**T.C.**

**SAKARYA ÜNİVERSİTESİ**

**BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ**

ISE 402 THESIS

SENSOR DATA ANALYSİS

B151200003 – Berk DOĞUŞ

B151200008 – Kaan Orhun KANAL

|  |  |  |
| --- | --- | --- |
| Department  Advisor | :  : | INFORMATION SYSTEMS ENGINEERING  Prof. Dr. İsmail Hakkı CEDİMOĞLU |

2019-2020 Bahar Dönemi

T.C.

SAKARYA ÜNİVERSİTESİ

**BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ**

SENSOR DATA ANALYSIS

ISE 402 - THESIS

Berk DOĞUŞ

Kaan Orhun KANAL

|  |  |  |
| --- | --- | --- |
| Fakülte Anabilim Dalı | : | BİLİŞİM SİSTEMLERİ MÜHENDİSLİĞİ |

Bu tez .. / .. / … tarihinde aşağıdaki jüri tarafından oybirliği / oyçokluğu ile kabul edilmiştir.

|  |  |  |
| --- | --- | --- |
|  |  |  |
| ………………. | ………………. | ………………. |
| Jüri Başkanı | Üye | Üye |

**FOREWORD**

At the age of ‘Internet of Things’, sensors have become more important than it was before. Collecting and analyzing sensor data in a right way is crucial for developing. As taking this situation into account, we have looked for the sensor data that we can collect and analyze. Thereafter, we have decided to collect and analyze accelerometer, gyroscope and GPS sensor data from our mobile phones by developed an Android application.

We would like to thank our esteemed advisor Prof. Dr İsmail Hakkı CEDİMOĞLU who helped us during our work.

**CONTENTS**

|  |  |
| --- | --- |
| FOREWORD..................................................................................................... | iii |
| CONTENTS…................................................................................................... | iv |
| SİMGELER VE KISALTMALAR LİSTESİ..................................................... | vi |
| FIGURES LİST…….......................................................................................... | vii |
| ABSTRACT....................................................................................................... | ix |
|  |  |
| [CHAPTER 1. ENTRY 10](#_Toc41402896)  [1.1. Sensor Data 10](#_Toc41402897)  [1.1.1. Accelerometer sensor data 10](#_Toc41402898)  [1.1.2. Gyroscope sensor data 10](#_Toc41402899)  [1.1.3. Global positioning system (GPS) sensor data 10](#_Toc41402900)  [CHAPTER 2. DATA COLLECTION 11](#_Toc41402901)  [2.1. MIT App Inventor 11](#_Toc41402902)  [2.2. Android App To Collect Data 13](#_Toc41402903)  [CHAPTER 3. DATA ANALYSIS TOOLS 14](#_Toc41402904)  [3.1. Python Data Science Libraries 14](#_Toc41402905)  [3.1.1. Pandas 14](#_Toc41402906)  [3.1.2. Sklearn 15](#_Toc41402907)  [3.1.3. Pandas profiling 15](#_Toc41402908)  [3.1.4. Matplotlib 15](#_Toc41402909)  [3.1.5. Folium 15](#_Toc41402910)  [3.1.6. Seaborn 15](#_Toc41402911)  [3.1.7. Tkinter 16](#_Toc41402912)  [3.1.8. Dask 16](#_Toc41402913)  [3.2. Data Analysis Methods 16](#_Toc41402914)  [3.2.1. Linear regression 16](#_Toc41402915)  [3.2.2. Decision tree 16](#_Toc41402916)  [3.2.3. Ridge regression 16](#_Toc41402917)  [3.2.4. Random forest 17](#_Toc41402918)  [3.3. Code Optimization 17](#_Toc41402919)  [3.4. Google Colab 17](#_Toc41402920)  [CHAPTER 4. DENEY DÜZENEĞİ VE SANAL LABORATUVAR İKİ SATIR YAZILSAYDI ŞU ŞEKİLDE OLURDU 18](#_Toc41402921)  [4.1. Deney Düzeneği 18](#_Toc41402922)  [4.2. Sanal Laboratuvar Karakteristiği 19](#_Toc41402923)  [4.3. Test Aşaması 19](#_Toc41402924)  [CHAPTER 5. VERİ GÜVENLİĞİ DEĞERLENDİRMESİ 22](#_Toc41402925)  [5.1. Veri Bozulması ve Elektriksel Parametrelerin İlişkisi 22](#_Toc41402926)  [5.2. Elektrik Dalgalanmasına Bağlı Güvenlik Kontrolünün Haberleşme Kalitesine Etkileri 22](#_Toc41402927)  [5.2.1. WH etkisi 22](#_Toc41402928)  [5.2.2. Güvenlik faktöründeki bozulmalar 23](#_Toc41402929)  [5.2.3. Veri hızının bulut teknolojisine etkisi 23](#_Toc41402930)  [5.3. Mobil Bozulma ve Veri Açığı İlişkisi 23](#_Toc41402931)  [CHAPTER 6. SONUÇLAR VE ÖNERİLER 24](#_Toc41402932) |  |

SİMGELER VE KISALTMALAR LİSTESİ

|  |  |
| --- | --- |
| A | : Numune kesit alanı |
| Av | : Sıkışma katsayısı |
| ASTM | : Amerikan standart |

**ŞEKİLLER LİSTESİ**

|  |  |  |
| --- | --- | --- |
| Figure 2.1. | MIT Inventor Loge………...……………………………... | 11 |
| Figure 2.2. | Designer Section of MIT App Inventor…………………... | 12 |
| Figure 2.3. | Blocks Editor Section of MIT App Inventor……………… | 12 |
| Figure 2.4. | User Interface of the app we created for collecting sensor data…………………………………………………….. | 13 |
| Figure 2.5. | Block logic of the app we created for collecting sensor data | 14 |
| Figure 3.1. | Importing Dask Library…………………………………… | 19 |
| Figure 3.2. | …………………………………………………………….. | 19 |
| Figure 3.3. | …………………………………………………………….. | 19 |
| Figure 3.4. | ……………………………………………………………... | 20 |
| Figure 3.5. | ……………………………………………………………... | 20 |
| Figure 3.6. | ……………………………………………………………... | 20 |
| Figure 3.7. | ……………………………………………………………... | 21 |
| Figure 3.8. | ……………………………………………………………... | 21 |
| Figure 3.9. | ……………………………………………………………... | 22 |
| Figure 3.10. | ……………………………………………………………... | 22 |
| Figure 3.11. | ……………………………………………………………... | 23 |
| Figure 3.12. | ……………………………………………………………... | 23 |
| Figure 3.13. | ……………………………………………………………... | 23 |
| Figure 3.14. | ……………………………………………………………... | 24 |
| Figure 3.15. | ……………………………………………………………... | 24 |
| Figure 4.1. | Importing libraries..........…………………………………... | 25 |
| Figure 4.2. | GUI codes…...……………………………………………... | 26 |
| Figure 4.3. | GPS data to showing on map……………….……………... | 26 |
| Figure 4.4. | Reading data……..………………………………………... | 27 |
| Figure 4.5. | Random Forest Regression codes…..…………………….. | 27 |
| Figure 4.6. | Random Forest Regression codes2……………………….. | 27 |
| Figure 4.7. | Ridge Regression codes…………………………………… | 28 |
| Figure 4.8. | Rdige Regression codes2………………………………….. | 28 |
| Figure 4.9. | Linear Regression………………………………………….. | 29 |
| Figure 4.10. | Decision Trees Regression………………………………… | 29 |
| Figure 4.11 | Decision Trees Regression codes2………………………… | 30 |
| Figure 4.12. | Exploratory data analysis codes…………………………… | 30 |
| Figure 4.13. | GUI………………………………………………………… | 31 |
| Figure 4.14. | GUI-2……………………………………………………… | 31 |
| Figure 4.15. | GUI-3……………………………………………………… | 32 |
| Figure 4.16. | Ridre Regression scores…………………………………… | 32 |
| Figure 4.17. | Ridge Regression Graphic…………………………………. | 33 |
| Figure 4.18. | CPU time calculating……………………………………… | 33 |
| Figure 4.19. | CPU times for exploratory data analysis…………………... | 34 |
| Figure 4.20. | CPU times for Linear Regression………………………….. | 34 |
| Figure 4.21. | CPU times for Decision Trees……………………………... | 34 |
| Figure 4.22. | CPU times for Random Forest Regression………………... | 35 |
| Figure 4.23. | CPU times for Ridge Regression………………………….. | 35 |
| Figure 4.24. | CPU times for showin GPS data on map………………….. | 35 |
| Figure 4.25. | Sitting dataset graphic……………………………………... | 36 |
| Figure 4.26. | Sitting dataset seaborn graphic…………………………….. | 36 |
| Figure 4.27. | Sitting dataset linear regression analysis…………………... | 37 |
| Figure 4.28. | Decision trees regression output of sitting dataset………… | 37 |
| Figure 4.29. | Random Forest Regression output of sitting dataset………. | 38 |
| Figure 4.30. | Ridge Regression output of sitting dataset………………… | 38 |
| Figure 4.31. | Walking data graphic……………………………………… | 39 |
| Figure 4.32. | Walking data graphic-2……………………………………. | 39 |
| Figure 4.33. | Linear regression output of walking dataset………………. | 40 |
| Figure 4.34. | Decision trees regression output of walking dataset………. | 40 |
| Figure 4.35. | Random forest regression output of walking dataset……… | 41 |
| Figure 4.36. | Ridge regression output of walking dataset……………….. | 41 |
| Figure 4.37. | Running dataset graphic…………………………………… | 42 |
| Figure 4.38. | Running dataset graphic-2…………………………………. | 42 |
| Figure 4.39. | Linear regression output of running dataset……………….. | 43 |
| Figure 4.40. | Decision trees regression output of running dataset……….. | 43 |
| Figure 4.41. | Random forest regression output of running dataset | 44 |
| Figure 4.42. | Ridge regression output of running dataset | 44 |
| Figure 4.43. |  |  |
| Figure 4.44. |  |  |
| Figure 4.45. |  |  |
| Figure 4.46. |  |  |
|  |  |  |

**ABSTRACT**

Key words: Sensor Data, Machine Learning, Data Analysis

Sensor data is the output of a device that detects some type of input from the physical environment. The aim of our study is to analyses accelerometer sensor data, gyroscope sensor data and GPS sensor data which were collected with the usage of a smartphone to detect a user’s activities. In order to collect sensor data, an Android application was built using MIT App Inventor.

User’s activities were examined in three condition; sitting, walking and running. The differences of all three condition has been observed and exploratory data analysis has been made. Afterwards, regression analysis has been applied to gyroscope sensor datas and accelerometer sensor datas with machine learning techniques(linear regression, decision trees, random forest and ridge regression). Collected GPS datas has been shown in a map.

To bring all these works together, we have created a graphical user interface. Users can reach the works easier with GUI.

# ENTRY

Sensor data are getting more important with the developing technology. Data analysis methods are useful to understand sensor data. We used Python language to analyze accelerometer, gyroscope and GPS data.

* 1. Sensor Data

Sensors are used to detect about any physical element. Sensor data is the output of a device that detects and responds to some type of input from the physical environment. Also, sensor data is an essential component of the Internet of Things environment.

* + 1. Accelerometer sensor data

The rate of change of velocity of the body with respect to time is called acceleration and Accelerometers are devices which measures acceleration. Accelerometers can measure two-dimensional and three-dimensional forms.

* + 1. Gyroscope sensor data

In order to get more accurate motion sensing, Gyroscope sensors are combined with Accelerometer sensors. Besides getting angular velocity data. Gyroscope sensors can also measure the motion of the object. With Gyroscope sensor in mobile phone, we can detect gestures and motion with our mobile phones.

* + 1. Global positioning system (GPS) sensor data

GPS is a satellite navigation system that detect location of users. GPS provides continuous real time, three-dimensional positioning and navigation worldwide. The working of GPS is based on the “trilateration” mathematical principle. GPS receiver takes the information from the satellite and uses the method of triangulation to determine a user’s exact position.

# DATA COLLECTION

For data collection, we need to create an android application which can collect data and save it. In order to create that application, MIT App Inventor is used.

* 1. MIT App Inventor

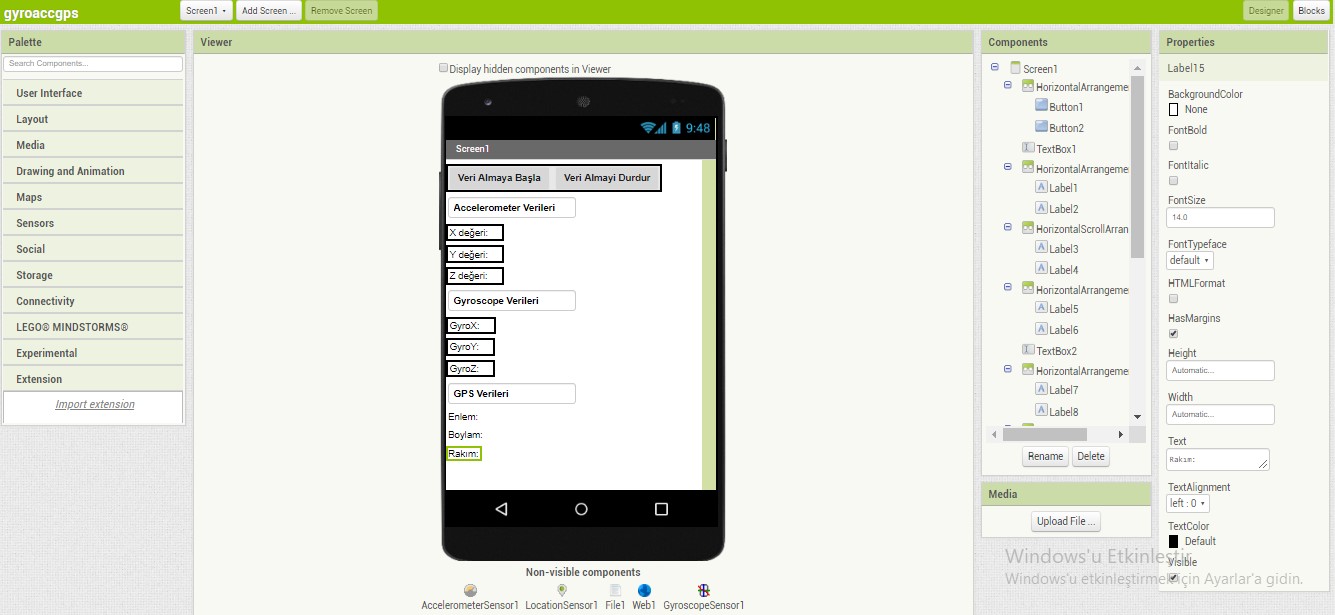
The application was made available through request on July 12, 2010 and released publicly on December 15, 2010. The App Inventor team was led by Hal Abelson and Mark Friedman.

Figure 2.1. MIT App Inventor logo

MIT App Inventor help you to develop Android applications by using a web browser. MIT App Inventor servers store creators’ works.

App Inventor has two sections which called The App Inventor Designer and The App Inventor Blocks Editor.

Designer section is the part that creators select the components for their app.

Figure 2.2 Designer Section of MIT App Inventor

The Blocks Editor section is the part that creators assemble program blocks which specify how the components should behave.

 Figure 2.3 Blocks Editor Section of MIT App Inventor

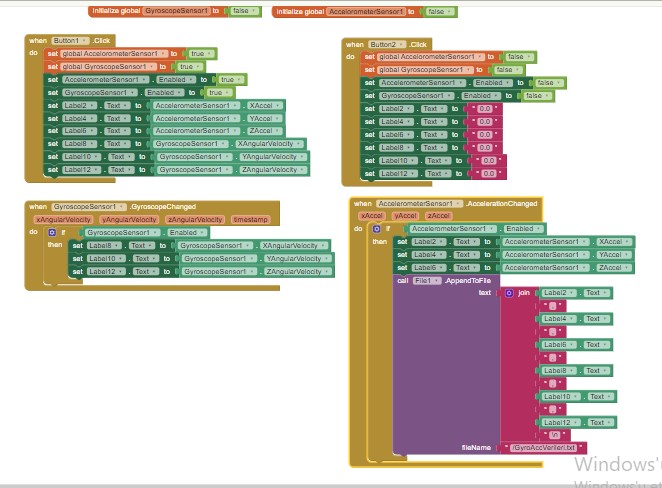
* 1. Android App To Collect Data

By using MIT App Inventor, we built an Android application helps us to collect and save accelerometer data, gyroscope data and GPS data.

Figure 2.4 User Interface of the app we created for collecting sensor data

The app we created is seen the photo above. When user push the ‘Veri Almaya Başla’ button, the app starts to collect and save data. If user push the ‘Veri Almayi Durdur’ button, the app stops data collection.

The block of app can be seen below. It shows the logic of the app created.

 Figure 2.5 Block logic of the app we created for collecting sensor data

# DATA ANALYSIS TOOLS

* 1. Python Data Science Libraries

To data analyze in python language, data scientists need libraries.

* + 1. Pandas

Pandas is a software library written for the Python programming language for data manipulation and analysis. It offers data structures and operations for manipulating numerical tables and time series.

* + 1. Sklearn

Sklearn (Scikit-learn) is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

* + 1. Pandas profiling

Pandas profiling provides analysis like type, unique values, missing values, quantile statistics, mean, mode, median, standard deviation, sum, skewness, frequent values, histograms, correlation between variables, count, heatmap visualization.

* + 1. Matplotlib

Matplotlib is a Python library which is using for 2D plotting. With using matplotlib, plots, histograms, power spectra, bar charts, error charts, scatterplots etc. can be generated easily.

* + 1. Folium

Folium library is a data visualization library for Pyhton programming language. It makes it easy to visualize data on an intereactive leaflet map. It provides binding of data to a map for choropleth visualizations. Folium supports Image, Video, GeoJSON and TopoJSON overlays.

* + 1. Seaborn

Seaborn is a Python library that based on matplotlib. Seaborn library is used to ease the challenging task of data visualization. Seaborn provides users to the creation of statistical graphics.

* + 1. Tkinter

Tkinter is a GUI library for Python. It provides lots of widgets such as buttons, labels, text boxes, scrollbar etc. in a GUI application.

* + 1. Dask

Dask library is a useful python library to work with large datasets. It is known as a pyhton paralel computing library. Dask library has a lot similarities with pandas library so that the users working with pandas library can easily work with dask library too. Unlike pandas library, dask library provides a much faster way to handle large and big data in python.

* 1. Data Analysis Methods
     1. Linear regression

Linear regression provides to users predict a dependent variable value based on a given independent variable. By using regression technique users can find a linear relationship between input and output.

* + 1. Decision tree

Decision tree is a supervised learning algorithm. It repeatedly splits the dataset into two more sub-nodes according to parameters. Decision trees consist leaves and decision nodes. The data is split on decision nodes and leaves are the decisions.

* + 1. Ridge regression

Ridge regression is regression technique that helps reduce the multicollinearity of variables in models. It also is used to quantify the overfitting of the data through measuring the magnitude of coefficients.

* + 1. Random forest

Random forests is a supervised learning algorithm. Random forest algorithm combines multiple algorithm of the same type multiple decision trees. It can be used for both regression and classification tasks.

* 1. Code Optimization

Nowadays, efficiency has become more crucial for software programs. To have a more efficient software, code optimization is an useful method. Optimized programs can consume less memory and executes more rapidly. With limited resources in terms of computing power or memory, code optimization would make sure that we can make do with the resources we have.

* 1. Google Colab

Google Colab is a cloud based Jupyter notebook environment which allows users to train their models on CPUs, GPUs and TPUs for free. Training models can take a lot of hours on CPUs but GPUs and TPUs, provided by Google Colab, can easily execute in a short time.

* 1. The Works Mentioned but Have not Used in Project
     1. Works with dask library

The command of reading csv files is the same with pandas library.

Figure 3.1. importing Dask library

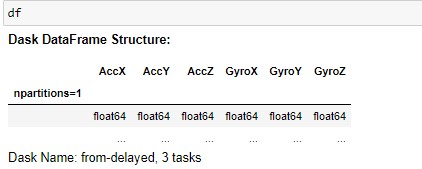
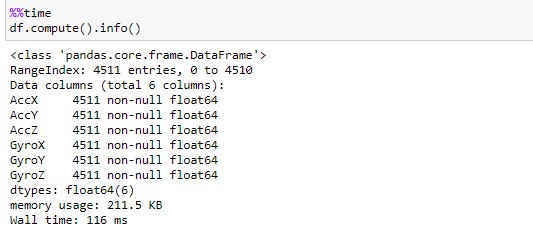
Dask library split the large dataframes to partitions to work with them efficiently, we can see the partitions count with df command just because our sensor datas not large enough to split, dask library doesn’t need to split it.

Figure 3.2

Unlike pandas library, you need to use compute() command to execute exploratory data analysis tools.

Figure 3.3.

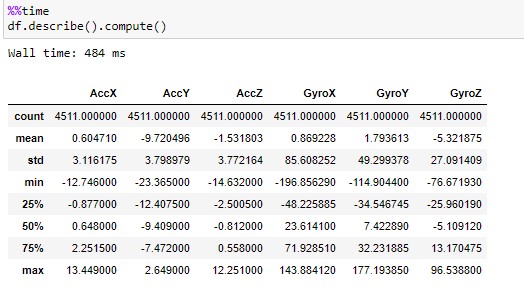


Figure 3.4

To showing the partitions function of dask library, we have found a larger dataset.

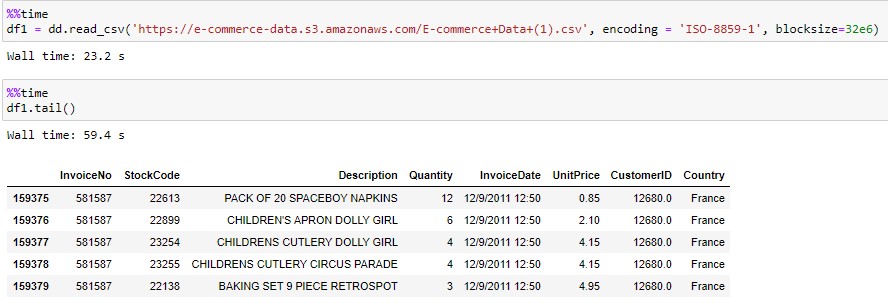


Figure 3.5.

As it seen with shape command df1 dataset is much larger than our sensor dataset df1.

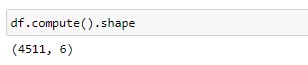
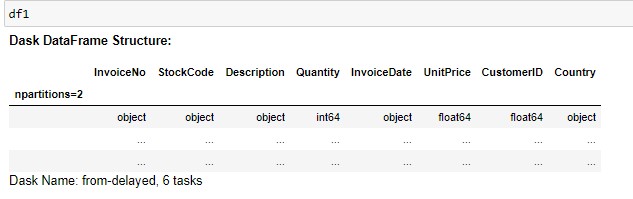
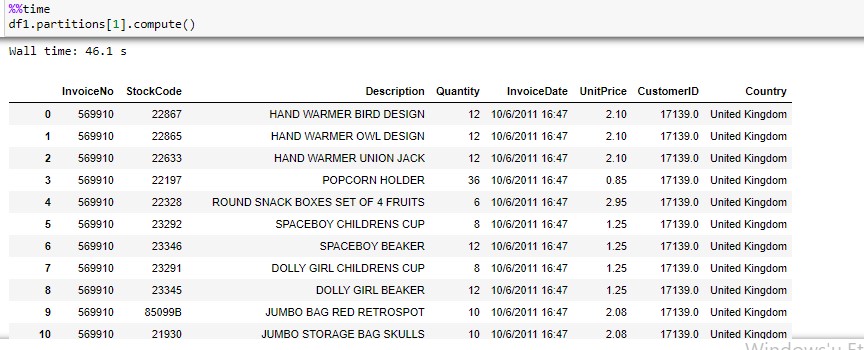
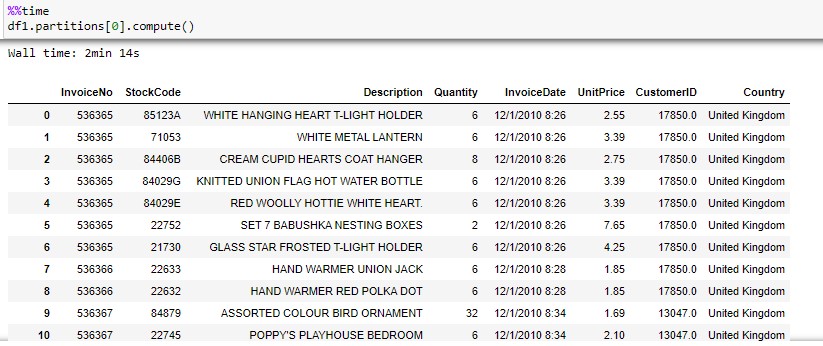


Figure 3.6.

Dask library split the df1 dataset to two partitions to work with the df1 dataset faster way.

Figure 3.7.

We can see each of the partitions with df1.partitions[indexcode].compute() command.

Figure 3.8.

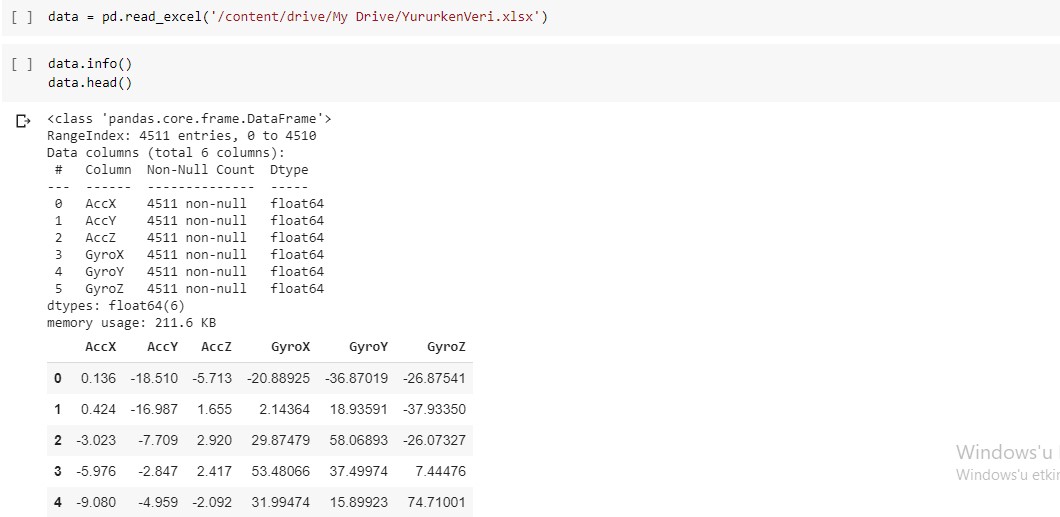
* + 1. Google colab work

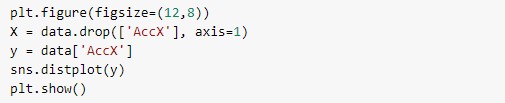
Google Colab’s Jupyter Notebook environment provided us a much stronger CPUs than our own CPUs so it’s very helpful for saving time. Furthermore, we can say that it’s so useful to make a group work.

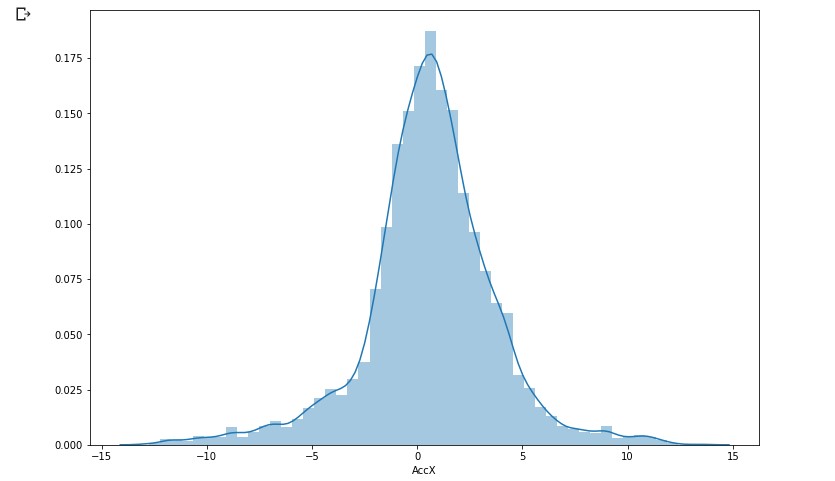
Google Colab notebook has some exclusive python libraries. We have used google.colab libraries to read datasets which is uploaded to Google Drive. We have moved our linear regression, random forest regression, decision trees regression and ridge regression analysis works to Google Colab environment.

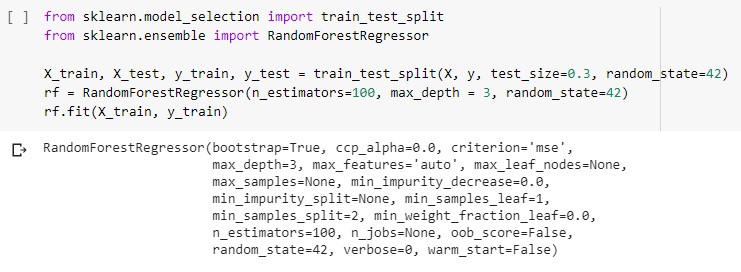
As an example of our works Random Forest Regression;

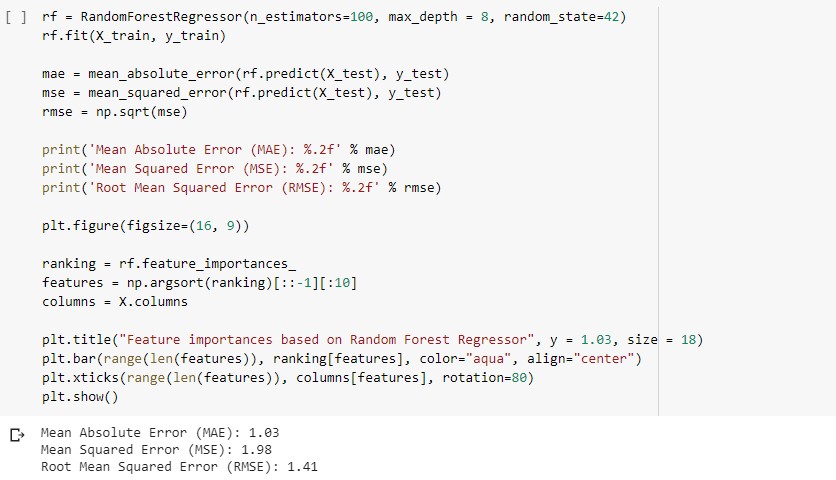
Figure 3.9.

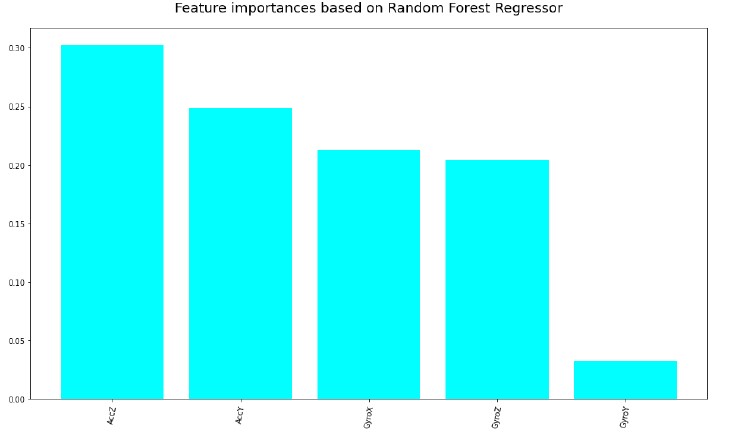
Figure 3.10.

Figure 3.11.

Figure 3.12.

Figure 3.13.

Figure 3.14.

Figure 3.15.

# DATA ANALYSIS WITH MACHINE LEARNING

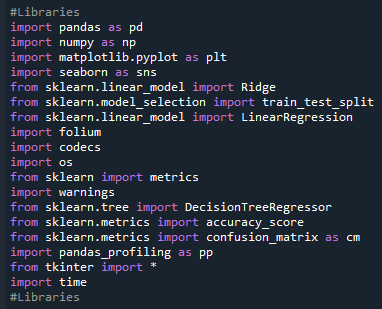
In order to optimizating our code, we have imported required libraries.

Figure 4.1. Importing libraries

We've used tkinter library for creating graphical user interface. We’ve gotten 2 dropdown list item.One of them is for selecting file. The other one for selecting function that applying to dataset. On interface we have 2 buttons. The one named "Calculate Selected" for getting all variables that we needed and start calculation. The other one is for show gps datas on map.

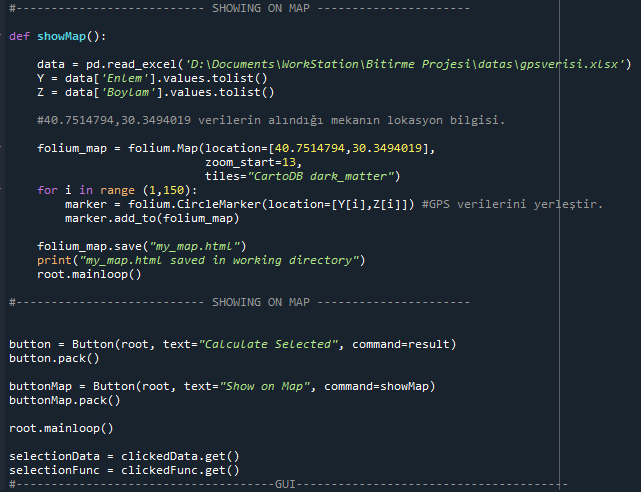
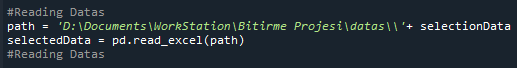
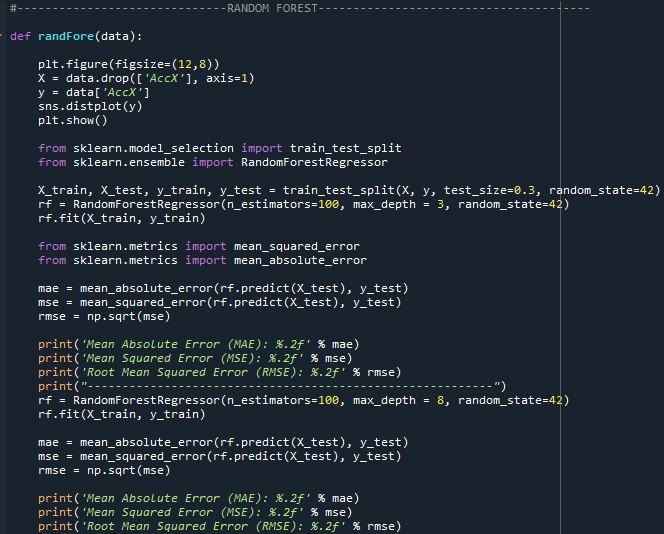
Figure 4.2. GUI codes

Figure 4.3. GPS data to showing on map

Figure 4.4. Reading data

We've optimized and speed up our python code, it has become more readable and agile. Code optimization have made work very useful for other function and dataset varieties.

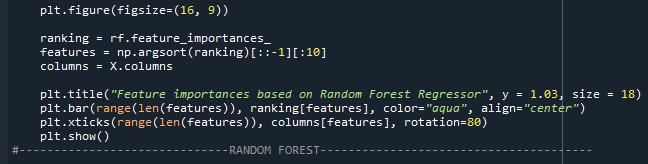
Figure 4.5. Random Forest Regression codes

Figure 4.6. Random Forest Regression codes2

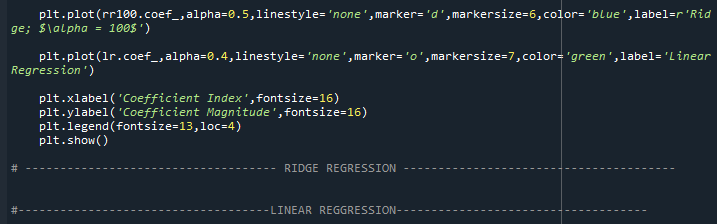
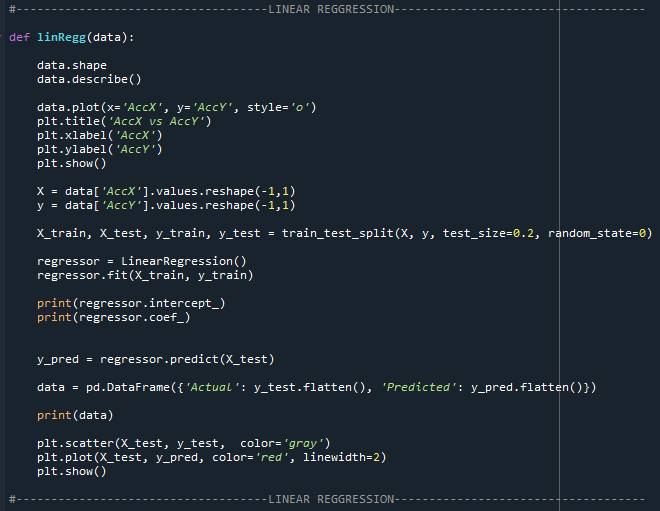
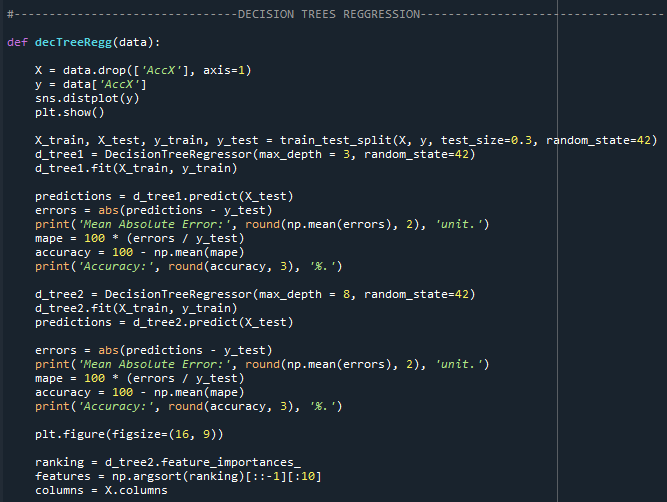
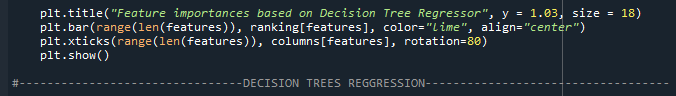
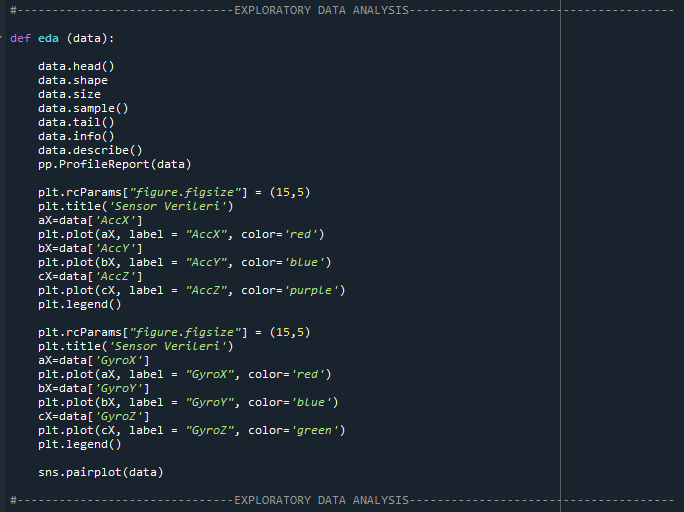
Figure 4.7. Ridge Regression codes

Figure 4.8. Ridge Regression Codes2

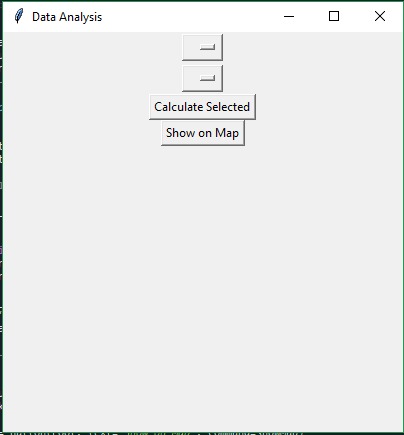
Figure 4.9. Linear Regression

Figure 4.10. Decision Trees Regression codes

Figure 4.11. Decision Trees Regression codes2

Figure 4.12. Exploratory Data Analysis codes

* 1. Graphical User Interface

The created GUI provide users to select the dataset(sitting, walking or running) and the regression method(Random Forest Regression, Decision Trees Regression, Linear Regression or Ridge Regression) or exploratory data analysis then, execute selected regression method on selected data.

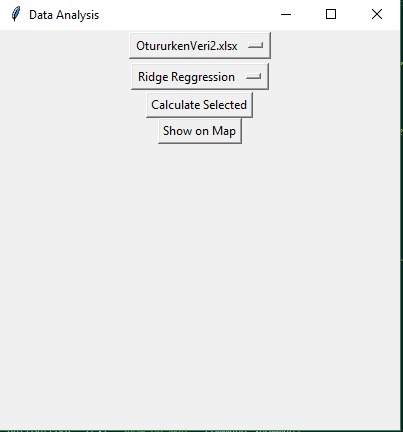
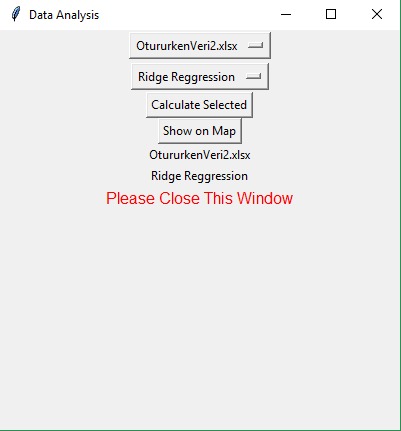
Figure 4.13. GUI

Figure 4.14. GUI-2

Figure 4.15. GUI-3

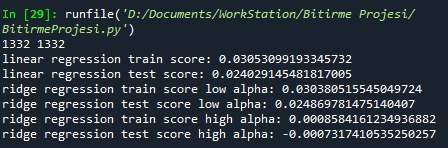
When the users completed their selections, they can execute it with using calculate selected button. On our example; user has selected ‘OtururkenVeri2’ and ‘Ridge Regression’. After using calculate button, user will see the outputs which is seen on the pictures below.

Figure 4.15. Ridre Regression scores

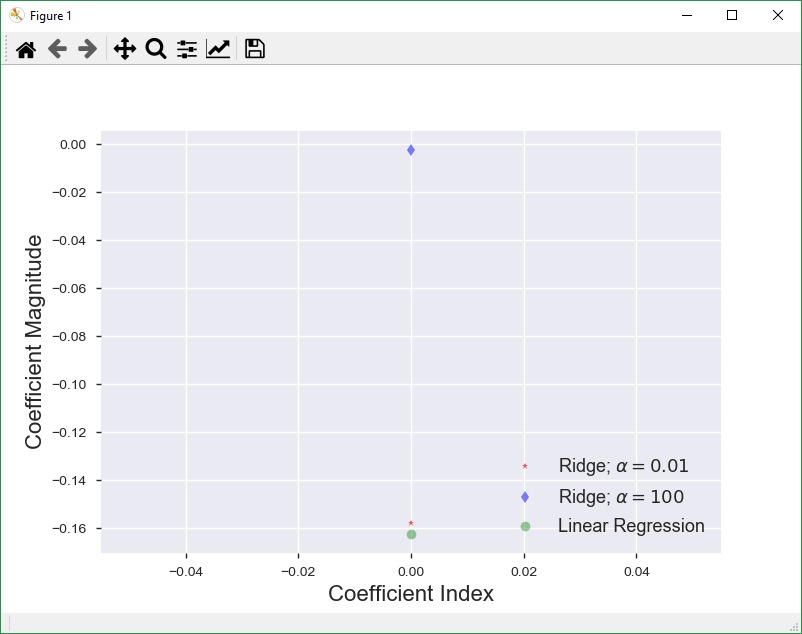


Figure 4.17. Ridge Regression Graphic

* 1. CPU Times

****The code provides us to see CPU time.

Figure 4.18. CPU time calculating

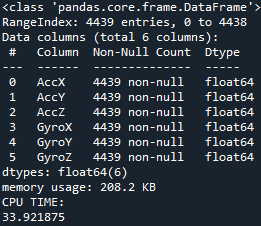
CPU times for Exploratory Data Analysis

Figure 4.19. CPU times for exploratory data analysis

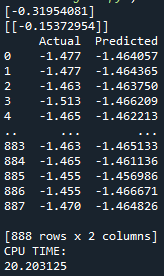
CPU times for Linear Regression

Figure 4.20. CPU times for Linear Regression

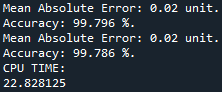
CPU times for Decision Trees Regression:

Figure 4.21. CPU times for Decision Trees

CPU times for Random Forest Regression

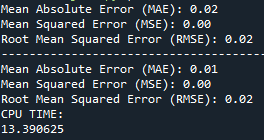


Figure 4.22. CPU times for Random Forest Regression

CPU times for Ridge Regression

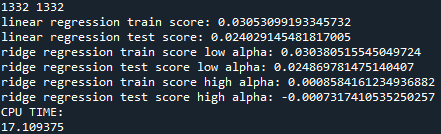


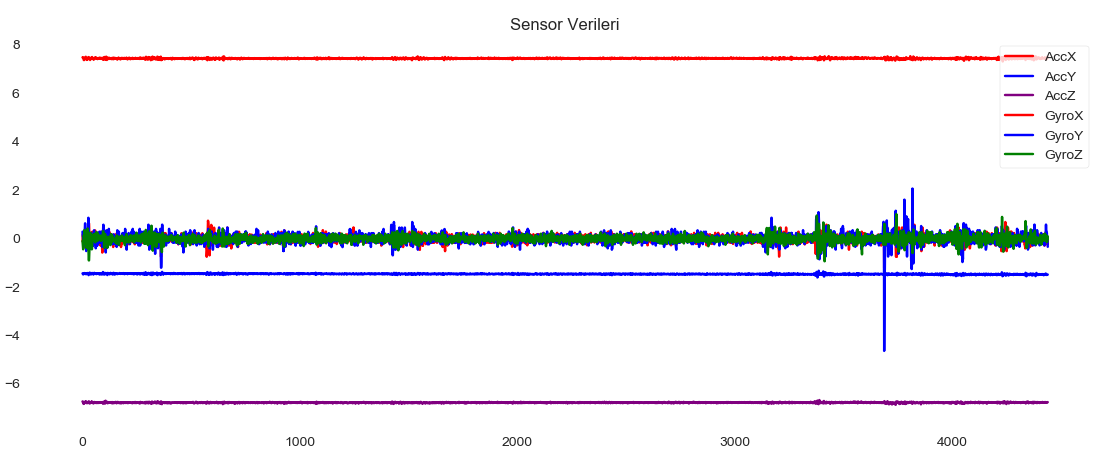
Figure 4.23. CPU times for Ridge Regression

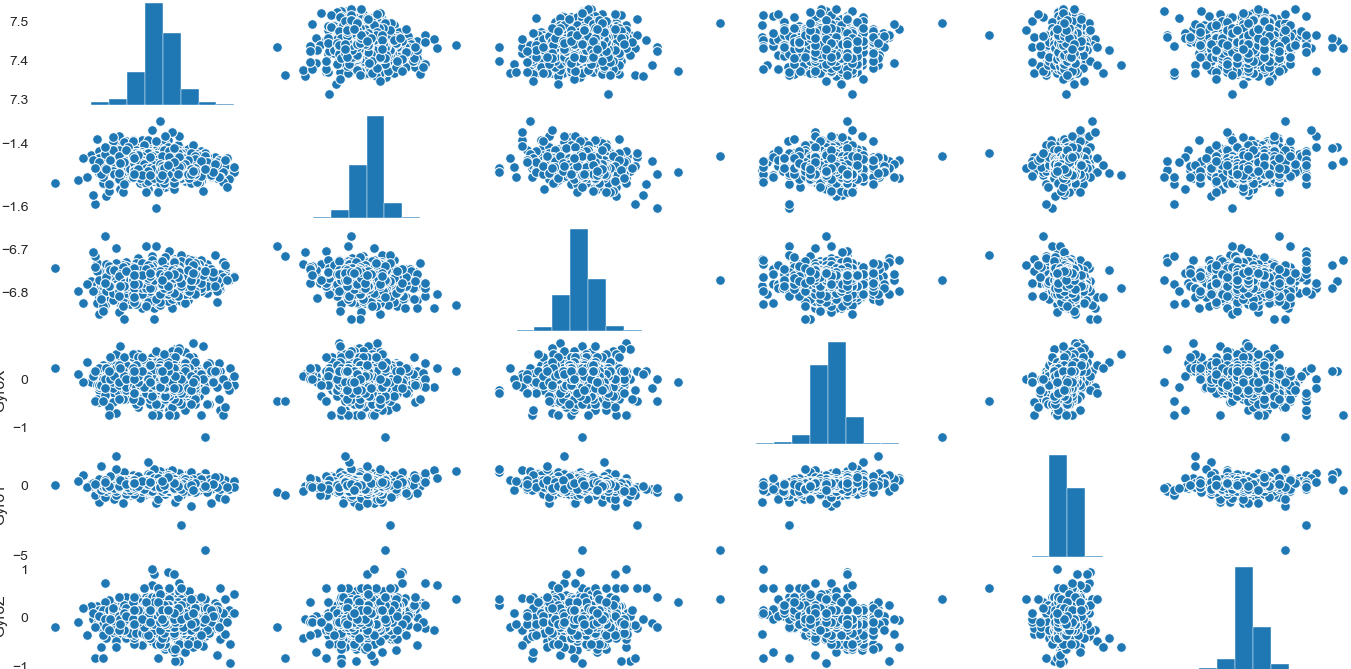
CPU times for GPS data on Map

Figure 4.24. CPU times for GPS data on map

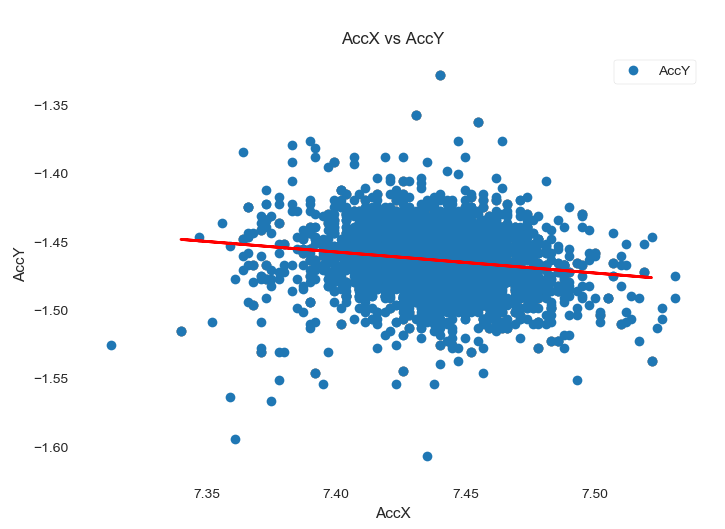
* 1. Outputs of Sitting Dataset Analysis

Sitting Dataset Graphic

Figure 4.25. Sitting Dataset Graphic

****Figure 4.26. Sitting dataset seaborn graphic

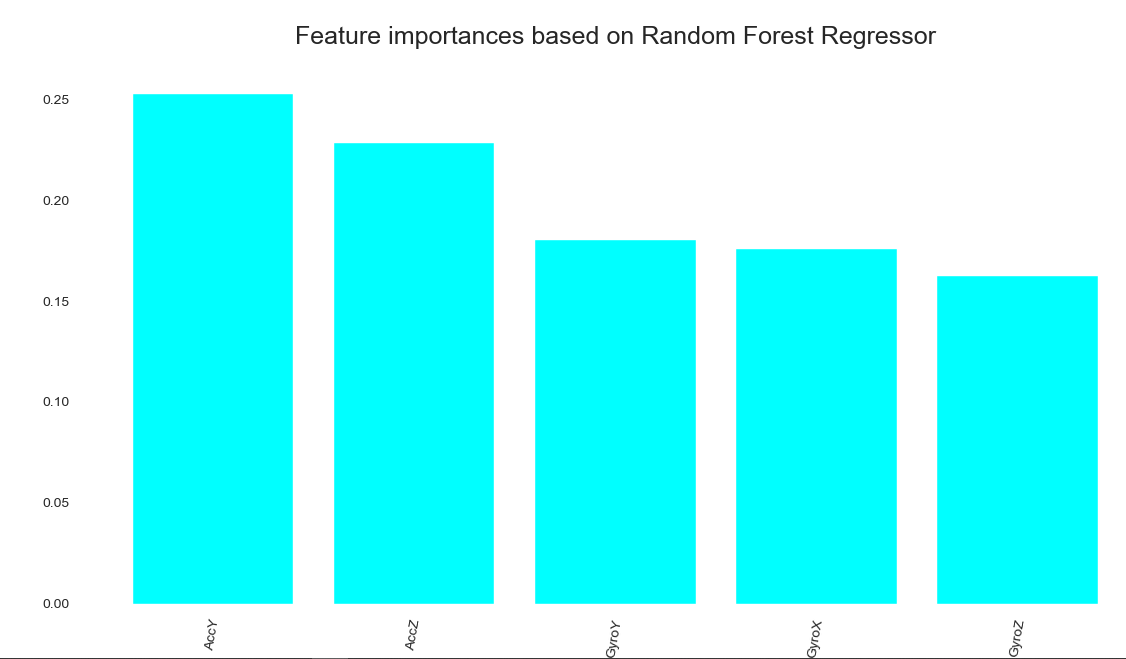
Linear Regression output of sitting dataset

Figure 4.27. Sitting Dataset Linear Regression analysis

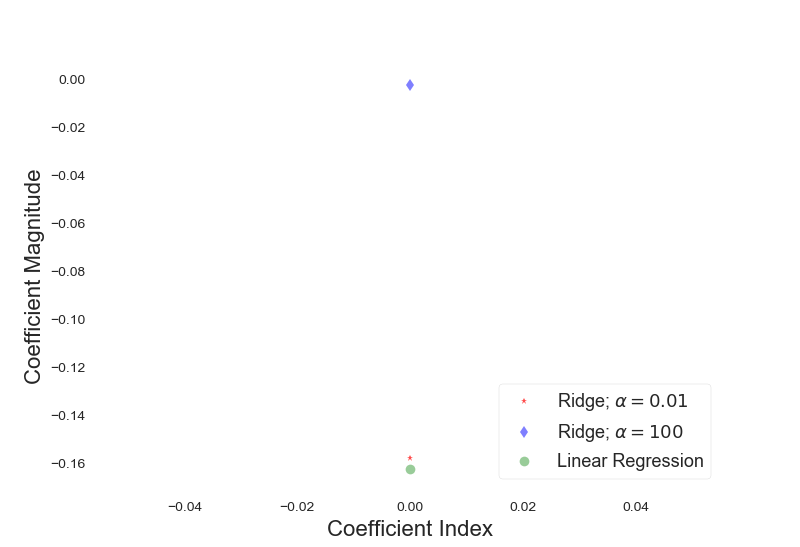
Decision Trees Regression output of sitting dataset

Figure 4.28. Decision Trees Regression output of sitting dataset

Random Forest Regression output of sitting dataset

Figure 4.29. Random Forest Regression output of sitting dataset

Ridge Regression output of sitting dataset

Figure 4.30. Ridge Regression output of sitting dataset

* 1. Outputs of Walking Dataset Analysis

Walking Dataset Graphics

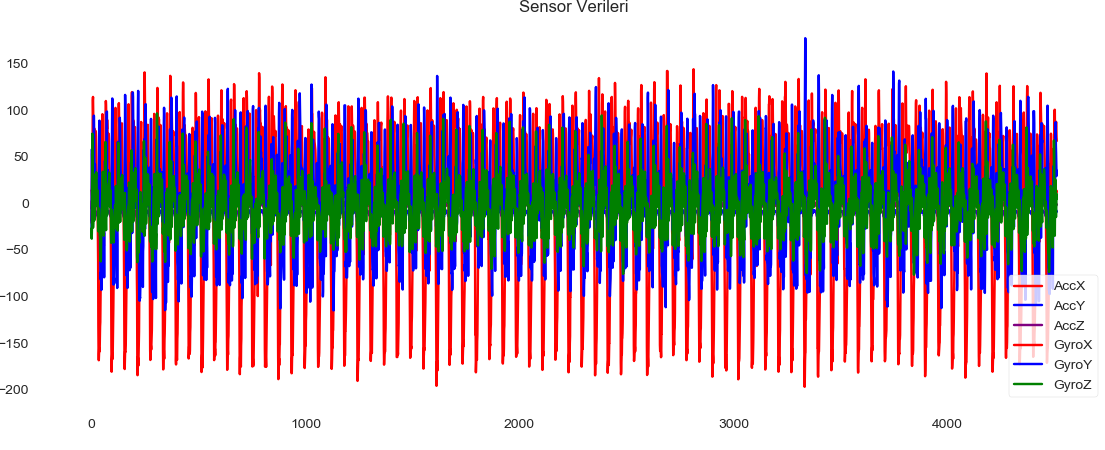
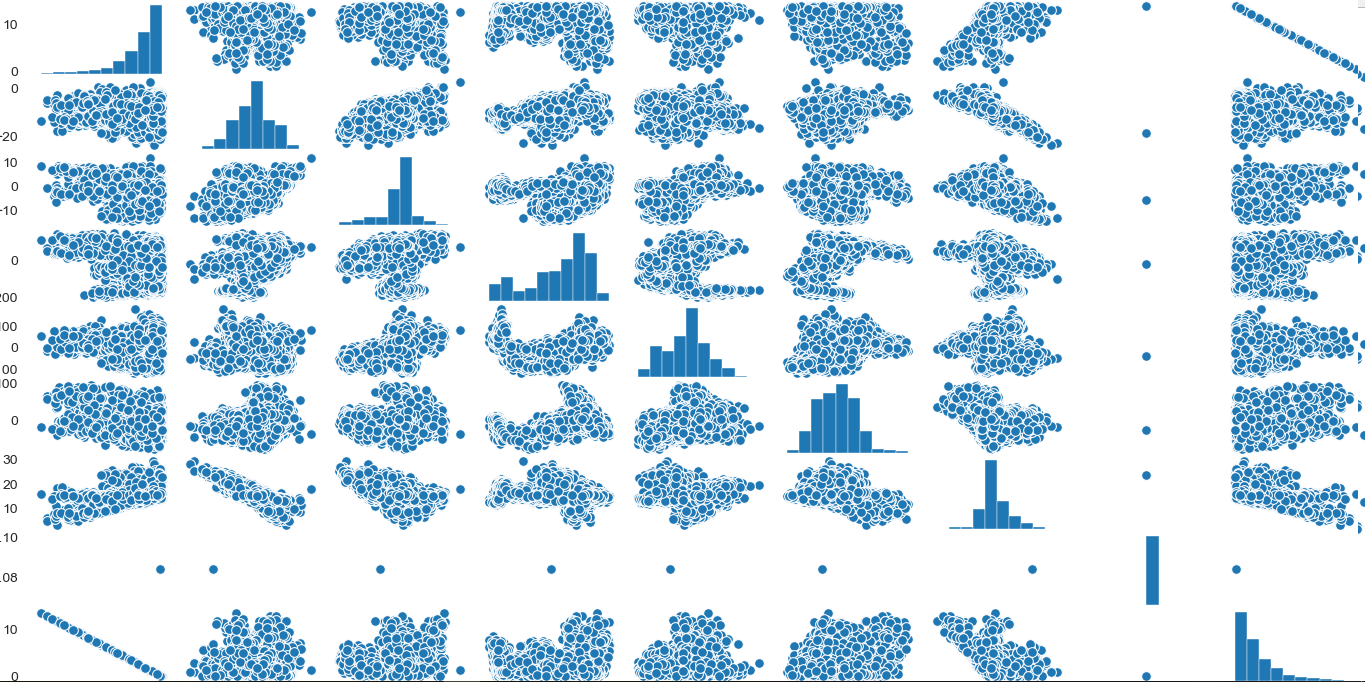
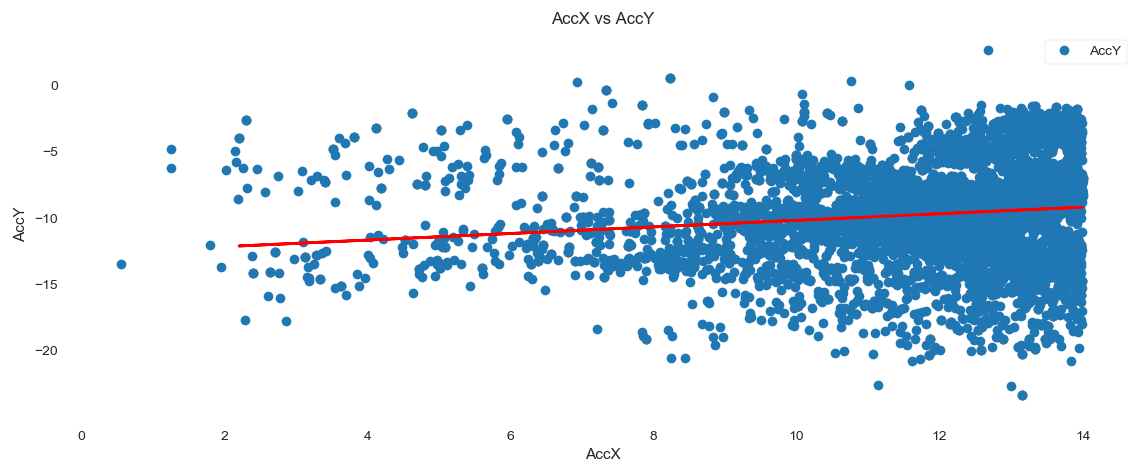


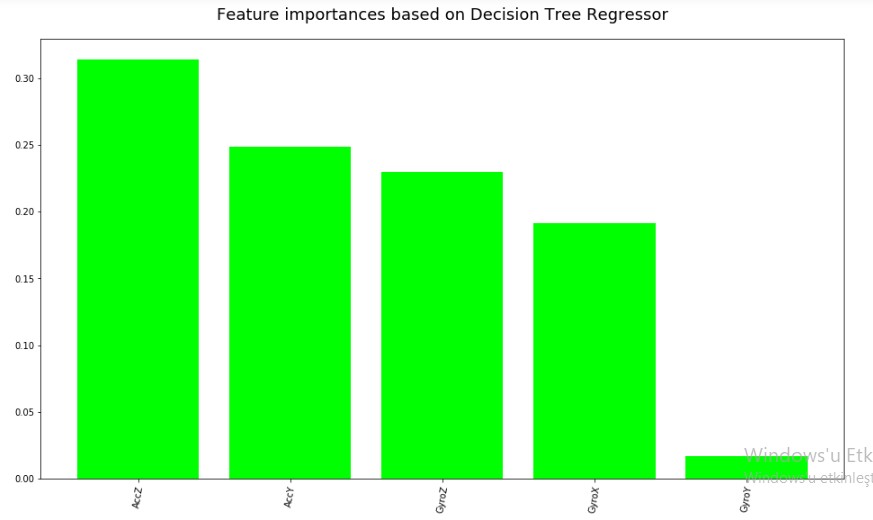
Figure 4.31. Walking Data Graphic

Figure 4.32. Walking Data Graphic2

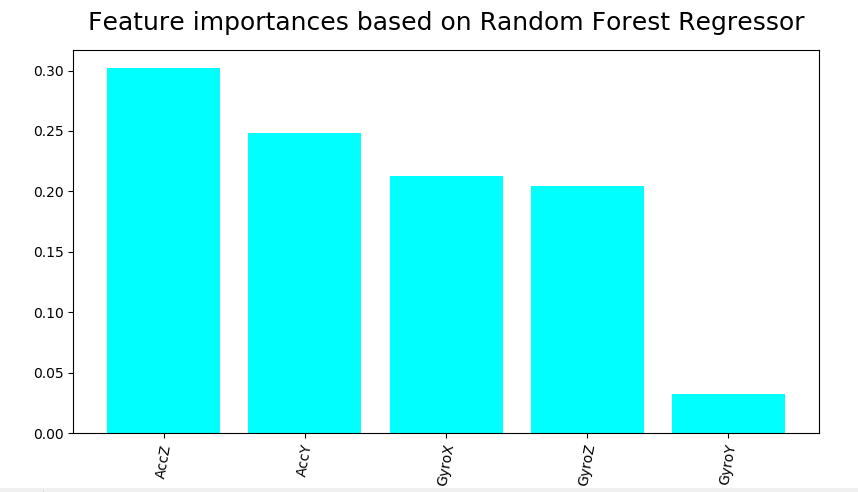
Linear Regression output of walking dataset

Figure 4.33. Linear regression output of walking dataset

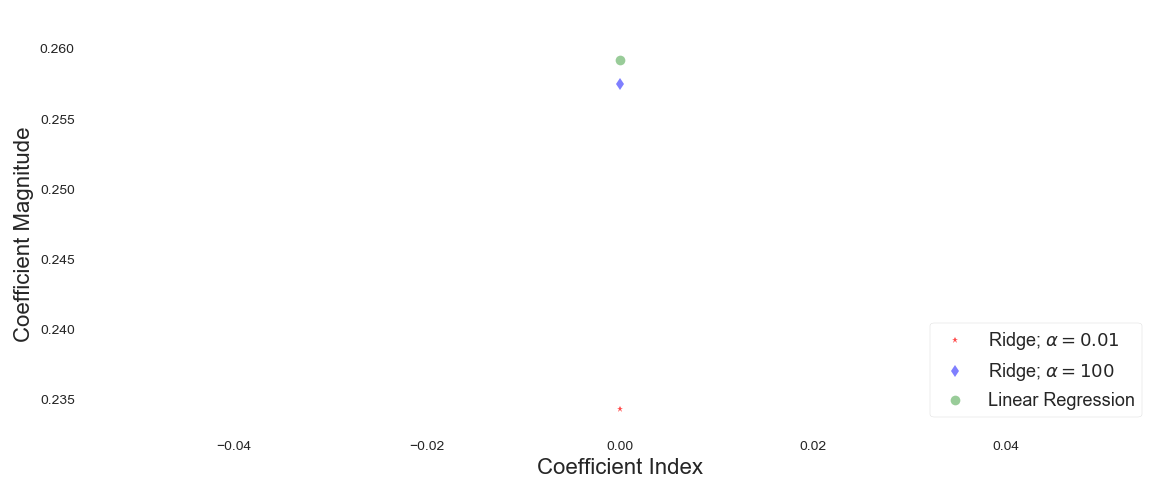
Decision Regression output of walking dataset

Figure 4.34. Decision Trees regression output of walking dataset

Random Forest Regression output of walking dataset

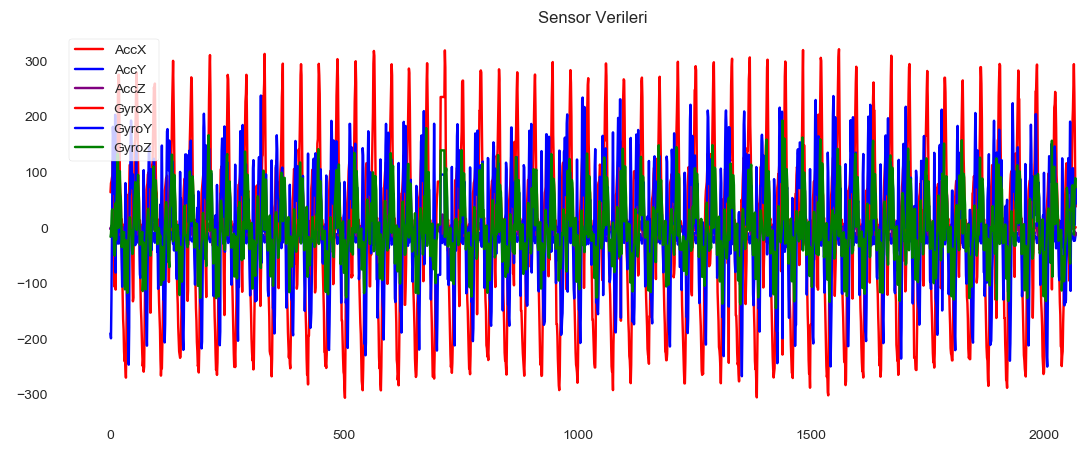
Figure 4.35. Random Forest Regression output of walking dataset

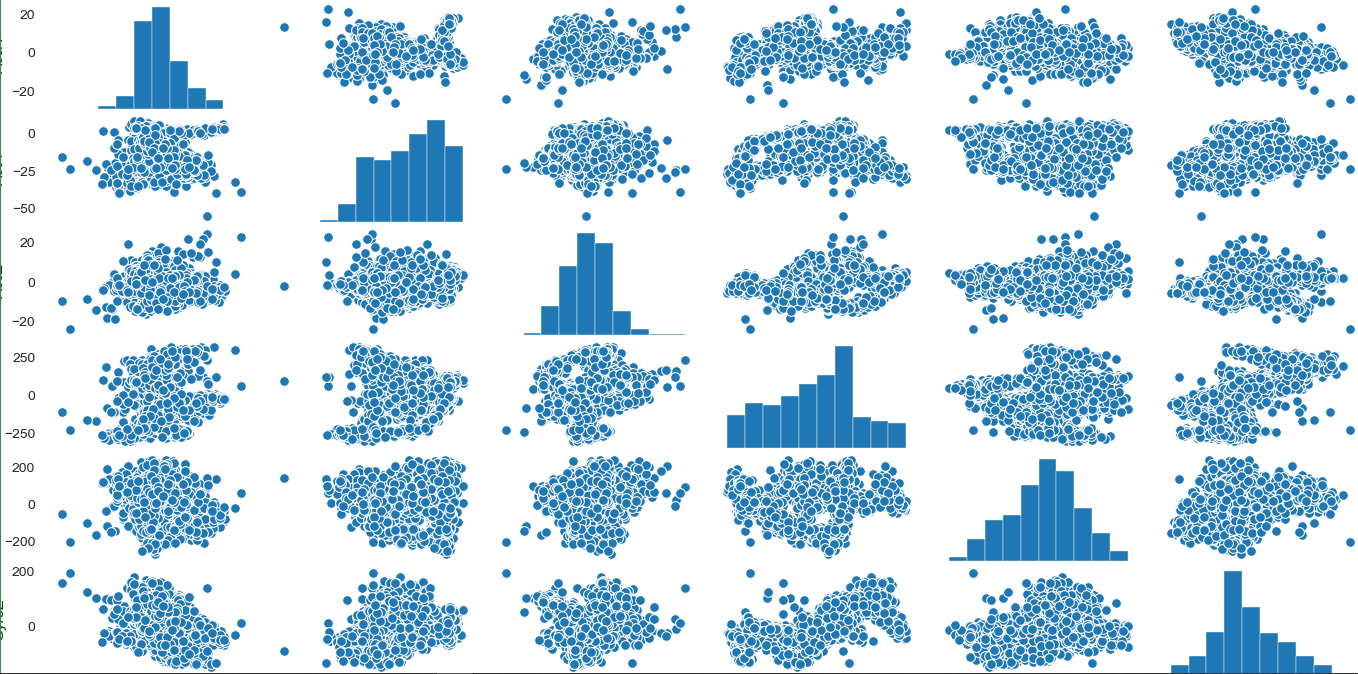
Ridge Regression output of walking dataset

Figure 4.36. Ridge Regression output of walking dataset

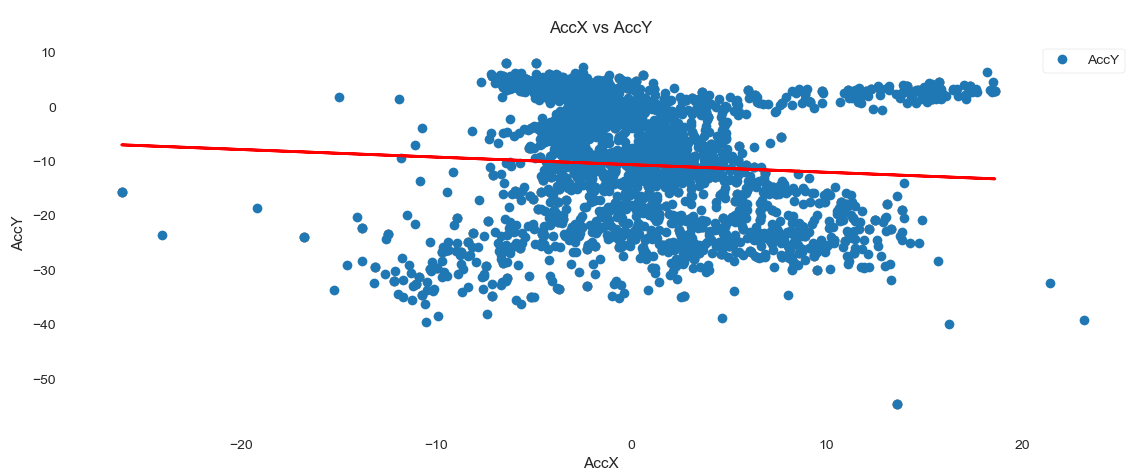
* 1. Output of Running Dataset Analysis

Running Dataset Graphics

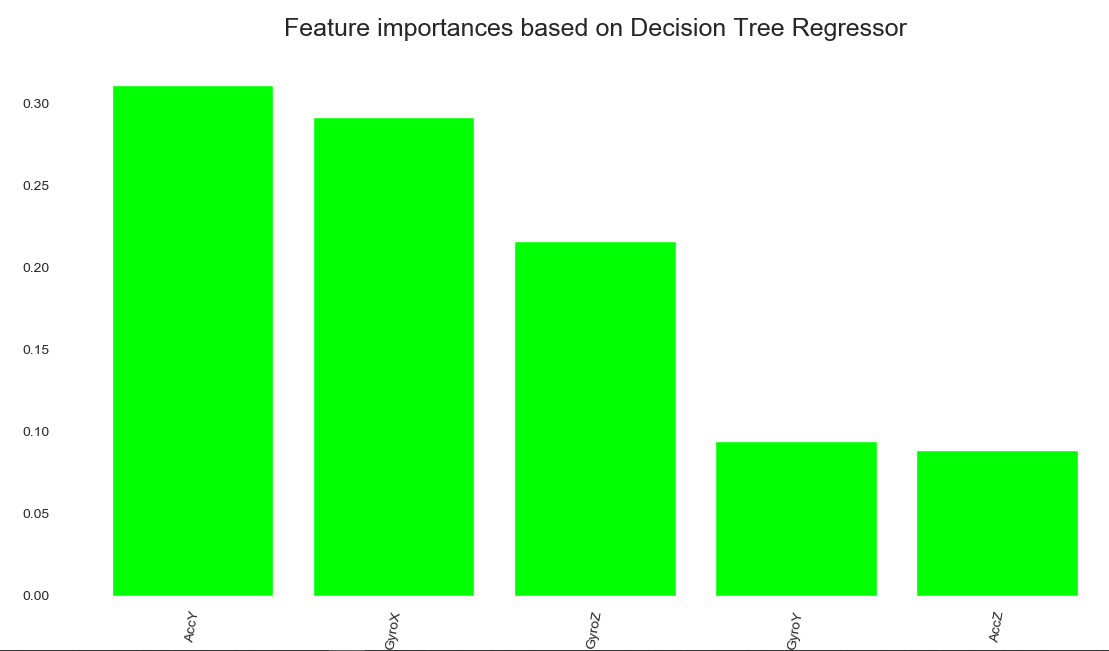
Figure 4.37. Running dataset graphic

Figure 4.38. Running dataset graphic-2

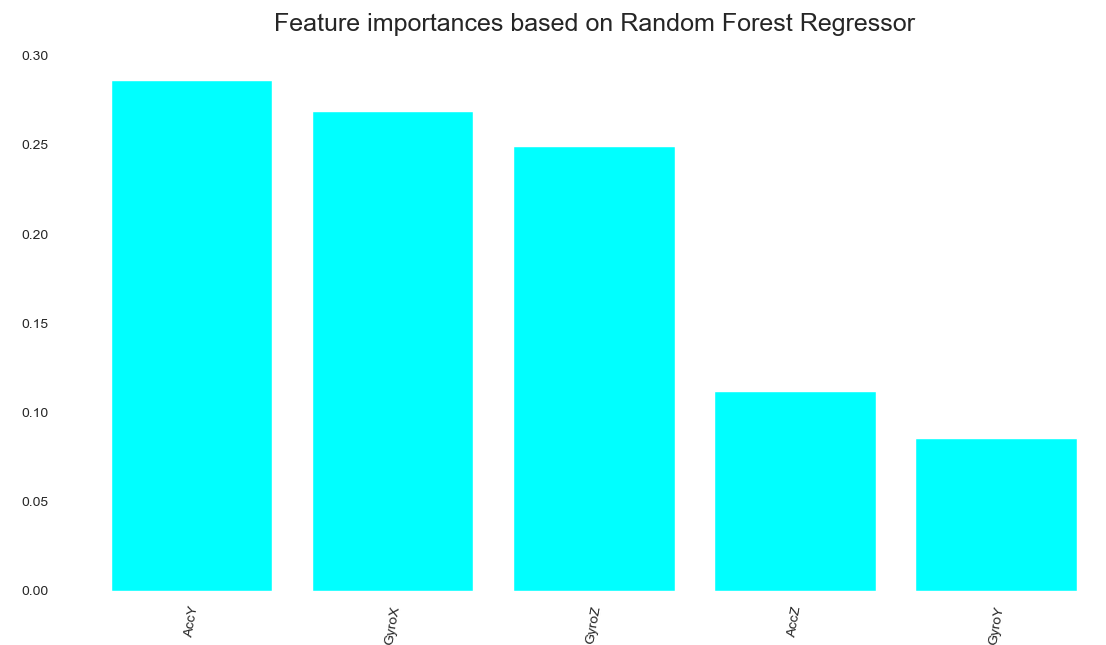
Linear Regression output of running dataset

Figure 4.39. Linear Regression output of running dataset

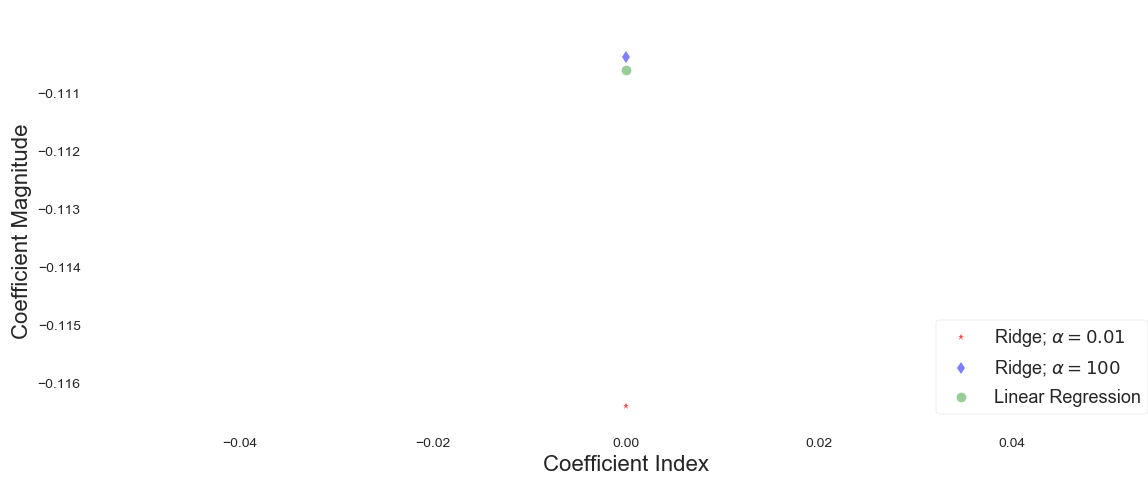
Decision Trees Regression output of running dataset

Figure 4.40. Decision Trees Regression output of running dataset

Random Forest Regression output of running dataset

Figure 4.41. Random Forest output of running dataset

Ridge Regression output of running dataset

Figure 4.42. Ridge Regression output of running dataset

# CONCLUSION

# SONUÇLAR VE ÖNERİLER

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gub rgren, no sea takimata sanctus est Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut lab ore sit et dolore magna.

**KAYNAKLAR**

|  |  |  |
| --- | --- | --- |
| [1] |  | TSOURIS,C., PORCELLI, JV., Process intensification − has its time finally come? Chem. Eng. Progr. 10:50−55, 2003. |
| [2] |  | WEGENG, RS., DROST, MK., BRENCHLEY, DL., Process intensification through miniaturization of chemical and thermal system in the 21st century. In Proc 3rd Int. Conf. Microreaction Technology (IMRET3), Ehrfeld W (ed); Springer: Berlin, pp. 2−13, 2000. |
| [3] |  | CHARPENTIER, J-C., Process intensification by miniaturization. Chem. Eng. Technol, 28(3):255−258, 2005. |
| [4] |  | SCHUBERT, K., BRANDNER, J., FICHTNER, M, LINDER, G., SCHYGULLA U, WENKA, A., Microstructure devices for applications in thermal and chemical process engineering. Microscale Therm. Eng, 5:17−39, 2001. |
| [5] |  | PFEIFER, P., HAAS-SANTO, K., GÖRKE, O., BOHN, L., SCHUBERT K., Fuel to hydrogen an overview over fuel conversion activities at the institute for micro process engineering. In Proc. AIChE Spring National Meeting / IMRET-8, Atlanta, U.S.A, April 11 - 14, 2005. |
| [6] |  | JÄHNISCH K, BAERNS M, HESSEL V, EHRFELD, W., HAVERKAMP, V., LÖWE, H., WILLE, C. G., A Direct fluorination of toluene using elemental fluorine in gas/liquid microreactors. J. Fluorine Chem., 105(1):117−128, 2000. |
| [7] |  | HAVERKAMP, V., HESSEL, V., LÖWE, H., MENGES, G., WARNIER MJF., REBROV, EV., de Croon MHJM, SCHOUTEN, JC., LIAUW, M., Hydrodynamics and mixer-induced bubble formation in microbubble columns with single and multiple channels. Chem. Eng. Technol., 29(9):1015−1026, 2006. |
| [8] |  | http://[www.google.com](http://www.google.com), Erişim Tarihi: 05.01.2013. |

**ÖZGEÇMİŞ**

Bilal Gates, 12.02.1993 de İstanbul’da doğdu. İlk, orta ve lise eğitimini Fatih’te tamamladı. 2011 yılında Yalova Teknik Lisesi, Bilgisayar Bölümü’nden mezun oldu. 2012 yılında Sakarya Üniversitesi Bilgisayar Mühendisliği Bölümü’nü kazandı. 2014 yılında MicSoft İmalat ve Tic. Ltd. Şirketinde yazılım stajını ve 2015 yılında da TUBİTAŞ-IBN Şirketinde donanım stajını yapmıştır. SAÜ Bilişim Sistemleri Mühendisliği Bölümünden 2016 yılında mezun olmuştur.

Berk DOĞUŞ was born in Saray/Tekirdağ on 27.05.1996. He completed his elementary, secondary and high school education here in 2014. He is still studying at Sakarya University Information Systems Engineering department. At 2018 and 2019, he completed his business and software development internship at “PAS South East Europe San ve Tic. Limited Sti.”.

at

**Özgeçmiş Sayfası Hakkında**Tez hazırlayan öğrenci, ÖZGEÇMİŞ başlığı altında kısa özgeçmişini, yaptığı stajları belirterek üçüncü şahıs ağızdan hazırlamalıdır. **Burayı çıktı almadan önce siliniz.**

**Özgeçmiş Sayfası Hakkında**Tez hazırlayan Yüksek Lisans veya Doktora öğrencisi, ÖZGEÇMİŞ başlığı altında kısa özgeçmişini üçüncü şahıs ağızdan hazırlamalı ve tezin en son sayfaları olarak vermelidir. **Burayı çıktı almadan önce siliniz.**

**ıse 402 BİTİRME ÇALIŞMASI**

**Değerlendİrme ve Sözlü Sınav Tutanağı**

KONU :

ÖĞRENCİLER (Öğrenci No/AD/SOYAD):

|  |  |  |  |
| --- | --- | --- | --- |
| Değerlendirme Konusu | İstenenler | Not Aralığı | Not |
| **Yazılı Çalışma** |  |  |  |
| **Çalışma klavuza uygun olarak hazırlanmış mı?** | x | 0-5 |  |
| **Teknik Yönden** |  |  |  |
| **Problemin tanımı yapılmış mı?** | x | 0-5 |  |
| Geliştirilecek yazılımın/donanımın mimarisini içeren blok şeması (yazılımlar için veri akış şeması (dfd) da olabilir) çizilerek açıklanmış mı? |  |  |  |
| Blok şemadaki birimler arasındaki bilgi akışına ait model/gösterim var mı? |  |  |  |
| Yazılımın gereksinim listesi oluşturulmuş mu? |  |  |  |
| Kullanılan/kullanılması düşünülen araçlar/teknolojiler anlatılmış mı? |  |  |  |
| Donanımların programlanması/konfigürasyonu için yazılım gereksinimleri belirtilmiş mi? |  |  |  |
| UML ile modelleme yapılmış mı? |  |  |  |
| Veritabanları kullanılmış ise kavramsal model çıkarılmış mı? (Varlık ilişki modeli, noSQL kavramsal modelleri v.b.) |  |  |  |
| Projeye yönelik iş-zaman çizelgesi çıkarılarak maliyet analizi yapılmış mı? |  |  |  |
| Donanım bileşenlerinin maliyet analizi (prototip-adetli seri üretim vb.) çıkarılmış mı? |  |  |  |
| Donanım için gerekli enerji analizi (minimum-uyku-aktif-maksimum) yapılmış mı? |  |  |  |
| Grup çalışmalarında grup üyelerinin görev tanımları verilmiş mi (iş-zaman çizelgesinde belirtilebilir)? |  |  |  |
| Sürüm denetim sistemi (Version Control System; Git, Subversion v.s.) kullanılmış mı? |  |  |  |
| Sistemin genel testi için uygulanan metotlar ve iyileştirme süreçlerinin dökümü verilmiş mi? |  |  |  |
| Yazılımın sızma testi yapılmış mı? |  |  |  |
| Performans testi yapılmış mı? |  |  |  |
| Tasarımın uygulamasında ortaya çıkan uyumsuzluklar ve aksaklıklar belirtilerek çözüm yöntemleri tartışılmış mı? |  |  |  |
| **Yapılan işlerin zorluk derecesi?** | x | 0-25 |  |
| **Sözlü Sınav** |  |  |  |
| **Yapılan sunum başarılı mı?** | x | 0-5 |  |
| **Soruları yanıtlama yetkinliği?** | x | 0-20 |  |
| **Devam Durumu** |  |  |  |
| **Öğrenci dönem içerisindeki raporlarını düzenli olarak hazırladı mı?** | x | 0-5 |  |
| **Diğer Maddeler** |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| **Toplam** |  |  |  |

Danışman (Jüri adına):

danışman imzası: