ANKARA UNIVERSITY

FACULTY OF ENGINEERING

DEPARTMENT OF COMPUTER ENGINEERING

COM3551 ARTIFICIAL INTELLIGENCE

PROJECT REPORT

IMAGE RECOGNITION
FOR DETECTION OF SUSPICIOUS PEOPLE

BENGÜ TERCAN 17290122

IRMAK GENÇ 17290100

YUSUF KAYSERİLİOĞLU 17290107

SUPERVISOR

ASSOC. DR. MEHMET SERDAR GÜZEL

Jan, 2021

TABLE OF CONTENTS

FIGURE LIST

1. INTRODUCTION

This report has been prepared to give information about the Image Recognition project of the course Artificial Intelligence. This course expects from us that develop a software program related to machine learning, deep learning or artificial intelligence using Python, Keras and TensorFlow or R language.

While determining the problem, the tendency of the market in this regard has been taken into consideration. In recent years, image recognition products have been more common in many fields such as image classification, face recognition, camera fields and so on. For this reason, the problem has been solved with image recognition.

Image recognition is the ability of a system or software to identify objects, people, places, and actions in images. It uses machine vision technologies with artificial intelligence and trained algorithms to recognize images through a file or camera system. The project was created by image classification to identify and categorize person in the image corresponding to their name.

In the first period of the project, the dataset was collected from the group members and it was created by the images of the group members named as Bengü, Irmak and Yusuf. After the data was made available to use, the CNN model was built and the model was trained. Then, after adding additional graphics, the predictions were made available for trials with usage of the Tkinter Application.

This report has detailed information about all the modules in the project. It will first present an overview of problem definition and aim of the project. Then, materials and methods will be described that creates the project. Additionally, use of the application will be explained in detail.

1.1. Problem Definition

The description of the problem is handled in this subsection.

Nowadays, many people attempt to commit an offense more than earlier as a result of the collapse of economy by the coronavirus. Because, people want to get rid of hunger and poverty that affect their lives adversely. Therefore; burglary, swindling and thuggery increased. In addition to this, workload on cops also increased.

In real-life, suspicious people such as burglars, murderers, impostors and swindlers can be diagnosed by the police officers by checking their personal identity cards one by one and searching their criminal records. But, this method is not practical way and it costs a lot of time and effort.

This project is developed to solve this important issue mentioned above. Using image classification brings more benefits for this problem.

1.2. Aim of the Project

The aim of this project is to increase the efficiency of time and offer convenience by creating a new method for detection of suspicious people.

Instead of checking these people's ID cards one by one, there is an easiest way that police men can take photos of the suspicious person and can detect if the suspicious person is guilty or not by loading his or her image into the application. Person in the image is found by the image recognition application and it shows a criminal record of his or her.

2. MATERIALS AND METHODS

2.1. Materials

This section focuses on platforms, libraries, software development environment and programming language which are used in this project.

The project is made of Python programming language because it includes various useful libraries and functions. There are many platforms on the market to develop project with python such as PyCharm, Jupyter Notebook, Visual Studio Code and Google Colab.

2.1.1. Jupyter Notebook

The Jupyter Notebook App is a server-client application that allows editing and running notebook documents via a web browser. The Jupyter Notebook App can be executed on a local desktop requiring no internet access or can be installed on a remote server and accessed through the internet.

In the project, Jupyter Notebook is prefered because it provides convenience to execution of the program line by line and instead of Google Colab, dataset can be accessed directly from the harddisk of computer or ssd. This advantages of The Jupyter Notebook have made the practice of this project easier.

2.1.2. Libraries

The libraries used to create the program and execute the CNN model are listed below:

- TensorFlow and Keras
- Numpy
- Tkinter
- Python Imaging Library (PIL)
- OS
- Matplotlib

2.1.3. Tkinter Graphical User Interface

Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit.

In the project, Tkinter converts the program usable for the users. It provides a practical way to execute the program for users.

2.2. Methods

In this part of the report, the technical methods and procedures which are used in the project will be discussed in detail.

The Convolutional Neural Network has both similarities and differences with human brain. Most particularly,it is used for image recognition.In this project, the CNN model is used to identify the images by separating them corresponding to their classes. There are many steps to create Convolutional Neural Network.

The first building block is convolution operation. This step includes usage of feature detectors, which basically serve as the neural network's filters. Also, it has discussion about feature maps, learning the parameters of such maps, how patterns are detected, the layers of detection, and how the findings are mapped out. The second part of this step will involve the Rectified Linear Unit or ReLU. It is covered with ReLU layers and explorement how linearity functions in the context of Convolutional Neural Networks.

The second building block is pooling. The project uses the max pooling. Max pooling is done to in part to help over-fitting by providing an abstracted form of the representation. It is the process of merging. So it’s basically for the purpose of reducing the size of the data.

The third building block is flattening. Flattening is converting the data into a 1-dimensional array for inputting it to the next layer. The output of the convolutional layers are flattened to create a single long feature vector and it is connected to the final classification model, which is called a fully-connected layer.

The last building block is full connection. In this part, everything that covered throughout in the first, second and third steps are merged together.

The investigation of these steps are more detailed as follows:

2.2.1. Convolution

Convolution is a mathematical operation on two objects to produce an outcome that expresses how the shape of one is modified by the other.

With this computation, we detect a particular feature from the input image and get the result having information about that feature. This is called “feature map”. Here are the three elements that enter into the convolution operation:

- Input image
- Feature detector
- Feature map

Input image can have any size. For instance, assume that image has 64x64 pixel data. If this image is not black and white, data has a 64x64x3 volume because of red layer,green layer and blue layer. In the project, input images have also 64x64x3 volume.

Feature detector can be 3x3, 5x5 or a 7x7 matrix but the project is made with the 3x3 feature detector. The feature detector is often referred to as a "kernel" or a "filter" in the project.

Passing the detector from left to right with movements across each pixels, the convolution value is taken for each time and filled in a matrix named as “feature map”. Feature map is also known as an “activation map”. In the project, one-pixel strides are used for feature mapping and that gives a fairly large feature map.

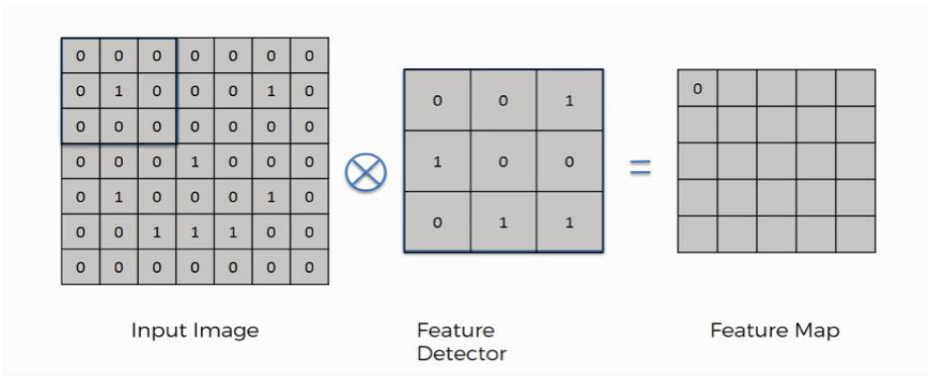


Figure 2.2.1.1 *Feature Mapping*

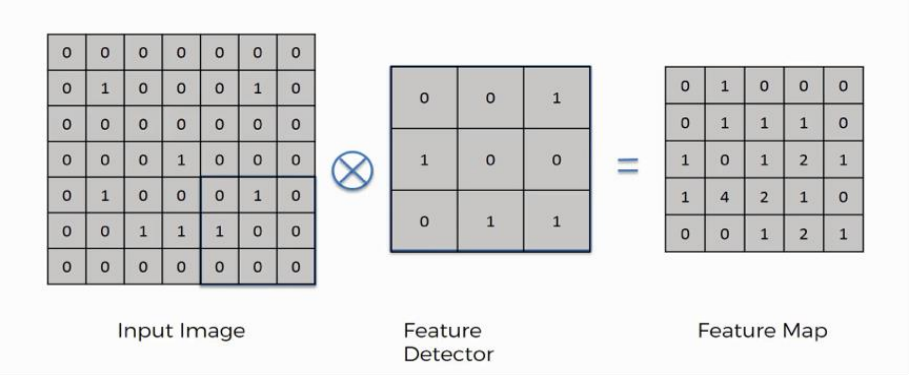


Figure 2.2.1.2 *Feature Mapping*

There are several gainings from deriving a feature map. Reducing the size of the input image is most important of them.

2.2.2. ReLU Layer

The Rectified Linear Unit, or ReLU, is a component of the CNN's process.

The purpose of applying the rectifier function is to increase the non-linearity in the project's images. The reason is that the images have lots of non-linear elements in transition between adjacent pixels.

When looking at any image, it will be seen that it contains a lot of non-linear features(e.g. the transition between pixels, the borders, the colors, etc.).

The rectifier serves to break up the linearity even further in order to make up for the linearity that might impose an image when putting it through the convolution operation.

In the project, 4 reLu layers are used and the number of filters is increasing in each layer to remove all the black elements from the picture and keeping only those carrying a positive value (the grey and white colors).

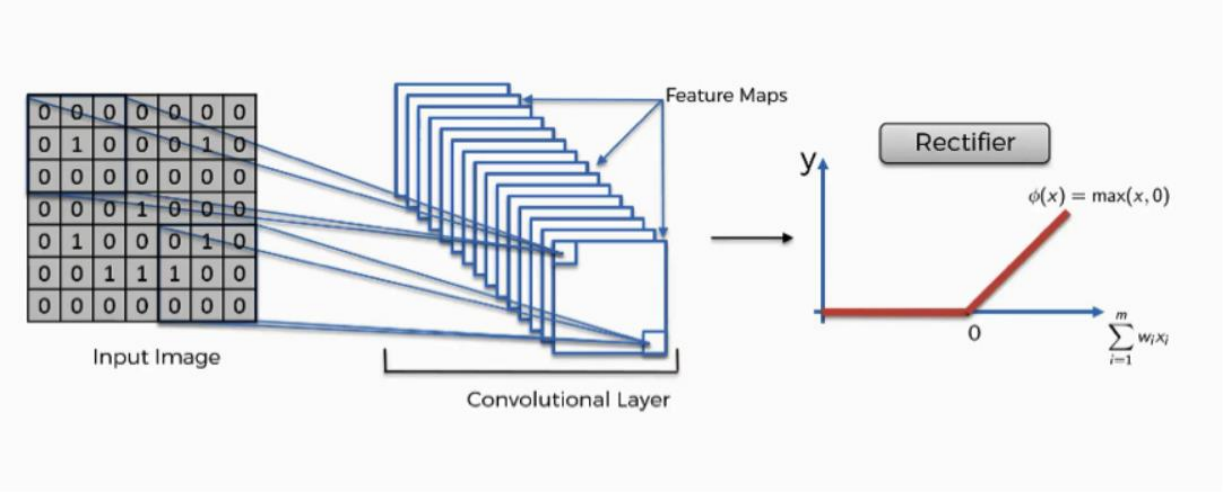


Figure 2.2.2.1 *ReLu Layer*

2.2.3. Pooling

Pooling is the process of merging. So it's basically for the purpose of reducing the size of the data.

If only the maximum value inside the frame is taken by sliding a window, this is 'max pooling'. The reason to extract the maximum value, which is actually the point from the whole pooling step, is to account for distortions.

The purpose of max pooling is enabling the CNN to detect image when the image is presented in any manner. For example, in the project, there are many types of photos for the "Irmak".

In the project, 4 max pooling layers are used. Pool size is the size of frame and the frame that will get to the maximum pixel of the 4 pixels inside. Therefore, pool size is 2. Strides is how far the pooling window moves for each pooling step. It moves 2 pixels for each pooling step.. Therefore, strides are 2. Then, getting a 3x3 pooled featured map.

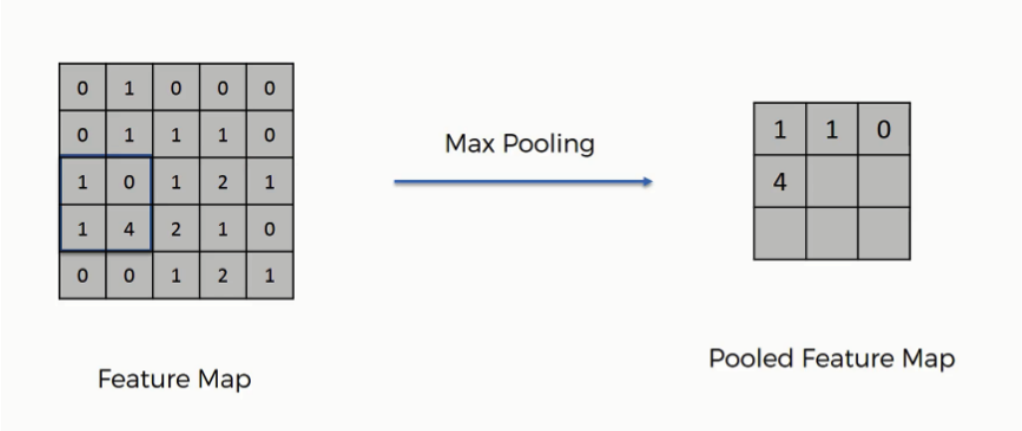


Figure 2.2.3.1 *Max Pooling*

2.2.4. Flattening

After finishing the previous two steps, it is necessary to have a pooled feature map by now. As the name of this step implies, the pooled feature map is literally flattened into a column like in the image below.

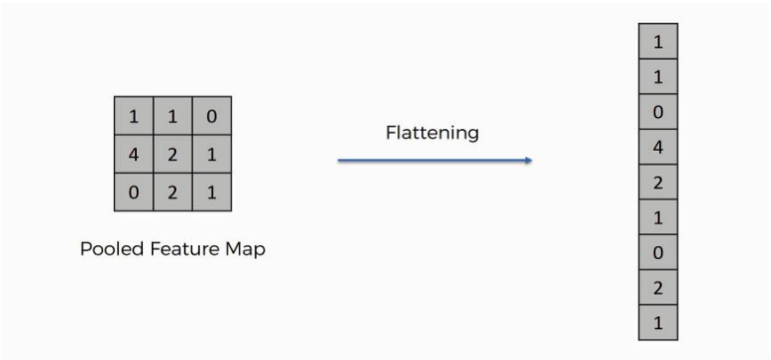


Figure 2.2.4.1 *Flattening*

This is because this data will need to be added to an ANN later.

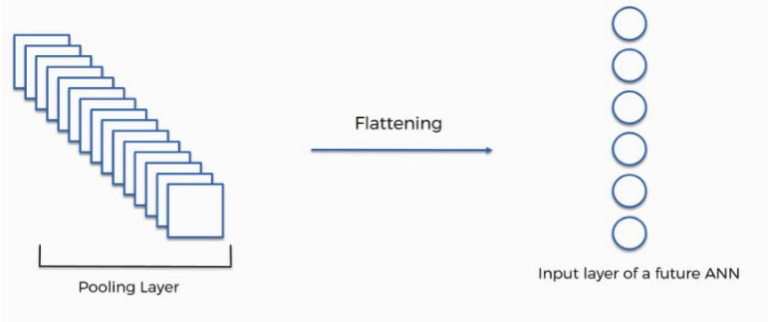


Figure 2.2.4.2 *Flattening*

2.2.5. Full Connection

In this step, adding whole artificial neural network(ANN) to convolutional neural network(CNN).

The aim of this step, the role of the artificial neural network is to take the data and combine the features into a wider variety of attributes that make the convolutional network more capable of classifying images, which is the whole purpose of creating a convolutional neural network.

Here is the three layers in the full connection step:

- Input layer
- Fully-connected layer
- Output layer

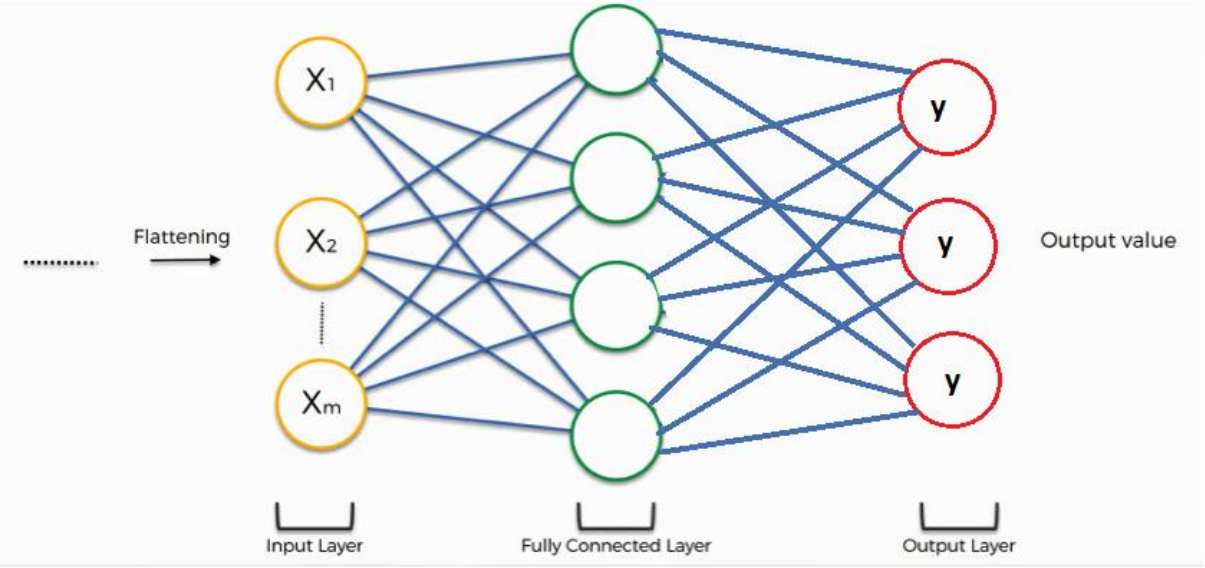


Figure 2.2.5.1 *Full Connection*

The input layer contains the vector of data that was created in the flattening step. The features that are distilled throughout the previous steps are encoded in this vector.

Fully-connected layer is the most specific type of hidden layer. In this layer, data is already sufficient for a fair degree of accuracy in recognized classes. Now, taking it to the next level in terms of complexity and precision.

Output layer is where getting the predicted classes.

In the project, information is processed from the moment it is inserted into the artificial neural network and until it develops its classes (Bengü, Irmak, Yusuf).

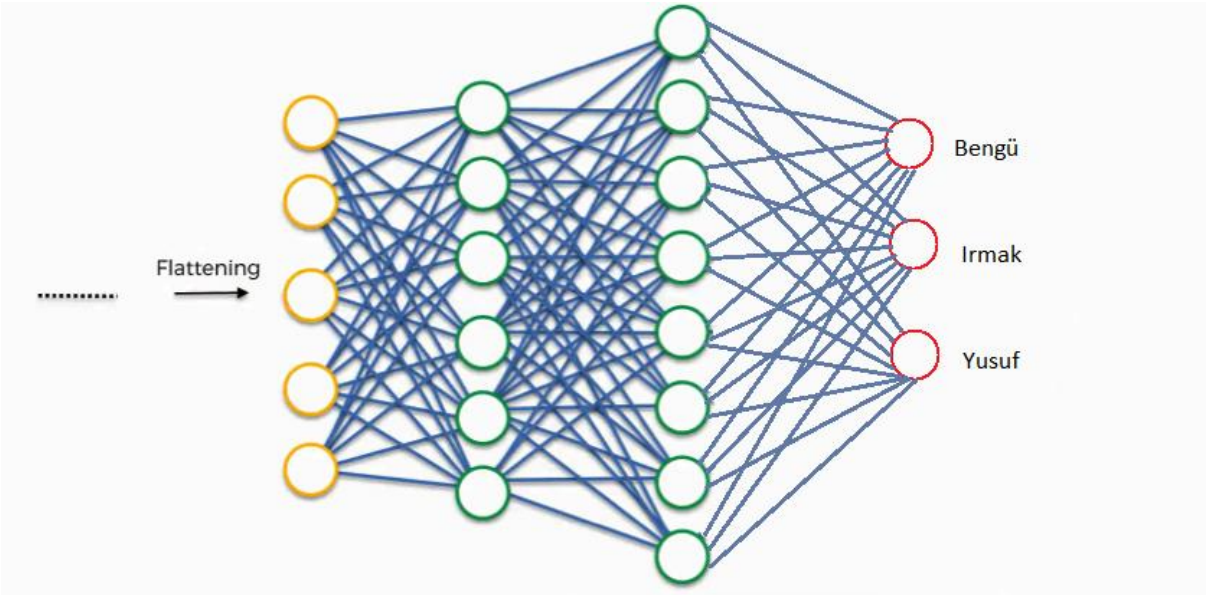


Figure 2.2.5.2 *Full connection process for the project*

In the project, for the full connection layer,units equals 128 because the larger number of hidden neurons are chosen. Also, rectifier activation function is used.

Then, adding output layer and units equals 3 because the number of neurons in the final output layer is 3 (Bengü, Irmak, Yusuf) and softmax activation function is used because multi-class classification is done.

2.2.6. Training the CNN

CNN is compiled and trained. Then obtaining accuracy for each data.

In this project, CNN is training until the number of epochs which is 83 and getting accuracy for each step. Also, loss is 'categorical_crossentropy' because of the multi-class classification.

3. APPLICATION PART

This Project can be demonstrated by a real-life PC Artificial Intelligent application. This section provides some information on how to use this application.

3.1. Setting of The Data

In this application, AI learns differences between images with Image Recognition. So, AI needs a dataset which contains a lot of different photos of each person. This part supplies this need and creates dataset using different photos of each person.

- **Step 1**

Create a dataset folder in the same direction with ".iynb" file.

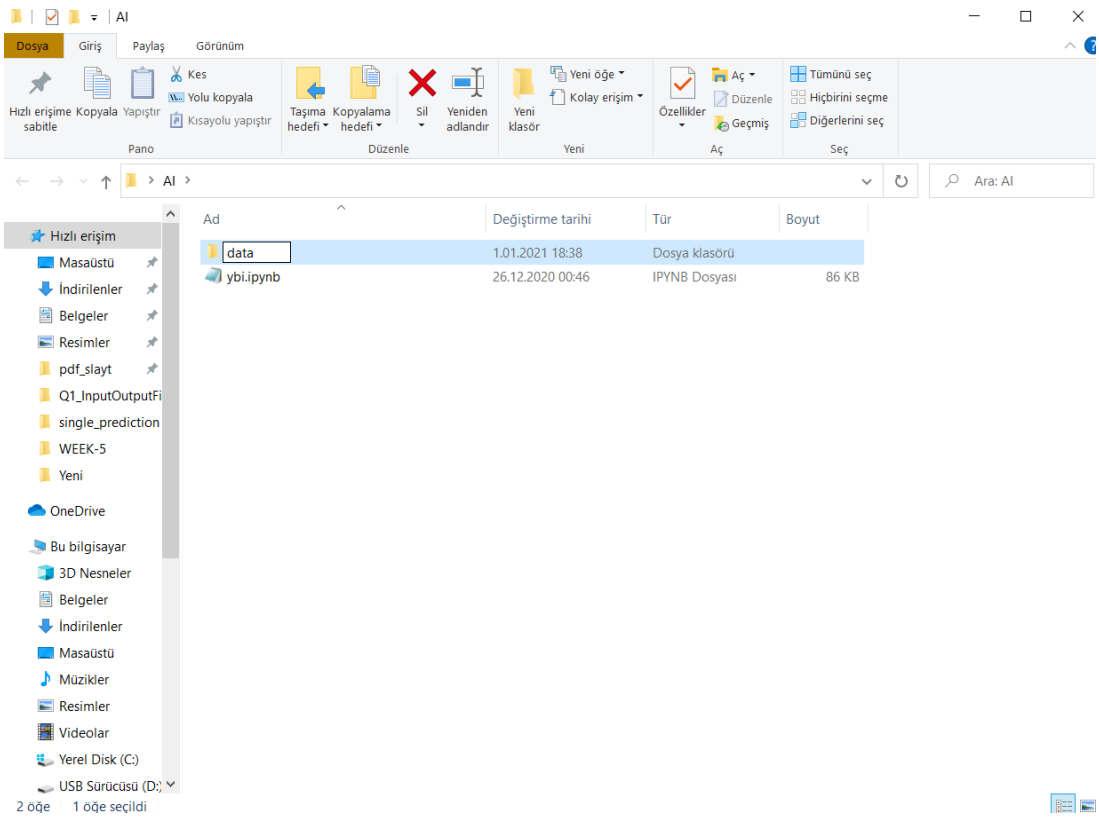


Figure 3.1.1 *Creation of a dataset folder*

- **Step 2**

Create folders for training, testing and single predicting. AI uses Training folder - “train”- for learn each person photo’s details. AI uses Testing folder – “test”- for testing information that learnt. User can put photos in - “single_prediction”- for using them in application.

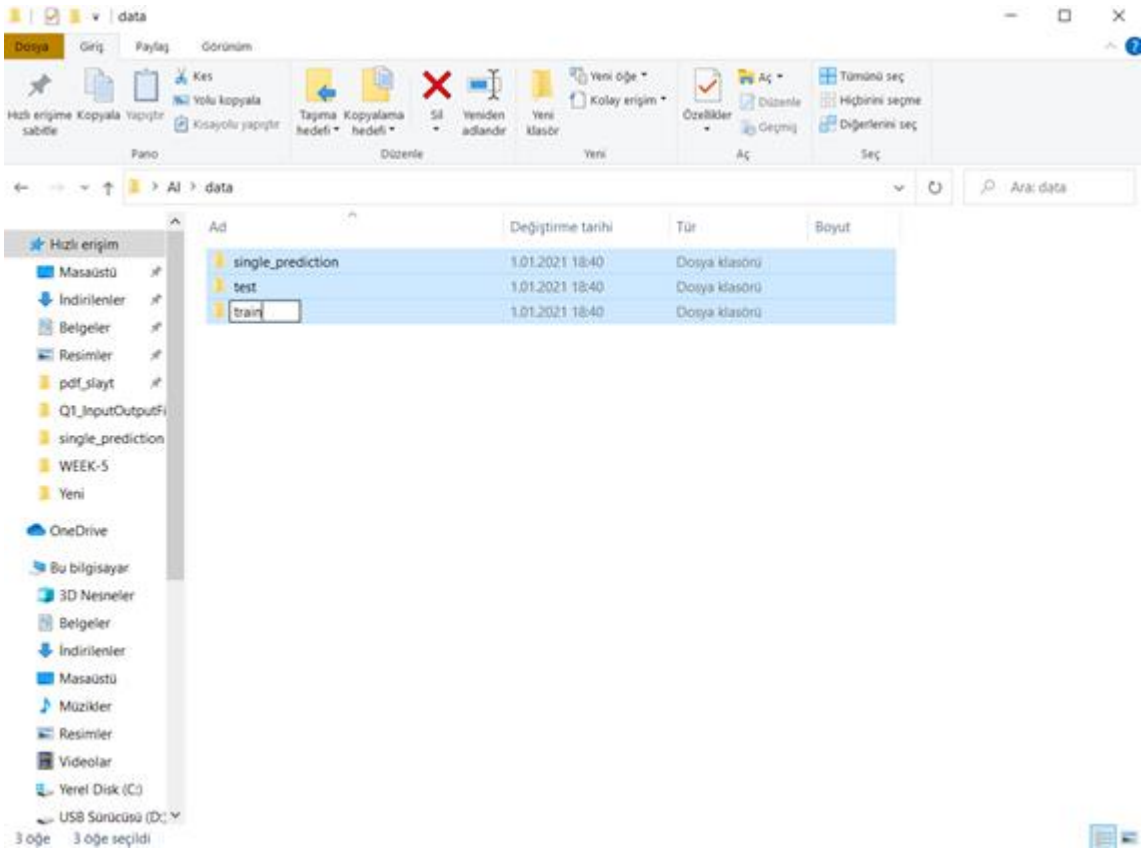


Figure 3.1.2 *Creation folders for training, testing and single predicting*

- **Step 3**

Create folders for each person in “train” and “test” folder. Fill these folders with photos that belong them. But there is an important rule that each person in “train” folder has 4 times more photo than “test” folder.

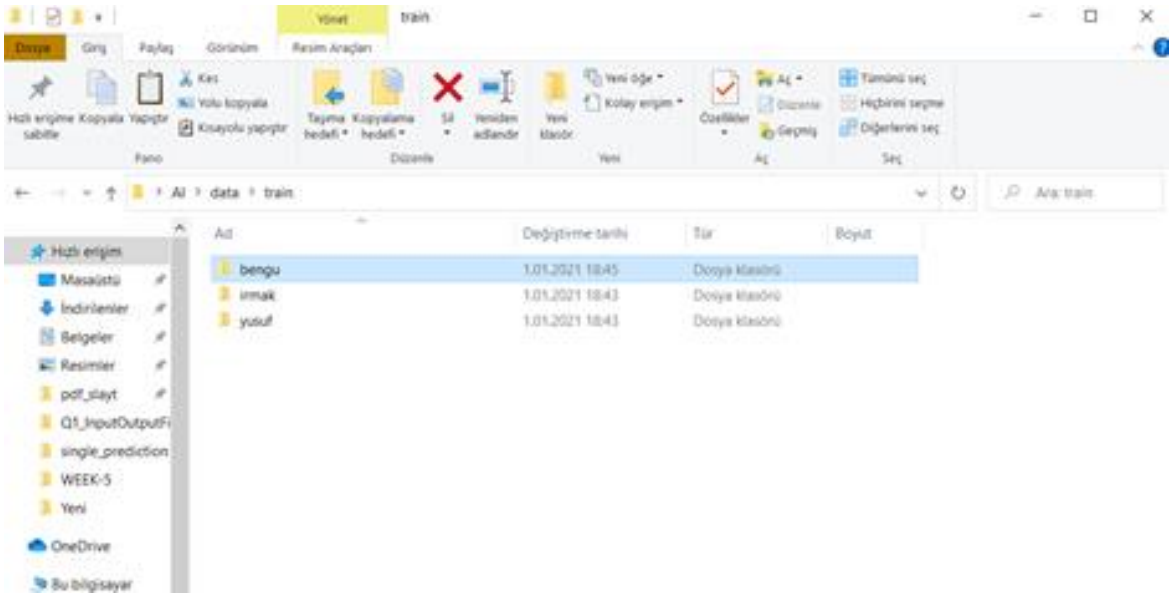


Figure 3.1.3 *Creation of train folder*

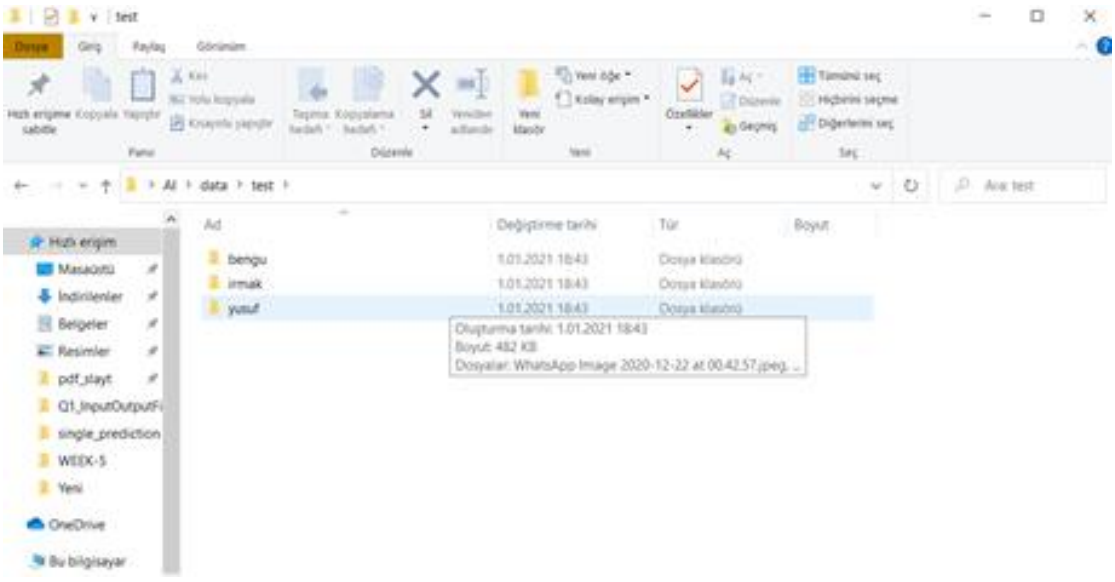


Figure 3.1.4 *Creation of test folder*

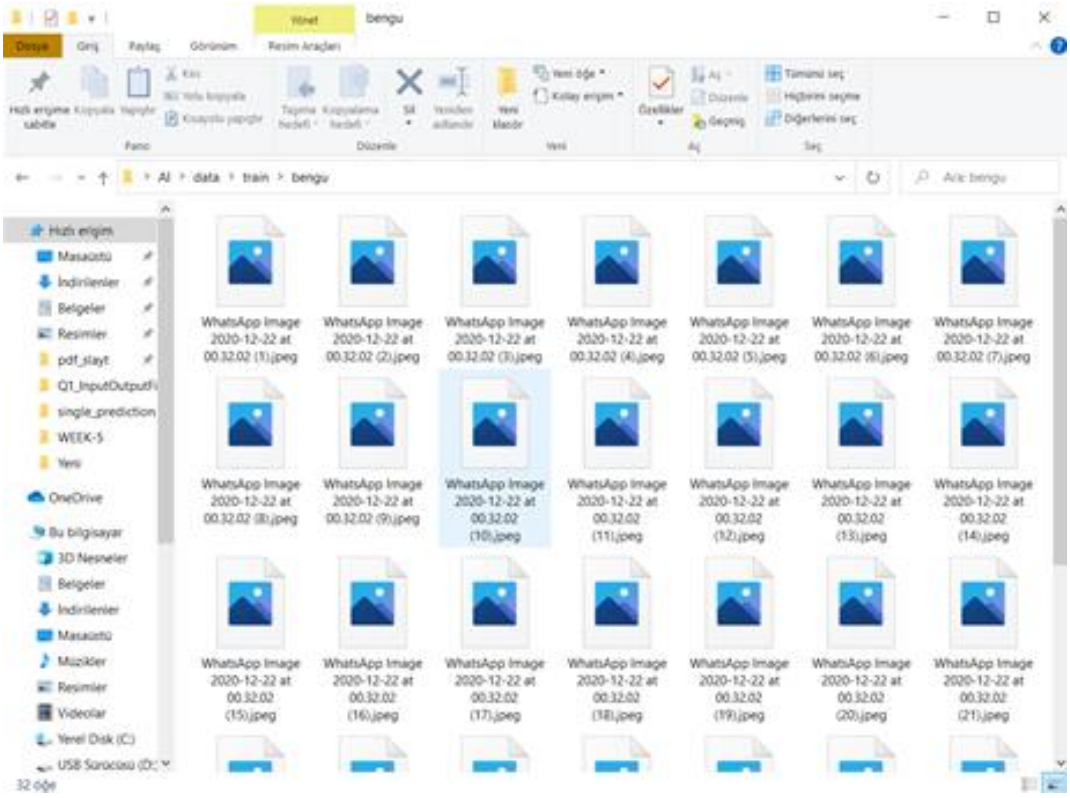


Figure 3.1.5 *Images in train folder for one example class*

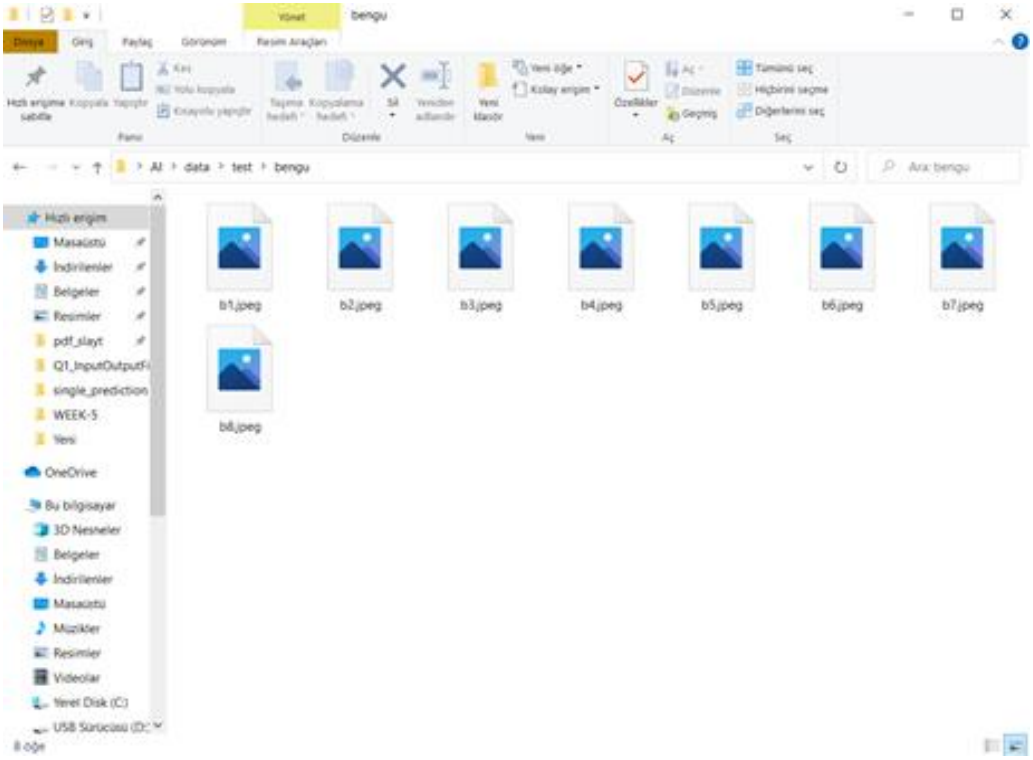


Figure 3.1.6 *Images in test folder for one example class*

- **Step 4**

The user puts the photos that he or she wants to use it in application into the “single_prediction” folder.

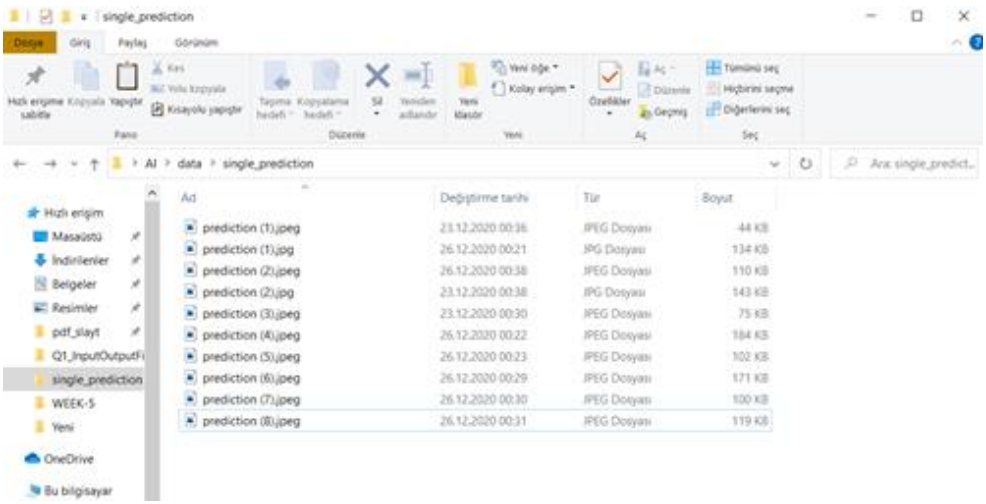


Figure 3.1.7 *Putting photos into the prediction folder*

3.2. Running AI

This project can only run on PC which has Tensorflow, Keras and Collab environment. So user should use some utilities. Anaconda is one of these utilities. This project should used with Anaconda.

- **Step 1**

After downloading Anaconda, open Anaconda Prompt Command:

>pip install keras

>pip install tensorflow

For using Tensorflow and Keras.

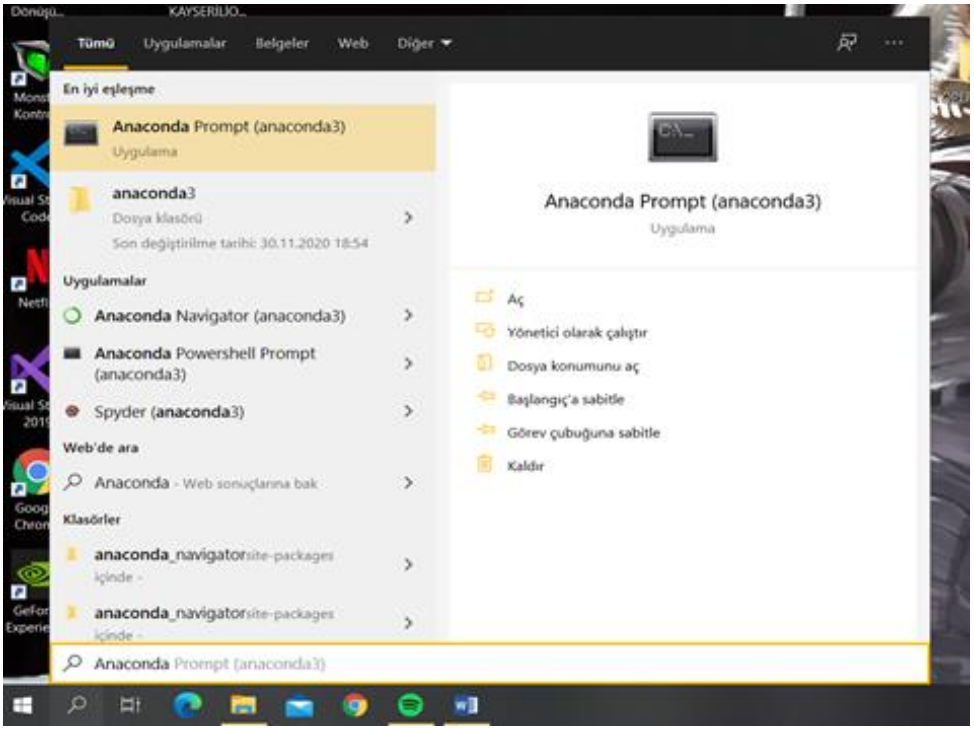


Figure 3.2.1 *Opening of Anaconda Prompt Command*

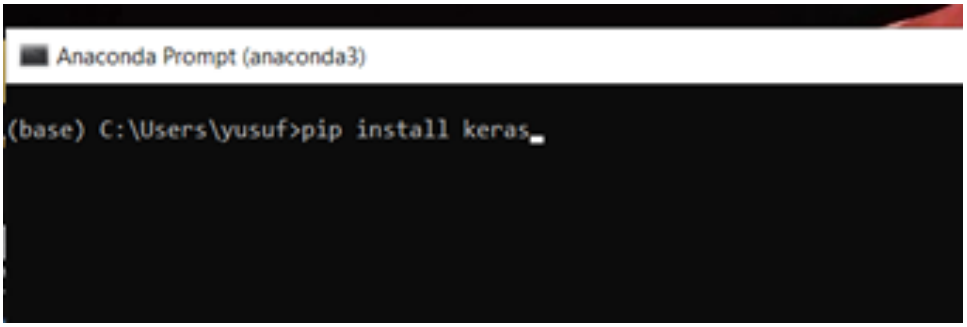


Figure 3.2.2 *pip install keras*

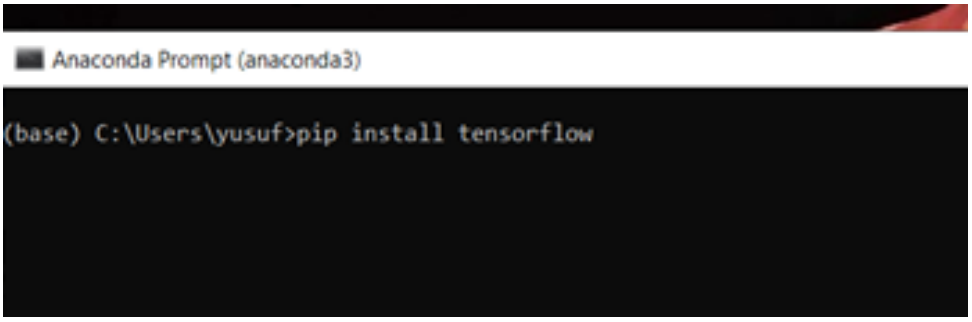


Figure 3.2.3 *pip install tensorflow*

- **Step 2**

User uses Anaconda Navigator for open .ipynb file.

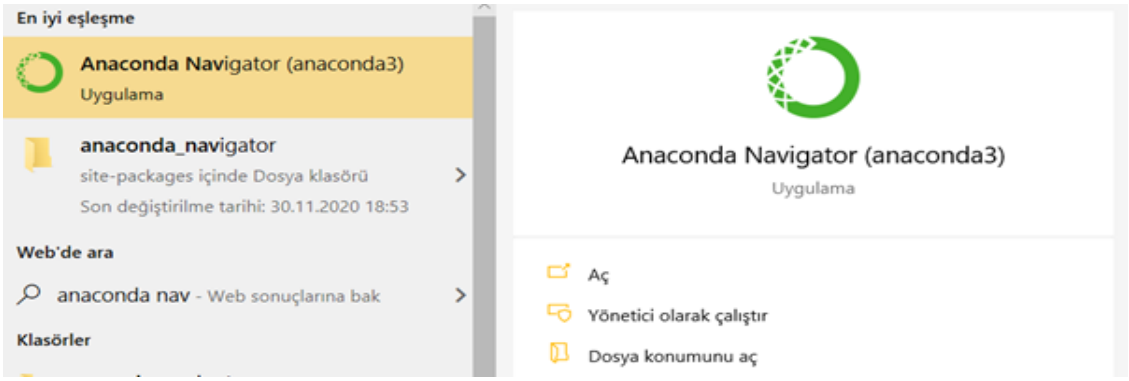


Figure 3.2.4 *Finding of Anaconda Navigator*

When Anaconda Navigator opened, Launch Jupyter Notebook.

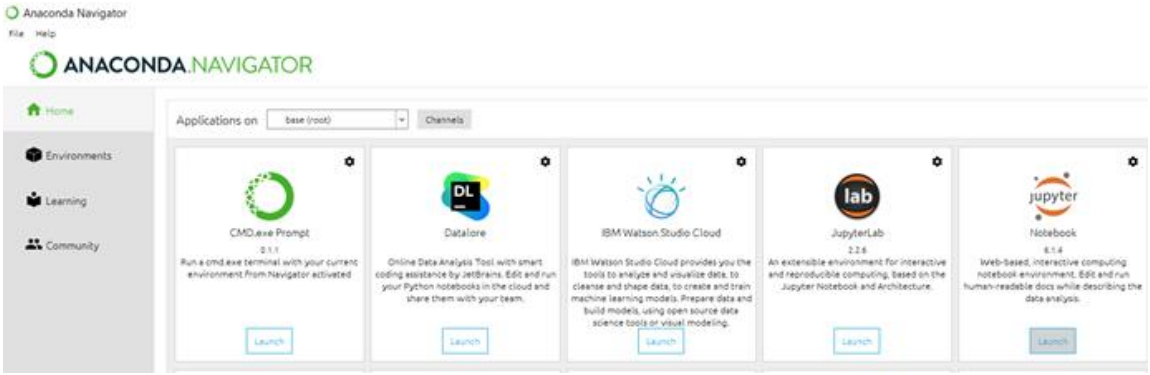


Figure 3.2.5 *Starting of Jupyter Notebook*

- **Step 3**

On Jupyter Notebook, open .ipynb file path.

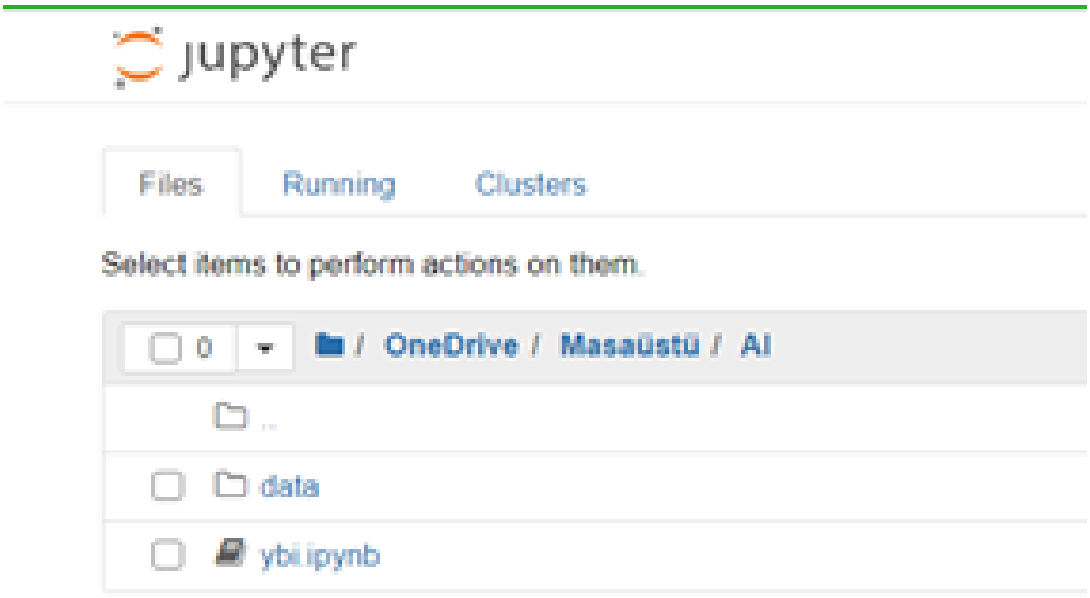


Figure 3.2.6 *Opening of .ipynb file path*

Open .ipynb file and Run every Cell.

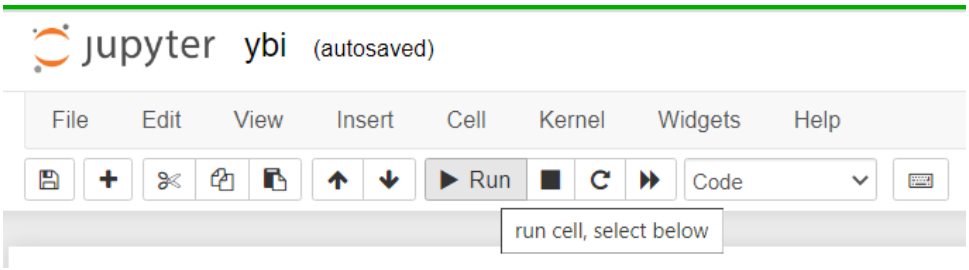


Figure 3.2.7 *Running every cell*

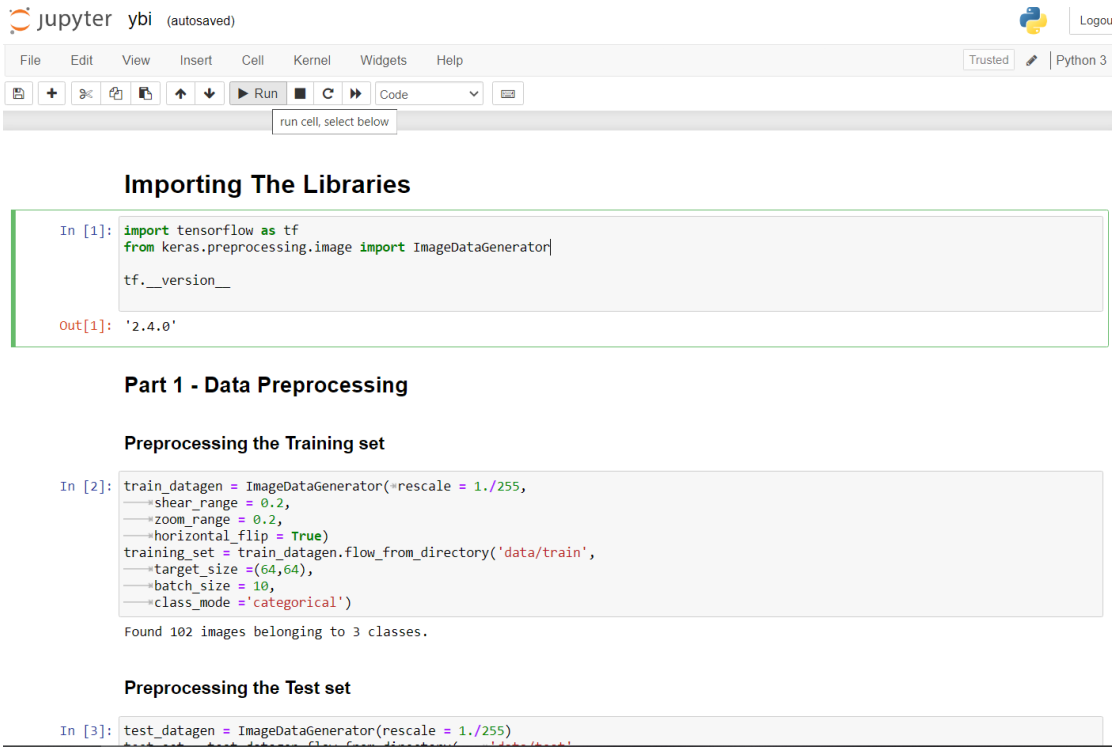


Figure 3.2.8 *Running every cell*

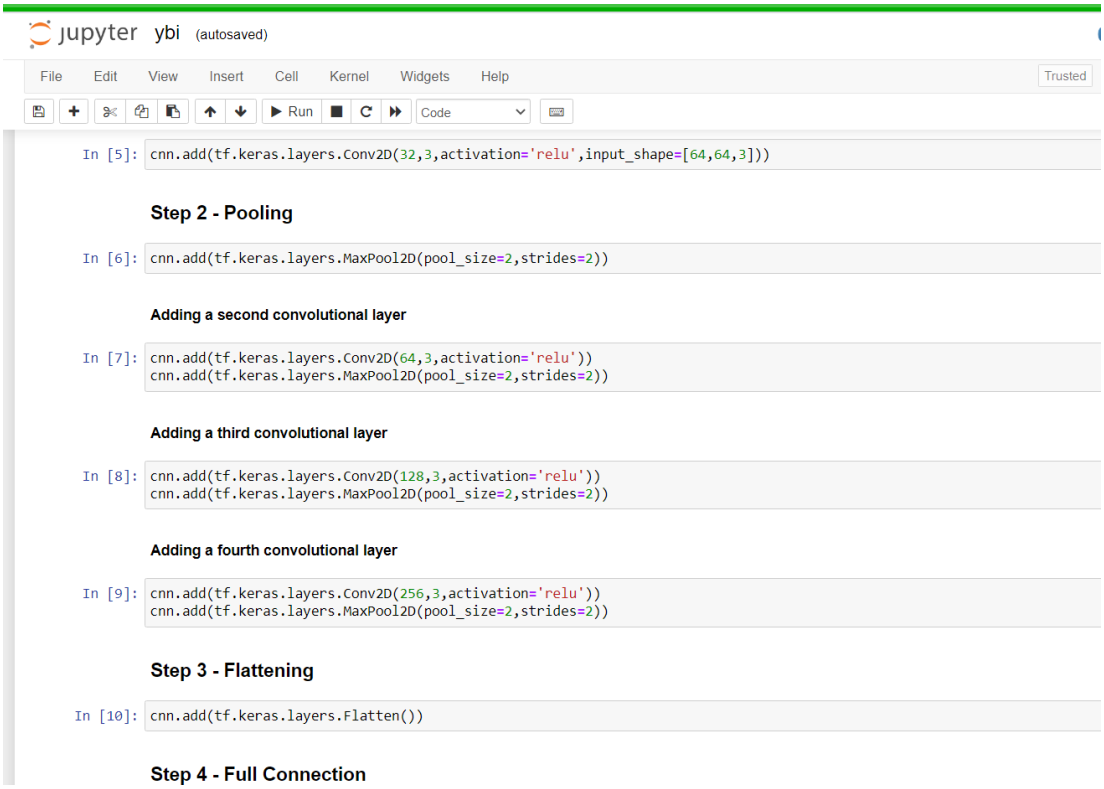


Figure 3.2.9 *Running every cell*

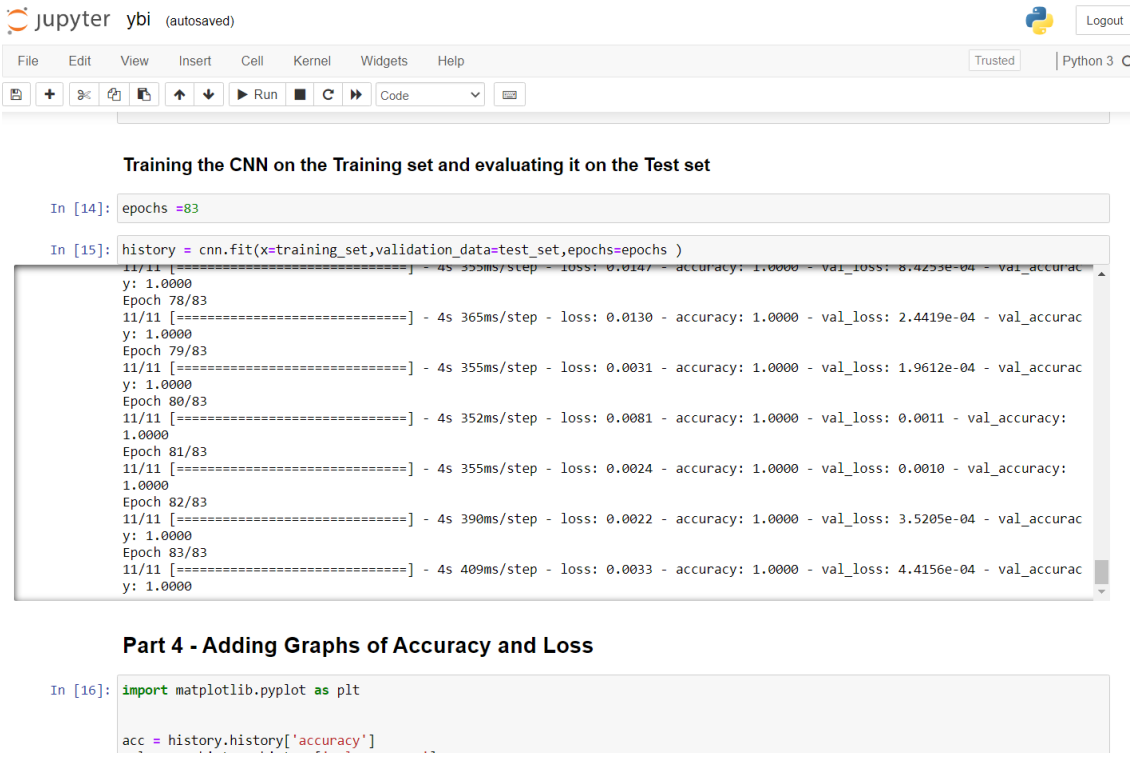


Figure 3.2.10 *Running every cell*

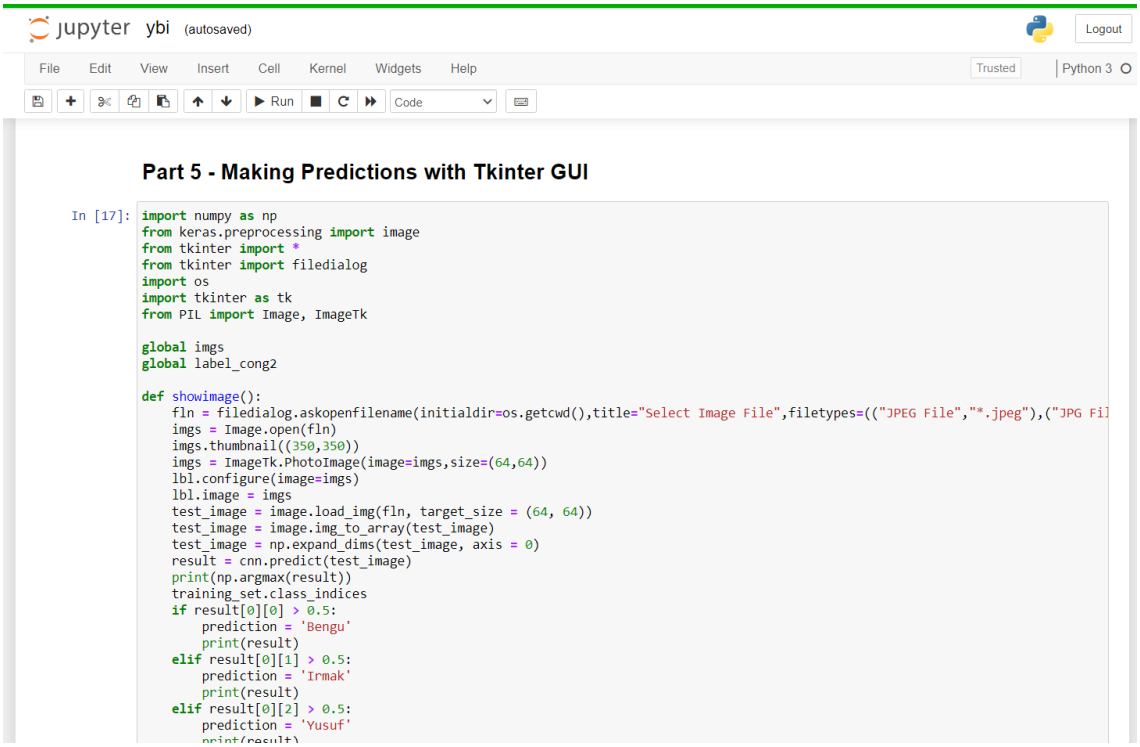


Figure 3.2.11 *Running every cell*

3.3. Application Interface

When Running AI part finished, Project's interface opens with 'WELCOME' label. In this area, there are two option, first button for browsing image to use it in program or second button for exit the program.

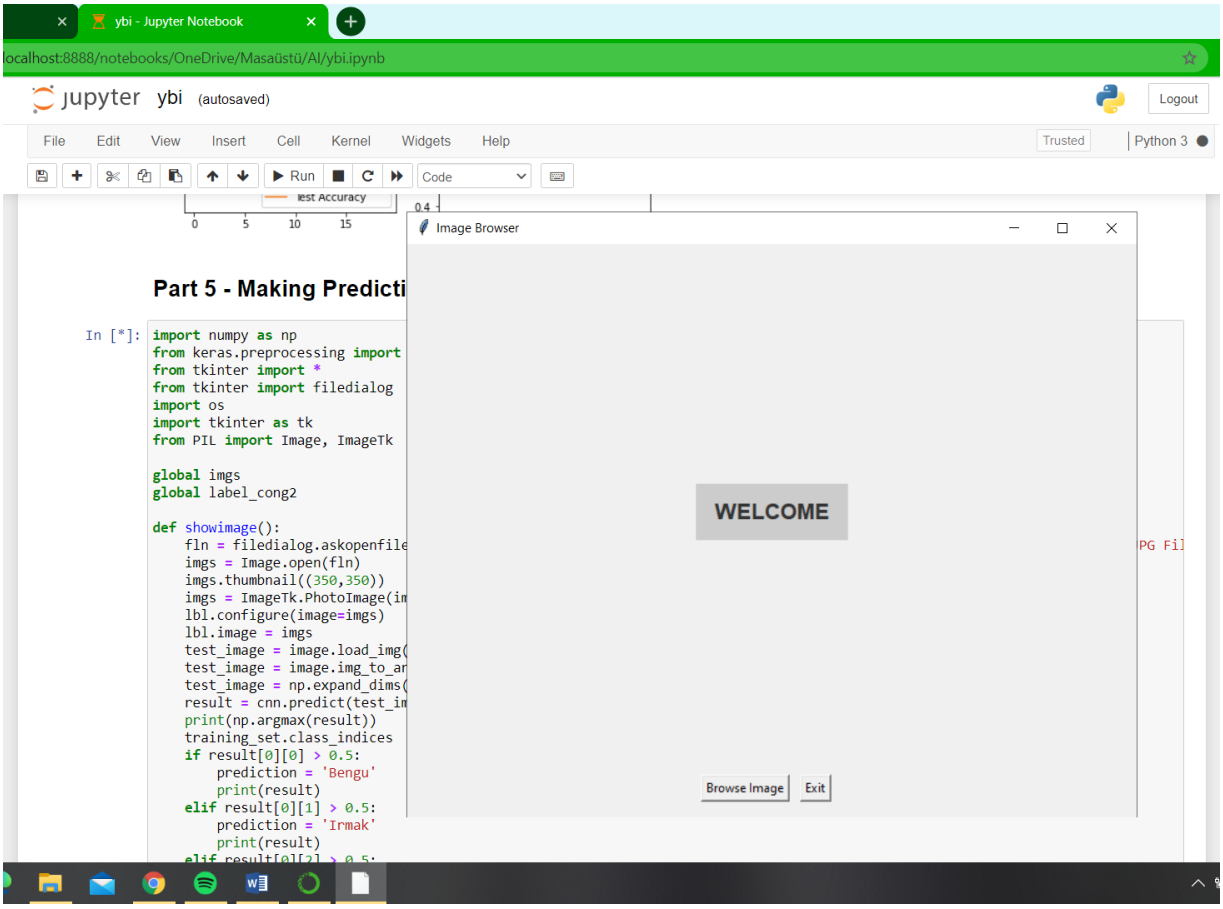


Figure 3.3.1 Application

3.4. Choosing Photo

When user clicks on Browse Image button, a window opens and user chooses photo from the single_prediction file that he or she want to use it in program.

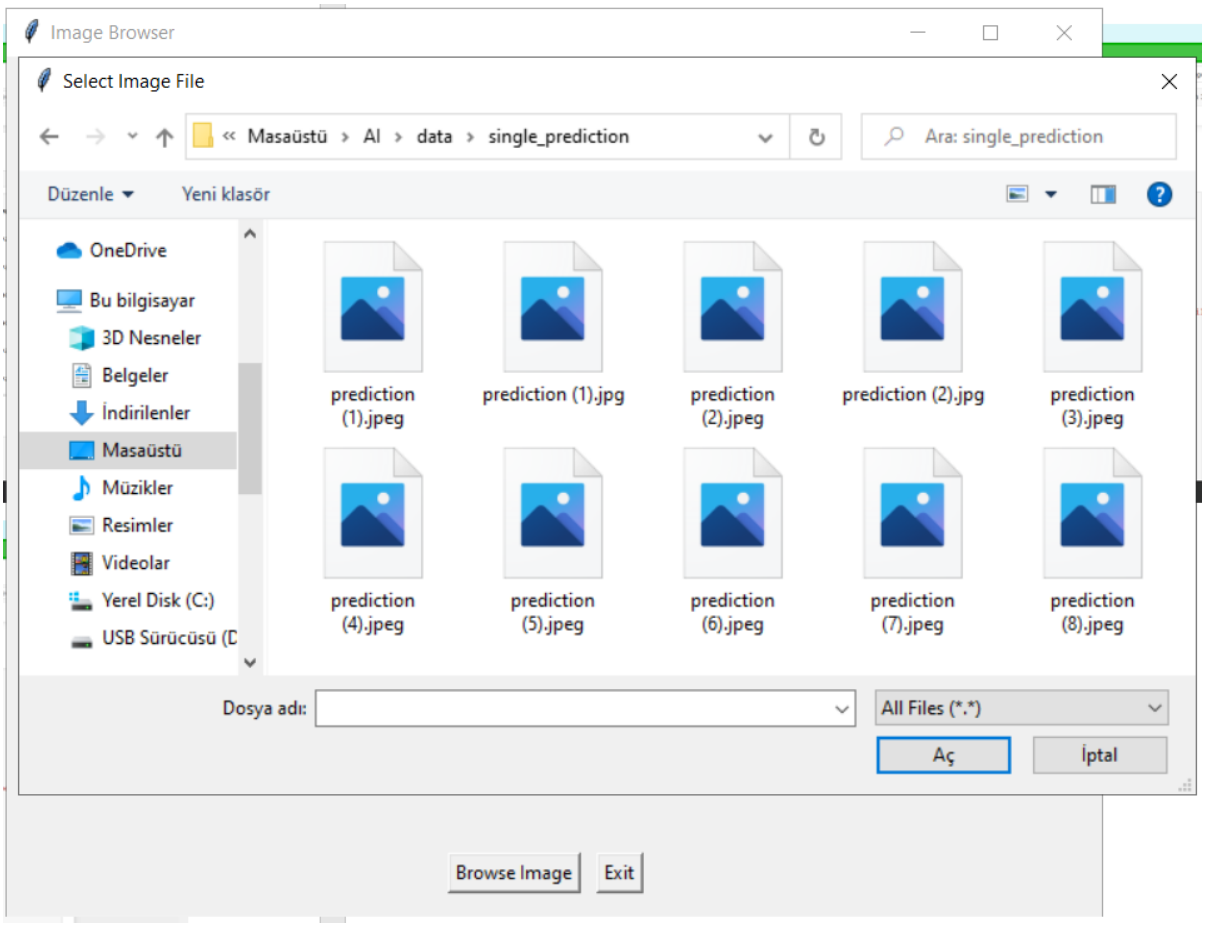
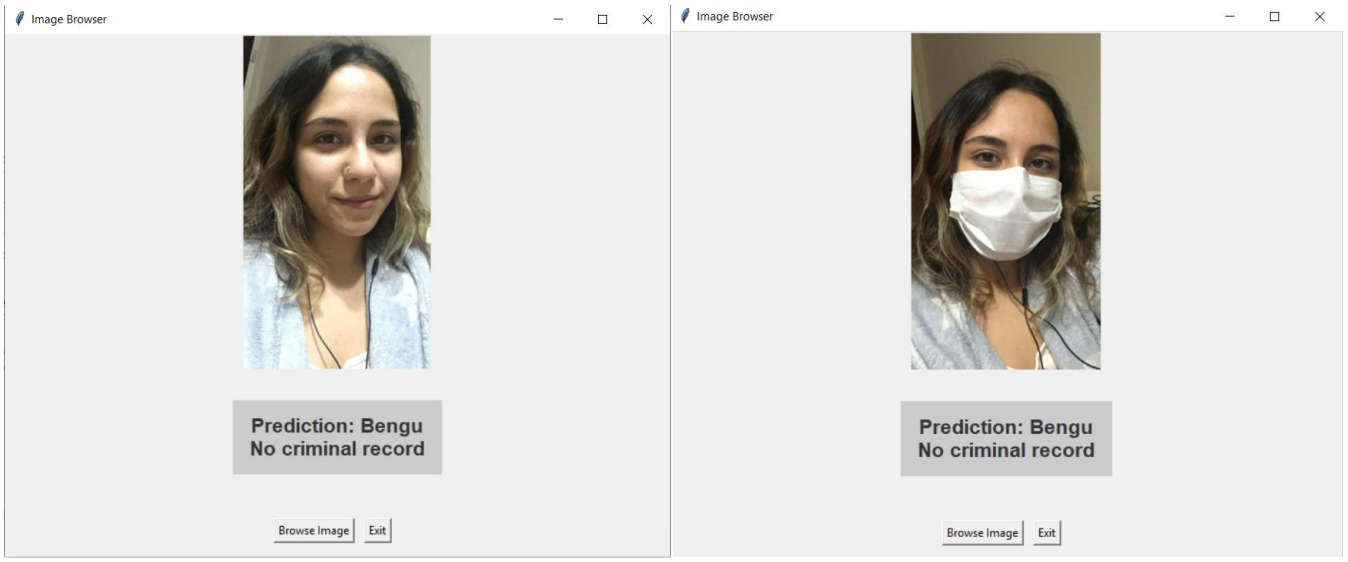


Figure 3.4.1 *Choosing a photo from the single_prediction folder*

3.5. Result

After choosing photo, AI (Image Recognition) runs and tests image. Finally, program prints Prediction.



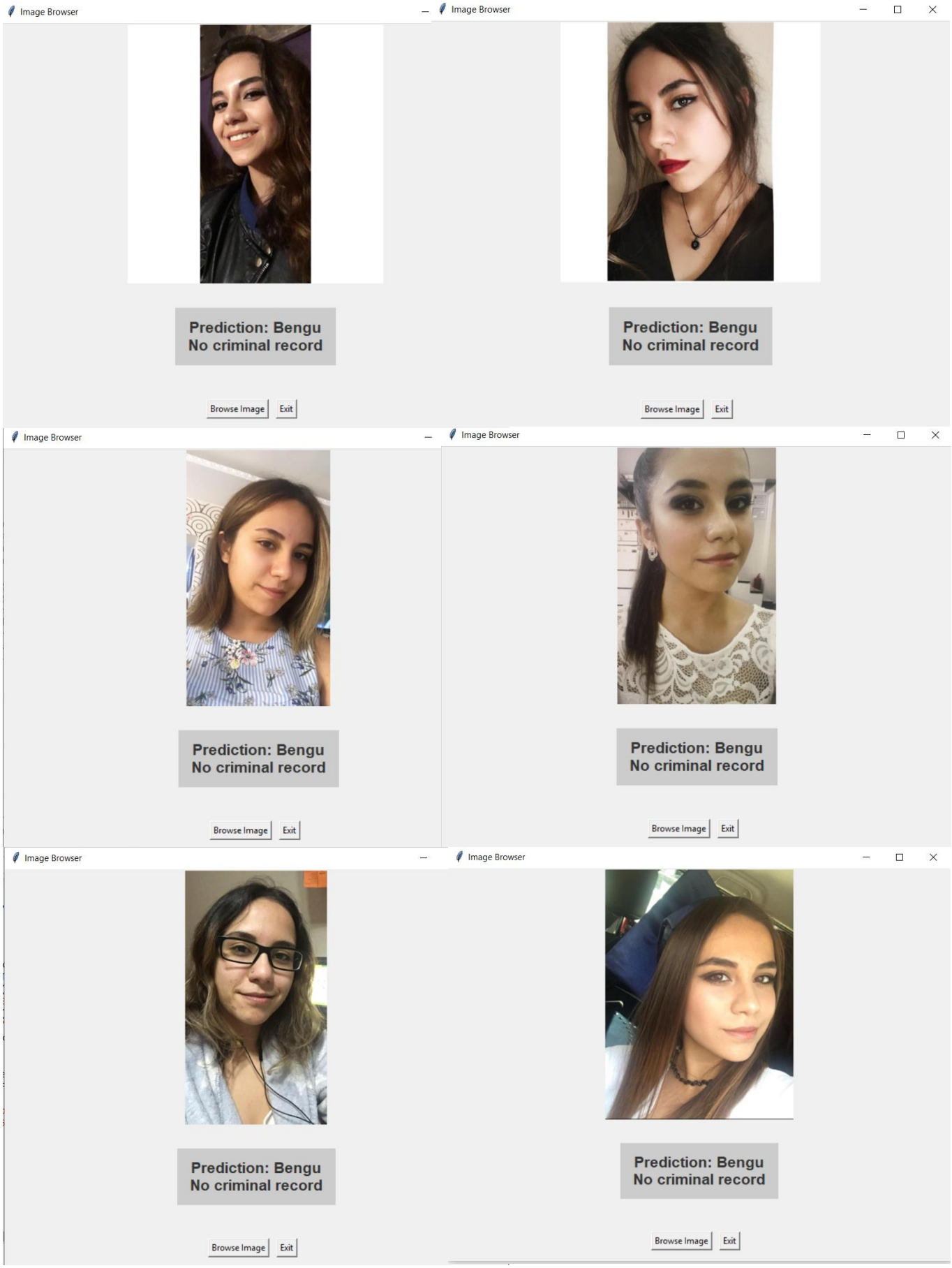


Figure 3.5.1 *Prediction results of the Bengü's pictures*

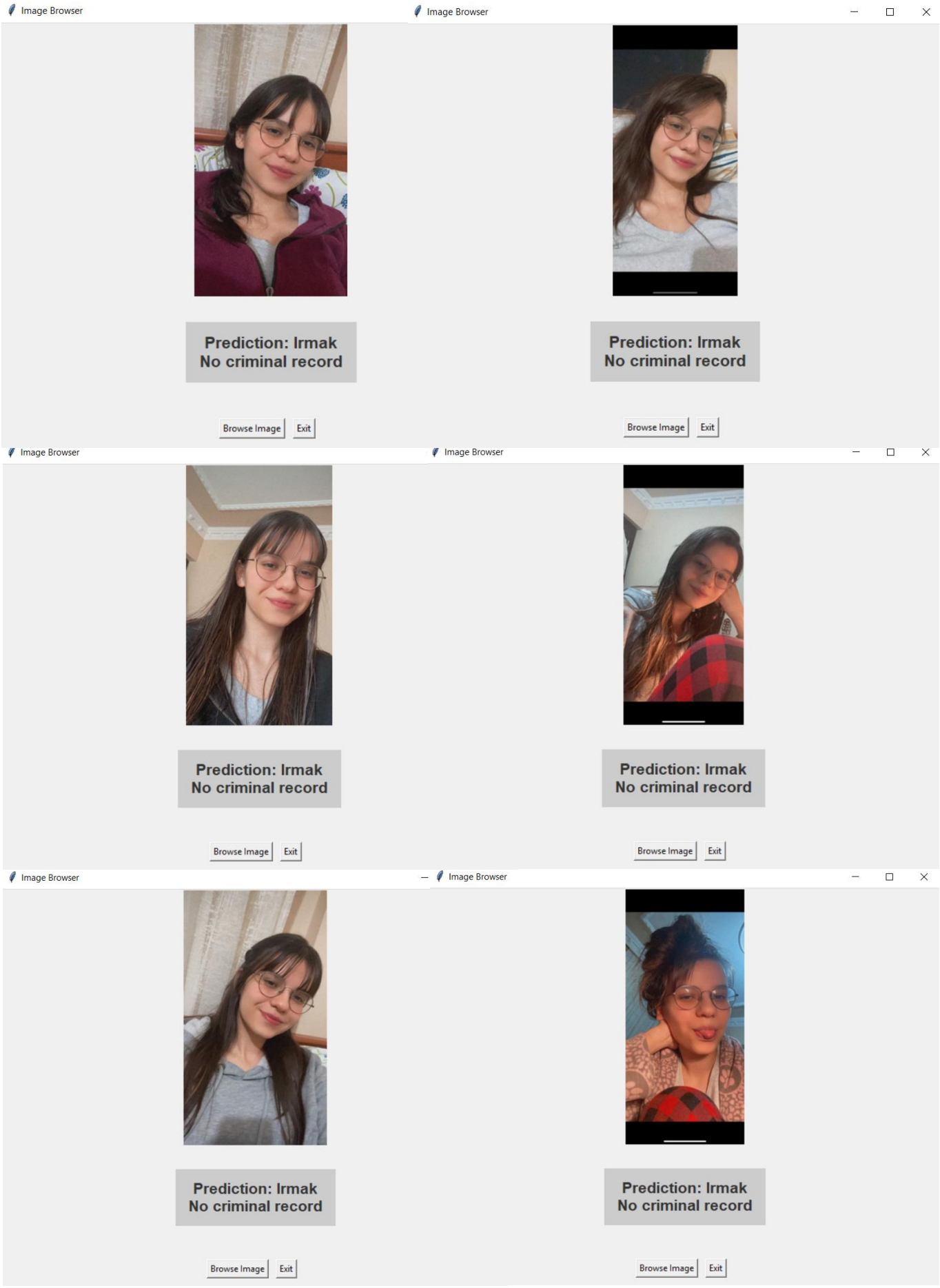
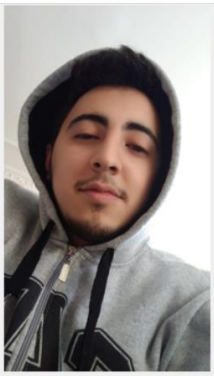


Figure 3.5.2 *Prediction results of the Irmak's pictures*

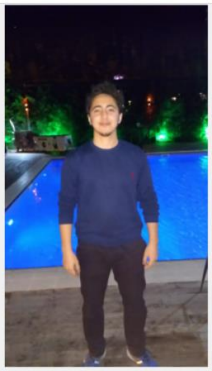
Image Browser



Prediction: Yusuf
No criminal record

Browse Image Exit

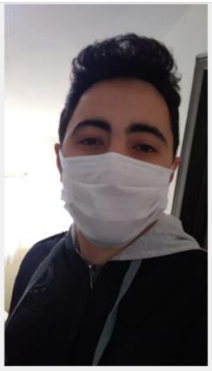
Image Browser



Prediction: Yusuf
No criminal record

Browse Image Exit

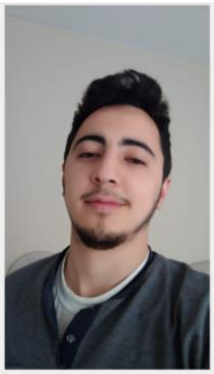
Image Browser



Prediction: Yusuf
No criminal record

Browse Image Exit

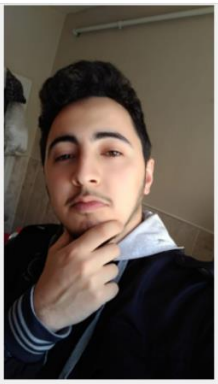
Image Browser



Prediction: Yusuf
No criminal record

Browse Image Exit

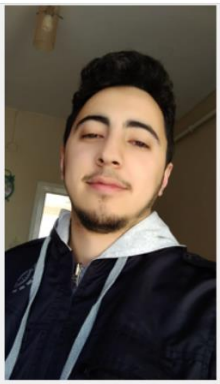
Image Browser



Prediction: Yusuf
No criminal record

Browse Image Exit

Image Browser



Prediction: Yusuf
No criminal record

Browse Image Exit

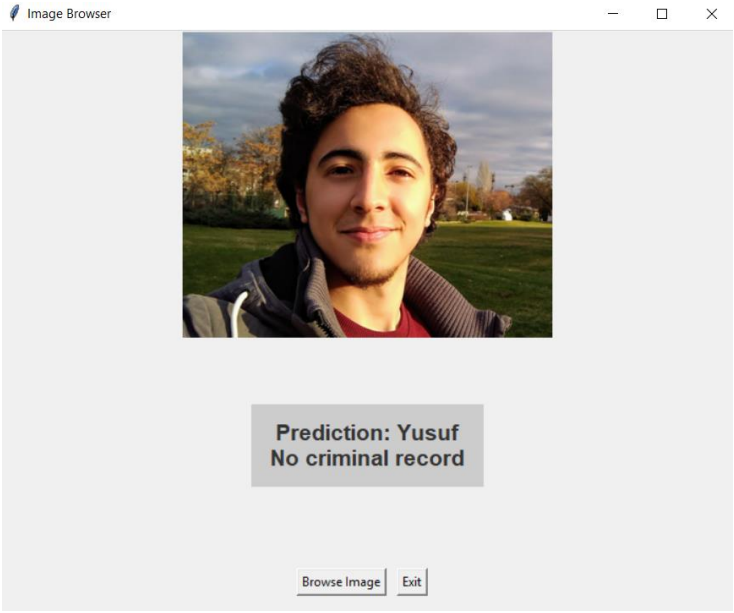


Figure 3.5.3 *Prediction results of the Yusuf's pictures*

In the epoch evaluation part of the code, Training & Test Accuracy and Training & Test Loss graphics are printed for 83 epochs.

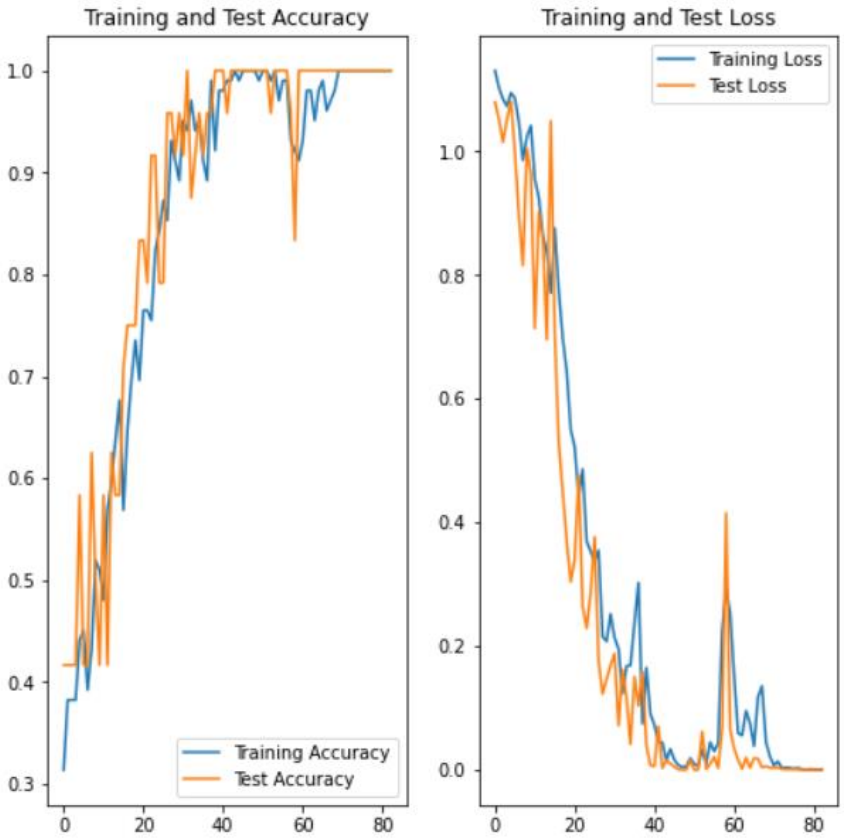


Figure 3.5.4 *Training and Test Accuracy / Loss Graphs*

4. DISCUSSION AND CONCLUSION

This report was written to explain the project given by Artificial Intelligence lesson. The definition and aim of the project were described clearly. The theoretical and technical informations of the project developed to classify the images by separating them corresponding to their classes were mentioned in the content of this report. Finally, information was given on use of the tkinter application and results were mentioned.

BIBLIOGRAPHY

1. https://visme.co/blog/report-writing-format/
2. https://www.superdatascience.com/blogs/the-ultimate-guide-to-convolutional-neural-networks-cnn
3. https://towardsdatascience.com/the-most-intuitive-and-easiest-guide-for-convolutional-neural-network-3607be47480
4. https://keras.io/api/
5. https://jupyter-notebook-beginner-guide.readthedocs.io/en/latest/what_is_jupyter.html