KADIR HAS UNIVERSITY
GRADUATE SCHOOL OF SCIENCE AND ENGINEERING

PREDICTING ELECTRICITY CONSUMPTION USING MACHINE LEARNING MODELS WITH R AND PYTHON

GRADUATE THESIS

MARYAM EL ORAIBY

August, 2016

PREDICTING ELECTRICITY CONSUMPTION USING MACHINE LEARNING MODELS WITH R AND PYTHON

MARYAM EL ORAIBY

Submitted to the Graduate School of Science and Engineering
in partial fulfillment of the requirements for the degree of
Master of Science
in
INFORMATION TECHNOLOGIES

KADIR HAS UNIVERSITY
August, 2016

KADIR HAS UNIVERSITY

GRADUATE SCHOOL OF SCIENCE AND ENGINEERING

PREDICTING ELECTRICITY CONSUMPTION USING MACHINE LEARNING MODELS WITH R AND PYTHON

MARYAM EL ORAIBY

APPROVED BY:

Prof. Dr. Hasan DAĞ (Advisor) [illegible]

Assoc. Prof. Mehmet N. AYDIN [illegible]

Asstoc. Prof. Söngül ALBAYRAK [illegible]

APPROVAL DATE: 10/08/2016

“I, Maryam El Oraiby, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.”

[illegible]

MARYAM EL ORAIBY

KADIR HAS UNIVERSITY

Abstract

Graduate School of Science and Engineering
Information Technologies

by MARYAM EL ORAIBY

Electricity load forecasting has become an important field of interest in the last years. Anticipating the energy usage is vital to manage resources and avoid risk. Using machine learning techniques, it is possible to predict the electricity consumption in the future with high accuracy.

This study proposes a machine learning model for electricity usage prediction based on size and time. For that aim, multiple predictive models are built and evaluated using two powerful open source tools for machine learning, R and Python. The data set used for modeling is publicly accessible and contains real electrical data usage of industrial and commercial buildings from EnerNOC. This type of analysis falls within the electricity demand management.

Keywords: machine learning, R, Python, predictive modeling, regression, electricity demand management.

Contents

List of Figures

List of Tables

Symbols

AI	Artificial Intelligence
DM	Data Mining
SVM	Support Vector Machine
K-NN	K-Nearest Neighbor
CART	Classification and Regression Tree
ANN	Artificial Neural Network
FCM	Fuzzy c-mean
HCA	Hierarchical Cluster Analysis
RF	Random Forest
CRAN	Comprehensive R Archive Network
GUI	Graphical User Interface
ML	Machine Learning
IDE	Integrated Development Environment
OS	Operating System
LM	Linear Model
GLM	Generalized Linear Model
SST	Sum of Squared Total
SSE	Sum of Squared Error

Chapter 1

Introduction

Just a few decades ago, it was hard to imagine a time where information would be as abundant and easily available as it is today. The pace of data generation is increasing dramatically, creating many challenging questionings related to the information explosion, such as, how to retrieve swiftly the right information and how to extract knowledge out of the enormous available amount of data.

While most of the challenges were before associated to data storage and data collection; nowadays, especially with big data and the increasing capacities of data storage, the challenges concern more and more data analysis with a tendency towards predictive analytics and artificial intelligence. In fact, big data presents many challenges related to data visualization and the need of real-time analytics to meet the need of the competitive business environment.

Recently, companies tend to invest more in data science. According to 451 Research [1], the total data market might almost double in size by 2019 and that from $60bn in 2014 to $115bn in 2019[2].

Data science has become essential to the organizations in diverse sectors. Data analysis including machine-learning techniques can help discover some previously unknown knowledge that can be very critical to the organization's success. From risk analysis and fraud detection to market analysis and customer profiling, this knowledge can lead the organization to achieve remarkable results and is, in itself, a competitive advantage that distinguishes the organization from its competitors. Many enterprises use the capacities of data analytics to create profitable businesses. For instance, in the domain of electricity demand management, there are companies offering energy intelligence solutions in order to enable industrial companies to optimize their energy consumption.

This study highlights the capacities of machine learning techniques, in terms of building predictive modeling using two open source tools R and Python. The approach used is this study aims to build multiple predictive models for electricity consumption based on time (months, hours and days) and size using a real energy data set from EnerNOC. The machine learning algorithms are implemented in both R and Python in the intentions of determining which models have the best performance in both tools and comparing the use of the two powerful programming language for machine learning. The goal is to choose the best model for the data set and evaluate its success.

1.1 Electricity Demand Management

The electricity demand management gained a lot of attention recently due to many factors. The growth of the world population led to an increase of energy consumption all over the globe, notably the electricity need and usage. Demand side management solutions for electricity aim to optimize the energy consumption by analyzing the sources of the electricity, the energy performance and the consumption over time in order to reduce the peaks and need for further generation resources. The electricity demand management helps reduce the electricity consumption, therefore decreases the costs and the need of new power sources.

The energy crisis of early 1970s caused an increase in energy prices, which forced the USA to develop the first demand side management in order to reduce the overall energy consumption[3]. In fact, governments can supervise the electricity consumption by imposing energy policies and controlling the prices. Industrial enterprises that are highly energy dependent are more interested in the electricity demand management in order to efficiently use the energy resources and reduce the costs.

To achieve that, many technologies and solutions are available. The choice of a specific solution depends on the characteristics of each case. Some solutions include simple measures: the reduction of consumption at peak times, by simply turning off some highly power consuming appliances selected in advance, such as the cooling system and encouraging the power consumption at low peak times. Other solutions are relatively more expensive, for instance replacing the electrical motors can be costly at first but highly efficient in the long term. In addition, raising awareness is a key factor to help engage the users in the process of energy saving.

1.2 Motivation

Although the process of collecting and sharing of data is not hard anymore, the analysis of this data is still problematic and many studies focus on this issue. Dealing with the data and trying to discover useful patterns is the challenge. Even more, the key step of data preparation that precedes the mining can be very tricky. Despite the fact that the tools and the techniques have progressed tremendously in the field of data analytics, finding the right algorithm and applying a process that suits each data set's unique characteristics in order to extract the most significant knowledge is far from being a simple task.

The first motivation of the thesis is to face the previously mentioned challenges; the second motivation is the capacity to experiment the power of machine learning models using only open source tools and publicly available data sets. EnerNOC, an energy intelligence software and services company has a publicly available assembly of data sets, which contains electrical consumption data of 100 different commercial buildings. The machine-learning techniques are applied using two open source tools: Python and R. After the implementation of the predictive models, an evaluation is indispensable. In addition, a comparison matrix of R and Python for machine learning is proposed. The objective is to select the best model based on both results in R and Python.

1.3 Research questions

This study mainly tries to answer the following questions:

- What is machine learning and how important is it?
- What are the main techniques and algorithms used in machine learning?
- How to build predictive models using R and Python?
- How accurate the predictive models are regarding the electrical data set?
- Which tool performed best on our data set R or Python?
- What is the best predictive model chosen for the electrical data set?

1.4 Thesis layout

The organization of thesis is the following:

Chapter2: This chapter introduces the concept of machine learning by proposing definitions and clarifying some misconceptions. This chapter also includes an overview of some popular machine learning algorithms.

Chapter3: This chapter explores the tools used for machine learning: R and Python, then proposes a comparison matrix of R and Python for machine learning.

Chapter4: This chapter includes the steps of building machine-learning models in both R and Python and evaluation of the performance based on time and R squared error. In addition to the elaboration of different plots and tests to compare the results and select the best model.

Chapter5: The final chapter provides a conclusion and propositions for future work.

Chapter 2

Machine learning

Nowadays, large amount of data is available and many solutions to process this data already exist. However, what matters most is to process this data fast enough to predict future actions and efficiently intercept risks in order to take the right decision at the right time. Since technology has dramatically improved, the tendency today is towards Artificial Intelligence in the ambition of making the machine think by itself and improve its program without a human intervention.

The technology is developing so fast that we do not perceive Artificial Intelligence as fiction anymore. The day when machine will be able to think, understand, learn and act by itself is coming for sure. Even though we are still unable to imitate the human intelligence, machine learning has achieved a part of the AI goal. In fact, machine learning automates the process of building models, which allows the machine to modify its code without the need of a human interaction. Since machines can analyze and process faster, machine learning proposes solutions to process data and generate models at large scale while being able to improve the result automatically. Thus, machine learning helps reduce the risk of errors and save time for a better decision making.

2.1 What is Machine Learning?

Machine learning intersects with many other sciences such as statistics, mathematics, computer science and natural sciences. Yet, it is also a subset of Artificial Intelligence. Machine learning is often confused -or used interchangeably- with predictive analytics and data mining.

There are many definitions of machine learning. An early definition of Arthur Samuel, 1959 define machine learning as follow: “Field of study that gives computers the ability to learn without being explicitly programmed”. SAS provides another definition: “Machine learning is a method of data analysis that automates analytical model building. Using algorithms that iteratively learn from data, machine learning allows computers to find hidden insights without being explicitly programmed where to look"[4]. Ethem Alpaydın states that "Machine learning is programming computers to optimize a performance criterion using example data or past experience"[5].

From the definitions above, in machine learning, the computer learns by examples. There is no program previously written to solve a certain problem. Though a selection of method is required, the computer creates and updates its own program based on the continuously provided examples. In fact, the main concern of machine learning is to build models that can learn from experience and adjust their actions when exposed to new data, by improving their algorithms without a direct human intervention. In practice, the concept of machine learning can sometimes be confused with artificial intelligence, predictive analytics or even data mining. Clarifying the similarities and differences seems crucial.

2.1.1 Machine Learning VS Artificial Intelligence

Artificial Intelligence has the objective of making the computers intelligent so they can think, communicate and learn the way human minds do. For that aim, a computer must be able to imitate the human behavior. A part of AI needs machine learning, in addition to understanding, reasoning and natural language processing. Machine learning on the other hand is concerned with the creating algorithms and models that can learn from examples and when exposed to new data. According to Neil Lawrence, "machine learning is the principal technology underpinning the recent advances in artificial intelligence" [6].

2.1.2 Machine Learning VS Predictive Analytics

Predictive analytics has a principal goal of building predictive models using different statistical techniques and methods also used in machine learning. Since the predictive modeling focuses on discovering new patterns for decision-making, it represents an important sub-field of data mining.

Generating predictive models can be a part of a machine-learning task in solving a specific problem. Still, machine learning's domain is larger; its goal goes beyond the prediction or the discovery of new patterns and it emphasis on "learning" by making the machine improve its algorithm from experience.

2.1.3 Machine Learning VS Data Mining

Machine learning and Data mining seem to have many common characteristics. However, the main goal of Data Mining is to discover new patterns from data sets in a way that is useful and tangible for the end user. Data Mining can only be successful if it discovers a previously unknown knowledge and an interpretable result for the human perception. Machine learning, besides of extracting knowledge -whether previously known or not- aims to develop complex programs for the own computer understanding, while to be able to improve when exposed to new data in the future.

Data mining and machine learning may seem similar, since data mining uses machine-learning algorithms, such as neural network. In fact, the computational methods used in Data Mining are derived from statistics, machine learning, and artificial intelligence.[7] Likewise, machine learning may have to use one or many data mining steps, like data preprocessing. In fact, Brett Lantz argues that " machine learning algorithms are a prerequisite for data mining, but the opposite is not true. In other words, you can apply machine learning to tasks that do not involve data mining, but if you are using data mining methods, you are almost certainly using machine learning".[8]

Overall, with the increase of data volume, machine learning gains more importance. Today, it is necessary to teach the machine how to learn by itself. Even though machine learning automates the process of learning, the choice of the most adequate algorithm depends on the data analyst.

2.2 Machine learning applications

Machine learning has many applications in different fields. In fact, we see machine learning applications almost everywhere and we use them on a daily basis. For instance, spam filtering. The algorithms enable the distinction of suspicious emails from actual ones, by either their sources or their contents. Spam filtering is one of the most known and important machine learning applications. Even though the algorithms for spam detection are always improving, the spammers are also developing their techniques to be undetectable. In fact, it can be dangerous to classify wrongly a suspicious email as a spam. Paul Graham states that "false positives are so much worse than false negatives that you should treat them as a different kind of error" [9].

Another popular application is web page ranking. Today, a huge amount of web pages is available; the indexed web only, contains at least 4.6 billion pages[10]. However, the retrieval of the best ones for a search request is challenging. For a search engines, the web page ranking is an important factor of success, which was the case for Google. In the same perspective of representing relevant results, there is an application named recommendation systems uses filtering algorithms to recommend targeted advertisements or suggestions to a user based on his own preferences, such as recommending a book from Amazon or suggesting new friends or groups in Facebook.

Fraud detection is another significant application. Multiple machine learning solutions exist for business purposes to protect companies especially in the domain of finance from fraudulent actions. National securities also ML algorithms to detect treats, possible attacks and suspicious individuals. There are many other applications of machine learning that are worth mentioning, including but not limited to speech and hand writing recognition, sentiment analysis, face recognition, weather forecast, health and medical prediction and games.

2.3 Machine learning cycle

The machine learning process depends on the type of problem to solve, however some specific steps are often required. The diagram 2.1 represents the overall cycle of machine learning. Depending on the data and type of the application, some steps can be discarded while other steps can be subject to additional development.

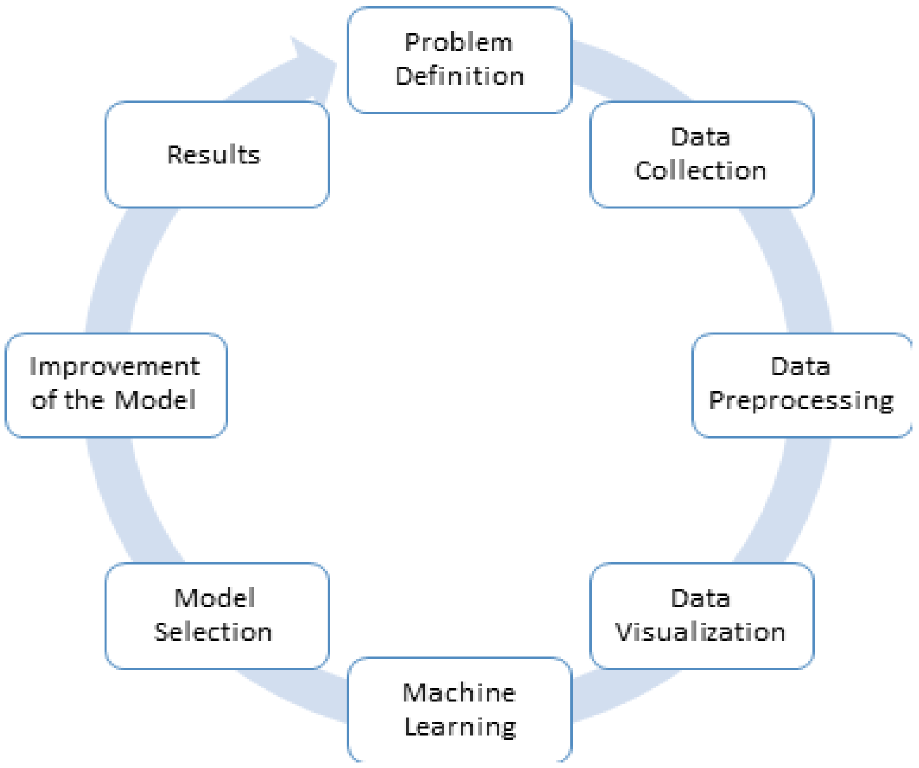


FIGURE 2.1: Machine Learning Cycle

Machine learning cycle consists of a number of steps; the description of each step can be find below:

1. Problem definition: this phase concerns the understanding of the strategy and the business problem.
2. Data Collection: this step is about data collection and aggregation - the data can sometimes be dispersed or in inadequate format. This step concerns the gathering of data for modeling.
3. Data preprocessing: data preprocessing involves of the data preparation and transformation - all the necessary steps to make data ready for processing including data cleaning, data reduction and feature selection.
4. Data visualization: this phase consists of data exploration, by creating plots and can take place after data collection or after data preprocessing, it helps gaining insight into the data and detecting anomalies.
5. Machine learning: this step is the core of machine learning and consists of building models - it includes the splitting of data into a training set and a testing set as well as the application of the different machine learning algorithms.
6. Model selection: This step consists of the evaluation of the models with the test dataset and the selection of the best one.
7. Improvement of the model: this step can be helpful if the selected model needs enhancements, and the development of the model when exposed to new data.
8. Results: The final phase consists of reporting the results/solutions for decision-making.

   The changes in the business strategy and the integration of new data imply systematically a redeployment of the model. In fact, the model must improve when exposed to new data.

2.4 Machine Learning: Model Types

The most known models types of machine learning are the supervised and the unsupervised learning.

2.4.1 Supervised learning

In the supervised learning, the training data is labeled in advance. In other terms, the class or the target of the training set is well defined. The main goal of the supervised learning is to build models for prediction[11]. For that reason, supervised learning is also known as predictive modeling.

Machine supervised learning models are in fact build based on the examples provided in the training data set. The predictive model supposes that the samples in the training data are most correctly classified; the model then learns from the training data set to predict the class of the test set. The best-known tasks used in supervised learning are Regression and Classification.

Classification involves the prediction of categorical or discrete outputs, for example, in the spam filtering application, the class information can be spam or “not spam”. If so, the model classifies the new instances into one of the two predefined classes. The same concept is behind regression; the only difference is that regression predicts continuous outcomes. Predicting the electricity consumption is a regression task, since the target class is a set of continuous values. The goal of regression consists of building a model/function that can predict most accurately this outcome.

2.4.2 Unsupervised learning

In the unsupervised learning, the data has no labels. In fact, the algorithms try to discover the similarities and the differences between the data samples, thus the effort is entirely exploratory and descriptive. Since the training data is unlabeled, we do not predict the class of the new instances as it is the case of supervised learning. For example, in clustering, the goal is to find patterns in order to group the data into clusters of instances that have some common characteristics. Those characteristics are extracted from the statistical distribution of the data. For instance, clustering is widely used in costumer segmentation and helps deliver better-targeted advertisements to distinct groups of customers based on their common behaviors and interests. Unlike clustering, association tries to find patterns between the variables. Association is also useful for analyzing and predicting customer behavior, where the focus is on the items rather than the customers. The association rules state that if a customer buys item X he is more likely to purchase item Y.

Besides the supervised and unsupervised learning mentioned above, other learning methods exist, such as semi-supervised learning and reinforced learning. Semi-supervised learning is a combination of supervised and unsupervised learning. It uses the same techniques of supervised learning. However, the training data in semi-supervised learning combines labeled data and unlabeled data.

Reinforced learning on the other hand, works with the punishment/reward technique. The goal of reinforced learning is to make the machine learn how to improve its actions by rewards. The machine will seek to improve its algorithm by checking the reward function, while trying to maximize the sum of rewards and avoid punishments. The reinforcement learning is mostly used in the development of games like chess and cards; in addition to other important applications such as network routing and control.

2.5 Machine learning algorithms

There is a great number of machine learning algorithms, which are continuously progressing. We provide below a non-exhaustive list of some widely used machine learning algorithms.

2.5.1 Supervised learning algorithms

2.5.1.1 Linear regression

Linear regression is one of the simplest and most popular models used for predictive analysis. The linear regression model consists of creating a linear function to fit the data. Consequently, In order to perform the linear regression, Y the dependent variable must have continuous outcomes. Linear regression supposes that all the variables are independent.
The linear regression equation is

$$(Y\_t = \beta_1 X_{1t} + \beta_2 X_{2t} + ...\beta_k X_{kt} + b\_0)$$

Where Y is the dependent variable to predict, X is the independent variables, β is the regression coefficient and b is the intercept.

2.5.1.2 Logistic regression

The logistic regression is quite similar to the linear regression. However, it is used when the dependent variable Y is categorical, most frequently binary, as 0 or 1. For example, when the outcome is either True or False, Yes or No, etc.

The logistic regression predicts the probability of the variable outcome being true.

2.5.1.3 Elastic net regression

The elastic net regression is a regularized regression has the advantage of its ability to avoid overfitting, using penalty functions to shrink the coefficients in order to reduce variance.

2.5.1.4 Support Vector Machine

Support vector machine is a classification and a regression algorithm that represents the data in term of points in space, and build a model that can separate categories of those points with the maximum margin as in 2.2.

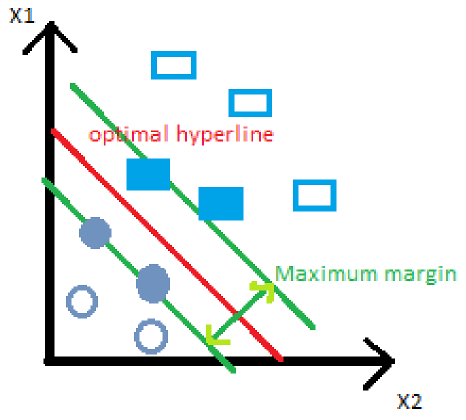
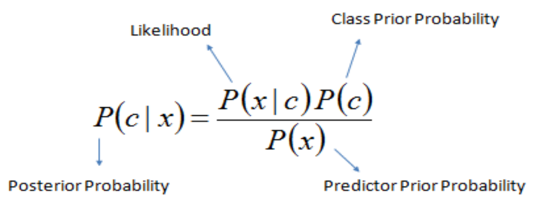


FIGURE 2.2: SVM depiction

2.5.1.5 Naive Bayes classifier

Naive Bayes Classifier is named after Thomas Bayes who proposed the Bayes Theorem[12]. The Naive Bayes Classifier is a probabilistic method that supposes the independence of variables. It consists of calculating the posterior probability of each class category in order to determine the likelihood of the new instances to belong to a certain class as in 2.3.



$$P(c \mid x) = \frac{P(x \mid c)P(c)}{P(x)}$$

$$P(c \mid X) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

FIGURE 2.3: Naive Bayes

2.5.1.6 K-Nearest Neighbor classifier

K-NN Classifier algorithm for classification and regression stores the available data instances and classifies the new instances based on a similarity measure as shown in figure 2.4. K-NN

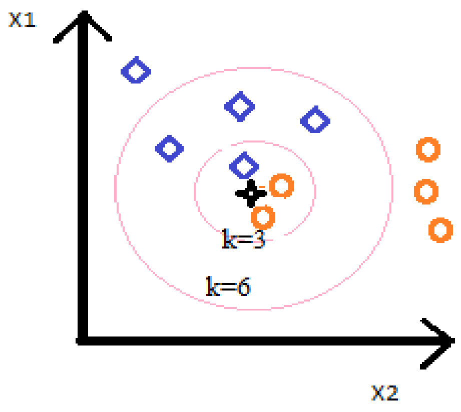


FIGURE 2.4: KNN Classifier with k=3 and k=6

classifies new cases by a majority vote of its k neighbors, using a distance measure calculation.

2.5.1.7 Decision trees

Decision trees are rule based models for classification that work with both categorical and continuous variables. It consists of creating a tree structure starting from a root node that denotes a test and continues in branching until it reaches the terminal node that holds a class label.

The first node must be the attribute that differentiates best the best the instances of the data set.

Below in figure 2.5 an example of a decision tree with R using Rpart package to generate a CART model. The dataset used to generate the decision tree is called Wine[13] and consists of chemical analysis of wines grown in the same region in Italy derived from three different cultivars 1-3. CART stands for Classification and Regression Tree and it is one of the most used algorithms for Decision Trees. CART can be used for both regression (predicting continuous values) and for decision trees model building (class identification).

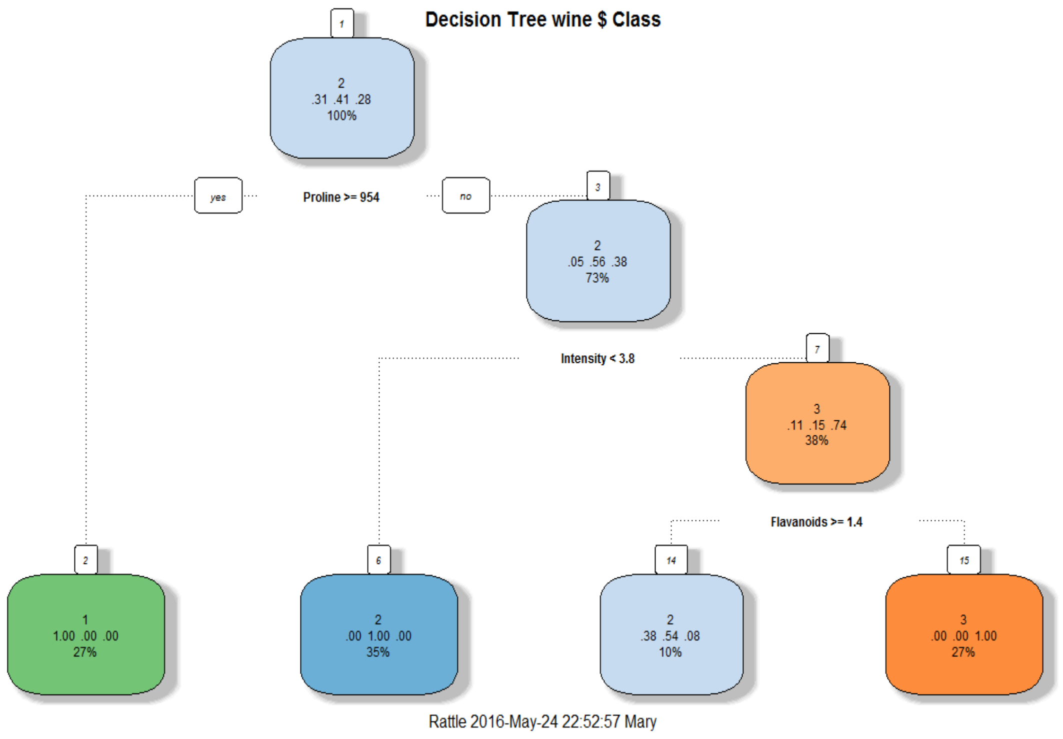


FIGURE 2.5: Decision tree using Rpart

2.5.1.8 Artificial Neural Network

ANN is in fact one of the most known algorithm in ML. That is due to the ANN philosophy, which is to make the machine behave like the human brain. The name of Neural Network comes from the biological neural network existing in the human body. It is an imitation of the network pattern between neurons, where the input and output data are the neurons and the lines representing the connections are the synapses.

In the Artificial Neural Network for supervised learning, the weights of connections between the input layers and the output layers are calculated in order to predict the outputs of the new data see figure 2.7. Artificial Neural Network is also used in unsupervised learning.

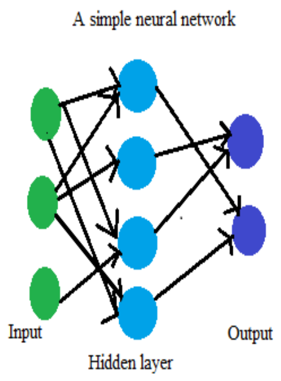


FIGURE 2.6: Illustration of a simple neural network

2.5.2 Unsupervised learning algorithms

2.5.2.1 K-means clustering

K-means clustering is the most used algorithm for clustering. In order to perform K-means clustering, the data must be numerical. The K-Means algorithm tries to find the best division of data points into K groups. See the formula 2.7. The number of K groups or clusters must be assumed in advance. The algorithm will then compute the distances in order to find an optimum centroid point for each cluster. The function describing the process is:

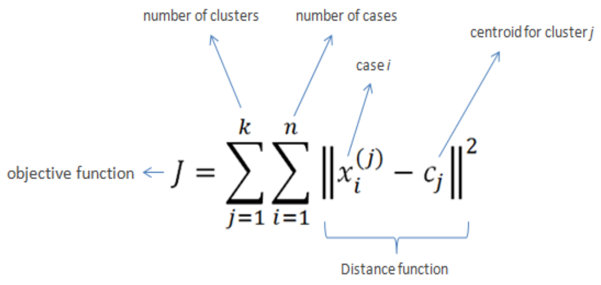


FIGURE 2.7: K-means clustering formula

2.5.2.2 Fuzzy c-mean clustering

FCM is a method of clustering that was developed from the C-Mean clustering method[14]. The difference resides in the fact that in fuzzy c-mean an object can belong to more than one cluster, which means that each data point in one cluster can somehow belong to other clusters at a certain degree. The calculation of the probabilities of each data point belonging to all the other clusters is included in the fuzzy c-mean algorithm.

2.5.2.3 Hierarchical clustering

Hierarchical Cluster Analysis algorithm starts by considering each object as a cluster, and then tries to find the closest object to each cluster to create a new cluster and so on. The clusters are gradually merged until we reach one cluster of all objects. This method is called bottom-up or agglomerative clustering. In contrast, HCA can also be performed in top-down way, which is a less common method, also known as divisive hierarchical clustering. as showen in figures 2.8 , the top-down clustering considers all the objects as one cluster, and then starts to split the cluster until each object defines a singleton cluster[15]. The hierarchical clustering is represented as a dendrogram.

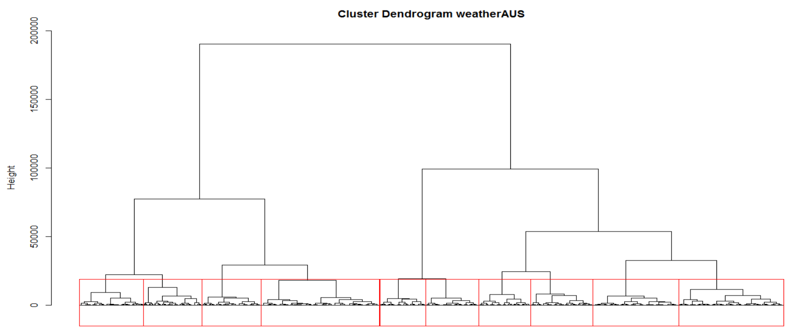


FIGURE 2.8: Example of a dendrogram using R

2.5.3 The Ensemble Learning Algorithms

The ensemble learning methods are different from the previously mentioned algorithms, as they construct multiple models in order to improve the accuracy of the prediction.

Among the most popular ensemble learning used today we find Boosting, Bagging and Random Forest.

2.5.3.1 Boosting

The boosting algorithm is used for classification and regression. The algorithm combines different weak learners in order to produce a good learner for an accurate prediction. There are many algorithms for boosting techniques including AdaBoost.

AdaBoost generates a weak learner generally a decision tree, applies it to the training data to find the ones wrongly classified, then assigns higher weights to the misclassified data in order to focus on those specific observations. Finally, the algorithm generates new learners trying to minimize the weights of misclassified observations until it finds a better model.

2.5.3.2 Bagging

Bagging stands for Bootstrap Aggregation algorithms. It can also be applied for both classification and regression. Bagging creates random samples of the training data with the respect of the size, which means that some observations are duplicated. The algorithm then applies the weak learner on each sample. Depending on the classification/regression results, it selects the best model using vote/average.

2.5.3.3 Random Forest

Random forest is an extension of the bagging method[16]. RF first creates random samples of data and random variables, then the algorithm constructs multiple and independent decision trees, and selects the best model based on a vote in order to improve the prediction accuracy.

In case of regression, random forest uses the average computation based on the generated decision trees in order to improve the results of the prediction.

Chapter 3

R and Python for Machine Learning

Today, we can assume that R and Python are the most popular open tools for Machine Learning. This Chapter presents a general overview of R and Python, their characteristics and usage for machine learning. Lastly, a comparison matrix of R and Python for ML is provided at the end of this chapter.

3.1 About R

R stands for both the statistical programming language and the software environment. It is one of the most popular statistical environment for machine learning and general data analysis. The R Foundation for Statistical Computing holds the copyright of R. R is free under the GNU General Public License.[5].

The source code of R is written in R, C and Fortran. R is free, open source and cross platform with a large community of users. Even though R programming is challenging, the CRAN packages reduce the number of lines coding, with a help command giving explanations and examples of use. R also gained popularity for its fancy visualizations. R is today one of the most -if not the most- powerful solutions for statistical programming and machine learning.

3.1.1 R History

R is a dialect of S programming language [17]. John Chambers initiated S in 1976 at Bell Laboratories, and was released years after as a commercial implementation called S-PLUS. In 1991, Ross Ihaka and Robert Gentleman created R at the University of Auckland in New Zealand, as open source implementation of S for exploratory purposes. When R was first announced in 1993, Martin Machler encouraged Ihaka and Gentleman to release the R source code as free software[18]. The source code of R was made available under the GNU general license of the Free Software Foundation in 1995. The first version 1.0.0 was released in 2000. R gained immediately the attention of researchers.

3.1.2 Why R for machine learning?

There are many advantages of using R for machine learning:

- R is cross platform and works perfectly with GNU/Linux, Mac and Windows.
- R is free and open source, allowing modification and development.
- R accepts different file types (CSV, TXT, SAS, SPSS, Microsoft Excel, Oracle, MySQL…)
- R is a great choice for novices in machine learning.
- R was initially designed for statistical computing and data analysis. R can handle different data structures, missing values, etc.
- Many tutorials and online courses exist on R, in addition to a good number of articles and books.
- R is constantly integrating new technologies and functionalities.
- R has a huge community of users, including academicians and statisticians. Any user can contribute to the development of R packages.
- R is great at visualizations. Creating impressive and high quality graphics is relatively easy in R. In addition, the plots can be imported easily in PDF, JPG or PNG formats.
- It is possible to run Python codes in R using rPython Package (the rPython package also enables calling python functions in R).

Even though R shows many strength points, some downsides can be listed:

- It takes some time to learn R and to get used to its functionalities.
- Some issues may appear occasionally related to the memory and to the availability of some packages.
- Other statistical languages are catching up with R!

3.1.3 R CRAN Packages

CRAN stands for Comprehensive R Archive Network, which is a "collection of sites which carry identical material, consisting of the R distribution(s), the contributed extensions, documentation for R, and binaries" [19].

The R Packages are a collection of functions and data that extend the functionalities of R. R repository disposes of more than 8465 available packages[20] as on May 2016. R comes with a number of preinstalled packages. Other packages can be installed as needed, using the install function: **install.packages(Package)**
Once installed, the package can be called anytime from the library as follows: **library(Package)**
There is a great number of packages for machine learning including:

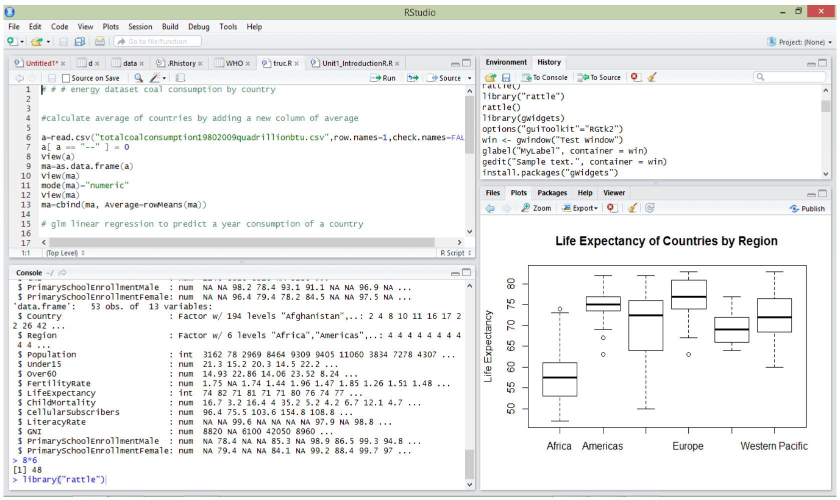
- **mlr:** for general Machine Learning algorithms.
- **caret:** stands for Classification And REgression Training, and it is one of the most used packages for Classification and Regression.
- **CORElearn:** for Classification and Regression. It includes also Feature Evaluation.

Specific algorithms can be installed individually. See the full list of R CRAN Packages at `https://cran.r-project.org/web/packages/`.

3.1.4 Using R

The user can download of R from CRAN via (`https://cran.r-project.org`). Once installed, the user can directly enter commands into the R console. R is command line based program.

Using a GUI like RStudio may be useful. RStudio is an integrated development environment (IDE) for R, written in the C++. It includes "a console, syntax-highlighting editor that supports direct code execution, as well as tools for plotting, history, debugging and workspace management"[21]. See the figure 3.1. RStudio can be downloaded from (`https://www.rstudio.com/products/rstudio/download/`).


RStudio
File Edit Code View Plots Session Build Debug Tools Help
Go to file/function
Project: (None)
Untitled1* d data .Rhistory WHO truc.R Unit1_IntroductionR.R
Source on Save
Run
Source
(Top Level)
R Script
Console ~/
Environment History
To Console To Source
Files Plots Packages Help Viewer
Zoom Export Publish
Life Expectancy of Countries by Region
Life Expectancy
80 75 70 65 60 55 50
Africa Americas Europe Western Pacific

```
# # # energy dataset coal consumption by country


#calculate average of countries by adding a new column of average

a=read.csv("totalcoalconsumption19802009quadrillionbtu.csv",row.names=1,check.names=FAL
a[ a == "--" ] = 0
View(a)
ma=as.data.frame(a)
View(ma)
mode(ma)="numeric"
View(ma)
ma=cbind(ma, Average=rowMeans(ma))

# glm linear regression to predict a year consumption of a country
```

```
 $ PrimarySchoolEnrollmentMale  : num  NA NA 98.2 78.4 93.1 91.1 NA NA 96.9 NA ...
 $ PrimarySchoolEnrollmentFemale: num  NA NA 96.4 79.4 78.2 84.5 NA NA 97.5 NA ...
'data.frame':   53 obs. of  13 variables:
 $ Country                      : Factor w/ 194 levels "Afghanistan",..: 2 4 8 10 11 16 17 2
2 26 42 ...
 $ Region                       : Factor w/ 6 levels "Africa","Americas",..: 4 4 4 4 4 4 4 4
4 4 ...
 $ Population                   : int  3162 78 2969 8464 9309 9405 11060 3834 7278 4307 ...
 $ Under15                      : num  21.3 15.2 20.3 14.5 22.2 ...
 $ Over60                       : num  14.93 22.86 14.06 23.52 8.24 ...
 $ FertilityRate                : num  1.75 NA 1.74 1.44 1.96 1.47 1.85 1.26 1.51 1.48 ...
 $ LifeExpectancy               : int  74 82 71 81 71 71 80 76 74 77 ...
 $ ChildMortality               : num  16.7 3.2 16.4 4 35.2 5.2 4.2 6.7 12.1 4.7 ...
 $ CellularSubscribers          : num  96.4 75.5 103.6 154.8 108.8 ...
 $ LiteracyRate                 : num  NA NA 99.6 NA NA NA NA 97.9 NA 98.8 ...
 $ GNI                          : num  8820 NA 6100 42050 8960 ...
 $ PrimarySchoolEnrollmentMale  : num  NA 78.4 NA NA 85.3 NA 98.9 86.5 99.3 94.8 ...
 $ PrimarySchoolEnrollmentFemale: num  NA 79.4 NA NA 84.1 NA 99.2 88.4 99.7 97 ...
> 8*6
[1] 48
> library("rattle")
```

```
rattle()
library("rattle")
rattle()
library(gWidgets)
options("guiToolkit"="RGtk2")
win <- gwindow("Test Window")
glabel("MyLabel", container = win)
gedit("Sample text.", container = win)
install.packages("gWidgets")
```

FIGURE 3.1: RStudio interface

3.1.5 Rattle

Rattle (R Analytical Tool To Learn Easily)[22] is GUI for data mining that is worth mentioning 3.2. It was developed using R by Graham Williams. Rattle can easily be installed using the following commands:

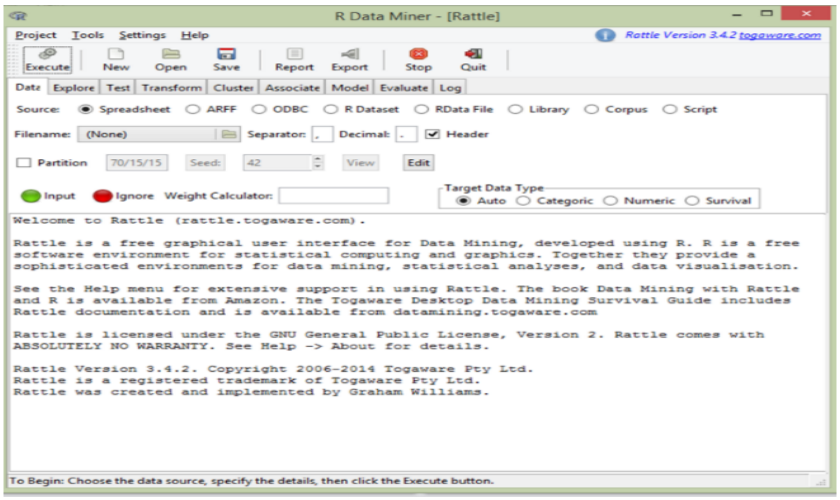
install.packages("RGtk2")

install.packages("rattle")

library("rattle")

rattle()

Rattle is itself an R package. It is built on the statistical language R, and an understanding of R is not required in order to use Rattle[23]. However, for more sophisticated data mining applications, the experienced user will progress to interacting directly with R.


R Data Miner - [Rattle]
Project Tools Settings Help
Rattle Version 3.4.2 togaware.com
Execute New Open Save Report Export Stop Quit
Data Explore Test Transform Cluster Associate Model Evaluate Log
Source: [✓ Spreadsheet] ARFF ODBC R Dataset RData File Library Corpus Script
Filename: (None) Separator: , Decimal: . [✓ Header]
Partition 70/15/15 Seed: 42 View Edit
Input Ignore Weight Calculator:
Target Data Type [✓ Auto] Categoric Numeric Survival

```
Welcome to Rattle (rattle.togaware.com).

Rattle is a free graphical user interface for Data Mining, developed using R. R is a free
software environment for statistical computing and graphics. Together they provide a
sophisticated environment for data mining, statistical analyses, and data visualisation.

See the Help menu for extensive support in using Rattle. The book Data Mining with Rattle
and R is available from Amazon. The Togaware Desktop Data Mining Survival Guide includes
Rattle documentation and is available from datamining.togaware.com

Rattle is licensed under the GNU General Public License, Version 2. Rattle comes with
ABSOLUTELY NO WARRANTY. See Help -> About for details.

Rattle Version 3.4.2. Copyright 2006-2014 Togaware Pty Ltd.
Rattle is a registered trademark of Togaware Pty Ltd.
Rattle was created and implemented by Graham Williams.
```

To Begin: Choose the data source, specify the details, then click the Execute button.

FIGURE 3.2: Rattle GUI

3.1.6 R Community

According to Revolution Analytics, R has a global community of more than 2 million users and developers[24] around the globe, who contribute to the extension and development of R. Inside this community there is the "R Core Group" of developers who have access to the source code of R to insure its development and sustainability.

3.1.7 Books on R

There are many books on R including general introductions to R, data mining/machine learning with R, R programming for specific applications, visualizations and statistical analysis with R, etc. Some interesting books are listed below:

- William N. Venables and David M. Smith (2004), **An Introduction to R** , Network Theory Ltd, ISBN: 978-0954161743
- Alain Zuur, Elena N. Ieno and Erik Meesters (2009), **A Beginner's Guide to R (Use R!)** , Springer, ISBN: 978-0387938363
- Gareth James, Daniela Witten, Trevor Hastie and Robert Tibshiran (2013) **An Introduction to Statistical Learning: with Applications in R** , Springer, ISBN: 978-1461471370
- John M. Chambers (2008), **Software for Data Analysis: Programming with R** . Springer, New York, ISBN 978-0-387-75935-7
- Peter Dalgaard (2008), **Introductory Statistics with R** , 2nd edition. Springer, ISBN 978-0-387-79053-4
- W. John Braun and Duncan J. Murdoch (2007), **A First Course in Statistical Programming with R** . Cambridge University Press, Cambridge, ISBN 978-0521872652.
- Max Kuhn and Kjell Johnson (2013), **Applied Predictive Modeling** . Spinger, ISBN: 978-1461468486

3.2 About Python

Python is a general-purpose, interpreted and object-oriented programming language. It was created by Guido van Rossum in 1989 based on the ABC language. Python was first published in 1991. It is highly readable with a simple and clear syntax.

Python is open source and cross-platform. It is today a strong tool for machine learning, and often compared to R in terms of popularity. It is the most common choice amongst accomplished developers and at the same time accessible to new programmers[25]. Python is used by hundreds of thousands of developers worldwide in different domains[26].

3.2.1 Why python for machine learning?

Many users prefer Python to perform machine learning for multiple reasons. Some of them are:

- Python is open source and free, cross platform
- Python is a fast and a powerful programming language with clear and simple syntax, very intuitive and easy to learn.
- Python allows flexibility and freedom of development.
- Python has reliable scientific libraries for computations and machine-learning algorithms, like Scikit-learn.
- Python has a good community of users.
- Some good books and tutorials are available concerning machine learning applications with Python.
- It can easily be integrated with other programming languages, such as C and Java.

Python is a great choice for general programming and for machine learning. However, non-developers are often reticent to use Python for machine learning. In addition, Python has a relatively limited documentation compared to its scope and capabilities.

3.2.2 Python libraries

Python has some useful libraries for ML, such as scikit-learn, PyBrain and mlpy. Scikit-learn is the most perfected library for machine learning and data analysis that incorporate a great number of algorithms. Scikit-learn has the following dependencies:

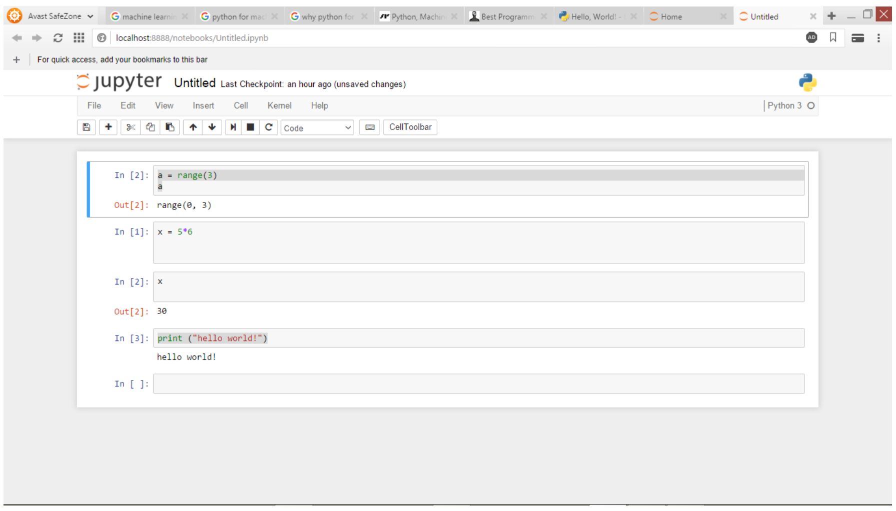
- **numpy:** a powerful library to support and manipulate data structures especially N-dimensional arrays.
- **scipy:** used for scientific computation and contains routines for numerical integration.
- **matplotlib:** produces high quality 2D plots.

3.2.3 Using Python

Python if free and cross-platform, it can be installed easily from the official website (https://www.python.org/). Python is generally installed by default in Linux based OS.

Python codes can be executed interactively from the python shell or by writing python scripts with .py extension. Another way is to use IPython (https://ipython.org).

To start with Python, it may seem helpful to download Anaconda, the scientific distribution of Python, which is a Modern open source analytics platform powered by Python [27]. It includes the IPython Notebook known now as the Jupyter Notebook (see figure 3.3), which is an interactive environment for python programming, data analysis and visualization. The installation requires Python 2.7 or Python 3.3 version and greater.



```
In [2]: a = range(3)
        a
Out[2]: range(0, 3)

In [1]: x = 5*6

In [2]: x
Out[2]: 30

In [3]: print ("hello world!")
        hello world!

In [ ]:
```

FIGURE 3.3: Jupiter Notebook

3.2.4 Community of Python

Several social media groups, blogs and forums propose solutions and help to Python users. There is a community of users called Python User Groups, where advanced python developers help and assist new python users. These groups also organize monthly meetings open to all. There are about 401 Python user groups around the world with an estimated 127,100 members [28].

The Python Software Foundation organizes the Python Community Awards for the members who show significant contribution to the Python community.

3.2.5 Books on Python

To start with python, here are some intereseting books for general Python programming:

- Lutz, Mark (2011), **Learning Python** . O'Reilly 5th Ed. ISBN: 978-1449355739
- Zelle, John M. (2010), **Python Programming: An Introduction to Computer Science**.Franklin. ISBN: 860-1200643879

For machine-learning with python, these books may be helpful:

- Raschka, Sebastian (2015),**Python Machine Learning** . ISBN: 978-1783555130
- Grus, Joel (2015). **Data Science from Scratch: First Principles with Python** . O'Reilly Media. ISBN: 978-1491901427

3.3 Comparison Matrix of R and Python for Machine Learning

Both R and Python are great solutions for machine learning. They are both open source, free and easy to install. The choice of a tool over the other is subjective and depends on the users' preference and personal experience. Python is a great solution if the user wants to learn a powerful programming language for not only machine learning and statistics. Besides, if the user is already familiar with python or with a similar programming language, Python seems a natural choice. Python is more flexible and gives the freedom of development. R on the other hand, is more suitable if the user is specifically interested in statistical computations and visualizations. For non-programmers, R can be very exciting; the user can easily do data explorations and create some amazing plots with little effort. Many tutorials and books are available for beginners, and the community is always welcoming new R users. Still, mastering R programming language demands a lot of time, patience and determination.

In machine learning, R disposes of a great collection of packages evolving almost every day. What makes R great is that any user can contribute in developing new packages. Python on the other hand has few packages for machine learning that endorse a good number of algorithms. Even that cannot determine which language is the best. In fact, either in R or in Python, the user can integrate foreign algorithms and data from other languages, such as rPthon in R and rPy in Python. When installing the packages in Python, the user must pay attention to the dependencies. In R, when installing new packages, the dependent packages are automatically suggested for installation. Yet, occasional incompatibilities of packages may occur if built on different versions of R. Even though R can be slow when dealing with large data set, it suggests different solutions to deal with the issue such as packages to increase speed, Packages for parallel computing (parallel) and RHadoop for Big Data (https://cran.r-project.org/web/views/HighPerformanceComputing.html).

		R	**Python**
Generalities	Purpose	Statistical computing	General purpose programming
	Open source and free	Yes	Yes
	Cross platform	Yes	Yes
	Graphical user interfaces	Yes, e.g.RStudio	Yes, e.g.IPython
	Creation date	1991	1989
	First release	1995	1991
	Current version	R 3.3.0	Python 3.5.1
	Language of development	Similar to S, Written in R, C and Fortran	Based on ABC, Written in C
	Core libraries under free license	Yes	yes
	Possibility of language integration	Yes, e.g. C, C++, Java, Python	Yes, e.g. C, C++, Fortran, Java, R
Knowledge and support	Users' preferences	Data analysts, Statisticians, Academicians	Developers, C programmers, Data analysts
	User's contribution in development	High	Moderate
	Community of support	Huge, e.g. R-Bloggers R User Groups	Good, e.g. Python User Groups
	Documentation	Abundant e.g. Rhelp RDocumentation	Good e.g. PyData
Syntax	Simple and easy to learn	Moderate	yes
	Concise and precise	Yes	Yes
	Easy to read	Moderate	Yes English-like
	Static/Dynamic typing	Dynamic	Dynamic

Machine Learning	Libraries/Packages For ML algorithms	Great number of packages for specific algorithms .e.g. rpart, glm, randomForest	Few libraries endorsing the main algorithms e.g. Scikit-learn, PyBrain
	Most used ML package	_	Scikit-learn
	Data visualization	Excellent e.g. ggplot2, ggvis	Very Good matplotlib
	ML performance with large data sets	Relatively slow	Very good
	Big data integration	Yes e.g. RHadoop	Yes e.g. Hadoopy
	Integration of ML algorithms from other tools/languages	Yes e.g. rWeka, rPython	Yes e.g. rpy2
	Books on Machine-Learning	Yes	Yes

TABLE 3.1: Comparison matrix of R and Python for ML

The comparison matrix is distributed into four principal modules as shown in the table 3.1:

1. Generalities: presents a general overview of R and python.
2. Syntax: describes the syntax of the languages.
3. Knowledge and support: shows the scope of the documentation and the community of support.
4. Machine learning: general comparison of using R and Python for machine learning.

Chapter 4

Machine learning models with R and Python to predict electricity usage

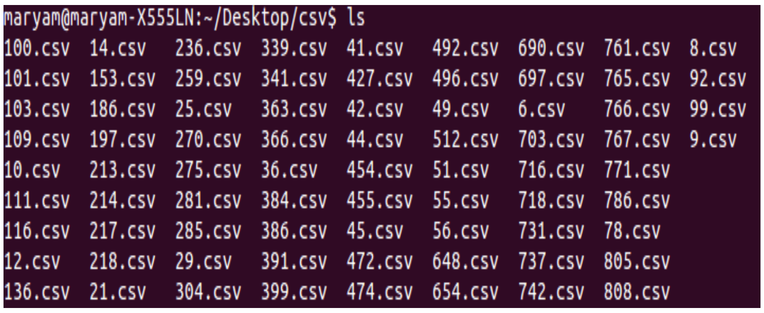
4.1 Introduction

R and Python are two powerful open source tools for applying machine learning algorithms and techniques. The objective of this chapter is to build different models in both R and Python for electrical power prediction. The determination of the performance of the models with both tools helps the selection of the best model fitting the data set. The process of this work is the following:

- Presentation of data set and the objectives.
- Data preprocessing and data integration.
- Data exploration.
- Presentation of the algorithms to use in both R and Python.
- Building models.
- Implementation of the models in R and Python.
- Evaluation and comparisons.
- Testing results.

4.1.1 The data set

The energy data set used for this study is the EnerNOC Open Project electrical data set of 100 buildings, accessible from the link:
https://open-enernoc-data.s3.amazonaws.com/anon/index.html
The data set contains electrical power usage for every 5-minute of 100 distinct commercial/industrial sites in different region of the USA in 2012. The data set folder contains 100 files, each file belongs to a unique building, named as Building_ID.csv. See 4.1. As for the start of this



```
maryam@maryam-X555LN:~/Desktop/csv$ ls
100.csv  14.csv   236.csv  339.csv  41.csv   492.csv  690.csv  761.csv  8.csv
101.csv  153.csv  259.csv  341.csv  427.csv  496.csv  697.csv  765.csv  92.csv
103.csv  186.csv  25.csv   363.csv  42.csv   49.csv   6.csv    766.csv  99.csv
109.csv  197.csv  270.csv  366.csv  44.csv   512.csv  703.csv  767.csv  9.csv
10.csv   213.csv  275.csv  36.csv   454.csv  51.csv   716.csv  771.csv
111.csv  214.csv  281.csv  384.csv  455.csv  55.csv   718.csv  786.csv
116.csv  217.csv  285.csv  386.csv  45.csv   56.csv   731.csv  78.csv
12.csv   218.csv  29.csv   391.csv  472.csv  648.csv  737.csv  805.csv
136.csv  21.csv   304.csv  399.csv  474.csv  654.csv  742.csv  808.csv
```

FIGURE 4.1: Data file names

study, there has been no article published yet on this data set, except for an exploratory work by Clayton Miller, who used this data set to explore the capacities of IPython. His work can be accessed via the link below:

http://nbviewer.jupyter.org/github/cmiller8/
EnerNOC-100-Building-Open-Dataset-Analysis

4.1.2 Objectives

The main objective of this work is to find the best model to predict the electricity usage. For this aim, it is necessary to build different predictive models and compare their performances. The models have to predict the electrical consumption based on the variables of time (months, days and hours) and the size (square_footage) of the buildings. Implementing the models on the test sets allows the evaluation of the performance; in addition to other parameters such as time and R squared error. Hence, it is possible to estimate which models predict best the electrical usage for this data set.

4.2 Data Preparation

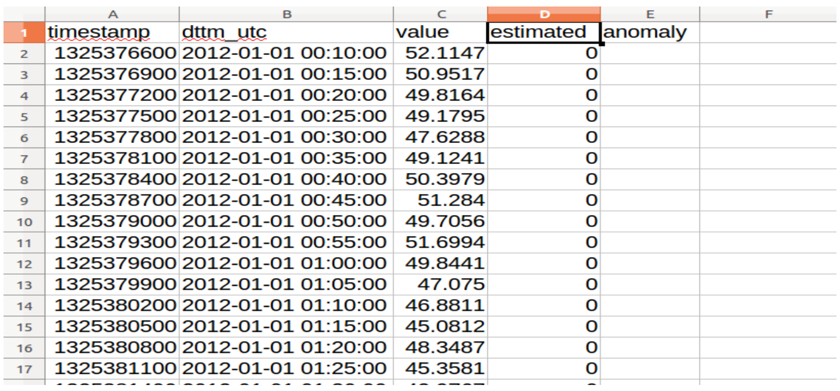
The data set must go through many steps before modeling, including data preparation, data reduction and data transformation.

4.2.1 Data preprocessing and integration

Initially, each data file contains the following features: Timestamp, Date/ time, reading value, estimated indicator and anomaly indicator.

- The "estimated indicator" indicates if the reading was estimated: 0 for yes, 1 for no.
- The "anomaly indicator" is non-blank if there is an error in reading.
- The "reading value" is the power usage measured in kWh.

In addition to the data files, there is a meta data folder that contains information on the buildings, including the type of the industry, square footage, lat/lng and timezone. It is necessary to add the feature of "square footage" to the data prior to modeling, since we intend to integrate buildings with different sizes. An example of the initial state of the data files can be observed in the figure 4.2.



	A	B	C	D	E	F
1	timestamp	dttm_utc	value	estimated	anomaly	
2	1325376600	2012-01-01 00:10:00	52.1147	0		
3	1325376900	2012-01-01 00:15:00	50.9517	0		
4	1325377200	2012-01-01 00:20:00	49.8164	0		
5	1325377500	2012-01-01 00:25:00	49.1795	0		
6	1325377800	2012-01-01 00:30:00	47.6288	0		
7	1325378100	2012-01-01 00:35:00	49.1241	0		
8	1325378400	2012-01-01 00:40:00	50.3979	0		
9	1325378700	2012-01-01 00:45:00	51.284	0		
10	1325379000	2012-01-01 00:50:00	49.7056	0		
11	1325379300	2012-01-01 00:55:00	51.6994	0		
12	1325379600	2012-01-01 01:00:00	49.8441	0		
13	1325379900	2012-01-01 01:05:00	47.075	0		
14	1325380200	2012-01-01 01:10:00	46.8811	0		
15	1325380500	2012-01-01 01:15:00	45.0812	0		
16	1325380800	2012-01-01 01:20:00	48.3487	0		
17	1325381100	2012-01-01 01:25:00	45.3581	0		

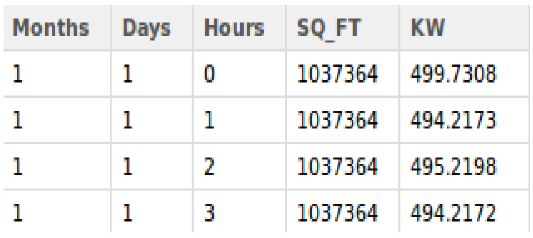
FIGURE 4.2: Example of file 6.csv

4.2.2 Feature reduction

The features of estimated indicator and anomaly indicator must be deleted (their values in all files have confirmed the correctness of the readings) and they will not be used for ML.

4.2.3 Data transformation

The time variable must be included as separate features of months, days and hours . In addition, there is a necessity of computing the sum of load consumption of every 5 minutes to one hour. Hence, transforming data into an hourly consumption measurement. We also need to include the square footage from the meta data into our data set. The final data set integrates all the



Months	Days	Hours	SQ_FT	KW
1	1	0	1037364	499.7308
1	1	1	1037364	494.2173
1	1	2	1037364	495.2198
1	1	3	1037364	494.2172

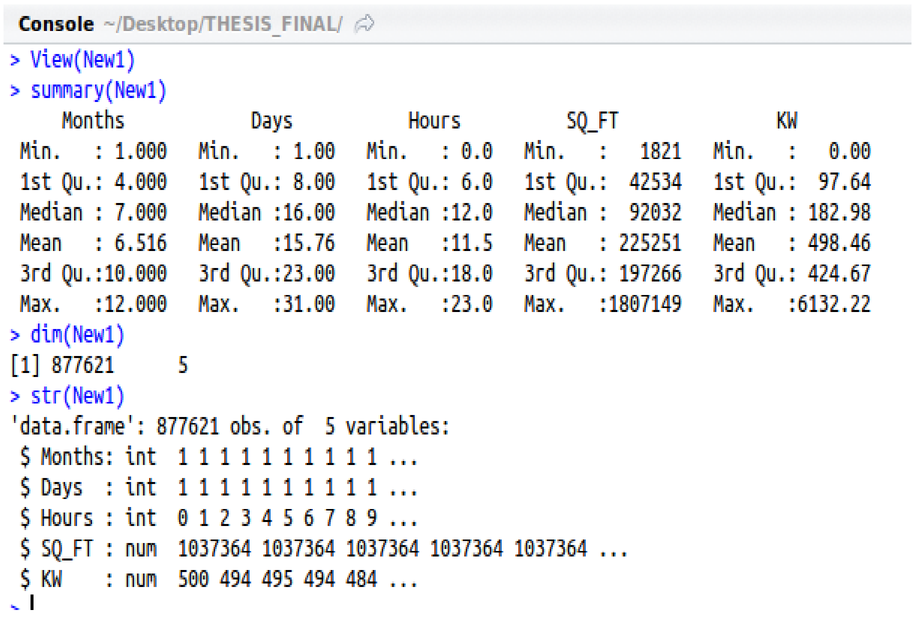
FIGURE 4.3: Overview of the final data set

building into one data set with a dependent variable of electricity consumption/hour. See the figure 4.3.

The data set has a total of 5 features and 877621 rows. The features are Months, Days, Hours, SQ_FT and KW for the power consumption.

4.3 Data Exploration

There are many ways to explore data in R using functions such as summary() and str(). These options offer a general overview of the data, such as min and max values, missing values, mean, median, quartiles, data types and data size, as show in the figure 4.4. At this point, it is possible to discover anomalies and irregularities.



```
Console ~/Desktop/THESIS_FINAL/
> View(New1)
> summary(New1)
     Months          Days           Hours          SQ_FT              KW
 Min.   : 1.000   Min.   : 1.00   Min.   : 0.0   Min.   :   1821   Min.   :   0.00
 1st Qu.: 4.000   1st Qu.: 8.00   1st Qu.: 6.0   1st Qu.:  42534   1st Qu.:  97.64
 Median : 7.000   Median :16.00   Median :12.0   Median :  92032   Median : 182.98
 Mean   : 6.516   Mean   :15.76   Mean   :11.5   Mean   : 225251   Mean   : 498.46
 3rd Qu.:10.000   3rd Qu.:23.00   3rd Qu.:18.0   3rd Qu.: 197266   3rd Qu.: 424.67
 Max.   :12.000   Max.   :31.00   Max.   :23.0   Max.   :1807149   Max.   :6132.22
> dim(New1)
[1] 877621      5
> str(New1)
'data.frame':	877621 obs. of  5 variables:
 $ Months: int  1 1 1 1 1 1 1 1 1 1 ...
 $ Days  : int  1 1 1 1 1 1 1 1 1 1 ...
 $ Hours : int  0 1 2 3 4 5 6 7 8 9 ...
 $ SQ_FT : num  1037364 1037364 1037364 1037364 1037364 ...
 $ KW    : num  500 494 495 494 484 ...
> |
```

FIGURE 4.4: Example of data exploration functions in R

It is also possible to generate plots to understand the distribution and the characteristics of the data set. As an example, we can generate a plot of Energy Consumption by months, see the figure 4.5. We first sum up the power consumption of each month and plot the results to see which month has the highest total of electricity consumption.

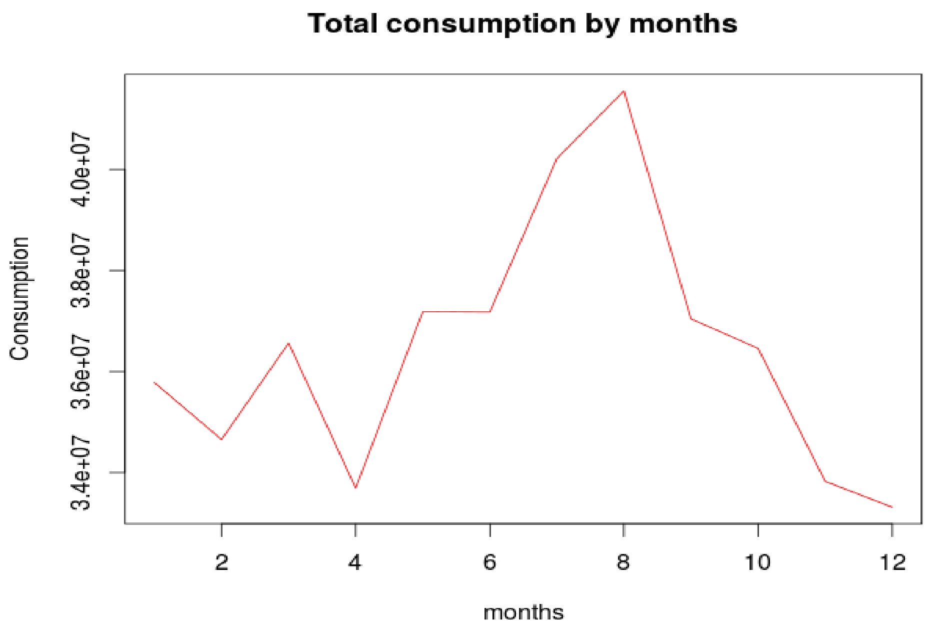


FIGURE 4.5: Plot of electricity consumption by months using R

From the plot we can observe that in the summer the electricity consumption is higher than it is in the other months and specially in August which reveals the highest peak of electricity consumption. The figure 4.6 also shows the standard deviation of the data per months.

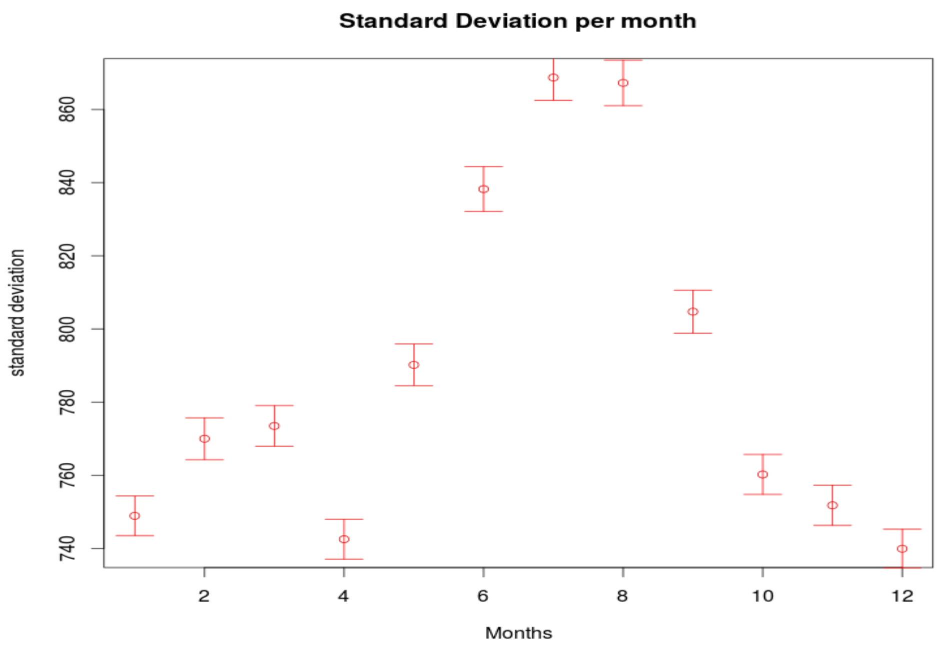


FIGURE 4.6: Plot of standard deviation by month

4.4 Algorithms for prediction

The algorithms belong to the family of supervised learning because the data has a well defined class information. Since the dependent variable KW has continuous values, regressions algorithms are the choice for the prediction.

There are many algorithms for regression: weak learners such as linear regressions, tree based algorithms and strong learners like random forest.

Different algorithms will be experimented in both R and Python. The choice of the models for this study is based on the selection of the models that perform successfully in both R and Python and show an R squared at least similar or higher than the R squared of a simple linear regression model.

The evaluation of the performance of each algorithm is based on the error calculation of R squared error and time spent on modeling and prediction, in addition to plots and error calculation on the test set to verify the correctness of the models. The following list presents the algorithms chosen to elaborate regression models in both R and Python:

- Linear Regression
- Elastic Net Regression
- Decision Tree
- Random Forest
- Bagging
- K-NN for Regression

4.5 implementation

Prior to modeling, splitting data into a training set and a test set is required. In fact, it is easy to split the data set into training and testing set with integrated split functions in both R and Python. For the sake of comparison, we use the same training data and test data for R and Python. With python, we split data into two separate files: train.csv and test.csv. The percentage of the split is 30% for test and 70% for training. The predictive models are built on the training sets using R and Python. The performance of every model is evaluated based on the following parameters:

- **Time complexity:** the total running time needed to elaborate the model on the training set and the prediction on the test set.The time calculation was run five times and the average time was computed for every model.
- **R squared error:** as for time, the R squared error calculation was run five time and the average was computed. R squared error indicate how good the model fits the data. R squared is always between 0 and 1, with 1 being the perfect fit.
  R2=1 SSE/SST , where **SSE** is the sum of squared error and **SSR** the sum of squared total.
- **Plots using the entire data set:** Fitting the models to the entire data set / months
  This method consists of using every model to predict the consumption on the entire dataset, and sum up the results by months.
- **Plots using the test set:** Fitting the models to the test set
  This method consists of comparing the predicted values to the real values for every instance in the test set. For aim of clarity in the visualization, a summation of the electricity consumption is performed every 20.000 row of the test set. The second part of this test consists of calculating the absolute error of every model on the test set.

The comparison of the models is discussed at each step of the evaluation. The best model is selected based on the overall performance.

4.5.1 Machine Learning with Python

4.5.1.1 The libraries

For ML in Python, Scikit-learn library contains a great number of algorithms. The following modules of machine learning algorithms must be imported from the Scikit-learn library.

***Random forest:**
```
from sklearn.ensemble import RandomForestRegressor
```
***K-NN regression:**
```
from sklearn import neighbors
```
***Linear model:**
```
from sklearn import linear_model, metrics
```
***Elastic Net regression:**
```
from sklearn.linear_model import ElasticNet
```
***Decision tree regression:**
```
from sklearn.tree import DecisionTreeRegressor
```
***Bagging regression:**

from sklearn.ensemble import BaggingRegressor

It is required to import the module r2_score in order to calculate the R squared error.

* **R squared error:**

from sklearn.metrics import r2_score

4.5.1.2 The algorithms

The following built in functions are used to elaborate the predictive models in Python. X represents the independent variables of the training set and y the dependent variable. Xt stands for the the independent variables of the test set. ***random forest regressor:**

rfr=RandomForestRegressor()

rfr.fit(X,y)

predictions=rfr.predict(Xt)

***K-NN regression:**

fknn1 = neighbors.KNeighborsRegressor()

knn1.fit(X, y)

predictions=knn1.predict(Xt)

***Linear model:**

regr1 = linear_model.LinearRegression(_X=True, fit_intercept=True, n_jobs=1, normalize=False)

regr1.fit(X, y)

predictions=regr1.predict(Xt)

***Elastic Net regression:**

enet = ElasticNet(alpha=0.1, l1_ratio=0.7)

enet.fit(X, y)

predictions=enet.predict(Xt)

***Decision tree regression:**

clf=DecisionTreeRegressor()

clf.fit(X, y)

predictions=clf.predict(Xt)

***Bagging regression:**

a=BaggingRegressor(DecisionTreeRegressor())

a.fit(X, y)

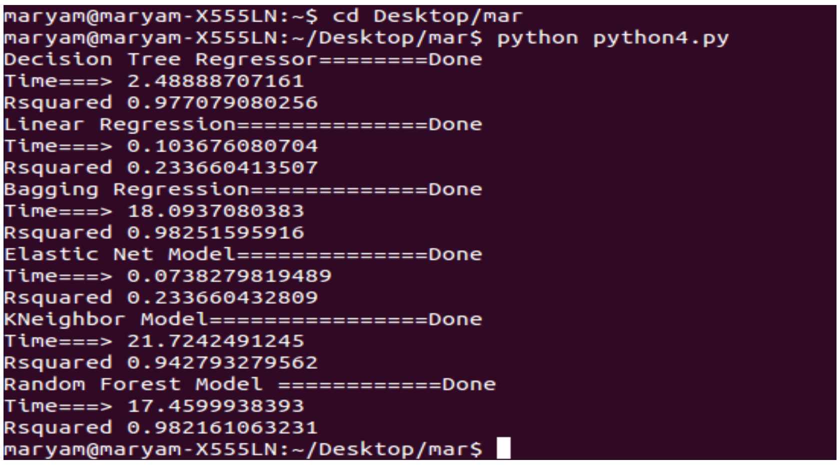
predictions=a.predict(Xt)

The performance of the model is measured in Time, R squared error and later on using examples to test the performance of the data.

In python, there is an integrated function of R squared error function under the **r2_score** module. **yt** refers to the real values of the independent variable in the test set, while predictions are the predicted values of the algorithm.

***R square error**
r2_score(yt, predictions)

***Time complexity**
Time is calculated using time function, which requires to import time and datetime modules. It starts the count of seconds with the command time() and stops with the same command. Two variables are generated of time calculation. The result is the difference between the two variables. The method of work chosen in python consists of using a python script executed from the shell4.7.
start = time.time()
### the algorithm
end = time.time()
The results of the algorithms are shown below. For every model, the time calculation and the R squared error in computed and returned. A good model must have a coefficient of determination close to 1.



```
maryam@maryam-X555LN:~$ cd Desktop/mar
maryam@maryam-X555LN:~/Desktop/mar$ python python4.py
Decision Tree Regressor========Done
Time===> 2.48888707161
Rsquared 0.977079080256
Linear Regression==============Done
Time===> 0.103676080704
Rsquared 0.233660413507
Bagging Regression=============Done
Time===> 18.0937080383
Rsquared 0.98251595916
Elastic Net Model==============Done
Time===> 0.0738279819489
Rsquared 0.233660432809
KNeighbor Model================Done
Time===> 21.7242491245
Rsquared 0.942793279562
Random Forest Model ============Done
Time===> 17.4599938393
Rsquared 0.982161063231
maryam@maryam-X555LN:~/Desktop/mar$ 
```

FIGURE 4.7: The Results of Python

4.5.1.3 Evaluation of the models

As we shown in table 4.1. The models with the highest **R Squared** are **Random Forest Model** and **Bagging regression** with **R2=0.98**. The fastest models are **Elastic Net Model** and the linear regression, but with a weak **R2** of **0.23**. The slowest model of the selection is **KNN** with **21.72s**. The R squared of this model is **0.94**.

Model	**Time**	**R-squared error**
Decision Tree Regressor	2.48888707161	0.977079080256
Linear Regression	0.103676080704	0.233660413507
Bagging Regression	18.0937080383	0.98251595916
Elastic Net Model	0.0738279819489	0.233660432809
KNeighbor Model Regression	21.7242491245	0.942793279562
Random Forest Model	17.4599938393	0.982161063231

TABLE 4.1: Comparison of error rate of the models in Python

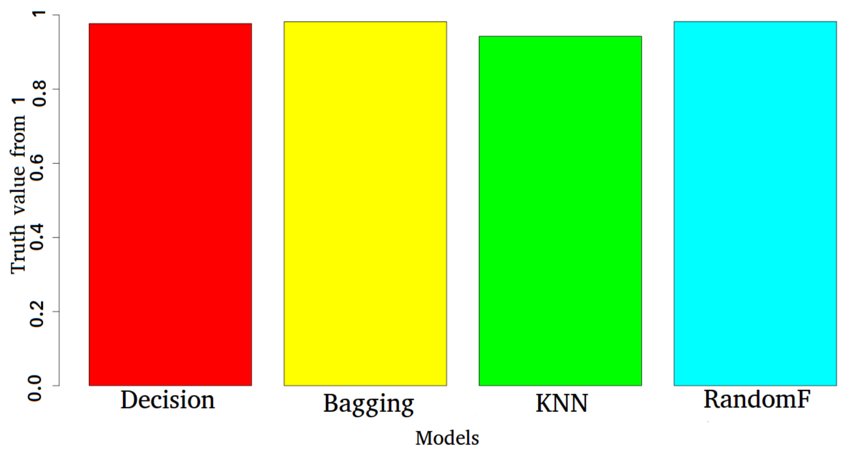


FIGURE 4.8: Comparative plot of R-squared error

The next steps of the evaluation and testing include only the models with an R squared higher than the R squared of a simple linear regression (0.233), i.e Decision tree, Random Forest, Bagging and KNN for regression. The results can be observed better with plots 4.8.
The figure 4.9 shows the comparative of time.

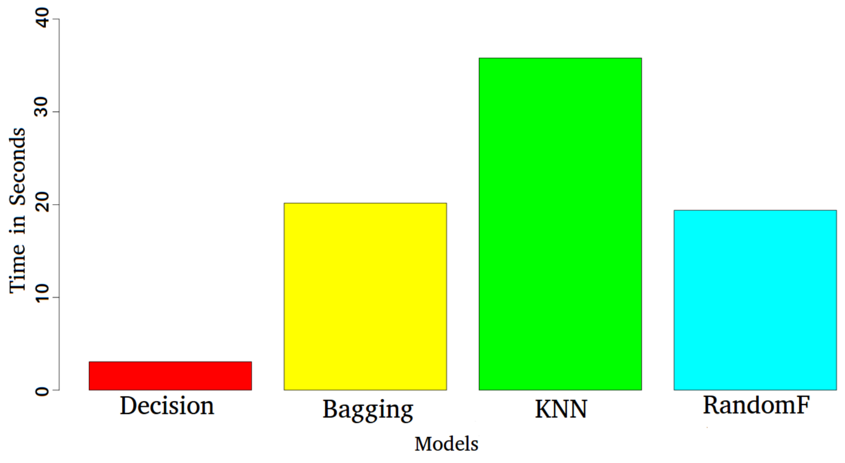


FIGURE 4.9: Comparative plot of Time

4.5.1.4 Testing the models

1- Results of the predictions on the entire data set/months

The plot 4.10 shows a close fit of Random Forest, bagging and decision tree to the real values of the data set, followed by KNN.

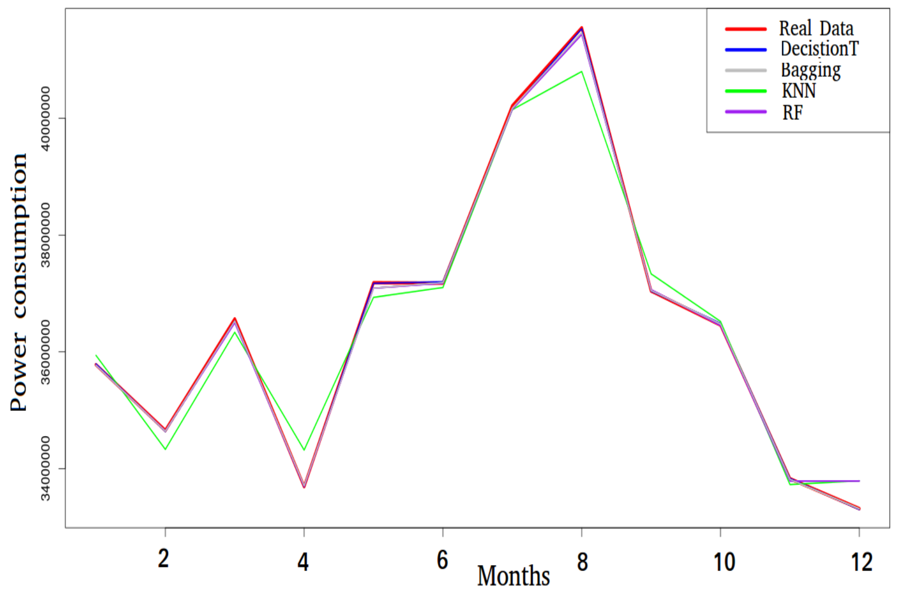


FIGURE 4.10: Comparative plot of the predictions per months using the entire data set in Python

2- Results of the predictions on the test set

We first elaborate a plot to compare the real values of the consumption in test set with the predicted values of the models. The figure 4.11 shows a good performance of the decision tree, random forest and bagging. The predictions of KNN on the other hand is not as good as the other tree based models. Te second part of the test consists of computing the absolute error of the models.

The second part of the test consists of computing the absolute error of every model. As a result of the prediction on the test set using Python, the best model was the decision tree showing the least absolute error, followed by bagging and the decision tree. KNN was far behind with the highest absolute error rate4.12.

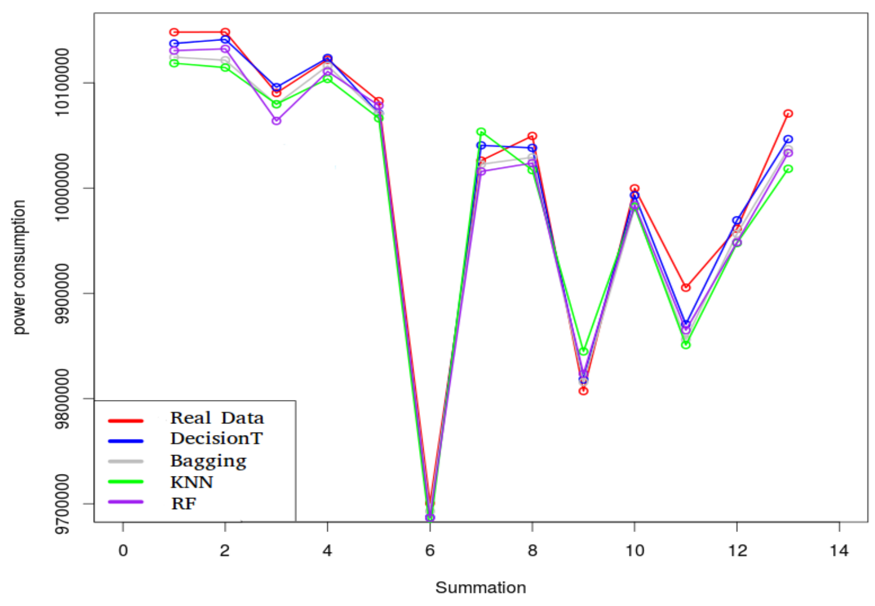


FIGURE 4.11: Comparative plot of the test prediction vs real values

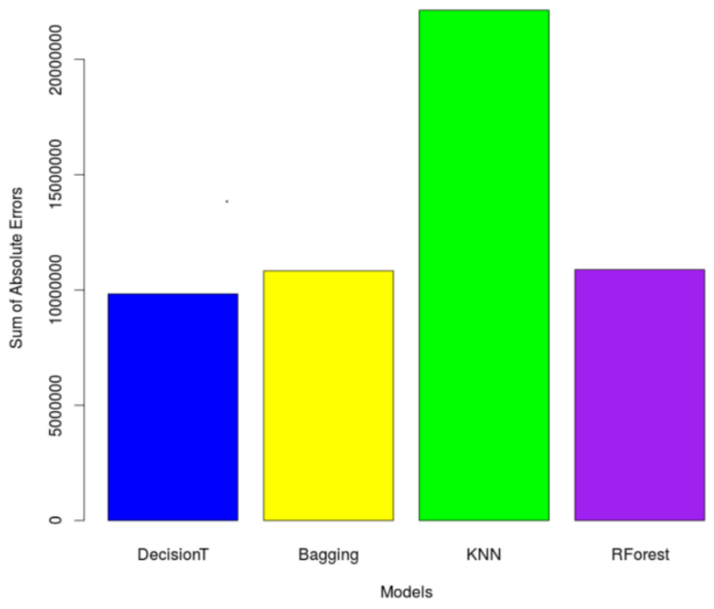


FIGURE 4.12: Comparative plot of the absolute error using python

4.5.2 Machine learning with R

4.5.2.1 The libraries

The libraries required for the data modeling in R are:

FNN:	Fast Nearest Neighbor Search Algorithms and Applications
rpart:	Recursive Partitioning and Regression Trees
randomForestSRC:	Random Forests for Survival, Regression and Classification
glmnet:	Lasso and Elastic-Net Regularized Generalized Linear Models
ipred:	Bagging for classification, regression and survival trees.

R squared error is calculated as follows,
error= function(predictions,y){
R2 =1 - (sum((y-predictions)^{2})$/sum((y-mean(y))^{2})$)
}

4.5.2.2 The algorithms

***Decision tree regression**

```
>t1=system.time({
+ Decision = vector()
+ fit = rpart(Y  .,data=Train , control=rpart.control(minsplit=15))
+ predictions = predict(fit, Test)
+ e1=error(predictions,Test$Y) +predict(fit,Test)
+ })
>e1
0.8032879
>t1
user system elapsed
15.487 0.000 15.488
```

***Linear regression**

```
>t2=system.time({
+ linear = vector()
+ fit = lm(Y  .,data=Train)
+ predictions = predict(fit, Test)
+ e2=error(predictions,Test$Y)
+})
>e2
0.2336604
>t2
user system elapsed
0.772 0.376 0.692
```

***Bagging**

```
+ bagging = vector()
+ fit = bagging(Y   .,data=Train)
+predictions = predict(fit, Test)
+ })
>e3
0.8032891
>t3
user system elapsed
65.760 0.099 65.833
```

*** Elastic Net regression**

```
>t4=system.time({
+ elastic = vector()
+x=as.matrix(Train[,1:4])}
+fit = glmnet(x,Train$Y, family="gaussian", alpha=0.7, lambda=0.1 )
+ xt=as.matrix(Test[,1:4])
+ predictions = predict(fit,xt,type="link")
+ e4=error(predictions,Test$Y)
+ })
>e4
0.2336604
>t4
user system elapsed
0.140 0.000 0.432
```

***KNN regression**

```
>t5=system.time({
+ knn¡-vector()
+ fit = knn.reg(train=Train[,1:4],test=Test[1:4],y=Train[,5],k=5)
+ predictions = fit$pred
+ e5=error(predictions,Test$Y)
+})
>e5
0.9410216
>t5
user system elapsed
2.460 0.000 2.462
```

*** Random Forest**

```
>rf=vector()
>t6=system.time({
+ fit= rfsrc(Y   .,data=Train,ntree=10)
+ v=predict(fit,Test[1:4])
+ e6=error(v$predicted,Test$Y)
+ })
>e6
0.9604368
>t6
user system elapsed
39.158 0.003 39.158
```

4.5.2.3 Evaluation of the models

Model	**Time**	**R-squared error**
Decision Tree Regressor	15.487	0.8032879
Linear Regression	0.772	0.2336604
Bagging Regression	65.760	0.8032891
Elastic Net Model	0.140	0.2336604
KNeighbor Model Regression	2.460	0.9410216
Random Forest Model	39.158	0.9604368

TABLE 4.2: Comparison of error rate of models in R

In R, Random Forest has the highest R squared error of 0.96, followed by KNN (0.941) with an impresive time of execution (1.87s), see table 4.2. Bagging is the slowest model of the selection (65.76s) and has almost the same R squared of the decision tree (0.8). The fastest models are the linear and the elastic network (0.77 and 0.14 seconds respectively), but both have very low R-squared of 0.23 and will not be selected for evaluation.

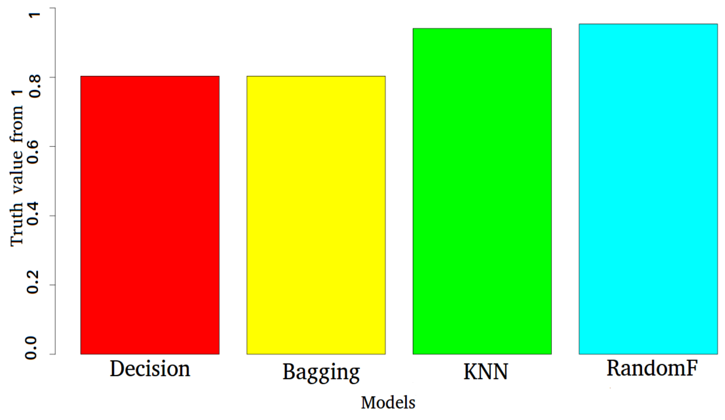


FIGURE 4.13: Comparative plot of time

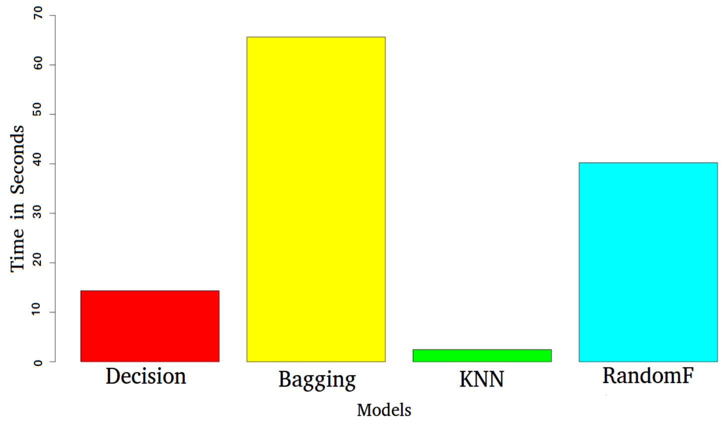


FIGURE 4.14: Comparative plot of R squared error in R

4.5.2.4 Testing the models

1- Results of the predictions on the entire data set/months.

The plot 4.15 shows how well the models perform with the data set. The Random forest shows the closest line to the real values. Even though decision tree has a high R squared, the performance of the model on the data is poor, which is due to the problem of underfitting.

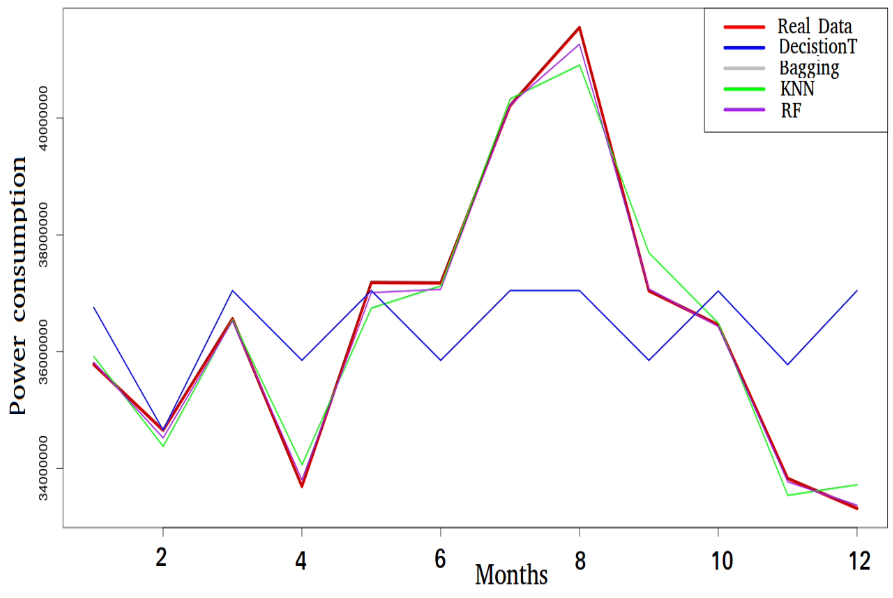


FIGURE 4.15: Comparative plot of the predictions per months using the entire data set in R

2- Results of the predictions on the test set

From the plot 4.16 of test prediction, when comparing the predicted values of the models to the real values in the test set, a different behavior of the models is observed. Random Forest seems to work better than the other models with the test set4.17.

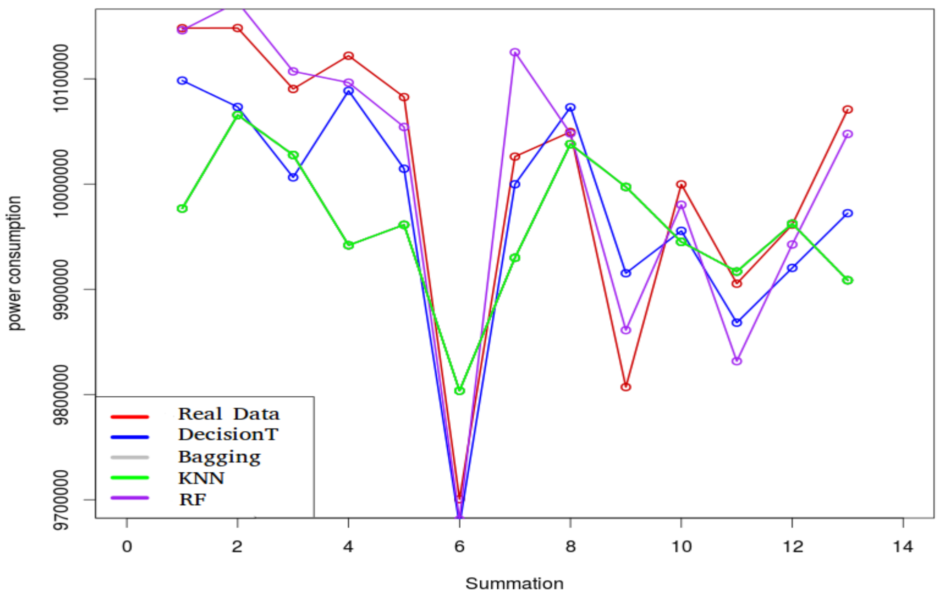


FIGURE 4.16: Comparative plot of the test prediction vs real values

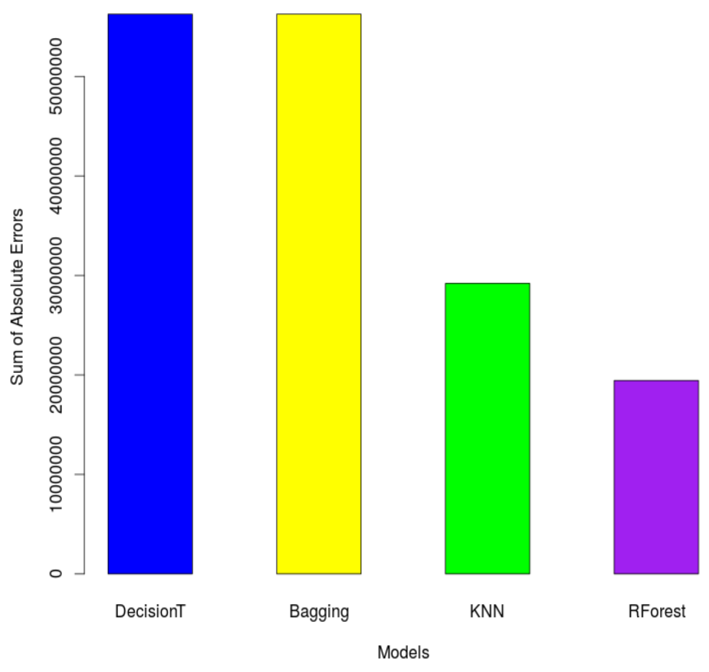


FIGURE 4.17: Comparative plot of the absolute error using R

4.6 Comparison of the models built with R and Python

4.6.1 Comparison based on time

The plot 4.18shows that in both R and Python, linear regression and elastic regression models were generated fast. Random Forest and Bagging were relatively slow using both tools. This is due to the nature of the models. The first two generate a simple equation that fits the data. In contrast, Random Forest and the Bagging generate multiple subsets of the data and different decision trees to select the best model based on average calculation.

The model that took most time in R was Bagging, while in Python, KNN regressor was the slowest one. KNN regressor can be slow, since it computes the distances between the data to find similarities. R was significantly faster than Python in KNN regression. However, python appears to be the largely faster with the other algorithms: bagging, decision tree and random forest.

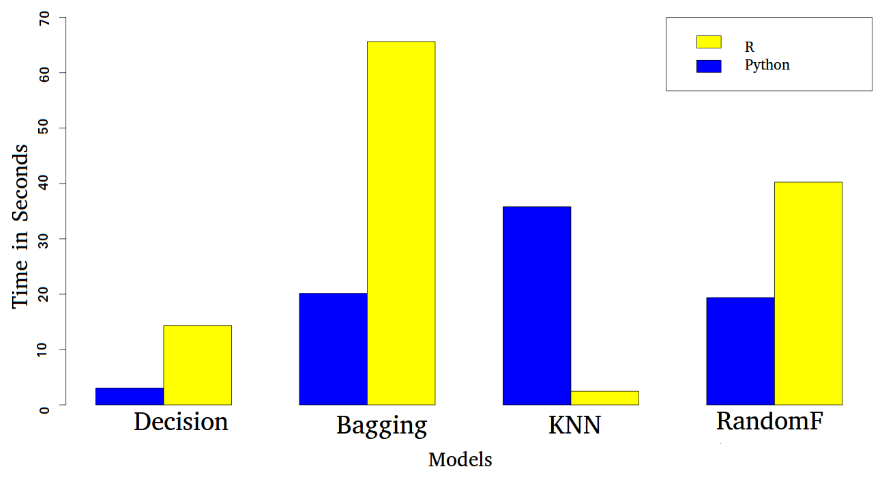


FIGURE 4.18: Comparative plot of Time performance in R and Python

4.6.2 Comparison based on R squared error

From the R squared comparison, both Python and R models seem to behave similarly in most cases, which reinforces the solidity of the models. The best model in both tools is the Random Forest. Nevertheless, time complexity and R squared error are not enough to determine the

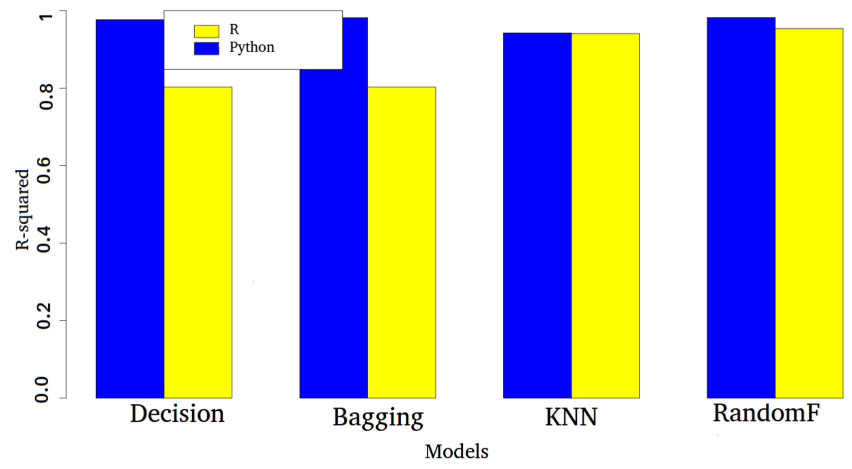


FIGURE 4.19: Comparative plot of error rate in R and Python

exactitude and the performance of the models. In fact, many experimentations must be performed on the data set, including comparisons, calculation of the errors on samples of the data and enhancement of the models by tuning the parameters.

To sum up the experience of using R and Python for machine learning, they both are powerful programming languages and provide many options and libraries for machine learning. Random Forest appears to be a solid model in both implementations. Based on R squared error rates4.19, time and tests on the models, it seems that python performed slightly better than R in this study.

Python is gaining more popularity in machine learning and R is still present with a strong community of users. The choice of the right algorithm for modeling depends on the data type and size. Some algorithms are more known to be more efficient than others. However, since every data has its own characteristics, the wise move is to try multiple algorithms and perform iterative tests before the implementation.

Chapter 5

Conclusion and future work

It is interest of every electricity market player to know how much electricity usage would be in the future in order to better manage and control the energy resources, reduce costs and avoid risk.

Although there are many models towards electricity load forecasting, the uniqueness of this study is the usage of both time and size as features of a real historical electricity usage data to build models based on machine learning, with two open source tools. This thesis is the first application of machine learning models on the EnerNOC Open Data Project. Multiple models were built using R and Python. The evaluation of the models' performance was based on time, R squared error and different tests on the data. Random Forest model appears to perform best throughout the process of evaluation.

As future work, we recommend the addition of data from next months and years, which would enhance the prediction and improve the model's performance. Other factors that influence the electricity usage such as weather may also reveal interesting patterns. As the size of the buildings is the most significant feature, the division of the data into subsets based on size may lead to a more precise prediction.

Bibliography

[1] 451 research, . URL https://451research.com. Accessed on February 14, 2016.

[2] 451 Research. Predicts total data market to hit 115 billion by 2019. URL https://451research.com/images/Marketing/press_releases/5.28.15_Total_Data_PR_Final.pdf. Accessed on February 14, 2016.

[3] *The History of Energy Efficiency.* Alliance Commission on National Energy Efficiency Policy, 1850 M Street, NW : Suite 600 : Washington, DC 20036, January 2013.

[4] Sas, . URL http://www.sas.com/en_us/insights/analytics/machine-learning.html.

[5] Alpaydın Ethem. *Introduction to Machine Learning.* The MIT Press Cambridge, Massachusetts, 2 edition, 2010. ISBN 978-0-262-01243-0.

[6] Neil Lawrence. What is the relation between artificial intelligence and machine learning? URL https://www.quora.com/What-is-the-relation-between-.....Artificial-Intelligence-and-Machine-Learning.

[7] Vijay Kotu and Bala Deshpande. *Predictive Analytics and Data Mining Concepts and Practice with RapidMiner.* 2014.

[8] Lantz and Brett. *Machine Learning with R.* 2 edition, October 2013. ISBN 978-1782162148.

[9] Graham Paul. Better bayesian filtering. *Spam Conference*, January 2003. URL http://www.paulgraham.com/better.html.

[10] World wide web size, . URL http://www.worldwidewebsize.com.

[11] S. B. Kotsiantis. *Supervised Machine Learning: A Review of Classification Techniques.* Informatica, 2007.

[12] Vijaykumar B, Vikramkumar, and Trilochan. Bayes and naive-bayes classifier. April 2014.

[13] Uci machine learning repository, . URL http://archive.ics.uci.edu/ml/datasets/Wine.

[14] Ghosh Soumi and Dubey Sanjay Kumar. Comparative analysis of k-means and fuzzy c-means algorithms. *International Journal of Advanced Computer Science and Applications*, 4(4):36, July 2013.

[15] Murtagh Fionn and Contreras Pedro. Algorithms for hierarchical clustering: an overview. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2(4):86–97, 2012. URL ttp://dx.doi.org/10.1002/widm.53.

[16] Zhi-Hua Zhou. *Ensemble Methods Foundations and Algorithms*. Taylor Francis Group, 2012. ISBN 978-1-4398-3005-5.

[17] Roger Peng. *Data Mining with Rattle and R: The Art of Excavating Data for Knowledge Discovery*. Springer, 3 edition, April 2016. ISBN 978-1365056826.

[18] Ihaka Ross. R : Past and future history. 1998. URL https://cran.r-project.org/doc/html/interface98-paper/paper.html.

[19] Hornik Kurt. Frequently asked questions on r. URL https://cran.r-project.org/doc/FAQ/R-FAQ.html#What-is-CRAN_003f.

[20] Cran r project, . URL https://cran.r-project.org/web/packages/.

[21] Rstudio, . URL https://www.rstudio.com/products/rstudio/.

[22] Williams Graham. Rattle: A data mining gui for r. *The R*, 1:45, December 2009.

[23] Graham Williams. *Data Mining with Rattle and R: The Art of Excavating Data for Knowledge Discovery*, volume 1/2. Springer, 3 edition, December 2011. ISBN 978-1441998897.

[24] Revolution analytic, . URL https://www.continuum.io/why-anaconda.

[25] Halterman Richard. Learning to program with python, 7 2011.

[26] Lutz Mark. *Programming Python*. O'Reilly, 3 edition, 2011. ISBN 978-0596009250.

[27] Anaconda, . URL https://www.continuum.io/why-anaconda. Accessed on March 6, 2016.

[28] Wikipython, . URL https://wiki.python.org/moin/LocalUser....Groups#fnref2c45ad947943f47963233f48588be434e138e001. Accessed on May 18, 2016.