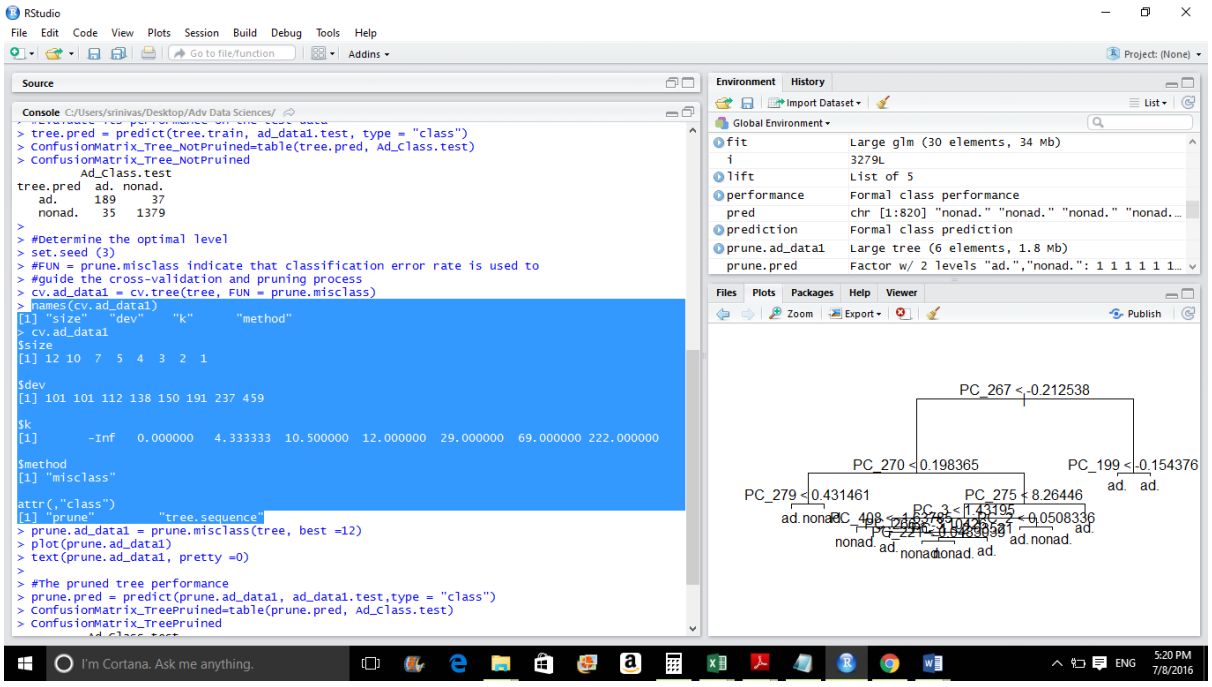
Error rate and accuracy in prediciting each individual class

- ✓ Error rate in Predicting Ad class= 37/226= 16.3%
- ✓ Accuracy in Predicting Ad class= 189/226= 83.6%
- ✓ Error rate in Predicitng NonAD class= 35/1414= 2.4%
- ✓ Accuracy in Predicitng NonAD class= 1379/1414= 97.5%

This tree model which is not pruned is more accurate in predicting NonAD classes.



In the above picture the highlighted blue part shows the confusion matrix of the tree model which is **not pruned**.



The above picture which is highlighted (Cross-Validation) gives the size: number of terminal nodes of each tree considered, k: the value of cost complexity parameter

b. Tree built after Pruning

Note : the number of terminal nodes were 12. Using Cross validation function checked the number of terminal nodes, the value of cost complexity (k)

When the model is built after pruning the tree then the overall error rate has decreased .

```
 ConfusionMatrix_TreePruined
             Ad_Class.test
prune.pred  ad. nonad.
    ad.     192     15
    nonad.   32   1401
```

As you can see above after the tree is pruned the number of nonad cases predicted correctly as nonad cases has increased(1379 to 1401) compared to the previous tree model without pruning.

In the previous tree model without pruning it predicted only 1379 nonad cases correctly as nonad

But whereas in the current model after the tree is pruned 1401 nonad cases are correctly predicted as nonad

Similarly the number of ad cases predicted correctly as ad cases also increased from 189 to 192

Hence this model which is pruned increases the accuracy of prediction of individual classes i.e Ad and NonAd

The overall error rate for the tree model which is pruned= ((32+15)/1640)*100

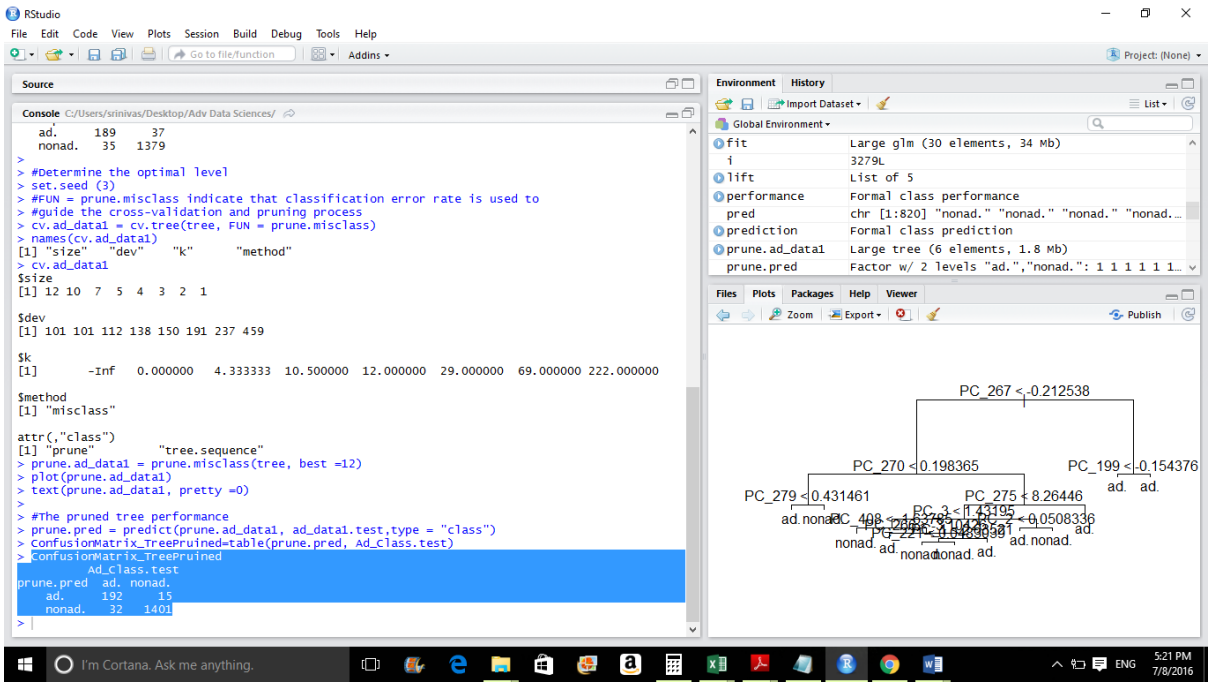
(b) Overall error rate_TreePruined=2.8%

NOTE: The overall error rate has come down after Pruning from 4.3% to 2.8%

Error rate and accuracy in prediciting each individual class

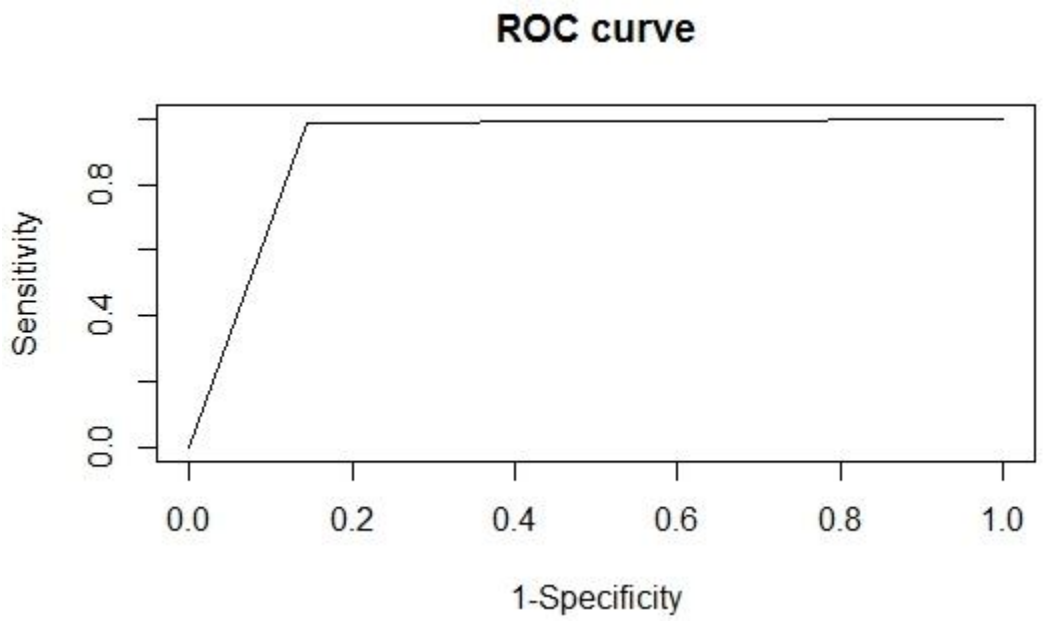
- ✓ Error rate in Predicting Ad class= 14/207= 7.2%
- ✓ Accuracy in Predicting Ad class= 187/207= 92.7%
- ✓ Error rate in Predicitng NonAD class= 32/1433= 2.2%
- ✓ Accuracy in Predicitng NonAD class= 1401/1433= 97.7%

This tree model which is not pruned is more accurate in predicting NonAD classes.

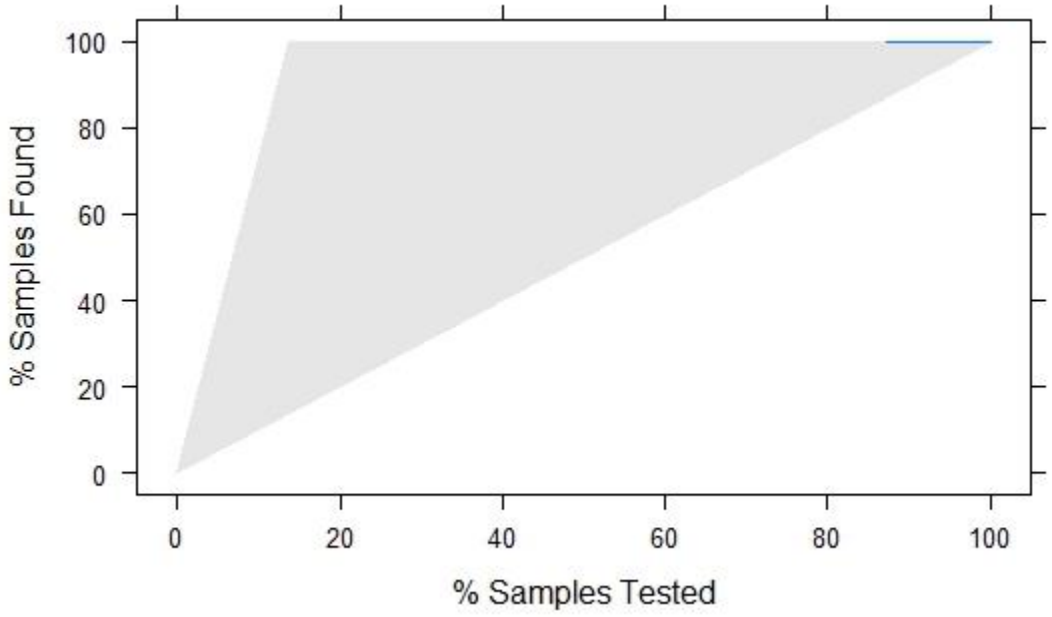


In the above picture the highlighted blue part shows the confusion matrix of the tree model which is **pruned**.

c. ROC CURVE FOR THE PRUNED TREE MODEL



d. LIFT CHART FOR PRUNED TREE MODEL



Conclusion from classification tree model: Since the performance metrics and confusion matrix looks better after the tree is pruned we are considering this model which is pruned as our final classification tree model. **The error rate from this model is 2.8%**

7. FINAL CONCLUSION FOR PROBLEM 2 which model we would choose

After comparing the performance metrics of all the three classification models for the dataset Internet Advertisements-Problem 2 we decided that we would choose **Classification trees** model. We have decided this because the overall error rate is less in that model compared to other two models. Also the accuracy and error rates in predicting the individual classes: Ad, NonAd is better in **Classification trees** model than the other two models.

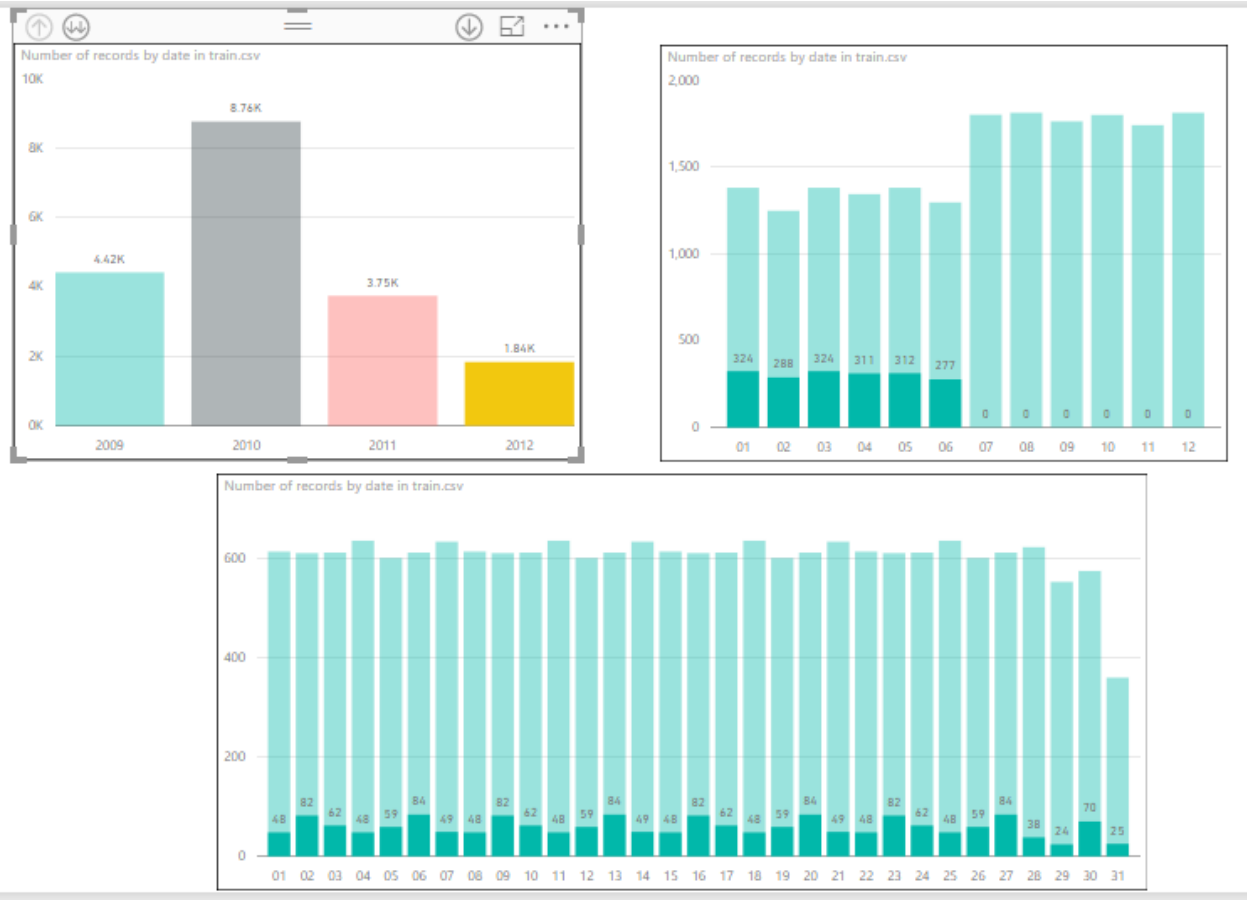
REFERENCES

1. http://www.fileformat.info/tip/web/imagesize.html
2. **http://www.r-bloggers.com/computing-and-visualizing-pca-in-r/**

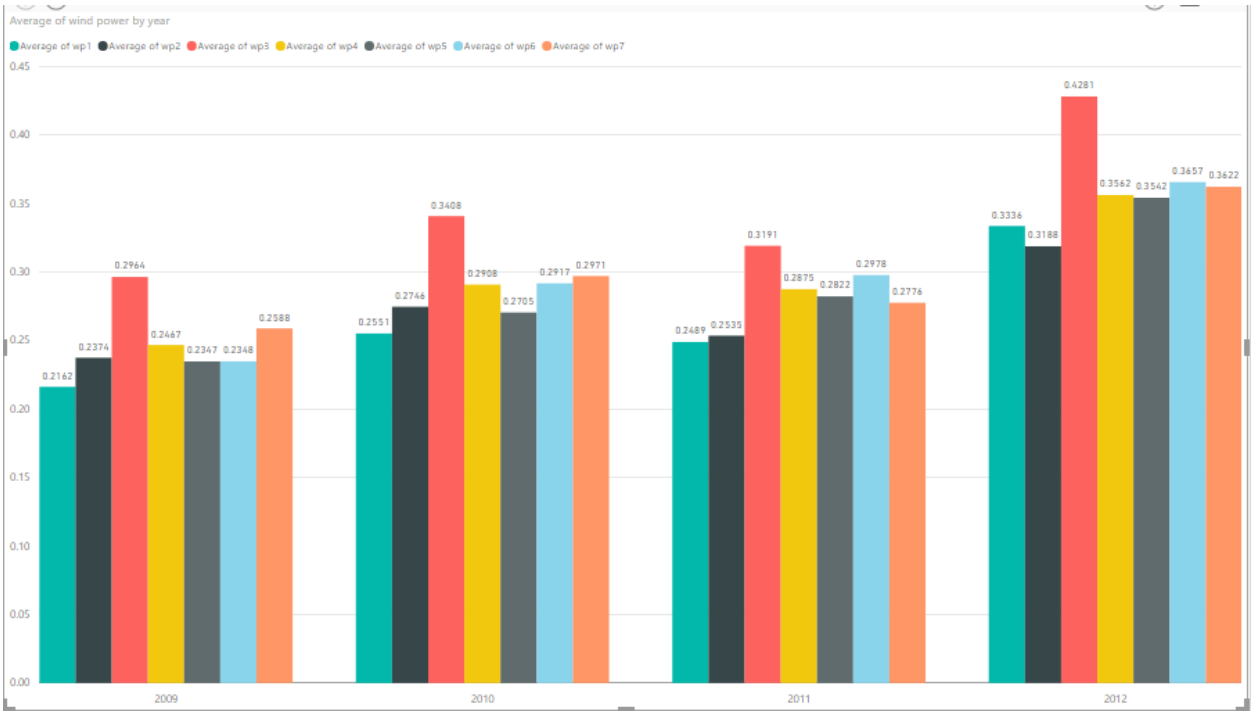
PROBLEM 3:

Observations:

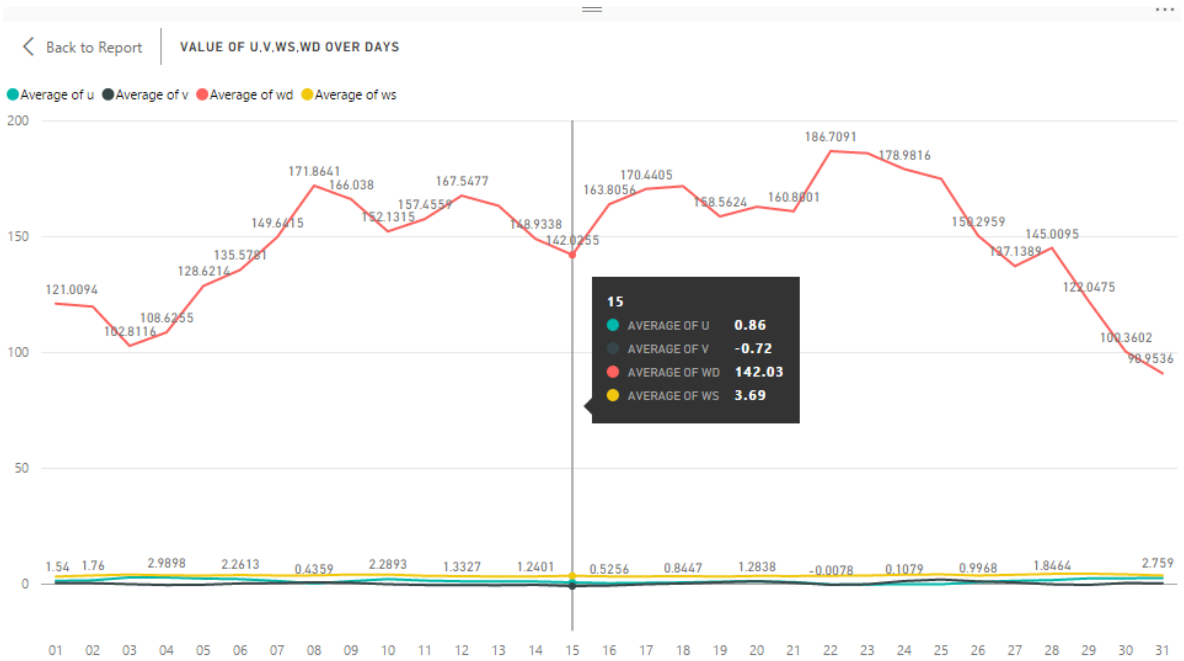
1. POWER BI:
   a. The datasets train and the weather forecasts files were available in .csv format which was easy to upload to POWER BI.
   b. Modified the dataset train in POWER BI to build better drilled down visualizations. Separated the date column to year, month, day and hour.
   c. Tried to create relationships between the weather forecasts datasets in POWER BI, but it restricts you from doing so saying “intermediate data is missing to connect the columns”.
2. Graphs:
   a. The following graph shows the missing / not missing data for specific years, months and days. For example: on selecting the year 2012, it shows that data for month’s August – December does not exist.



   b. The following graph shows the average windpower for each farm per year.



c. The following graph displays on a yearly average u, v, ws, wd measurements for a particular day.



1)Challenges faced while cleansing and pre-processing the data

a) Replacing the “0” values by NA in continuous data i.e for the wp1-wp7 columns in the train.csv dataset
b) Finding relation between windforecast dataset and train.csv dataset

c) Finding missing values in the train.csv dataset.
d) Handling NA values in the windforecast_WF1-WF7 datasets.

The following steps are followed in overcoming the challenge (a)

- ✓ Treating each wind farm data separately and and build separate models for each one of them.
- ✓ Conversion of the date column into character datatype.
- ✓ Split the hours and merge 4 row values by aggregating the values of "u", "v", "ws", "wd" to 1 single row. This resulted in reduced set of rows.
- ✓ For inference: 1st 4 rows aggregated values will be equivalent to the date 1st July 2009 00th hour and so on.
- ✓ Conversion of date into character format and performed substring operation to separate the hour column.
- ✓ The hour column was iterated through the last column from 00 to 23 values
- ✓ Split the date into day, month year to consider the date for Feature engineering.
- ✓ Considering the respective "windpower" value and "date" from the train.csv and doing a windforecast_wf1 with **left join** on train.csv
- ✓ The data consisted a lot 0 values. To remove those zero values in all the columns, running a for loop to make the "0" to "NA" and using zoo library doing NA_fill to fill the NA's
- ✓ Removed the date column and split the data into train(2009-10) and test(2011-12) based upon the given condition as mentioned the brackets

The following steps are followed in overcoming the challenge (b)

- ✓ As the values of the windforecast file are prediction of 48 hours, this gives insight on the data that for
  1. 20090701 00-12 hours and 20120628 12-24 there are 1 set of predicted values(u,v,ws,wd).
  2. 20090701 12-24 hours and 20120628 00-12 there are 2 set of predicted values(u,v,ws,wd).
  3. 20090702 00-12 hours and 20120627 12-24 there are 3 set of predicted values(u,v,ws,wd).
  4. Rest other dates have 4 predicted values
- ✓ For the missing values in train.csv dataset there were corresponding NA values in the Windforecast files of the 7 farms.

The following steps are followed in overcoming the challenge (c)

- ✓ For missing values in the train.csv file. As we performed the Left Join the corresponding missing values were made "NA"
- ✓ Benchmark file was also taken for reference.

The following steps are followed in overcoming the challenge (d)

- ✓ The data consisted a lot 0 values. To remove those zero values in all the columns, running a for loop to make the "0" to "NA"
- ✓ Using zoo library doing NA_fill to fill the NA's

Cleansing:

a. In the cleansing process we merged the train dataset with each weather forecast dataset in order to add more features to the model and obtain better results, also a process known as Feature Engineering.
b. After merging the 2 datasets, we eliminate the zeroes and NA's which are present for the various columns. We achieved this by converting all the zeroes to NA's and then using na.fill() function in R to eliminate all the NA's.

Building the Prediction Model using Regression

We choose regression for predicting as it showed comparatively less values of errors

IE: MAE, MAPE and RMS Values.

Key things noted while performing regression:

- ✓ Not all features are dependent.
- ✓ Few windfarms prediction was not relying on wind direction.
- ✓ None of the windfarms prediction was relying on the day on the year.
- ✓ Every wind farm has different dataset behavior and we made this conclusion based on the errors which varied from 300-500 value range.

Merging:

Merging the data after prediction to form benchmark.csv

WINDFORCAST1_WINDPOWER1

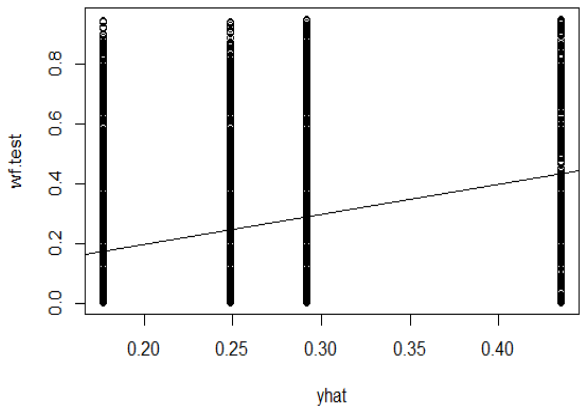
1. Linear regression

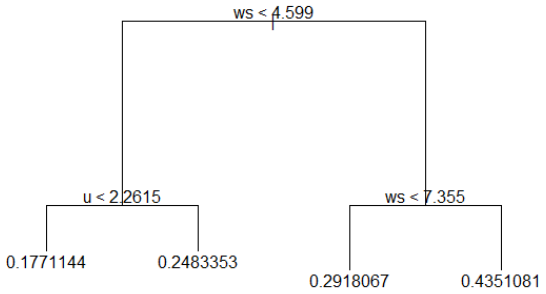
```
Call:
lm(formula = wp1 ~ . - day, data = wf1_wp1_training)

Residuals:
       Min          1Q      Median          3Q         Max
-0.55640703 -0.16271530 -0.06339265  0.11712971  0.76075124

Coefficients:
                    Estimate      Std. Error  t value             Pr(>|t|)
(Intercept) -58.39781227844   9.30804281011 -6.27391       0.00000000036312 *
**
hour          0.00101563011   0.00028279597  3.59139             0.00033012 *
**
year          0.02910570130   0.00463075111  6.28531       0.00000000033752 *
**
month         0.00247881234   0.00066192682  3.74484             0.00018128 *
**
u             0.00612577315   0.00112773259  5.43194       0.00000005673567 *
**
v            -0.00366313693   0.00072449594 -5.05612       0.00000043362583 *
**
```

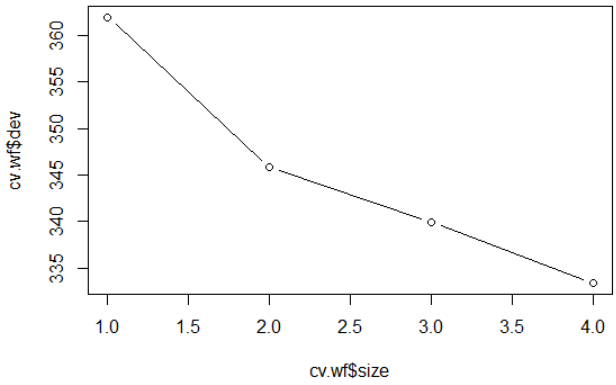
```
ws            0.03139724100   0.00124116922 25.29650 < 0.000000000000000222 *
**
wd           -0.00014243030   0.00003565005 -3.99523       0.00006498188885 *
**
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.2246904 on 13168 degrees of freedom
Multiple R-squared:  0.1031021,       Adjusted R-squared:  0.1026253
F-statistic:  216.245 on 7 and 13168 DF,  p-value: < 0.00000000000000022204



                      ME        RMSE          MAE          MPE        MAPE
Test set -0.01699120728 0.210518832 0.1711572784 -318.3813001 345.1425378
```

2. Regression Tree

```
Regression tree:
tree(formula = wp1 ~ ., data = wf1_wp1_training, subset = train)
Variables actually used in tree construction:
[1] "ws" "u"
Number of terminal nodes:  4
Residual mean deviance:  0.04993693 = 328.7848 / 6584
Distribution of residuals:
       Min.     1st Qu.      Median        Mean     3rd Qu.        Max. -0.43
010810 -0.16211440 -0.06211442  0.00000000  0.10961790  0.76988560
```








```
mean((yhat -wf.test)^2)
[1] 0.05299158023

mean((yhat -wf.test)^2)
[1] 0.05299158023
```

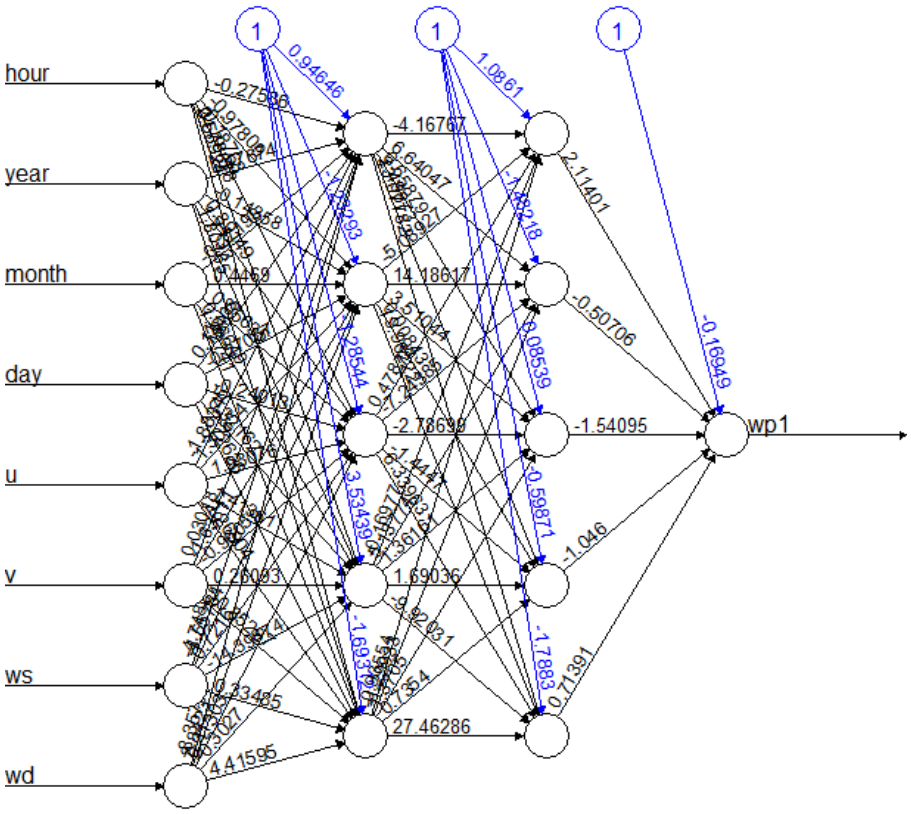
```
regtree
                      ME          RMSE           MAE           MPE          MAPE
Test set 0.008810648161 0.2301990014 0.1828134443 -456.4245724 490.7715326
```

3. Neural Networks

```
Call:
glm(formula = wp1 ~ . - day, data = wf1_wp1_training)

Deviance Residuals:
       Min          1Q      Median          3Q         Max
-0.55640703  -0.16271530  -0.06339265   0.11712971   0.76075124

Coefficients:
                    Estimate      Std. Error  t value             Pr(>|t|)
(Intercept) -58.39781227844   9.30804281011 -6.27391       0.00000000036312 *
**
hour          0.00101563011   0.00028279597  3.59139             0.00033012 *
**
year          0.02910570130   0.00463075111  6.28531       0.00000000033752 *
**
month         0.00247881234   0.00066192682  3.74484             0.00018128 *
**
u             0.00612577315   0.00112773259  5.43194       0.00000005673567 *
**
```

```
v            -0.00366313693   0.00072449594 -5.05612       0.00000043362583 *
**
ws            0.03139724100   0.00124116922 25.29650 < 0.00000000000000222 *
**
wd           -0.00014243030   0.00003565005 -3.99523       0.00006498188885 *
**
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 0.05048579386)

    Null deviance: 741.21805  on 13175  degrees of freedom
Residual deviance: 664.79693  on 13168  degrees of freedom
AIC: -1942.5041

Number of Fisher Scoring iterations: 2
```

```
Call: neuralnet(formula = f, data = train_, hidden = c(5, 5), threshold = 0.5
,     linear.output = F)

1 repetition was calculated.

        Error Reached Threshold Steps
1 495.3966914      0.4877164458  1334
```



Error: 495.396691 Steps: 1334

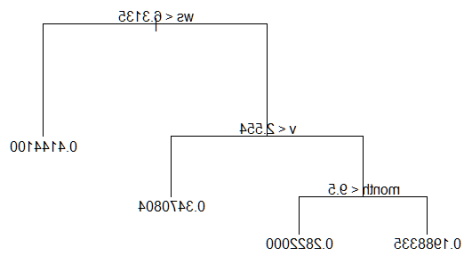
1. Linear regression

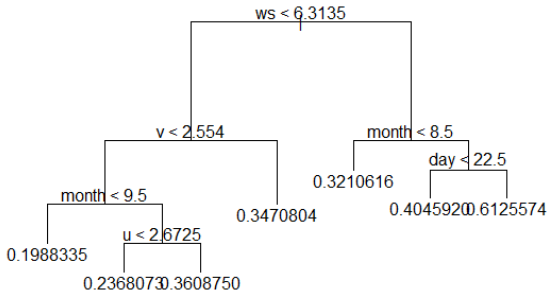
```
Call:
lm(formula = wp2 ~ . - day, data = wf2_wp2_training)

Residuals:
       Min         1Q     Median         3Q        Max
-0.58938421 -0.18418477 -0.07827996  0.13313365  0.81655643

Coefficients:
                     Estimate       Std. Error   t value                Pr(>|t|
)
(Intercept) -132.61932608467   10.56322274628 -12.55482 < 0.00000000000000022
2 ***
hour          -0.00155684772    0.00032093071  -4.85104       0.00000124221371
7 ***
year           0.06600190577    0.00525520311  12.55934 < 0.00000000000000022
2 ***
month          0.00999198736    0.00075118697  13.30160 < 0.00000000000000022
2 ***
u              0.00775482133    0.00127980616   6.05937       0.00000000140383
3 ***
v              0.01452804286    0.00082219346  17.66986 < 0.00000000000000022
2 ***
ws             0.03298144798    0.00140853961  23.41535 < 0.00000000000000022
2 ***
wd             0.00027245575    0.00004045742   6.73438       0.00000000001714
6 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.2549897 on 13168 degrees of freedom
Multiple R-squared:  0.1074973,       Adjusted R-squared:  0.1070228
F-statistic: 226.5738 on 7 and 13168 DF,  p-value: < 0.00000000000000022204
```

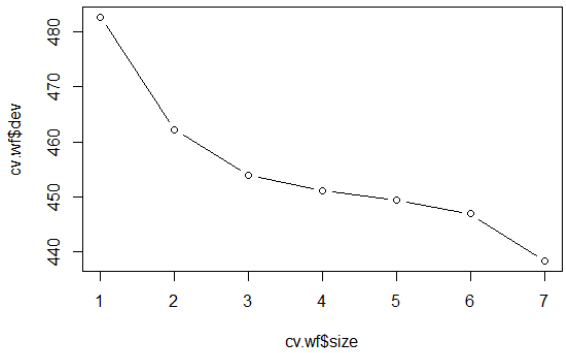
```
                    ME         RMSE          MAE          MPE      MAPE
Test set -0.0760176958 0.2562800574 0.2208869875 -614.0206594 635.77218
```

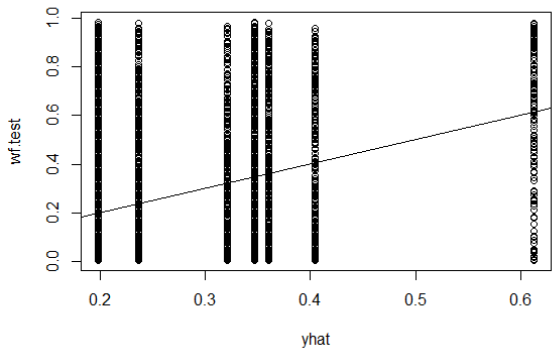
2. Regression Trees

```
Regression tree:
tree(formula = wp2 ~ ., data = wf2_wp2_training, subset = train)
Variables actually used in tree construction:
[1] "ws"    "v"     "month"
Number of terminal nodes:  6
Residual mean deviance:  0.0634658 = 417.7319 / 6582
Distribution of residuals:
       Min.     1st Qu.      Median        Mean     3rd Qu.        Max.
-0.56866390 -0.17321060 -0.07985127  0.00000000  0.13291040  0.80478940
```









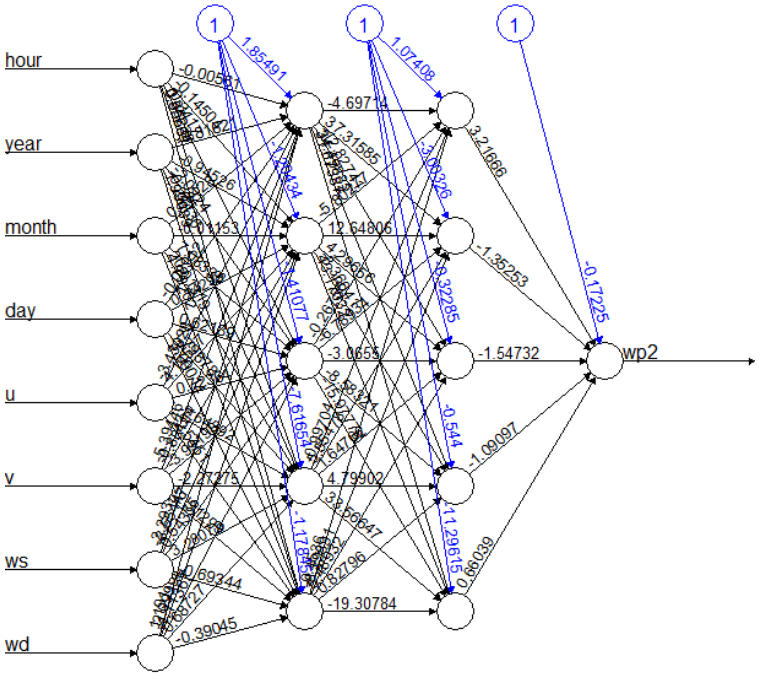

```
mean((yhat -wf.test)^2)
[1] 0.06488203411

mean((yhat -wf.test)^2)
[1] 0.06680416837
```

```
                      ME          RMSE           MAE           MPE        MAPE
Test set -0.001236647949 0.2584650235 0.2091916873 -538.5510493 571.589764
```

3. Neural Networks

```
Call:
glm(formula = wp2 ~ . - day, data = wf2_wp2_training)

Deviance Residuals:
```

```
        Min          1Q      Median          3Q         Max
-0.58938421  -0.18418477  -0.07827996   0.13313365   0.81655643

Coefficients:
                      Estimate       Std. Error   t value                Pr(>|t|
)
(Intercept) -132.61932608467   10.56322274627 -12.55482 < 0.00000000000000022
2 ***
hour          -0.00155684772    0.00032093071  -4.85104      0.00000124221371
7 ***
year           0.06600190577    0.00525520311  12.55934 < 0.00000000000000022
2 ***
month          0.00999198736    0.00075118697  13.30160 < 0.00000000000000022
2 ***
u              0.00775482133    0.00127980616   6.05937      0.00000000140383
3 ***
v              0.01452804286    0.00082219346  17.66986 < 0.00000000000000022
2 ***
ws             0.03298144798    0.00140853961  23.41535 < 0.00000000000000022
2 ***
wd             0.00027245575    0.00004045742   6.73438      0.00000000001714
6 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 0.06501975389)

    Null deviance: 959.30254  on 13175  degrees of freedom
Residual deviance: 856.18012  on 13168  degrees of freedom
AIC: 1391.0126

Number of Fisher Scoring iterations: 2
```



Error: 558.340601 Steps: 3597

WINDFORCAST3_WINDPOWER3

1. Linear regression

```
Call:
lm(formula = wp3 ~ . - day, data = wf3_wp3_training)

Residuals:
       Min         1Q     Median         3Q        Max
-0.85305831 -0.20039033 -0.08008114  0.18343275  0.84848121

Coefficients:
                    Estimate      Std. Error  t value               Pr(>|t|)
(Intercept) -72.34165585640  11.63516420960 -6.21750       0.00000000052039 *
**
hour         -0.00127076288   0.00035574399 -3.57213             0.00035536 *
**
year          0.03605656929   0.00578840214  6.22911       0.00000000048338 *
**
month         0.00176205344   0.00083167024  2.11869             0.03413514 *
u             0.00432837599   0.00106091535  4.07985       0.00004533019596 *
**
v            -0.00335057485   0.00074718277 -4.48428       0.00000737799332 *
**
ws            0.05075164434   0.00114578866 44.29407 < 0.00000000000000222 *
**
wd           -0.00020029236   0.00004178322 -4.79361       0.00000165598849 *
**
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.2826515 on 13168 degrees of freedom
Multiple R-squared:  0.1750178,       Adjusted R-squared:  0.1745793
F-statistic: 399.0796 on 7 and 13168 DF,  p-value: < 0.00000000000000022204
```

```
                      ME         RMSE          MAE          MPE        MAPE
Test set -0.03635507731 0.2699360242 0.2288923019 -316.1639134 342.8249857
```

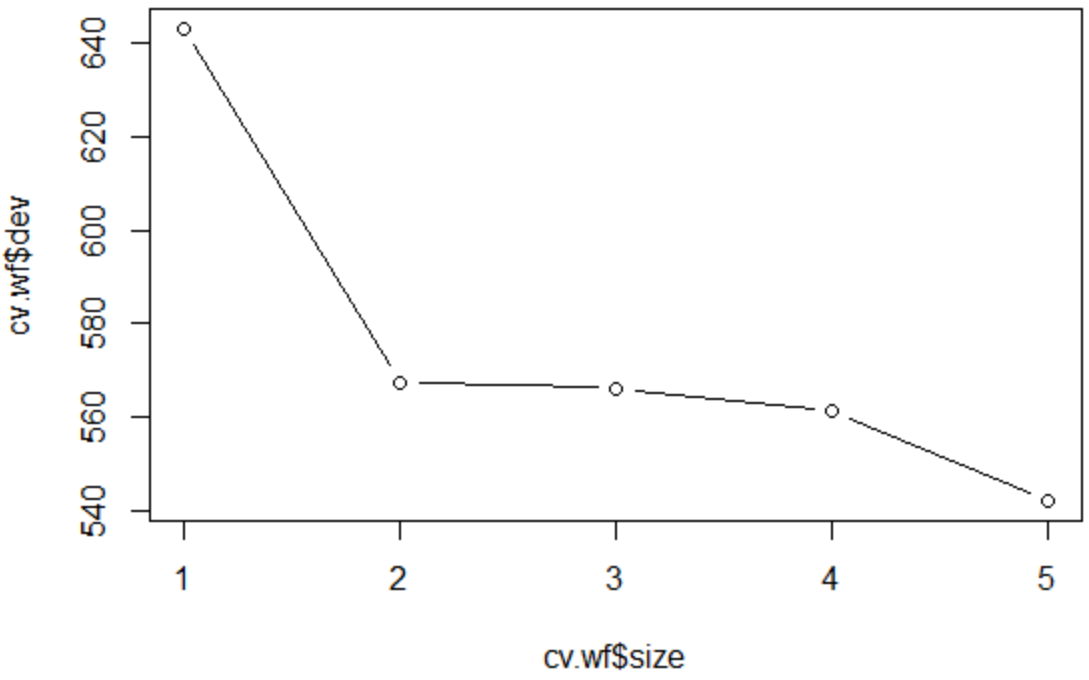
Regression Trees

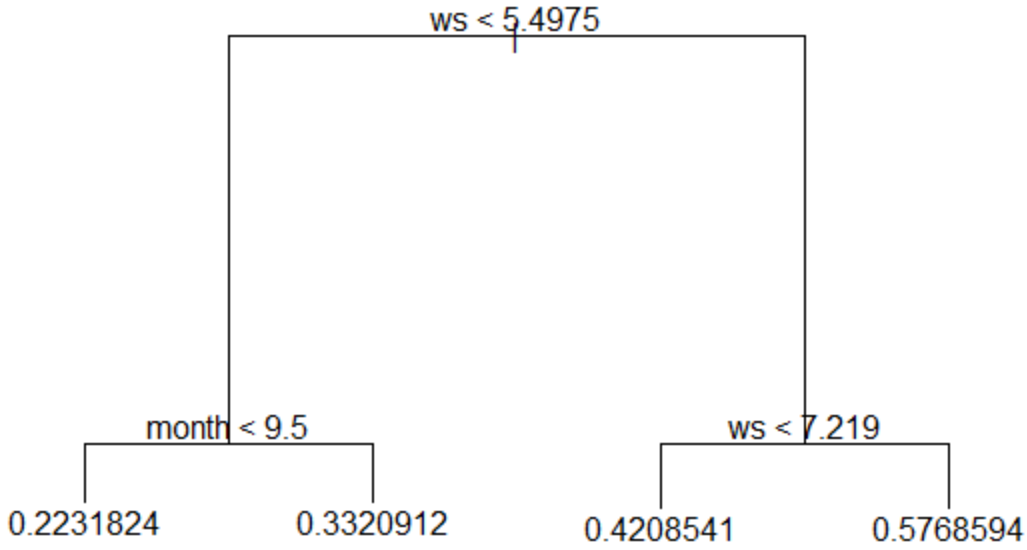
```
Regression tree:
tree(formula = wp3 ~ ., data = wf3_wp3_training, subset = train)
Variables actually used in tree construction:
[1] "ws"    "month"
Number of terminal nodes:  5
Residual mean deviance:  0.08050204 = 529.9449 / 6583
Distribution of residuals:
       Min.     1st Qu.      Median        Mean     3rd Qu.        Max.
-0.56685940 -0.19560770 -0.07509118  0.00000000  0.19190880  0.80422730
```

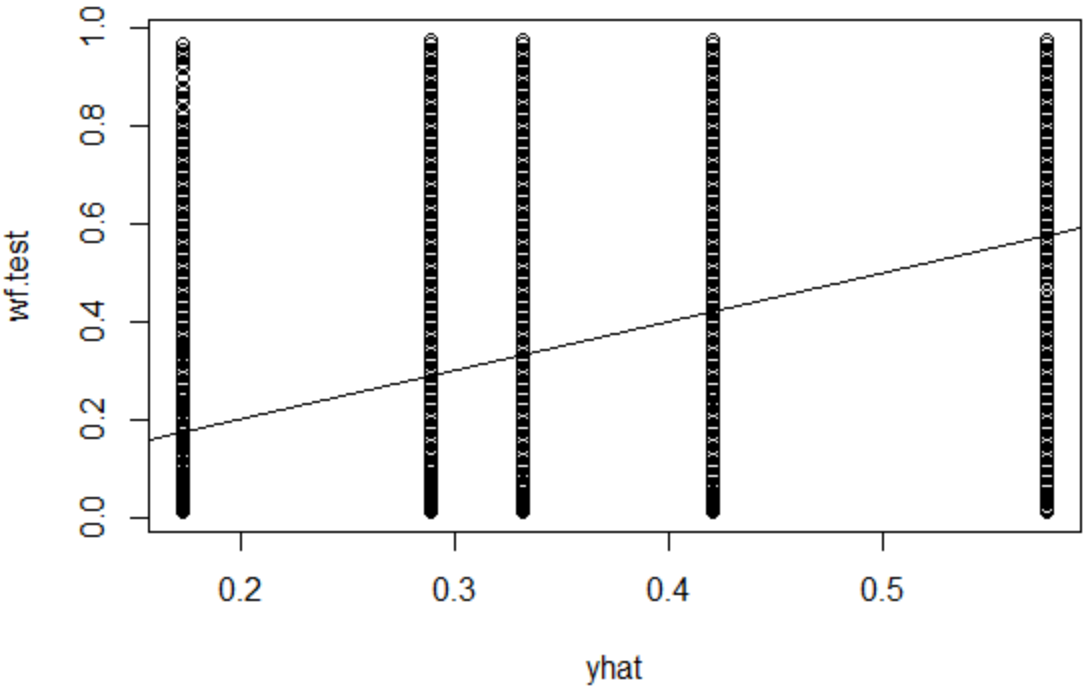
```
mean((yhat -wf.test)^2)
[1] 0.08065279286
```

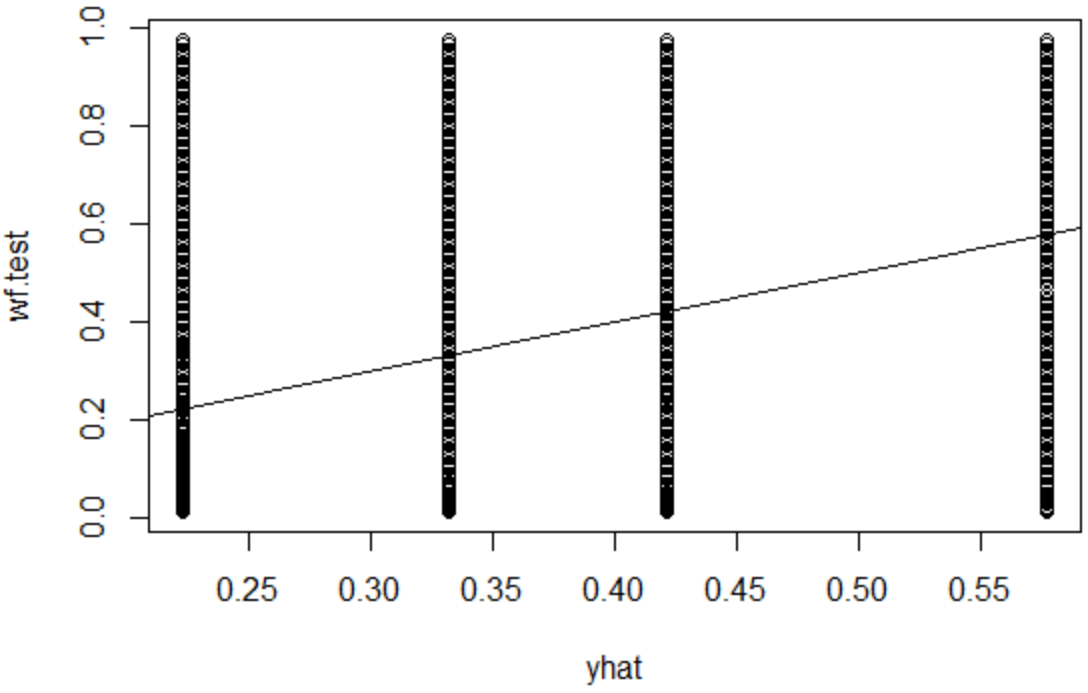
```
mean((yhat -wf.test)^2)
[1] 0.0820510557
```

```
ME          RMSE           MAE           MPE        MAPE
Test set 0.0004895362947 0.2864455545 0.2344040792 -335.8236797 368.6495668
```


cv.wf$dev
640
620
600
580
560
540
1
2
3
4
5
cv.wf$size


ws < 5.4975
month < 9.5
ws < 7.219
0.2231824
0.3320912
0.4208541
0.5768594

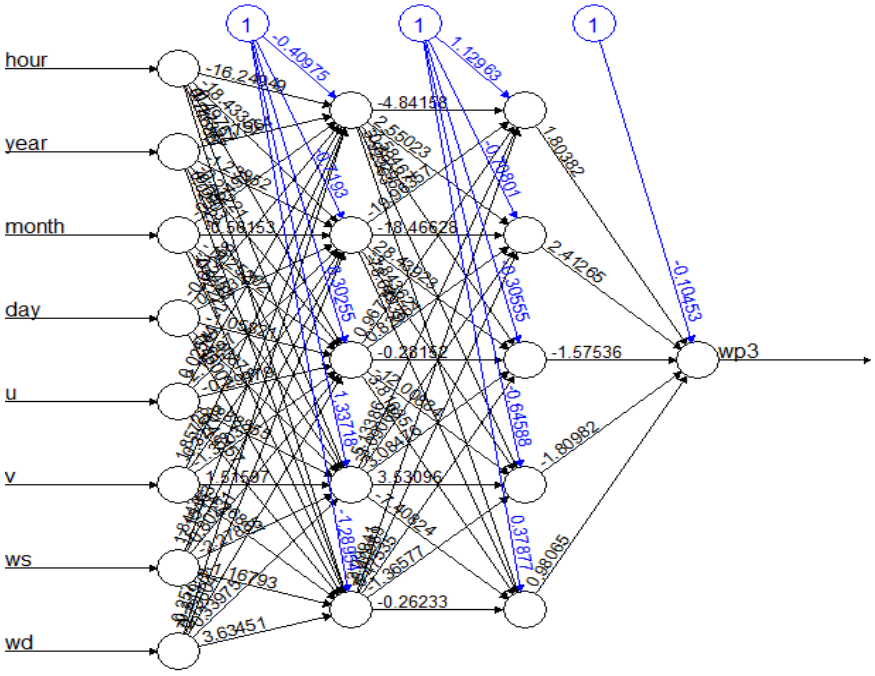

wf.test
0.0
0.2
0.4
0.6
0.8
1.0
0.2
0.3
0.4
0.5
yhat


wf.test
0.0
0.2
0.4
0.6
0.8
1.0
0.25
0.30
0.35
0.40
0.45
0.50
0.55
yhat

3. Neural Networks

```
Call:
glm(formula = wp3 ~ . - day, data = wf3_wp3_training)

Deviance Residuals:
        Min           1Q       Median           3Q          Max
-0.85305831  -0.20039033  -0.08008114   0.18343275   0.84848121

Coefficients:
                    Estimate      Std. Error  t value               Pr(>|t|)
(Intercept) -72.34165585640  11.63516420960 -6.21750       0.00000000052039 *
**
hour          -0.00127076288   0.00035574399 -3.57213             0.00035536 *
**
year           0.03605656929   0.00578840214  6.22911       0.00000000048338 *
**
month          0.00176205344   0.00083167024  2.11869             0.03413514 *
u              0.00432837599   0.00106091535  4.07985       0.00004533019596 *
**
v             -0.00335057485   0.00074718277 -4.48428       0.00000737799332 *
**
ws             0.05075164434   0.00114578866 44.29407 < 0.000000000000000222 *
**
wd            -0.00020029236   0.00004178322 -4.79361       0.00000165598849 *
**
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 0.0798918668)

    Null deviance: 1275.1986  on 13175  degrees of freedom
Residual deviance: 1052.0161  on 13168  degrees of freedom
AIC: 4105.0436

Number of Fisher Scoring iterations: 2


Call: neuralnet(formula = f, data = train_, hidden = c(5, 5), threshold = 0.5
,     linear.output = F)

1 repetition was calculated.

        Error Reached Threshold Steps
```

```
1 702.5487993       0.4838796531  1623
```



Error: 702.548799 Steps: 1623

WINDFORCAST4_WINDPOWER4

1. Linear regression

```
Call:
lm(formula = wp4 ~ . - day - wd, data = wf4_wp4_training)

Residuals:
       Min          1Q      Median          3Q         Max
-0.75753528 -0.18244095 -0.07306806  0.16212072  0.87422240

Coefficients:
                    Estimate     Std. Error  t value                Pr(>|t|)
(Intercept) -92.8876110319  10.6851285798 -8.69317 < 0.00000000000000022 ***
hour         -0.0007342933   0.0003228937 -2.27410                0.022976 *
year          0.0462256852   0.0053160407  8.69551 < 0.00000000000000022 ***
month         0.0090212238   0.0007588975 11.88728 < 0.00000000000000022 ***
u             0.0096449698   0.0007475528 12.90206 < 0.00000000000000022 ***
v            -0.0029016857   0.0006318734 -4.59219           0.0000044267 ***
ws            0.0437621838   0.0011330347 38.62387 < 0.00000000000000022 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.2565489 on 13169 degrees of freedom
Multiple R-squared:  0.1781864,       Adjusted R-squared:  0.1778119
F-statistic: 475.8857 on 6 and 13169 DF,  p-value: < 0.00000000000000022204
```

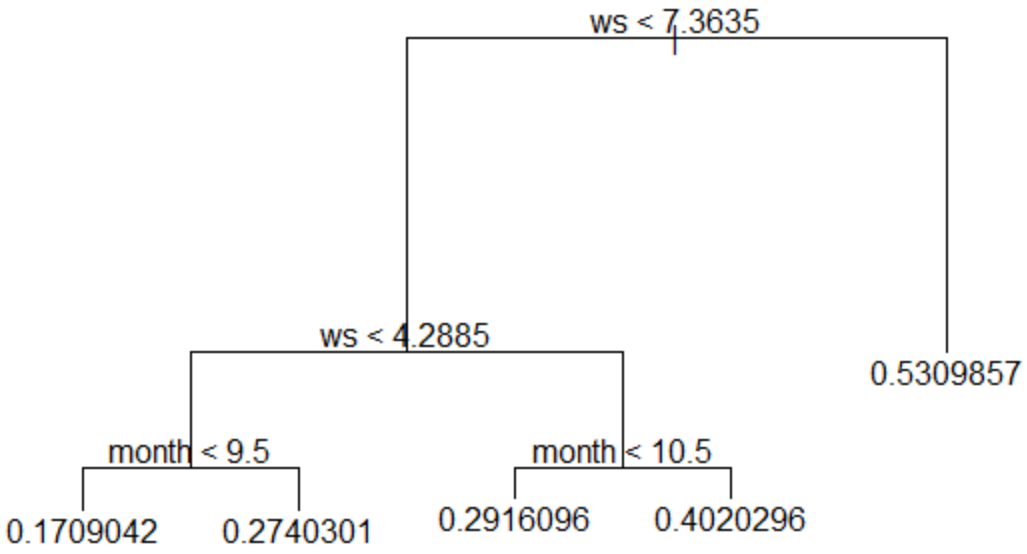
```
                       ME          RMSE          MAE          MPE        MAPE
Test set -0.03325152957 0.2538314469 0.2105435097 -570.3906287 596.9841164
```

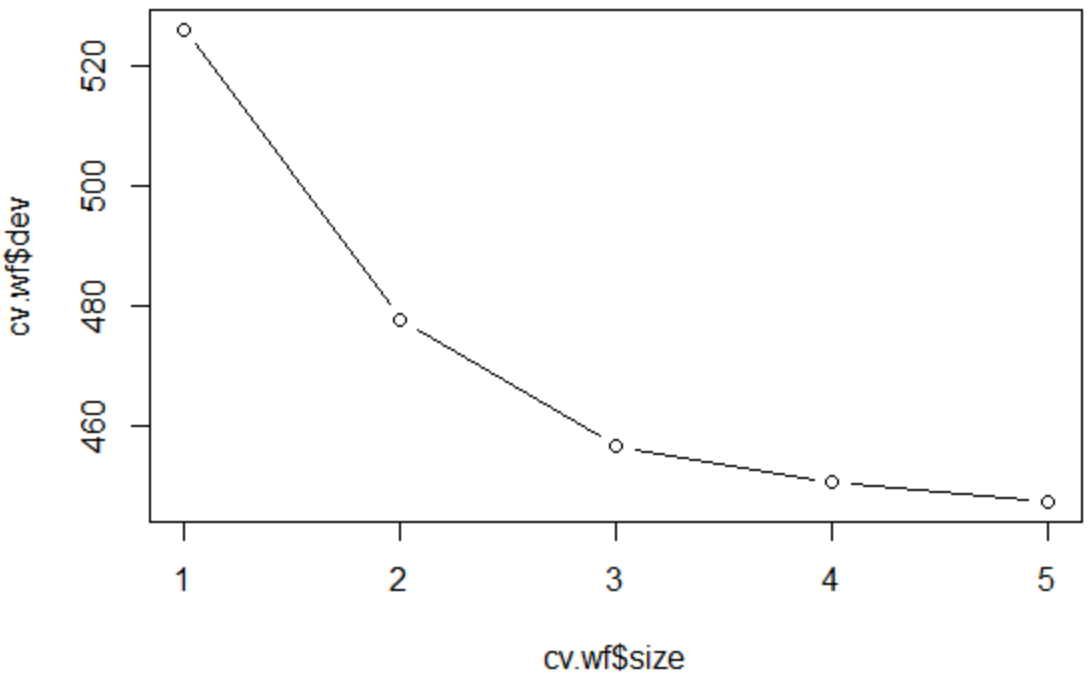
2. Regression Tree

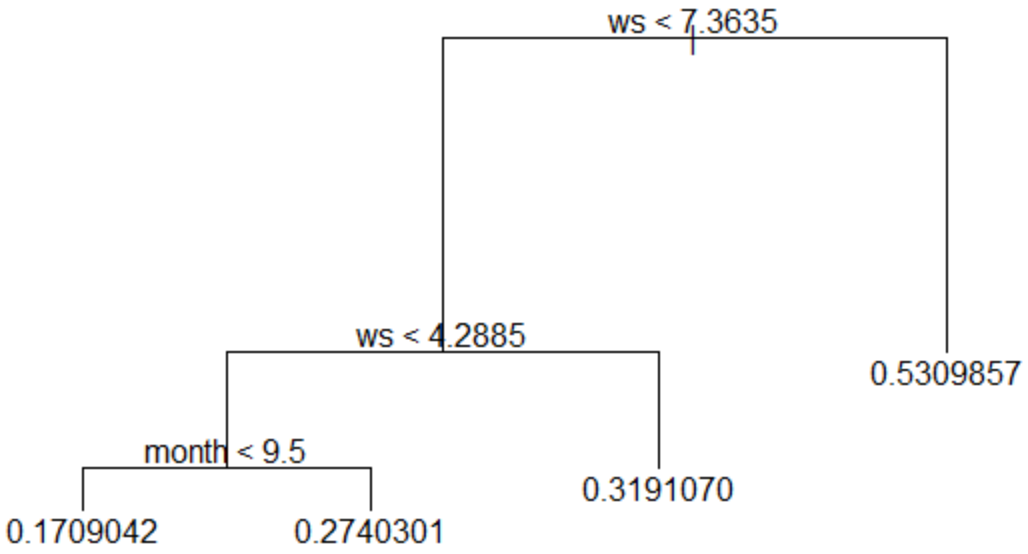
```
Regression tree:
tree(formula = wp4 ~ ., data = wf4_wp4_training, subset = train)
Variables actually used in tree construction:
[1] "ws"    "month"
Number of terminal nodes:  5
```

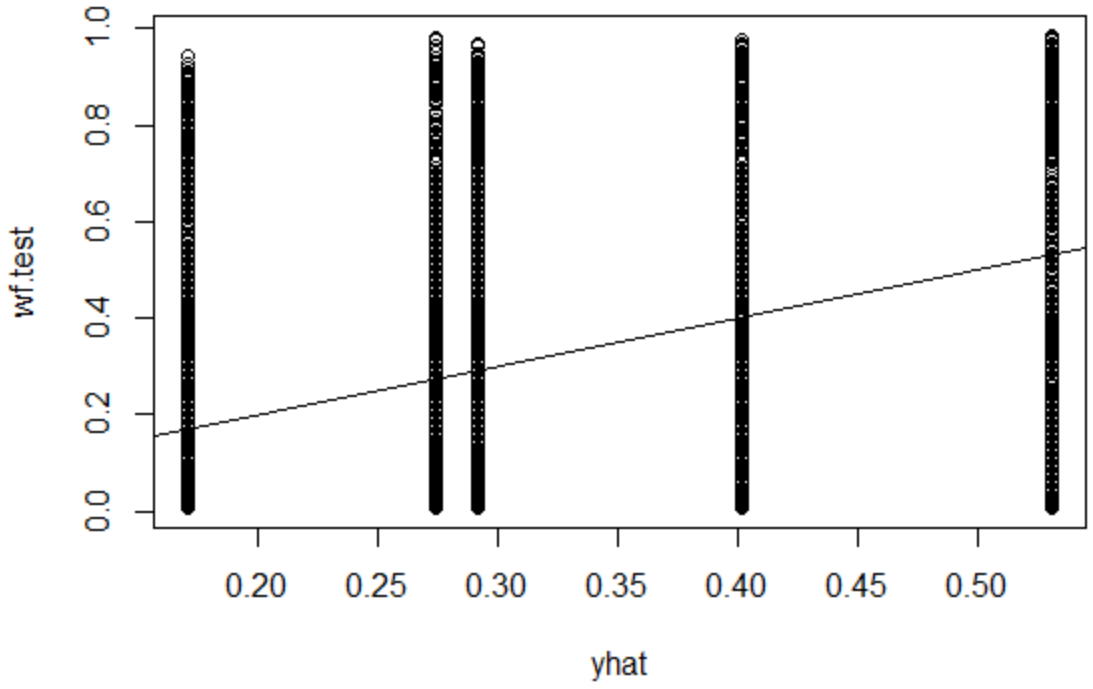
```
Residual mean deviance:  0.06662173 = 438.5709 / 6583
Distribution of residuals:
      Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
-0.5249857 -0.1649042 -0.0769042  0.0000000  0.1630143  0.8100958
```

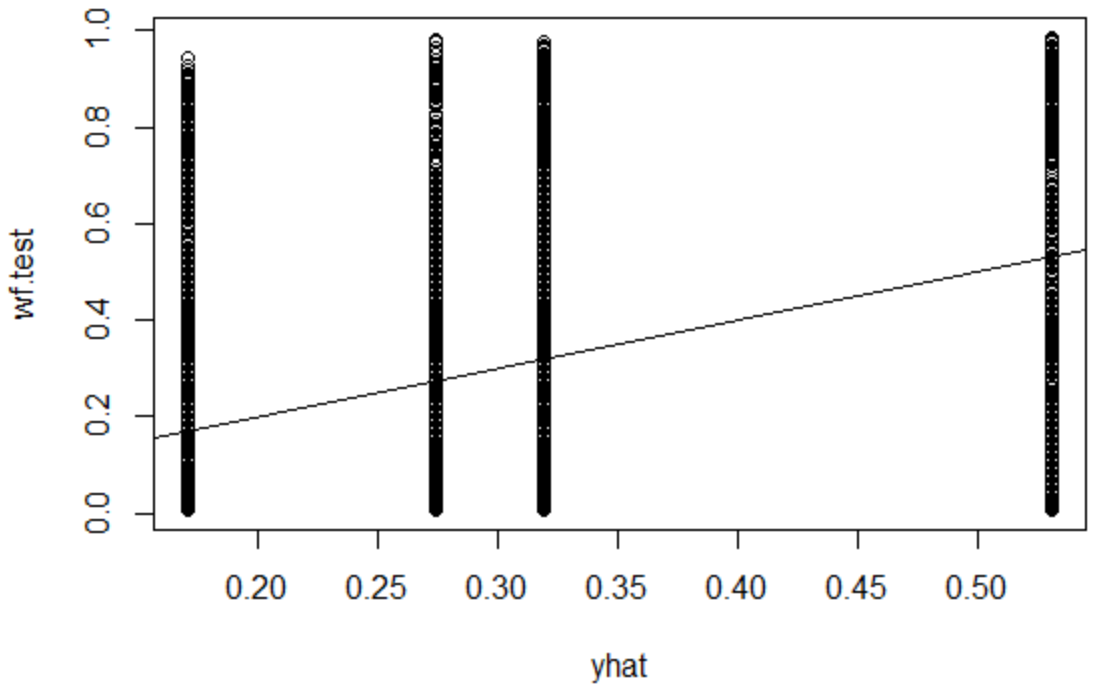
	ME	RMSE	MAE	MPE	MAPE
Test set	0.001945137797	0.2615205415	0.2142294052	-546.1986825	580.9484296


ws < 7.3635
ws < 4.2885
0.5309857
month < 9.5
month < 10.5
0.1709042
0.2740301
0.2916096
0.4020296


cv.wf$dev
460
480
500
520
1
2
3
4
5
cv.wf$size


ws < 7.3635
ws < 4.2885
0.5309857
month < 9.5
0.3191070
0.1709042
0.2740301

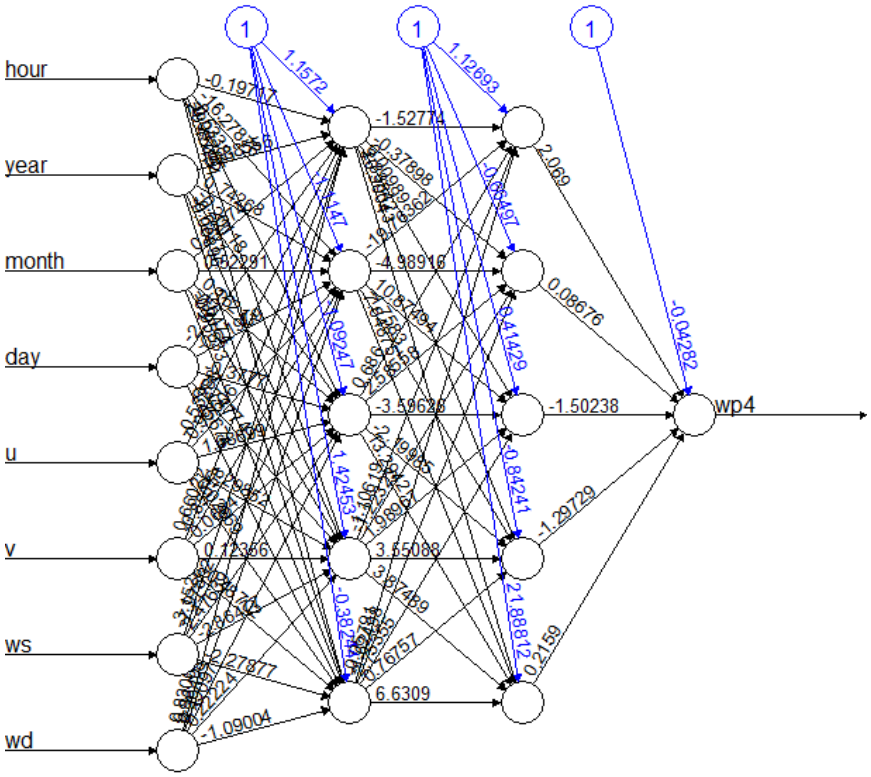

wf.test
0.0
0.2
0.4
0.6
0.8
1.0
0.20
0.25
0.30
0.35
0.40
0.45
0.50
yhat



3. Neural Networks

```
Call:
glm(formula = wp4 ~ . - day - wd, data = wf4_wp4_training)

Deviance Residuals:
        Min           1Q       Median           3Q          Max
-0.75753528  -0.18244095  -0.07306806   0.16212072   0.87422240

Coefficients:
                   Estimate     Std. Error  t value                Pr(>|t|)
(Intercept) -92.8876110319  10.6851285798 -8.69317 < 0.00000000000000222 ***
hour         -0.0007342933   0.0003228937 -2.27410                0.022976 *
year          0.0462256852   0.0053160407  8.69551 < 0.00000000000000222 ***
month         0.0090212238   0.0007588975 11.88728 < 0.00000000000000222 ***
u             0.0096449698   0.0007475528 12.90206 < 0.00000000000000222 ***
v            -0.0029016857   0.0006318734 -4.59219           0.0000044267 ***
ws            0.0437621838   0.0011330347 38.62387 < 0.00000000000000222 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 0.06581731732)

    Null deviance: 1054.67737  on 13175  degrees of freedom
Residual deviance:  866.74825  on 13169  degrees of freedom
AIC: 1550.653

Number of Fisher Scoring iterations: 2
```

```
Call: neuralnet(formula = f, data = train_, hidden = c(5, 5), threshold = 0.5
,     linear.output = F)

1 repetition was calculated.

        Error Reached Threshold Steps
1 630.2489674      0.4392417142   522
```



Error: 630.248967 Steps: 522

WINDFORCAST5_WINDPOWER5

1. Linear regression

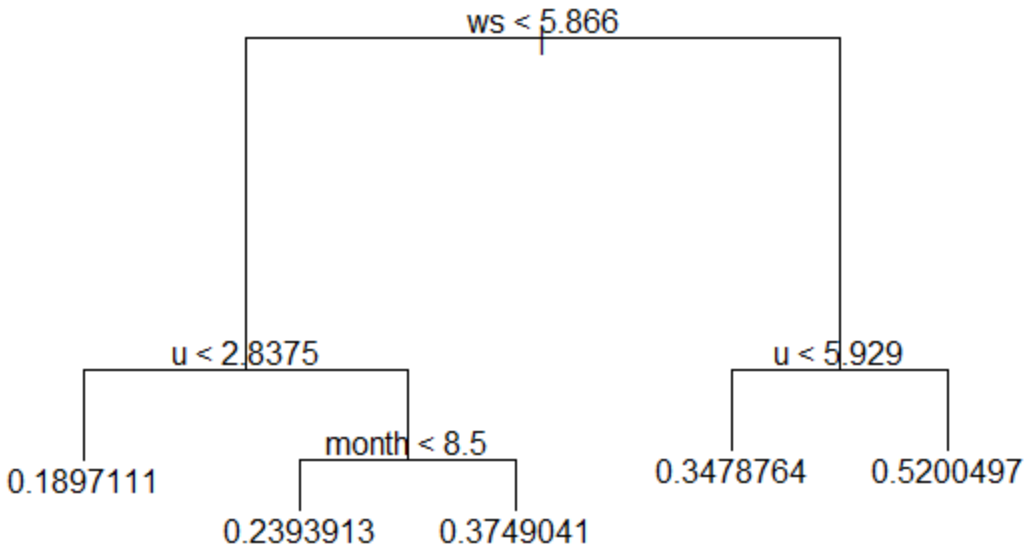
```
Call:
lm(formula = wp5 ~ . - day - wd - hour, data = wf5_wp5_training)

Residuals:
        Min          1Q      Median          3Q         Max
-0.72506484 -0.17650223 -0.07811524  0.11844692  0.84466002

Coefficients:
                    Estimate      Std. Error   t value                Pr(>|t|)
(Intercept) -116.2193416573   10.6904694660 -10.87130 < 0.000000000000000222
***
year           0.0578411654    0.0053183085  10.87586 < 0.000000000000000222
***
month          0.0063783431    0.0007614767   8.37628 < 0.000000000000000222
***
u              0.0125562619    0.0007623441  16.47060 < 0.000000000000000222
***
v             -0.0048812890    0.0006844421  -7.13178      0.0000000000010425
***
ws             0.0385749634    0.0012122612  31.82067 < 0.000000000000000222
***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.2596923 on 13170 degrees of freedom
Multiple R-squared:  0.1346889,       Adjusted R-squared:  0.1343604
F-statistic: 409.9919 on 5 and 13170 DF,  p-value: < 0.00000000000000022204
```

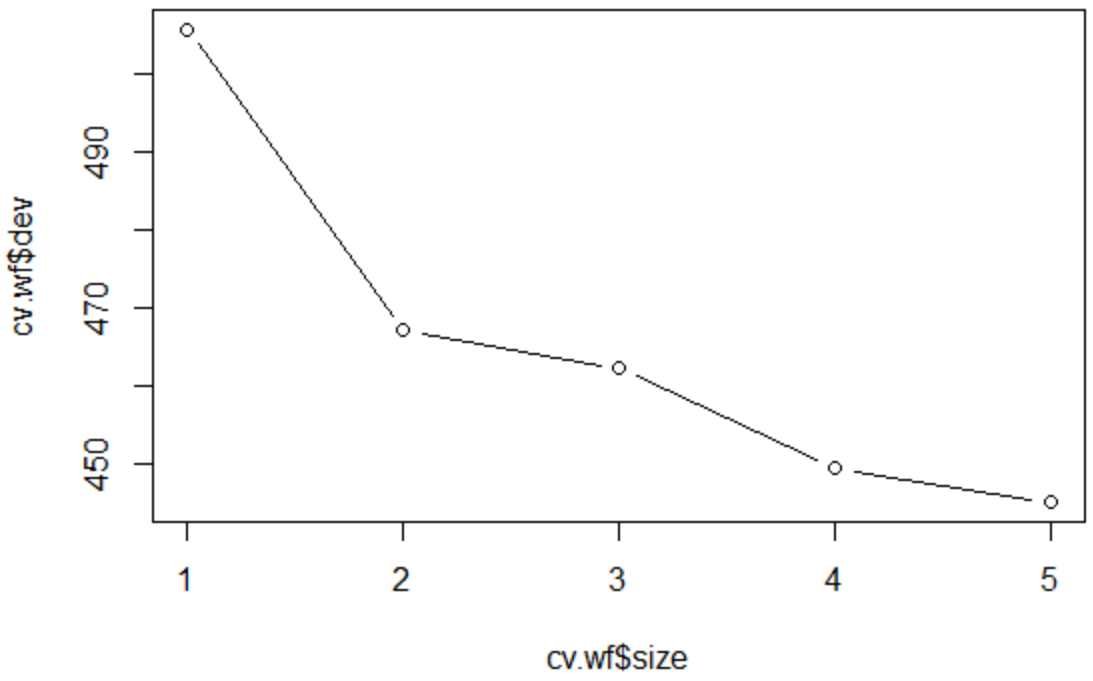
```
                      ME          RMSE           MAE           MPE        MAPE
Test set -0.03391770093 0.2596316125 0.2171041735 -542.3727209 568.7013946
```

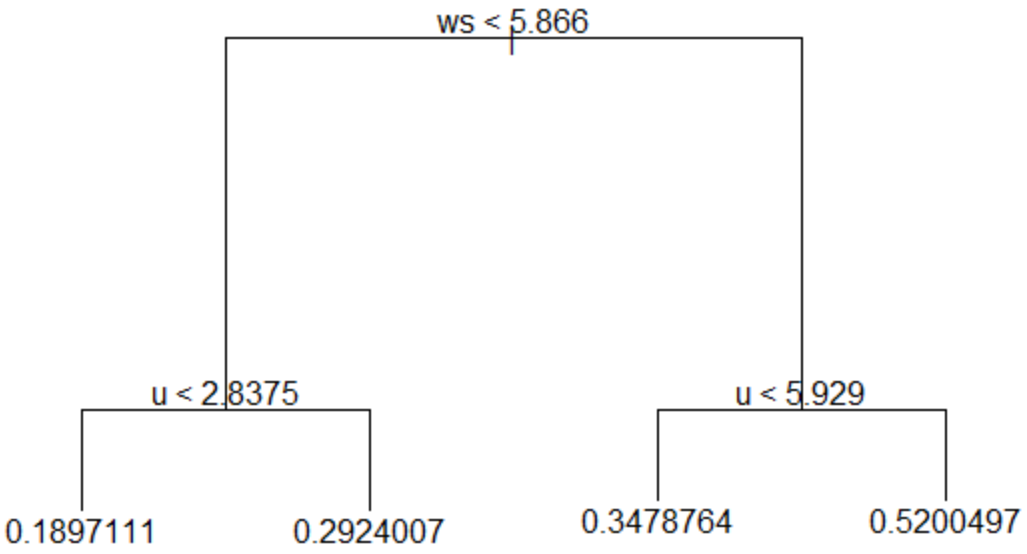
2. Regression Tree

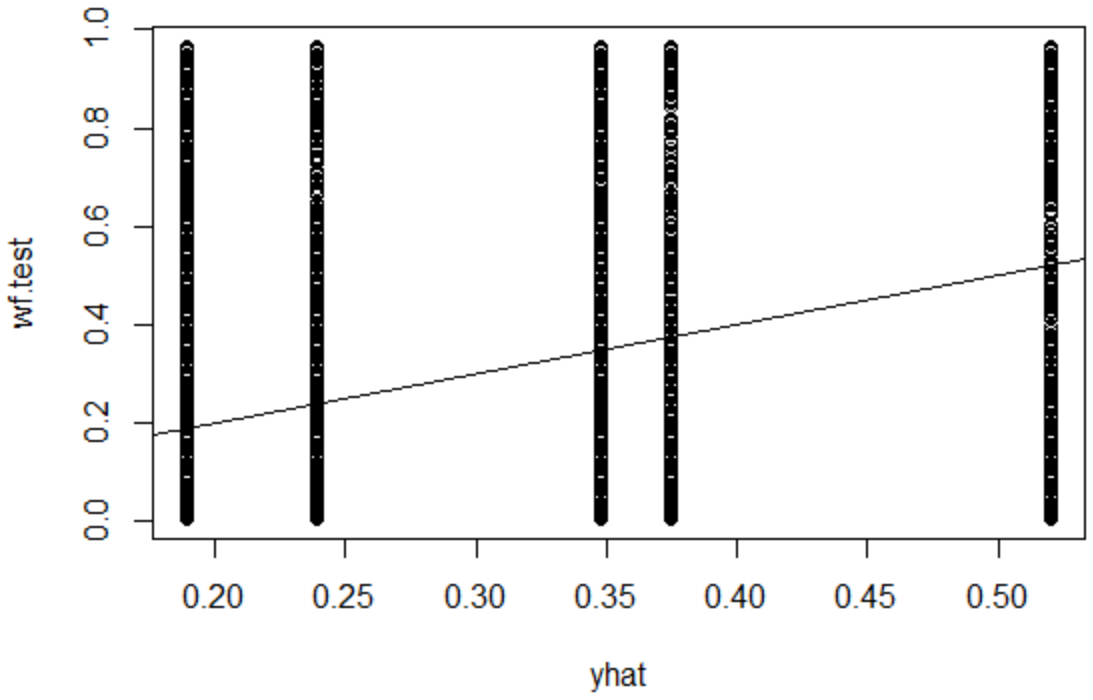
```
Regression tree:
tree(formula = wp5 ~ ., data = wf5_wp5_training, subset = train)
Variables actually used in tree construction:
[1] "ws"    "u"     "month"
Number of terminal nodes:  5
Residual mean deviance:  0.0669271 = 440.5811 / 6583
Distribution of residuals:
       Min.     1st Qu.      Median        Mean     3rd Qu.        Max.
-0.51504970 -0.17471110 -0.08371114  0.00000000  0.12699360  0.77628890
```

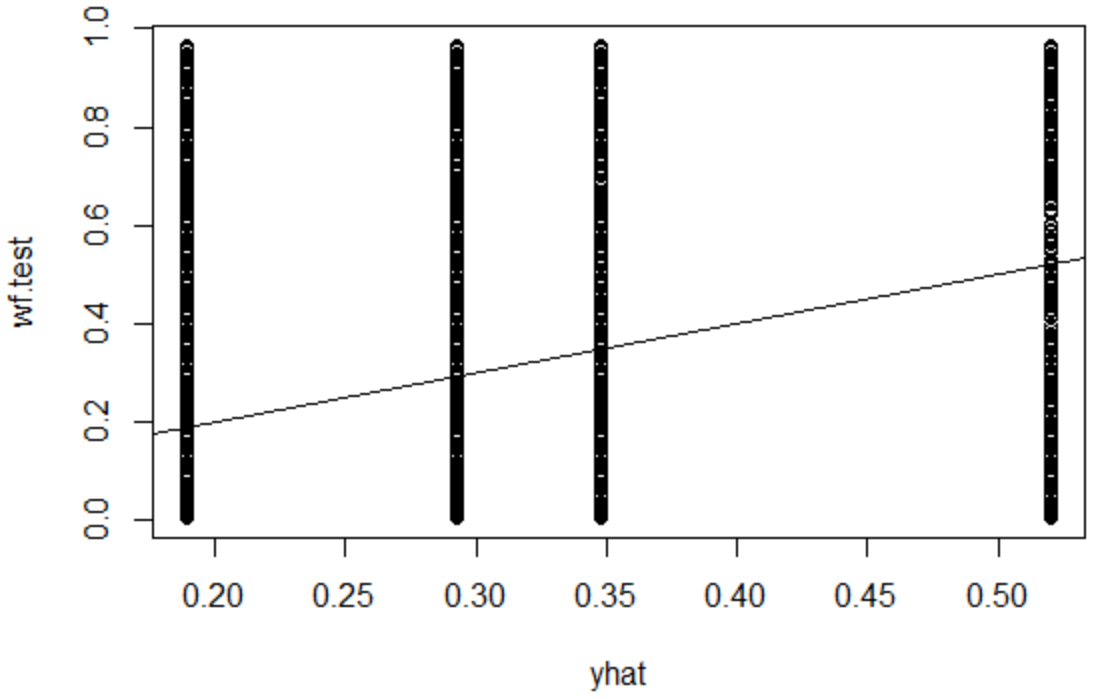
```
                     ME        RMSE           MAE           MPE        MAPE
Test set 0.003809033866 0.267051861 0.2095449639 -532.2896787 565.2044116
```


ws < 5.866
u < 2.8375
u < 5.929
month < 8.5
0.1897111
0.2393913
0.3749041
0.3478764
0.5200497


cv.wf$dev
450
470
490
1
2
3
4
5
cv.wf$size

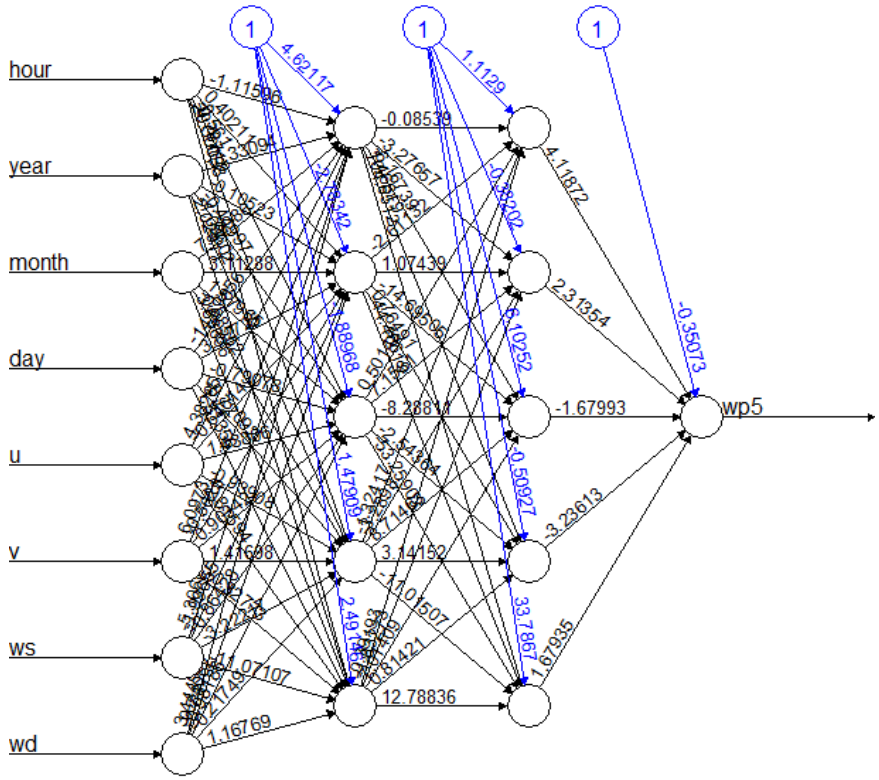

ws < 5.866
u < 2.8375
u < 5.929
0.1897111
0.2924007
0.3478764
0.5200497


wf.test
0.0
0.2
0.4
0.6
0.8
1.0
0.20
0.25
0.30
0.35
0.40
0.45
0.50
yhat



3. Neural Network

```
Call:
glm(formula = wp5 ~ . - day - wd - hour, data = wf5_wp5_training)

Deviance Residuals:
        Min           1Q       Median           3Q          Max
-0.72506484  -0.17650223  -0.07811524   0.11844692   0.84466002

Coefficients:
                    Estimate      Std. Error   t value               Pr(>|t|)
(Intercept) -116.2193416573   10.6904694660 -10.87130 < 0.000000000000000222 ***
year           0.0578411654    0.0053183085  10.87586 < 0.000000000000000222 ***
month          0.0063783431    0.0007614767   8.37628 < 0.000000000000000222 ***
u              0.0125562619    0.0007623441  16.47060 < 0.000000000000000222 ***
v             -0.0048812890    0.0006844421  -7.13178     0.0000000000010425 ***
ws             0.0385749634    0.0012122612  31.82067 < 0.000000000000000222 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 0.06744008979)

    Null deviance: 1026.43544  on 13175  degrees of freedom
Residual deviance:  888.18598  on 13170  degrees of freedom
AIC: 1870.5769

Number of Fisher Scoring iterations: 2


Call: neuralnet(formula = f, data = train_, hidden = c(5, 5), threshold = 0.5,     linear

1 repetition was calculated.

        Error Reached Threshold Steps
1 657.2507165      0.4986928763  6053
```



WINDFORCAST7_WINDPOWER7

1. Linear regression

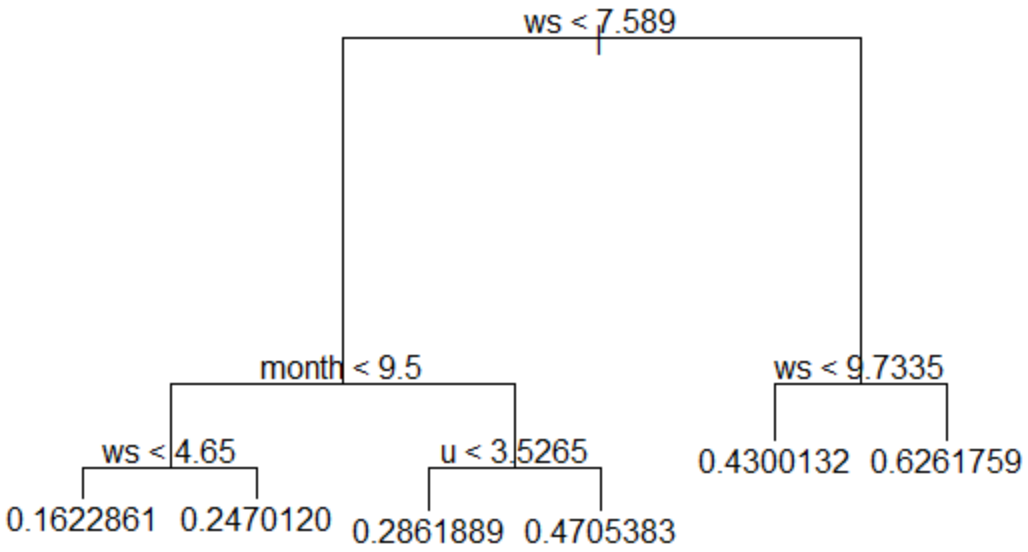
```
Call:
lm(formula = wp7 ~ . - hour - day - v, data = wf7_wp7_training)

Residuals:
       Min          1Q      Median          3Q         Max
-0.84563300 -0.17933486 -0.07103305  0.16340310  0.82736686

Coefficients:
                    Estimate      Std. Error  t value                Pr(>|t|)
(Intercept) -84.34131543285  10.68295065578 -7.89495  0.0000000000000031333 *
**
year          0.04195109824   0.00531486363  7.89317  0.0000000000000031781 *
**
month         0.00847582076   0.00076532098 11.07486 < 0.00000000000000222 *
**
u             0.01299829601   0.00086688388 14.99428 < 0.00000000000000222 *
**
ws            0.04022286443   0.00090076027 44.65435 < 0.00000000000000222 *
**
wd            0.00016839979   0.00003330497  5.05630  0.0000004332209643909 *
**
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.2571437 on 13170 degrees of freedom
Multiple R-squared:  0.2211961,       Adjusted R-squared:  0.2209004
F-statistic: 748.1094 on 5 and 13170 DF,  p-value: < 0.00000000000000022204
```

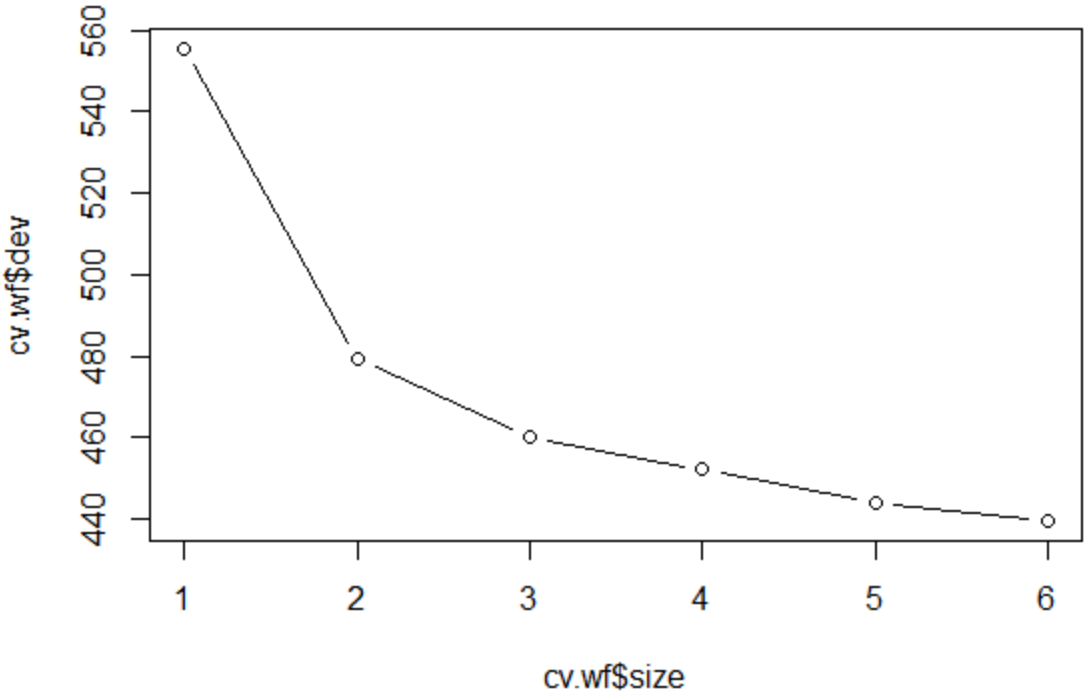
```
                      ME        RMSE          MAE           MPE        MAPE
Test set -0.05678008543 0.261535252 0.2210903422 -437.7348054 461.9548179
```

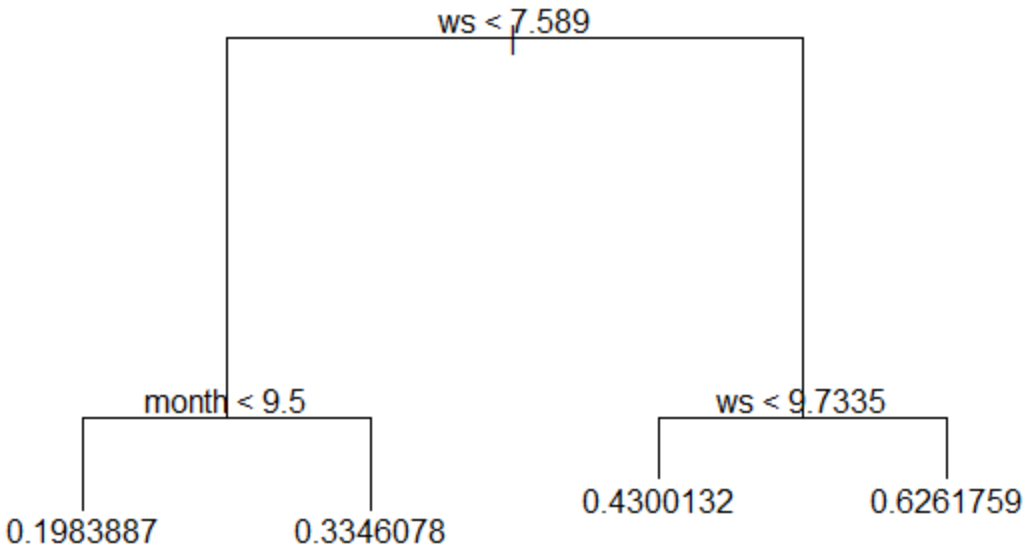
2. Regression Tree

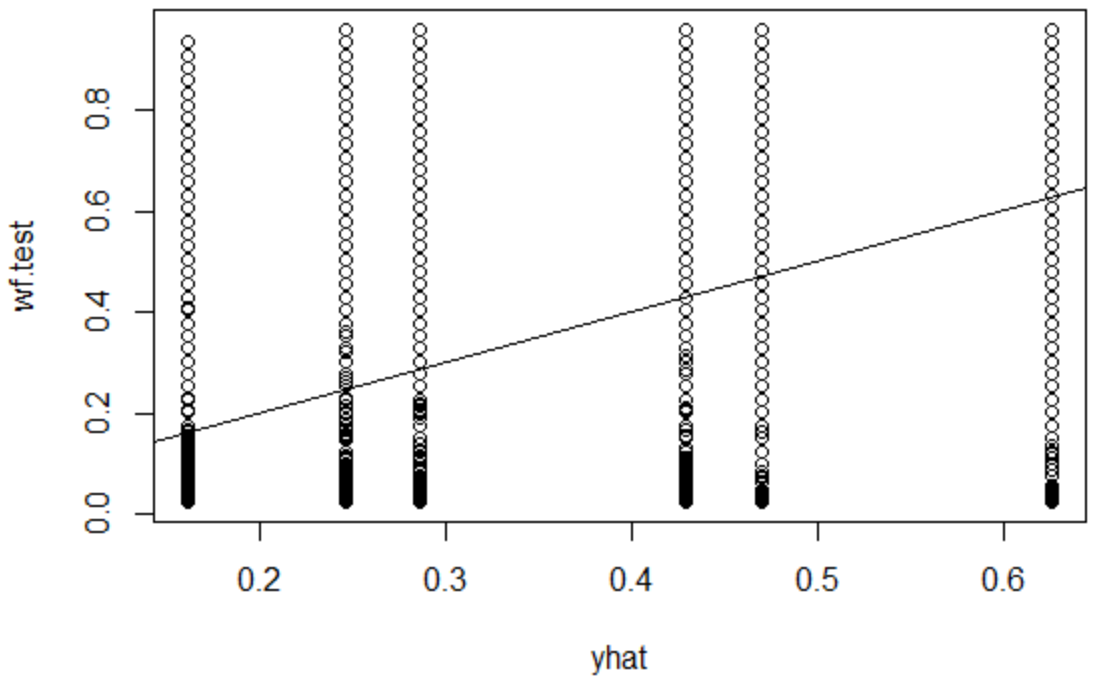
```
Regression tree:
tree(formula = wp7 ~ ., data = wf7_wp7_training, subset = train)
Variables actually used in tree construction:
[1] "ws"    "month" "u"
Number of terminal nodes:  6
Residual mean deviance:  0.0651041 = 428.5152 / 6582
Distribution of residuals:
       Min.     1st Qu.      Median        Mean     3rd Qu.        Max.
-0.60117590 -0.16018890 -0.07601323  0.00000000  0.15698800  0.77171390
```

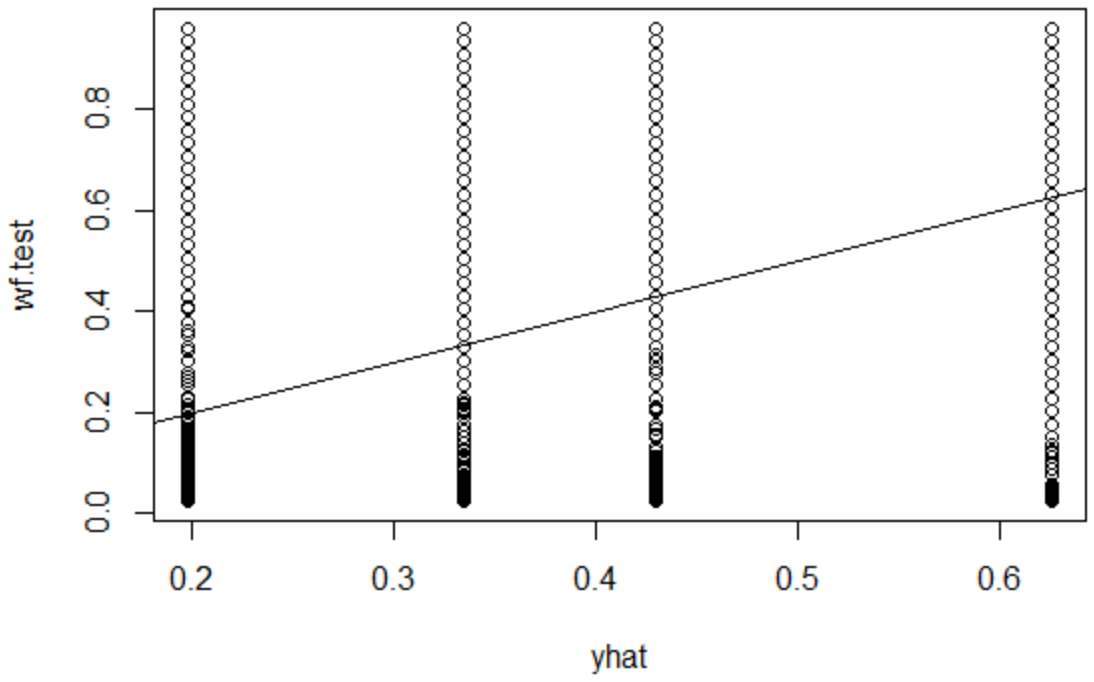
```
   ME       RMSE          MAE          MPE        MAPE
Test set 0.004002024254 0.26347363 0.2145010487 -215.8635908 249.5108729
```


ws < 7.589
month < 9.5
ws < 9.7335
ws < 4.65
u < 3.5265
0.1622861
0.2470120
0.2861889
0.4705383
0.4300132
0.6261759


cv.wf$dev
440
460
480
500
520
540
560
1
2
3
4
5
6
cv.wf$size

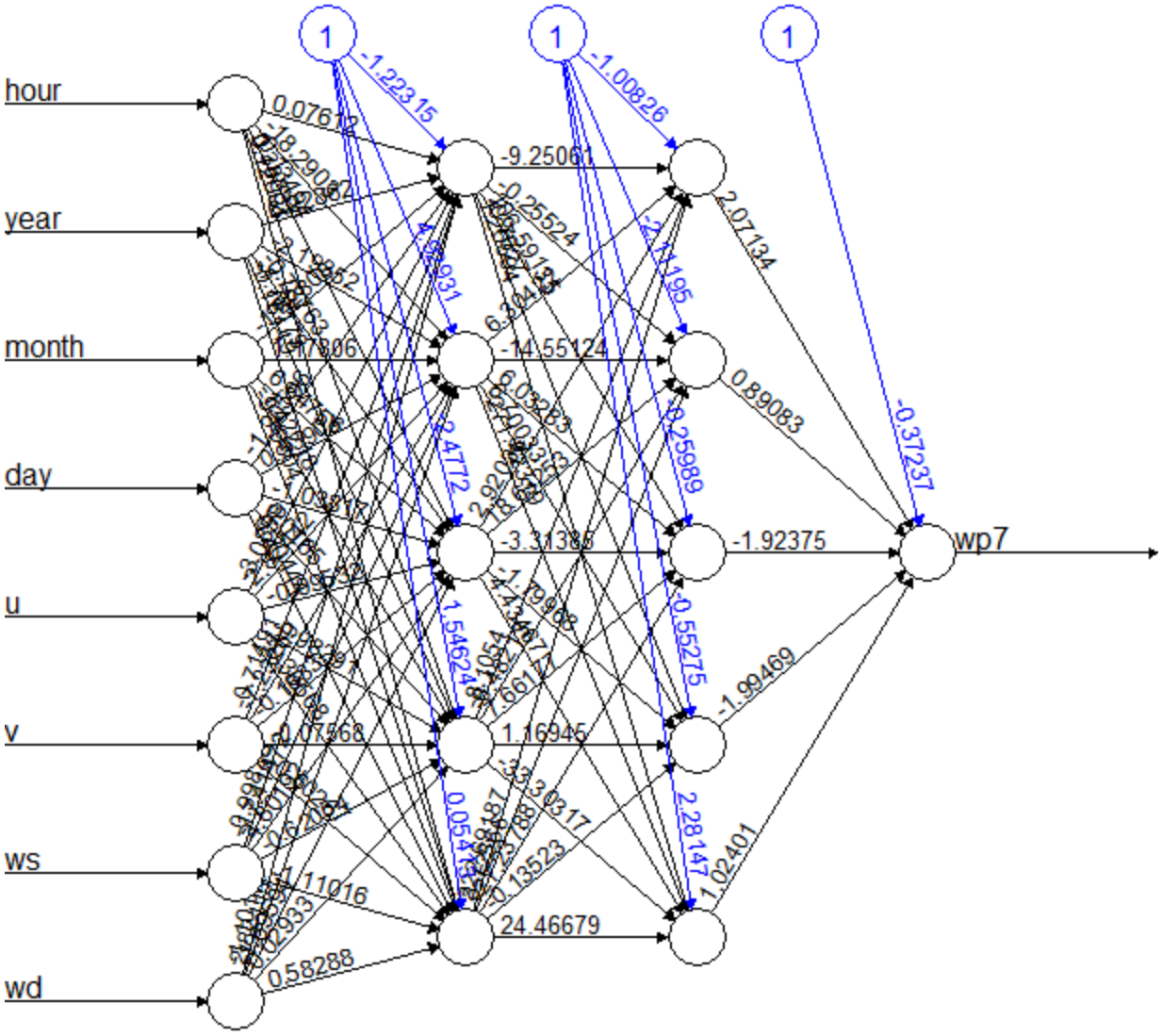

ws < 7.589
month < 9.5
ws < 9.7335
0.1983887
0.3346078
0.4300132
0.6261759


wf.test
0.0
0.2
0.4
0.6
0.8
0.2
0.3
0.4
0.5
0.6
yhat



3. Neural Network

```
Call:
glm(formula = wp7 ~ . - hour - day - v, data = wf7_wp7_training)

Deviance Residuals:
        Min           1Q       Median           3Q          Max
-0.84563300  -0.17933486  -0.07103305   0.16340310   0.82736686

Coefficients:
                    Estimate      Std. Error  t value                Pr(>|t|)
(Intercept) -84.34131543285  10.68295065579 -7.89495  0.0000000000000031333 *
**
year          0.04195109824   0.00531486363  7.89317  0.0000000000000031781 *
**
month         0.00847582076   0.00076532098 11.07486 < 0.00000000000000222 *
**
u             0.01299829601   0.00086688388 14.99428 < 0.00000000000000222 *
**
ws            0.04022286443   0.00090076027 44.65435 < 0.00000000000000222 *
**
wd            0.00016839979   0.00003330497  5.05630  0.0000004332209643958 *
**
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 0.066122871)

    Null deviance: 1118.17391  on 13175  degrees of freedom
Residual deviance:  870.83821  on 13170  degrees of freedom
AIC: 1610.6809

Number of Fisher Scoring iterations: 2
```



Error: 610.99974 Steps: 3378