A

Project Report

on

Brain Age Estimation using Machine Learning

submitted in partial fulfillment of the requirements

for the award of the degree of

Bachelor of Technology

in

INFORMATION TECHNOLOGY

submitted by

Sainath D. Rudrurkar(20150751)

Atish R. Bagate (20150702)

Shivam D. Bobde(20160772)

Sana Fakir(20160774)

Trupti U. Bandgar(20150703)

Under the guidance of

prof. S.R.Hivre

DEPARTMENT OF INFORMATION TECHNOLOGY

DR. BABASAHEB AMBEDKAR TECHNOLOGICAL UNIVERSITY

Lonere-402103, Tal. Mangaon, Dist. Raigad (MH) INDIA

2018-2019

DR. BABASAHEB AMBEDKAR TECHNOLOGICAL UNIVERSITY
Lonere-402103,Tal-Mangaon,dist raigad(MH),INDIA
DEPARTMENT OF INFORMATION TECHNOLOGY

Certificate

The Project Report entitled **" Brain Age Estimation using Machine Learning"** submitted by **Sainath D. Rudrurkar(20150751), Atish Radhakishan Bagate (Roll.No:20150702), Shivam D. Bobde(20160772), Sana Fakir(20160774), Trupti U. Bandgar(20150703)** is approved for the partial fulfillment of the requirement for the award of the degree of Bachelor of Technology in Information Technology of the Dr.Babasaheb Ambedkar Technological University,Lonere is a Bonafide work carried out during the acadamic year 2018-2019.

prof.S.R.Hivre	**Dr. S.M. Jadhav**
(Project Report Guide)	(Head Of Department)
Information Technology	Information Technology

External Faculty:-

(-)

(-)

Acknowledgment

We are pleased to present this project report entitled **"Brain Age Estimation using Machine Learning"**. It is indeed a great pleasure and a moment of immense satisfaction for me to express my sense of profound gratitude and indebtedness towards our guide **Prof.S.R.Hivre** whose enthusiasm is the source of inspiration for us . we are extremely thankful for the guidance and untiring attention, which she bestowed on me right from the beginning. Her valuable and timely suggestions at crucial stages and above all her constant encouragement have made it possible for us to achieve this work. we would like to give sincere thanks to **Dr.S.M.JADHAV** Head Of **INFORMATION TECHNOLOGY** for necessor help for providing us required facilities for completion of this project report.I woud like to thank the entire teaching staff who are directly or indirectly involved in the various data collection and software assistance to bring forward this project report. I express my deep sense of gratitude towards my parents for their sustained cooperation and wishes, which have been a prime source of inspiration to take this Internship work to its end without any hurdles. Last but not the least, We would like to thank all my B.Tech. colleagues for their co-operation and useful suggestions and all those who have directly or indirectly helped me in completion of this work.

Place: Dr. Babasaheb Ambedkar Technological University, Lonere.
Date:

Abstract

The Project **"Brain Age Estimation using Machine Learning"** in Machine learning analysis of neuroimaging data can accurately predict age in healthy people and deviations from healthy brain ageing have been associated with cognitive impairment and disease. Here we sought to further establish the credentials of 'brain-predicted age' as a biomarker of individual differences in the brain ageing process, using a predictive modelling approach based on deep learning, and specifically convolutional neural networks (CNN), and applied to both pre-processed and raw T1-weighted MRI data.

Firstly, we aimed to demonstrate the accuracy of CNN brain-predicted age using a large dataset of healthy adults. Next, we sought to establish the heritability of brain-predicted age using a sample of MRI scan.

CNN accurately predicted chronological age using GM (correlation between brain-predicted age and chronological age r = 0.96, mean absolute error [MAE] and raw (r) data.

Brain-predicted age represents an accurate, highly reliable and genetically-valid phenotype, that has potential to be used as a biomarker of brain ageing. Moreover, age predictions can be accurately generated on raw T1-MRI data, substantially reducing computation time for novel data, bringing the process closer to giving real-time information on brain health in clinical settings.

Contents

List of Figures

Chapter 1

INTRODUCTION

The human brain changes across the adult lifespan. This process of brain ageing occurs in accord with a general decline in cognitive performance, cognitive ageing . Although the changes associated with brain ageing are not explicitly pathological, with increasing age comes increasing risk of neurodegenerative disease and dementia. However, the wide range of onset ages for age-associated brain diseases indicates that the effects of ageing on the brain vary greatly between individuals. Thus, advancing our understanding of brain ageing and identifying biomarkers of the process are vital to help improve detection of early-stage neurodegeneration and predict age-related cognitive decline.

One promising approach to identifying individual differences in brain ageing derives from the research showing that neuroimaging data, most commonly T1-weighted MRI, can be used to accurately predict chronological age in healthy individuals, using machine learning . By 'learning' the correspondence between patterns in structural or functional neuroimaging data and an age 'label', machine-learning algorithms can formulate massively high-dimensional regression models, fitting large neuroimaging datasets as independent variables to predict chronological age as the dependent variable. The resulting brain-based age predictions are generally highly accurate, particularly when algorithms learn from large training datasets and are applied to novel or 'left-out' data (i.e., test datasets).

Beyond improving clinical applicability, a biomarker of brain ageing needs to relate

to naturally occurring variation, such as that caused by genetic factors. Many aspects of brain ageing and susceptibility to age-related brain disease are thought to be under genetic influence . Therefore, demonstrating a brain ageing biomarker is sensitive to genetic influences gives some external, genetic, validity to the measure. Furthermore, if a neuroimaging biomarker is significantly heritable, this motivates further research into specific candidate genes, or sets of genes, that may affect this aspect of brain ageing. These candidate genes can then, in turn, provide biological targets for pharmacological interventions which aim to improve brain health in older adults.

- Through analysis of structural magnetic resonance imaging (MRI) images, classification and prediction of the age of adults were implemented to make analysis of the age-related.
- Due to the distributed nature of the aging spatial pattern in human brain, a multivariate voxel selection method based on sparse representation was introduced to identify the most discriminative brain regions.
- It can effectively pick out the isolating voxels as well as the clustered voxels which contribute enormously to the classification and age prediction.
- Difference trend of the brain development between the senior and the junior was observed.
- Four different models were used to fit the predicted age of all subjects in space.

Chapter 2

SYSTEM ANALYSIS

2.1 SOFTWARE AND HARDWARE REQUIREMENTS

2.1.1 HARDWARE REQUIREMENTS

- Processor : Pentium 3 or higher. item RAM : more than 2 GB.
- Hard Disk : at least 4 GB for application.
- Graphics Card : AMD Redon or Nvidia at least 1 GB.

2.1.2 SOFTWARE REQUIREMENTS

- Operating system : Windows XP, UBANTU, MAC, ANDROID, WINDOWS 10.
- Back End :TENSERFLOW.
- IDE : WEB-BROWSER.
- GOOGLE COLAB NOTEBOOK.

Chapter 3

PROBLEM STATEMENT:

- In this project we apply regression techniques to predict a person's age from a three-dimensional magnetic resonance image(MRI) of their brain.
- Magnetic Resonance Imaging (MRI) is a key technology in medical imaging as it provides a tool to investigate sensitive organs such as the brain.
- MRI typically operates on a medium length scale which corresponds to a resolution of about 1mm.
- We will learn how to deal with real medical image data, and how to process it to apply machine learning techniques successfully.
- In this project we start with a more modest goal and try to predict the age of a person from their brain MR image.

Chapter 4

LIBRARIES USED:

- numpy.
- pandas.
- Scikit-learn.
- matplotlib.pylab.
- seaborn.
- NiBabel.
- (CSV file.) IXI-MRI : Large-scale MRA Dataset.
- GITHUB- for database availability.

4.1 GOOGLE COLAB NOTEBOOK

Colaboratory is a Google research project created to help disseminate machine learning education and research. It's a Jupyter notebook environment that requires no setup to use and runs entirely in the cloud.

Whether a student interested in exploring Machine Learning but struggling to conduct simulations on enormous datasets, or an expert playing with ML desperate for extra computational power, Google Colab is the perfect solution. Google Colab or "the Colaboratory" is a free cloud service hosted by Google to encourage Machine Learning and Artificial Intelligence research, where often the barrier to learning and success is the requirement of tremendous computational power.

4.1.1 Benefits of Colab

Besides being easy to use, the Colab is fairly flexible in its configuration and does much of the heavy lifting.

- Python 2.7 and Python 3.6 support.
- Free GPU acceleration.
- Pre-installed libraries: All major Python libraries like TensorFlow, Scikit-learn, Matplotlib among many others are pre-installed and ready to be imported.
- Google Colab notebooks are stored on the drive.

4.2 numpy

Numpy is a math library for python. It enables us to do computation efficiently and effectively. It is better than regular python because of it's amazing capabilities.

numpy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

NumPy by incorporating features of the competing Numarray into Numeric, with extensive modifications. NumPy is open-source software and has many contributors.

4.2.1 Benefits of numpy

NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

- A powerful N-dimensional array object.
- Sophisticated (broadcasting) functions.
- Tools for integrating C/C++ and Fortran code.
- Useful linear algebra, Fourier transform, and random number capabilities.

Installation:Mac and Linux users can install NumPy via pip command:–¿ pip install numpy .

4.3 pandas

Pandas stands for "Python Data Analysis Library"

pandas is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language.

What's cool about Pandas is that it takes data (like a CSV or TSV file, or a SQL database) and creates a Python object with rows and columns called data frame that looks very similar to table in a statistical software (think Excel or SPSS for example. o.3inThis is so much easier to work with in comparison to working with lists and/or dictionaries through for loops or list comprehension.

4.3.1 Benefits of numpy

- Loading and Saving Data with Pandas.
- Viewing and Inspecting Data.
- Selection of Data.
- Filter, Sort and Groupby.
- Data Cleaning.

4.4 Scikit-learn

it is a free software machine learning library for the Python programming language.

Scikit-learn is one the most popular ML libraries. It supports many supervised and unsupervised learning algorithms. Examples include linear and logistic regressions, decision trees, clustering, k-means and so on.

It builds on two basic libraries of Python, NumPy and SciPy. It adds a set of algorithms for common machine learning and data mining tasks, including clustering, regression and classification. Even tasks like transforming data, feature selection and ensemble methods can be implemented in a few lines.

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.

Scikit-learn is largely written in Python, with some core algorithms written in Cython to achieve performance. Support vector machines are implemented by a Cython wrapper around LIBSVM; logistic regression and linear support vector machines by a similar wrapper around LIBLINEAR.

4.4.1 Scikit-learn features:

- Classification.
- Regression.
- Clustering.
- Dimensionality reduction.
- Preprocessing.

4.5 Matplotlib

it takes you through the basics Python data visualization: the anatomy of a plot, pyplot and pylab, and much more.

his library is very flexible and has a lot of handy, built-in defaults that will help you out tremendously.

you need to make the necessary imports, prepare some data, and you can start plotting with the help of the plot() function! When you're ready, don't forget to show your plot using the show() function.

4.5.1 Matplotlib features:

- Plot creation.which could raise questions about what module you exactly need to import (pylab or pyplot?), how you exactly should go about initializing the figure and the Axes of your plot, how to use matplotlib in Jupyter notebooks, etc.
- Plotting routines,from simple ways to plot your data to more advanced ways of visualizing your data.
- Basic plot customizations, with a focus on plot legends and text, titles, axes labels and plot layout.
- Saving, showing, clearing, ... your plots: show the plot, save one or more figures to, for example, pdf files, clear the axes, clear the figure or close the plot, etc.
- you can customize Matplotlib: with style sheets and the rc settings.

4.5.2 Tensorflow

The interesting thing about Tensorflow is that when you write a program in Python, you can compile and run on either your CPU or GPU. So you don't have to write at the C++ or CUDA level to run on GPUs.

It uses a system of multi-layered nodes that allows you to quickly set up, train, and

deploy artificial neural networks with large datasets. This is what allows Google to identify objects in photos or understand spoken words in its voice-recognition app.

4.6 Seaborn

inOne of the best but also more challenging ways to get your insights across is to visualize them: that way, you can more easily identify patterns, grasp difficult concepts or draw the attention to key elements. When you're using Python for data science, you'll most probably will have already used Matplotlib, a 2D plotting library that allows you to create publication-quality figures. Another complimentary package that is based on this data visualization library is Seaborn, which provides a high-level interface to draw statistical graphics.

Seaborn is complimentary to Matplotlib and it specifically targets statistical data visualization. But it goes even further than that: Seaborn extends Matplotlib and that's why it can address the two biggest frustrations of working with Matplotlib. Or, as Michael Waskom says in the "introduction to Seaborn": "If matplotlib "tries to make easy things easy and hard things possible", seaborn tries to make a well-defined set of hard things easy too.

One of these hard things or frustrations had to do with the default Matplotlib parameters. Seaborn works with different parameters, which undoubtedly speaks to those users that don't use the default looks of the Matplotlib plots.

4.6.1 Benefits of Seaborn:

- Nicer Default Aesthetics-One of the biggest advantages of seaborn is that its default aesthetics are much more visually appealing than matplotlib.

- Easily Customizable Aesthetics-If you want to change either the background or the colors of all your graphs, you can do so easily with two commands: sns.set$_s$*tyleandsns.set*$_p$*alette.Statistically-MindedPlots.*

4.7 NiBabel

NiBabel supports an ever growing collection of neuroimaging file formats. Every file format format has its own features and pecularities that need to be taken care of to get the most out of it. To this end, NiBabel offers both high-level format-independent access to neuroimages, as well as an API with various levels of format-specific access to all available information in a particular file format.

When loading an image, NiBabel tries to figure out the image format from the filename. An image in a known format can easily be loaded by simply passing its filename to the load function.

To start the code examples, we load some useful libraries:
import os pip install nibabel
Then we fine the nibabel directory containing the example data:
from nibabel.testing import $data_path$

4.8 GITHUB- for database availability.

GitHub was founded back in 2007 by PJ Hyett.

GitHub is a web-based hosting service for version control using Git. It is mostly used for computer code. It offers all of the distributed version control and source code management (SCM) functionality of Git as well as adding its own features.

The company has its own private repository on GitHub to develop GitHub, of course.

GitHub was created initially by us, for us, and we are all basically software engineers and we want to use a good tool to do this.

It provides access control and several collaboration features such as bug tracking, feature requests, task management, and wikis for every project. GitHub offers plans for enterprise, team, pro and free accounts which are commonly used to host open-source software projects.

4.8.1 Benefits of GITHUB

- Documentation, including automatically rendered README files in a variety of Markdown-like file formats (see README files on GitHub).
- Issue tracking (including feature requests) with labels, milestones, assignees and a search engine
- Pull requests with code review and comments.
- Commits history.

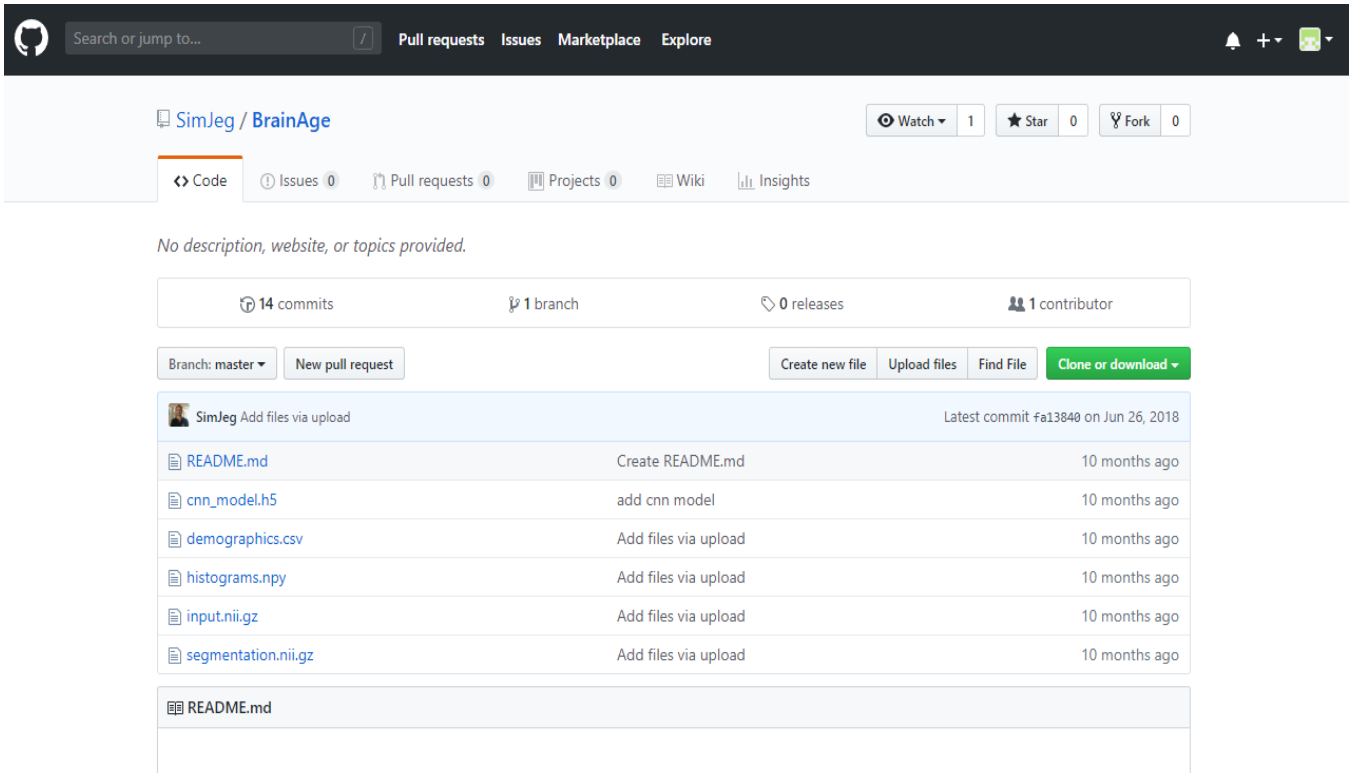


Figure 4.1: GITHUB user interface

Chapter 5

DATASET:

5.1 What is dataset:

One of the hardest problems to solve in deep learning has nothing to do with neural nets: it's the problem of getting the right data in the right format.

Machine learning typically works with two data sets: training and test. All three should randomly sample a larger body of data.

The first set you use is the training set, the largest of the three. Running a training set through a neural network teaches the net how to weigh different features.

The second set is your test set. It functions as a seal of approval, and you don't use it until the end. After you've trained and optimized your data, you test your neural net against this final random sampling.

	id	dataset	sex	hospital	age	path_process_data
0	0	ixi	f	Guys	35.800137	IXI002/
1	1	ixi	m	HH	38.781656	IXI012/
2	2	ixi	m	HH	46.710472	IXI013/
3	3	ixi	f	HH	34.236824	IXI014/
4	4	ixi	m	HH	24.284736	IXI015/
5	5	ixi	m	Guys	55.167693	IXI016/
6	6	ixi	f	Guys	29.092402	IXI017/
7	7	ixi	m	Guys	58.658453	IXI019/
8	8	ixi	m	Guys	39.466119	IXI020/
9	9	ixi	f	Guys	21.566051	IXI021/

Figure 5.1: dataset parameters

```
There are 1597 patients
There are 25 hospitals
There are 2 datasets
There are 55.48% of women
Age statistics :
count    1597.000000
mean       35.599132
std        17.491271
min        18.000000
25%        22.000000
50%        27.000000
75%        48.052019
max        86.318960
Name: age, dtype: float64
```

Figure 5.2: dataset

Chapter 6

Exploration of the dataset

We can use the packages matplotlib (plt) and seaborn (sns) to plot figures. Figure 1 shows that patients are not equally distributed across hospitals. Figure 2 shows that ages are not equally distributed across hospitals.

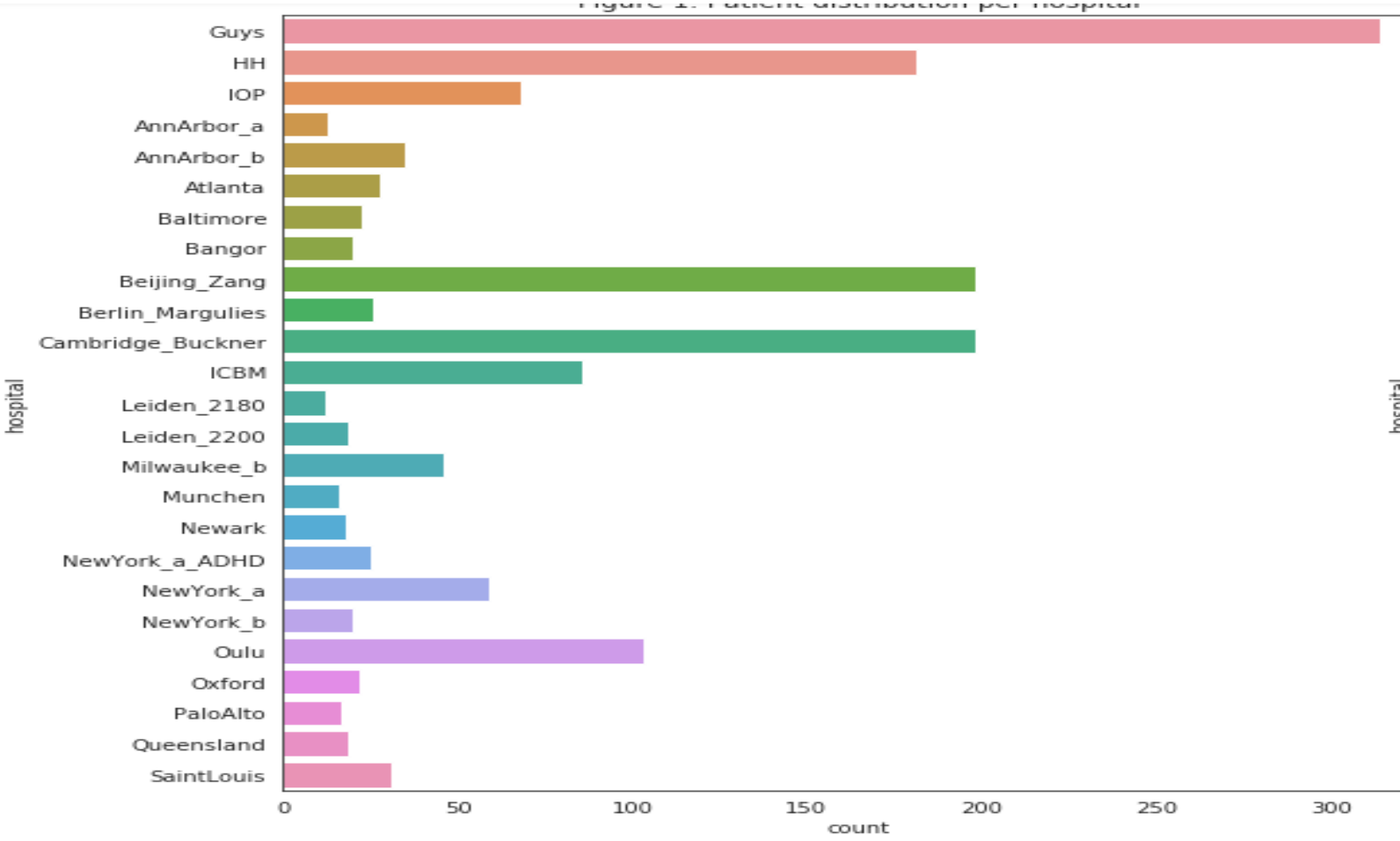


Figure 6.1: dataset according to patient count over hospitals

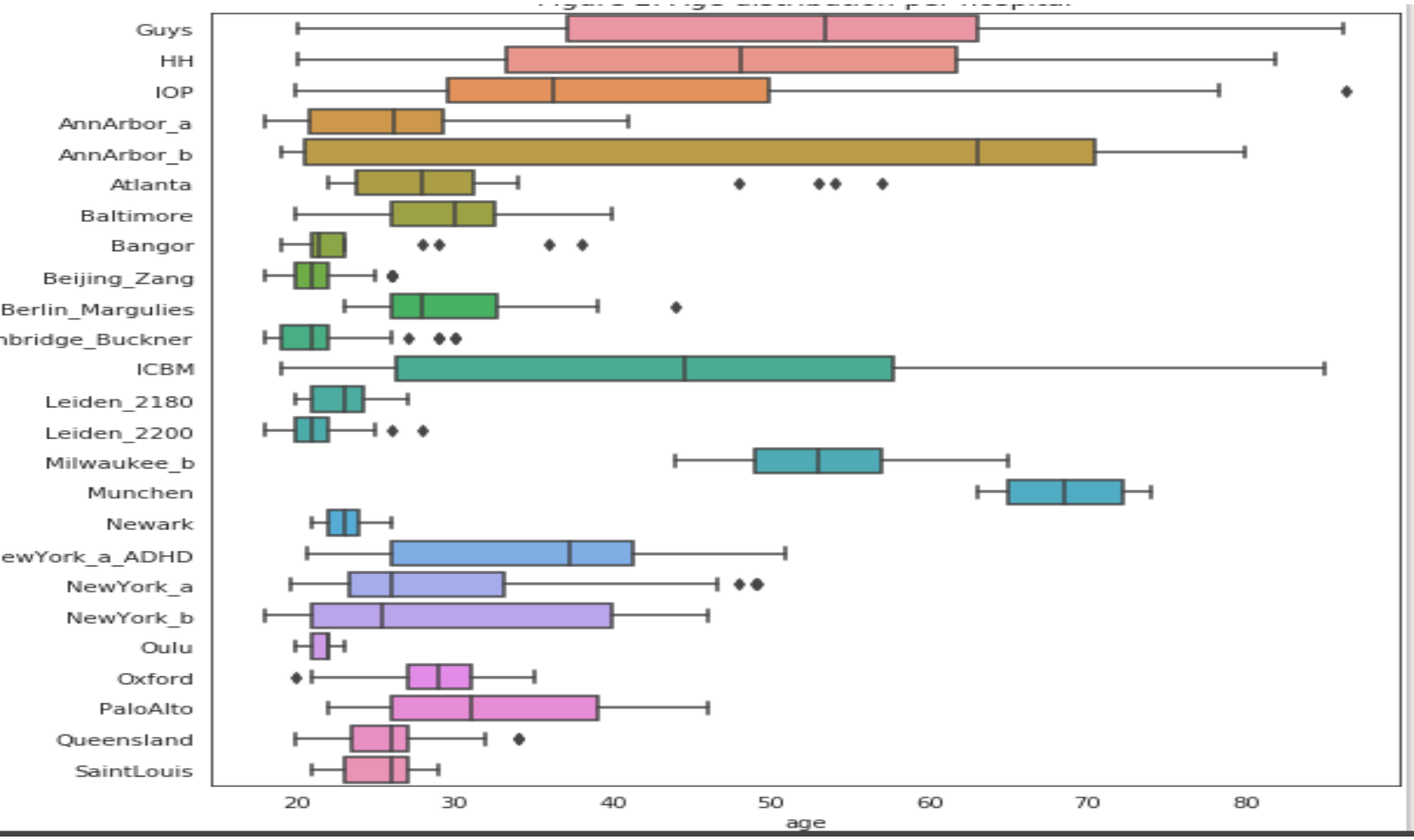


Figure 6.2: dataset according the age distribution over hospital

Chapter 7

one plot of MRI of a subject.

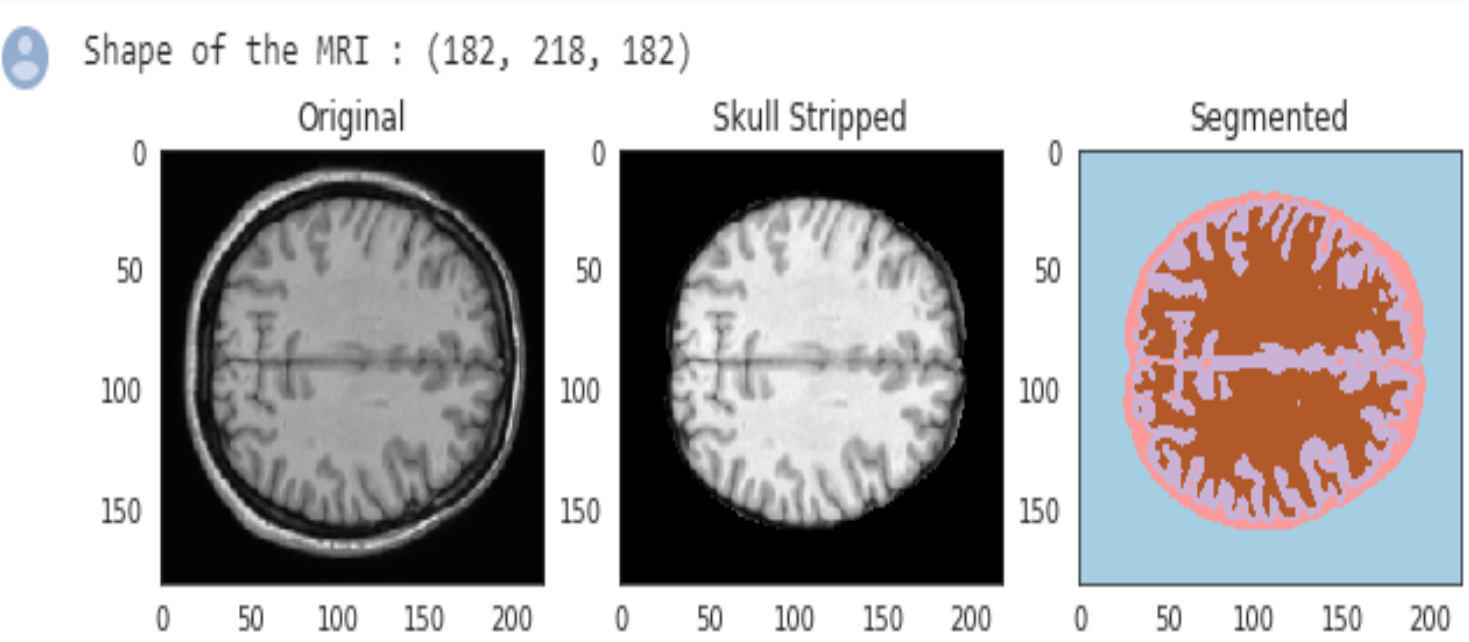


Figure 7.1: shape of MRI brain

Chapter 8

Load and visualize histograms

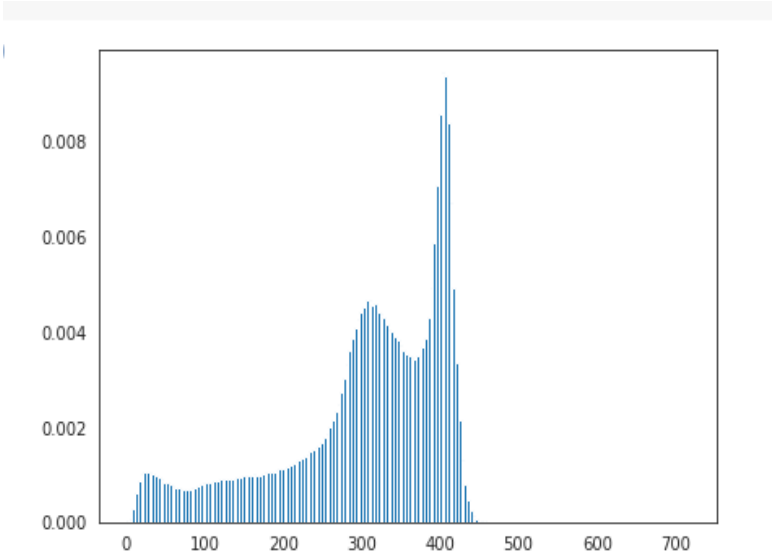


Figure 8.1: USE CASE DIAGRAMS

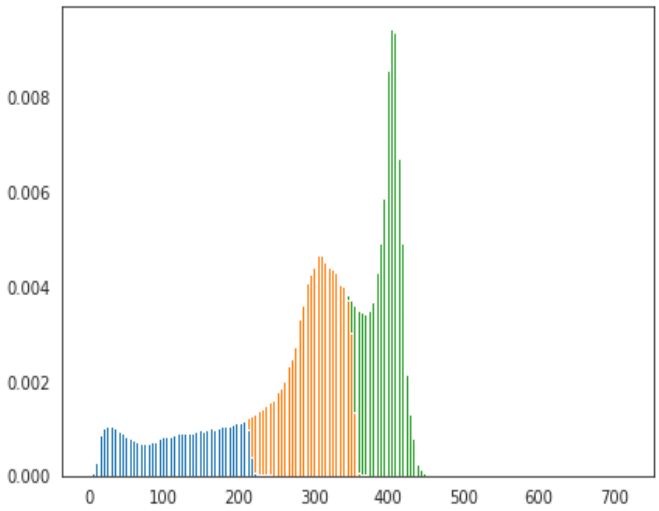


Figure 8.2: USE CASE DIAGRAMS

8.1 average histograms per hospital.

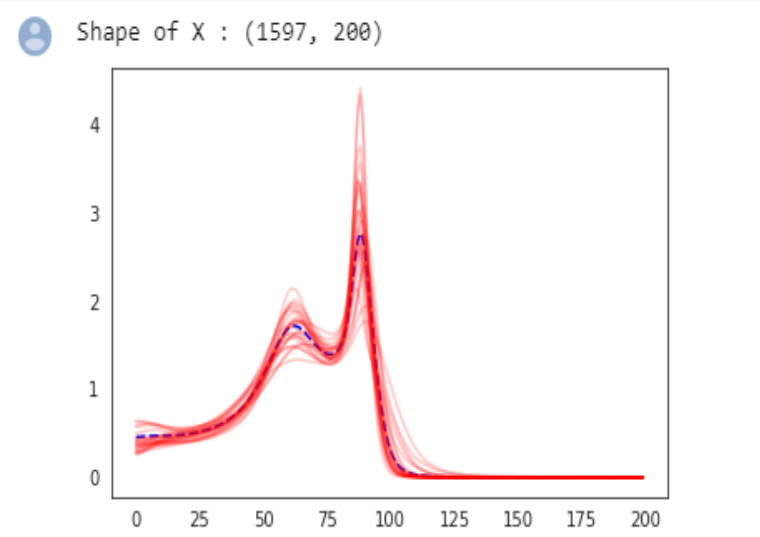


Figure 8.3: USE CASE DIAGRAMS

Chapter 9

Train a linear model

9.1 random training model

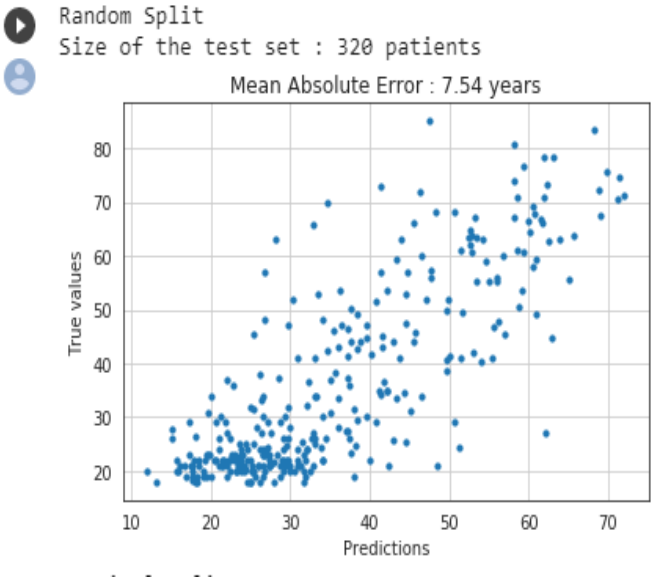


Figure 9.1: USE CASE DIAGRAMS

9.2 hospital spilt

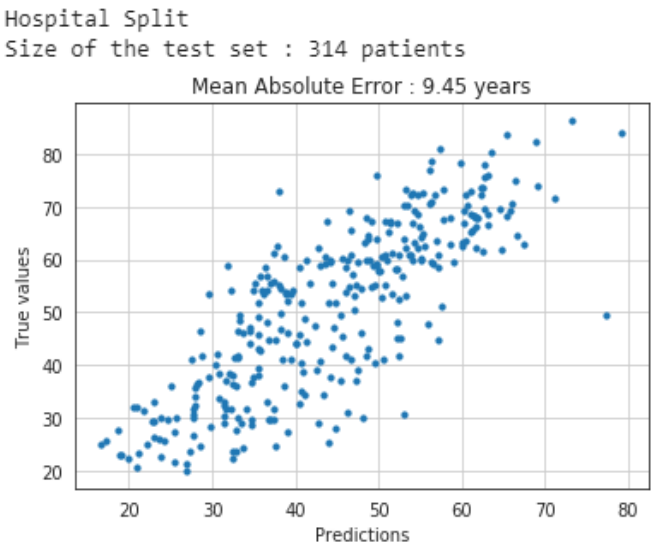


Figure 9.2: USE CASE DIAGRAMS

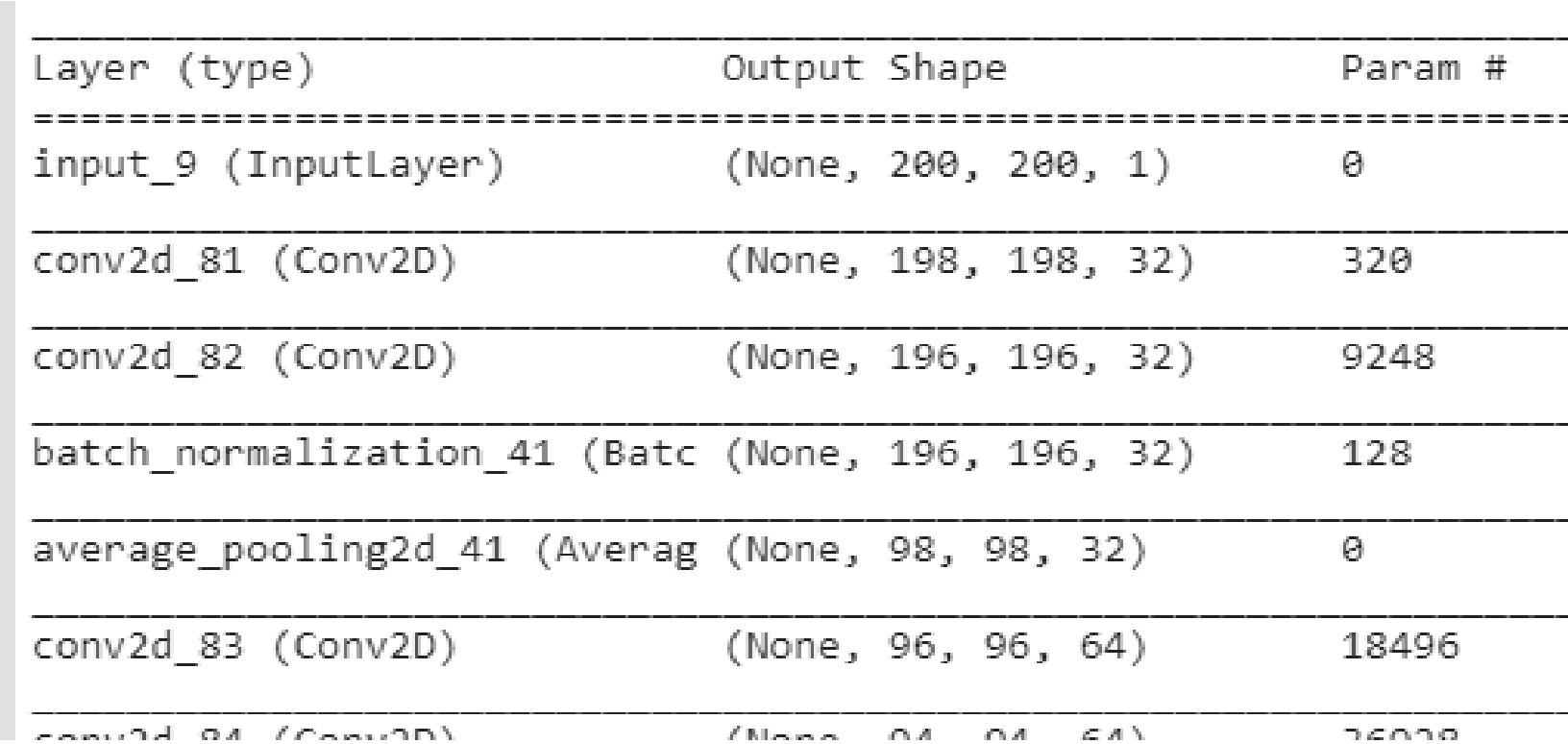
Chapter 10

Convolutional Neural Networks

10.1 tenserflow use in backend.

```
Using TensorFlow backend.
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow/python/framework/op_def_library.py:263: colocate_with (from tensorfl
Instructions for updating:
Colocations handled automatically by placer.
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow/python/ops/math_ops.py:3066: to_int32 (from tensorflow.python.ops.ma
Instructions for updating:
Use tf.cast instead.
```

Figure 10.1: USE CASE DIAGRAMS



Layer (type)	Output Shape	Param #
input_9 (InputLayer)	(None, 200, 200, 1)	0
conv2d_81 (Conv2D)	(None, 198, 198, 32)	320
conv2d_82 (Conv2D)	(None, 196, 196, 32)	9248
batch_normalization_41 (Batc	(None, 196, 196, 32)	128
average_pooling2d_41 (Averag	(None, 98, 98, 32)	0
conv2d_83 (Conv2D)	(None, 96, 96, 64)	18496
conv2d_84 (Conv2D)	(None, 94, 94, 64)	36928

Figure 10.2: USE CASE DIAGRAMS

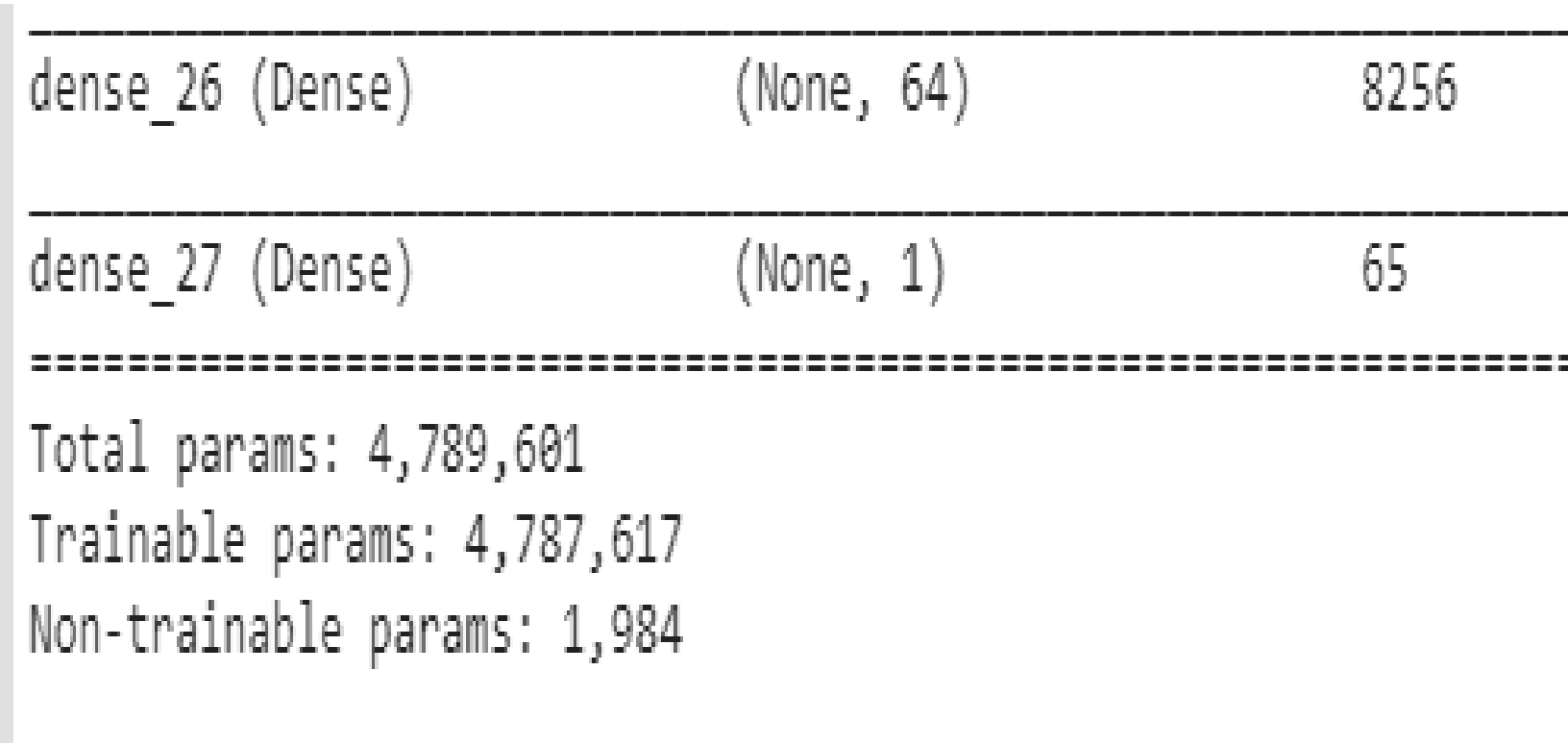


Figure 10.3: USE CASE DIAGRAMS

Chapter 11

OUTPUT

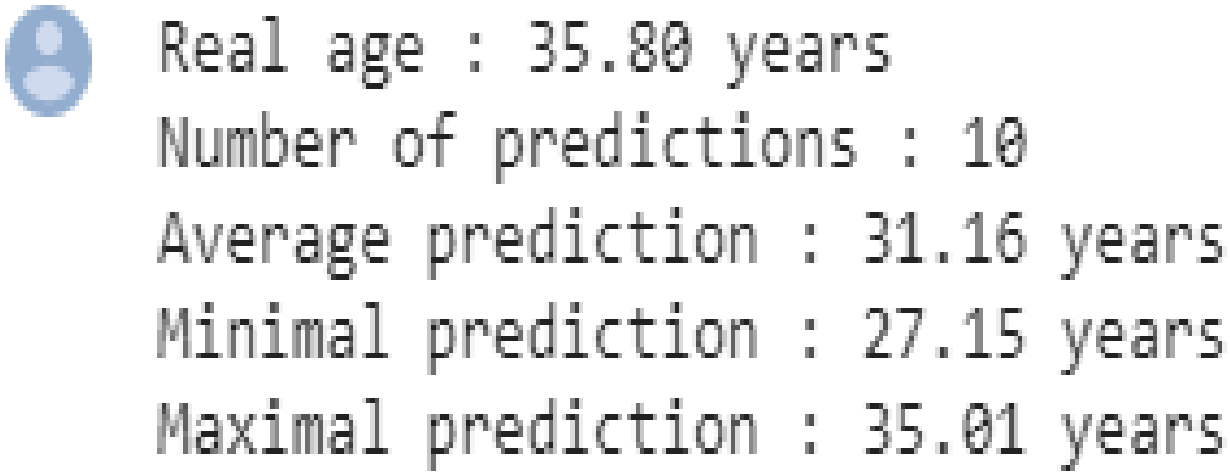


Figure 11.1: USE CASE DIAGRAMS

]

Chapter 12

SNAPSHOOT

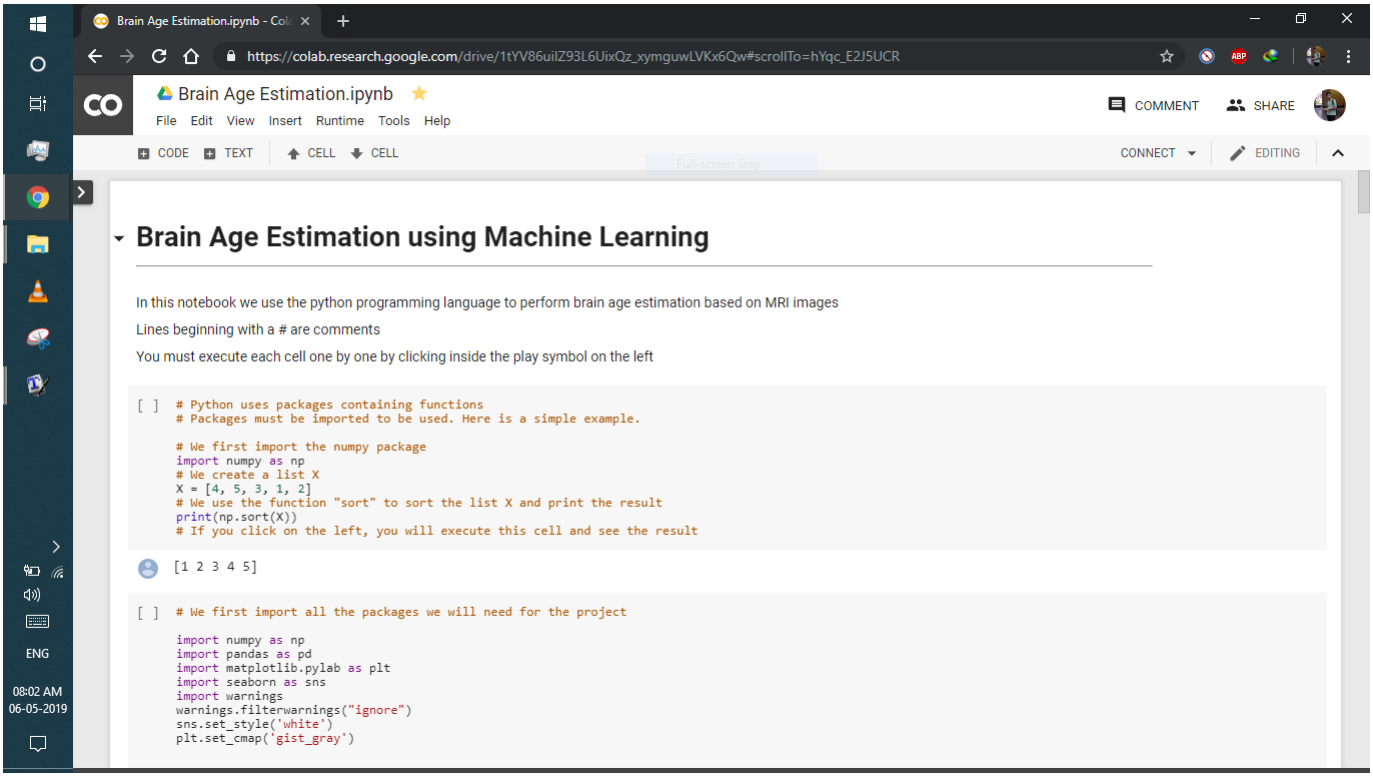


Figure 12.1: USE CASE DIAGRAMS

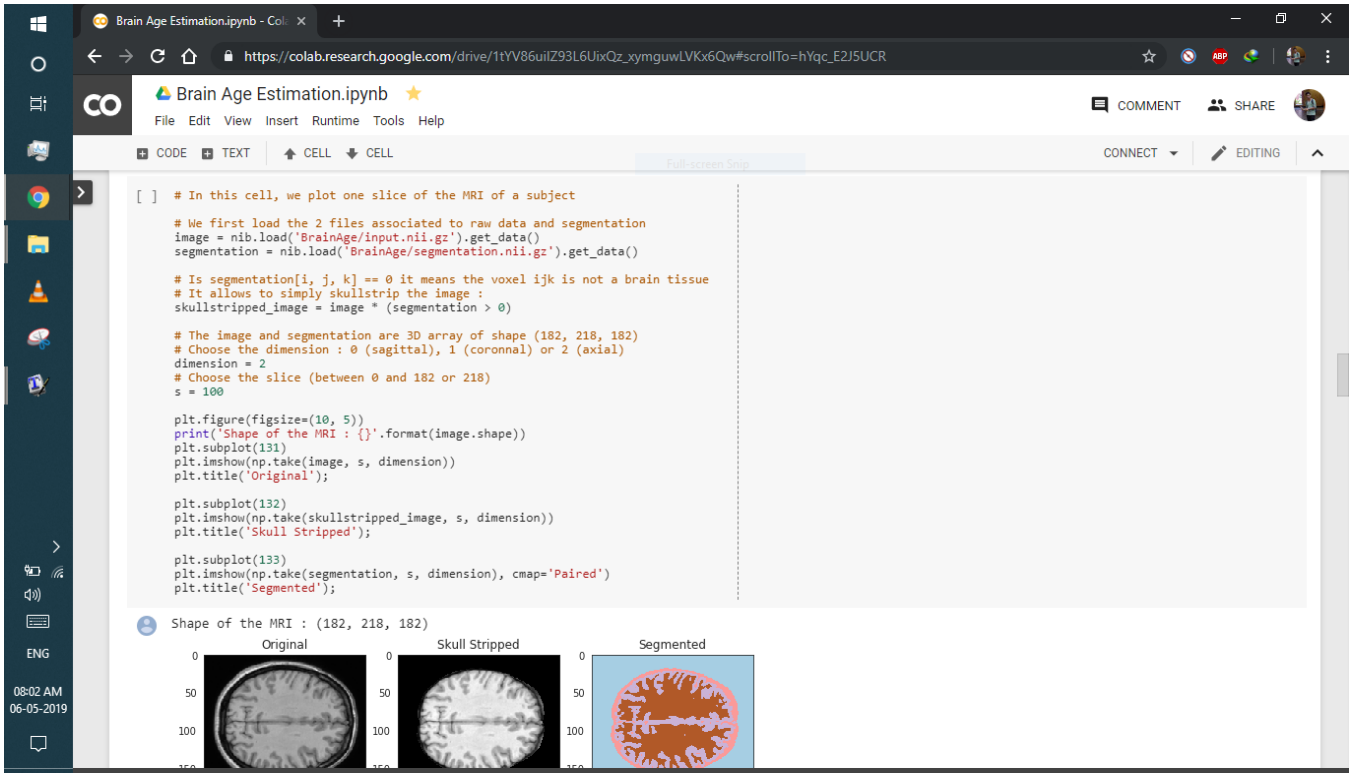


Figure 12.2: bill for print

Histograms

In this section we will create the histograms of the MRIs and train a linear model on it to predict the age

Load and visualize histograms

```
[ ]  # We first show how an histogram looks like

     plt.figure(figsize=(15, 5))
     plt.subplot(121)
     # We only use POSITIVE values (background removed) to compute the histogram
     plt.hist(skullstripped_image[skullstripped_image > 0], bins=200, density=True);

     # Now use the segmentation and superpose the histograms of CSF (blue),
     # white matter (green) and grey matter (red) intensity values
     plt.subplot(122)
     plt.hist([skullstripped_image[segmentation == i] for i in [1, 2, 3]],
              stacked=True, bins=200, density=True);
```

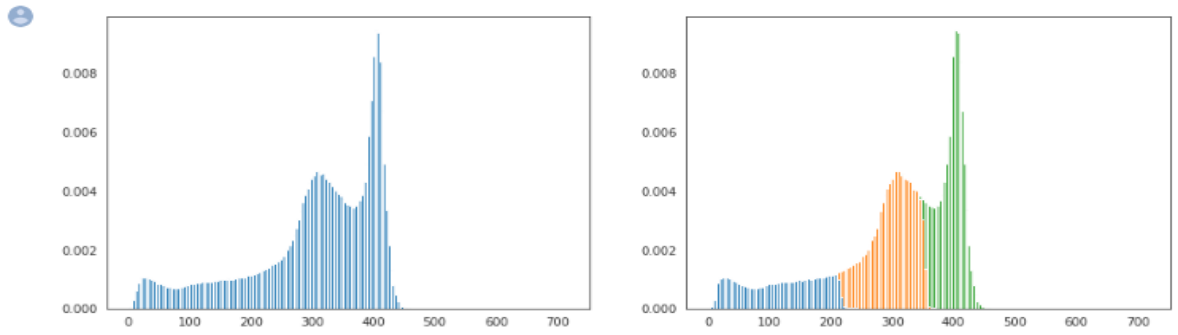


Figure 12.3: Admin Login

Train a linear model

```
[ ]  10**np.linspace(-3.0, 2.0, 10)
```

```
[ ]  # Scikit Learn is the most popular package for machine learning in python
     # We use it to train a linear model with L2 regularization (Ridge Regression)

     from sklearn.model_selection import train_test_split
     from sklearn.linear_model import RidgeCV
     n_patients = len(df)

     # We create a function to train and test a linear model

     def train_linear_regression(train, test):
         # Create model
         model = RidgeCV()

         # Train on the training set : fit the models with pairs (X, Y)
         model.fit(X[train], Y[train])

         # Predict on the test set
         predictions = model.predict(X[test])
         true_values = Y[test]

         # Plot the results and compute the MAE
         plt.scatter(predictions, true_values, s=10)
         plt.grid()
         plt.xlabel('Predictions')
         plt.ylabel('True values')
         mae = np.mean(np.abs(predictions - true_values))
         plt.title('Mean Absolute Error : {:.2f} years'.format(mae))
         plt.show()


     # We don't perform 5 fold cross validation here, but a single split

     # 1. Random split
     # We randomly split the data between train and test with 80/20 proportions
     train, test = train_test_split(np.arange(n_patients), test_size=0.2,
                                    random_state=0)
     print('Random Split')
     print('Size of the test set : {} patients'.format(len(test)))
     # We use our function on this split
     train_linear_regression(train, test)

     # 2. Hospital split
     # We split the hospitals using Guys hospital as the test set
     train = np.where(df.hospital != 'Guys')[0]
     test = np.where(df.hospital == 'Guys')[0]
     print('Hospital Split')
     print('Size of the test set : {} patients'.format(len(test)))
     train_linear_regression(train, test)
```

Figure 12.4: gallery and information

Convolutional Neural Networks

In this section we load a trained convolutional neural network and apply it on a single patient

```
[ ] # We will use the keras package for deep learning models
    from keras.models import load_model

    # Training a model is very long, so here we simply load a trained model
    model = load_model('BrainAge/cnn_model.h5')

    # We show the model summary
    # The input is an image of size (200, 200, 1) and the output the age of size 1
    # You can see different layers : Convolutional, Pooling, Dense, and BatchNorm

    model.summary()
```

```
Using TensorFlow backend.
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow/python/framework/op_def_library.py:263: colocate_with (from tensorflow.python.framework.ops) is deprecated and will be removed in
Instructions for updating:
Colocations handled automatically by placer.
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow/python/ops/math_ops.py:3066: to_int32 (from tensorflow.python.ops.math_ops) is deprecated and will be removed in a future version
Instructions for updating:
Use tf.cast instead.
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_9 (InputLayer)         (None, 200, 200, 1)       0
_________________________________________________________________
conv2d_81 (Conv2D)           (None, 198, 198, 32)      320
_________________________________________________________________
conv2d_82 (Conv2D)           (None, 196, 196, 32)      9248
_________________________________________________________________
batch_normalization_41 (Batc (None, 196, 196, 32)      128
_________________________________________________________________
average_pooling2d_41 (Averag (None, 98, 98, 32)        0
_________________________________________________________________
conv2d_83 (Conv2D)           (None, 96, 96, 64)        18496
```

Figure 12.5: Stocks and Records

```
[ ]  # Get predictions : we first need to prepare the input image

     X = skullstripped_image

     # Step 1 : Normalize Intensities
     h, t = np.histogram(X[segmentation == 3], bins=100)
     peak_wm = t[np.argmax(h)]
     h, t = np.histogram(X[segmentation == 2], bins=100)
     peak_gm = t[np.argmax(h)]

     X /= peak_wm
     C = peak_gm / peak_wm
     a = (0.75 - C**2) / (C - C**2)
     X[X < 1] = a * X[X < 1] + (1 - a) * X[X < 1]**2
     X = np.clip(X, 0, 1.5) / 1.5

     # Step 2 : resize the image to get (200, 200) shape

     X = X[:, 10:-8, 80:90]
     X = np.pad(X, ((9,9), (0,0), (0,0)), mode='constant')
     X = np.rollaxis(X, 2, 0)[..., None]

     # Step 3 : use the trained model to make the predictions

     predictions = 10 * model.predict(X)[:, 0]
```

```
[ ]  # The image provided here is the first of the dataset
     # This image was NOT in the training set of the  model

     real_age = df.age[0]
     print('Real age : {:.2f} years'.format(real_age))

     # Our model made a prediction for each axial slice (10 in total)
     # That's fact was a bit hidden in the blog post !
     print('Number of predictions : {}'.format(len(predictions)))
     print('Average prediction : {:.2f} years'.format(np.mean(predictions)))
     print('Minimal prediction : {:.2f} years '.format(np.min(predictions)))
     print('Maximal prediction : {:.2f} years '.format(np.max(predictions)))
```

```
Real age : 35.80 years
Number of predictions : 10
Average prediction : 31.16 years
Minimal prediction : 27.15 years
Maximal prediction : 35.01 years
```

Figure 12.6: Customer Data

Chapter 13

CONCLUSION

"Billing Software For Cloth Center" is successfully designed anddeveloped to fullling the necessary requirements, as identied in therequirements analysis phase, such as the system is very much user friendly, form level validation and eld level validation are performing very eciently.The new computerized system was found to be much faster and reliableand user friendly then the existing system, the system has been designed and developed step by step and tested successfully. It eliminates the human error that are likely.

"Billing Software For Cloth Center" software devoloped for an institute has been designed to achive maximum eciency and reduce the time taken to handle the storing activity.

It is designed to replace and existing managment system thereby reducing time taken for caculations and for storing data

REFRENCES

1. https://developer.ibm.com/articles/os-android-devel/

2. https://en.wikipedia.org/wiki/$_{(}software$).
   $https://code.visualstudio.com/docs/editor/whyvscode$

3. https://developer.android.com/studio/install

4. https://www.youtube.com