

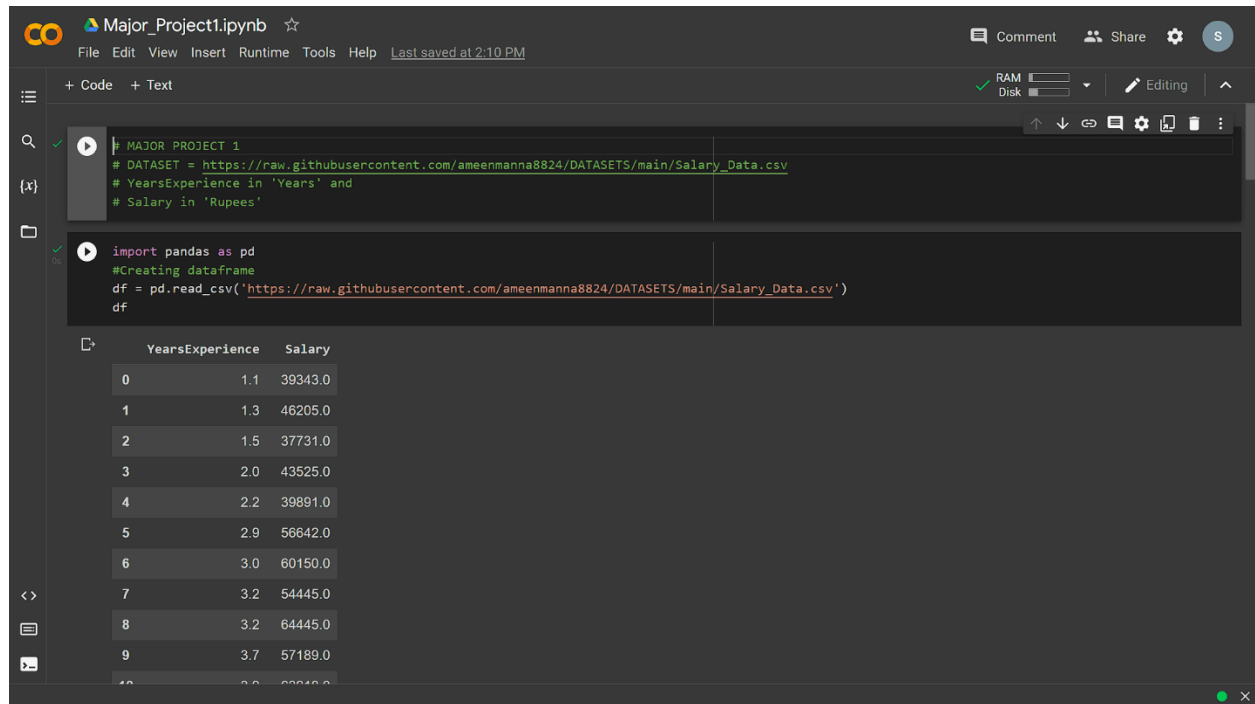
Name :- Kaivalya Hemant Vanmali

**College :- Veermata Jijabai Technological Institute ,
Matunga**

Year :- 3rd Year , B.tech , Civil Engineering

Course :- Machine Learning

Major Project 1 Screenshots :



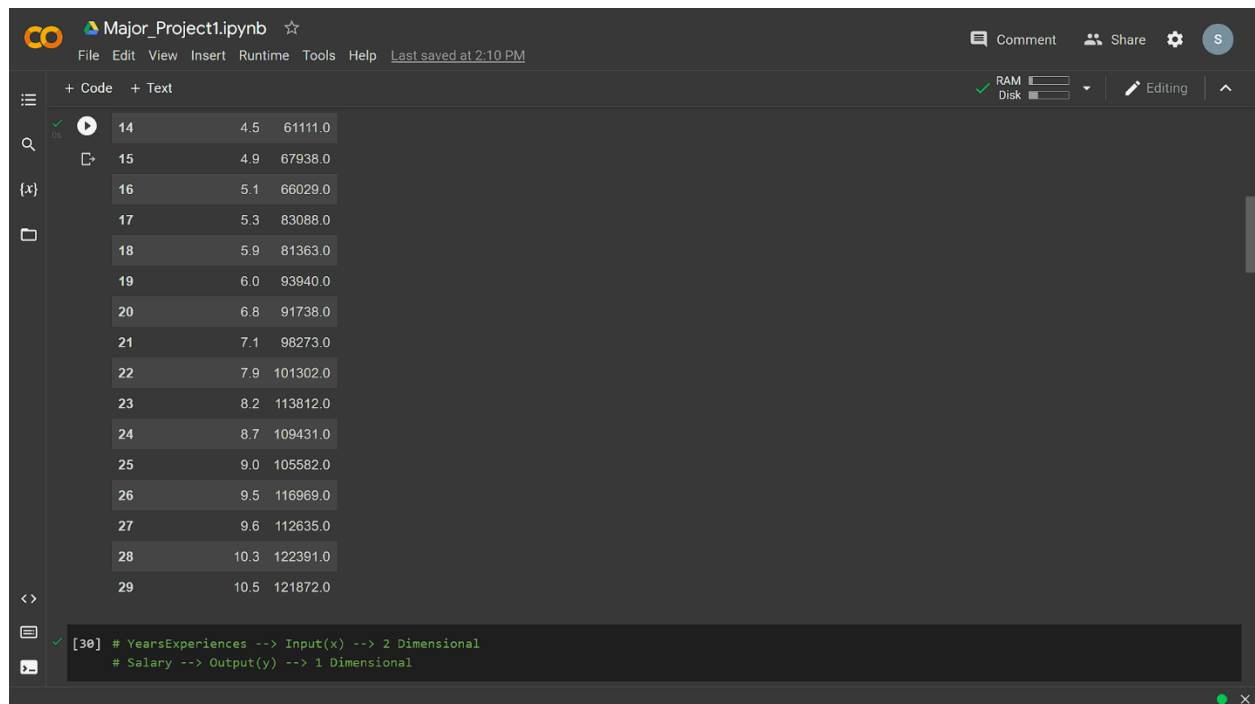
The screenshot shows a Jupyter Notebook titled 'Major_Project1.ipynb'. The code cell contains the following Python code:

```
MAJOR PROJECT 1
# DATASET = https://raw.githubusercontent.com/ameenmanna8824/DATASETS/main/Salary_Data.csv
# YearsExperience in 'Years' and
# Salary in 'Rupees'

import pandas as pd
#Creating dataframe
df = pd.read_csv('https://raw.githubusercontent.com/ameenmanna8824/DATASETS/main/Salary_Data.csv')
df
```

The output of the code is a DataFrame with 10 rows and 2 columns: 'YearsExperience' and 'Salary'.

| | YearsExperience | Salary |
|---|-----------------|---------|
| 0 | 1.1 | 39343.0 |
| 1 | 1.3 | 46205.0 |
| 2 | 1.5 | 37731.0 |
| 3 | 2.0 | 43525.0 |
| 4 | 2.2 | 39891.0 |
| 5 | 2.9 | 56642.0 |
| 6 | 3.0 | 60150.0 |
| 7 | 3.2 | 54445.0 |
| 8 | 3.2 | 64445.0 |
| 9 | 3.7 | 57189.0 |



The screenshot shows the same Jupyter Notebook. The code cell contains the following Python code:

```
14      4.5  61111.0
15      4.9  67938.0
16      5.1  66029.0
17      5.3  83088.0
18      5.9  81363.0
19      6.0  93940.0
20      6.8  91738.0
21      7.1  98273.0
22      7.9 101302.0
23      8.2 113812.0
24      8.7 109431.0
25      9.0 105582.0
26      9.5 116969.0
27      9.6 112635.0
28     10.3 122391.0
29     10.5 121872.0

[30] # YearsExperiences --> Input(x) --> 2 Dimensional
      # Salary --> Output(y) --> 1 Dimensional
```

The output of the code is a DataFrame with 16 rows and 2 columns: 'YearsExperience' and 'Salary'.

| | YearsExperience | Salary |
|----|-----------------|----------|
| 14 | 4.5 | 61111.0 |
| 15 | 4.9 | 67938.0 |
| 16 | 5.1 | 66029.0 |
| 17 | 5.3 | 83088.0 |
| 18 | 5.9 | 81363.0 |
| 19 | 6.0 | 93940.0 |
| 20 | 6.8 | 91738.0 |
| 21 | 7.1 | 98273.0 |
| 22 | 7.9 | 101302.0 |
| 23 | 8.2 | 113812.0 |
| 24 | 8.7 | 109431.0 |
| 25 | 9.0 | 105582.0 |
| 26 | 9.5 | 116969.0 |
| 27 | 9.6 | 112635.0 |
| 28 | 10.3 | 122391.0 |
| 29 | 10.5 | 121872.0 |

The final cell shows the dimensions of the input and output variables:

```
[30] # YearsExperiences --> Input(x) --> 2 Dimensional
      # Salary --> Output(y) --> 1 Dimensional
```

Major_Project1.ipynb

File Edit View Insert Runtime Tools Help Last saved at 2:10 PM

+ Code + Text

RAM Disk

Editing

[30] # YearsExperiences --> Input(x) --> 2 Dimensional
Salary --> Output(y) --> 1 Dimensional

'.values' convert dataframe into array

x = df.iloc[0:,1].values
column has ':', therefore it is 2 dim
x

array([[1.1],
[1.3],
[1.5],
[2.],
[2.2],
[2.9],
[3.],
[3.2],
[3.2],
[3.7],
[3.9],
[4.],
[4.],
[4.1],
[4.5],
[4.9],
[5.1],
[5.3],
[5.9],
[6.],
[6.8],
[7.1],
[7.9]])

Major_Project1.ipynb

File Edit View Insert Runtime Tools Help Last saved at 2:10 PM

+ Code + Text

RAM Disk

Editing

'.values' convert dataframe into array

x = df.iloc[0:,1].values
column has ':', therefore it is 2 dim
x

array([[1.1],
[1.3],
[1.5],
[2.],
[2.2],
[2.9],
[3.],
[3.2],
[3.2],
[3.7],
[3.9],
[4.],
[4.],
[4.1],
[4.5],
[4.9],
[5.1],
[5.3],
[5.9],
[6.],
[6.8],
[7.1],
[7.9]])

[34] y = df.iloc[0:,1].values
column not has ':', therefore it is 1 dim
y

array([39343., 46205., 37731., 43525., 39891., 56642., 60150.,
54445., 64445., 57189., 63218., 55794., 56957., 57081.,
61111., 67938., 66029., 83088., 81363., 93940., 91738.,
98273., 101302., 113812., 109431., 105582., 116969., 112635.,
122391., 121872.])

[36] # RUNNING A CLASSIFIER / REGRESSOR
sklearn.linear_model - package (collection of libraries)
LinearRegression - Library
from sklearn.linear_model import LinearRegression
model = LinearRegression()

[37] model.fit(x,y) # mapping the values of x and y in the LinearRegression Library

LinearRegression()

Major_Project1.ipynb

File Edit View Insert Runtime Tools Help Last saved at 2:10 PM

Comment Share Settings S

+ Code + Text

RAM Disk Editing

[37] model.fit(x,y) # mapping the values of x and y in the LinearRegression Library

LinearRegression()

[38] y_predict = model.predict(x)

using the input values to predict the output

y_predict

array([36187.15875227, 38077.15121656, 39967.14368085, 44692.12484158, 46582.11730587, 53197.09093089, 54142.08716303, 56032.07962732, 56032.07962732, 60757.06078805, 62647.05325234, 63592.04948449, 63592.04948449, 64537.04571663, 68317.03064522, 72097.0155738 , 73987.00803809, 75877.00050238, 81546.97789525, 82491.9741274 , 90051.94398456, 92886.932681 , 100446.90253816, 103281.8912346 , 108006.87239533, 110841.86109176, 115566.84225249, 116511.83848464, 123126.81210966, 125016.80457395])

[39] y

#these are the Actual output

array([39343., 46205., 37731., 43525., 39891., 56642., 60150., 54445., 64445., 57189., 63218., 55794., 56957., 57081., 61111., 67938., 66029., 83088., 81363., 93940., 91738., 98273., 101302., 113812., 109431., 105582., 116969., 112635., 122391., 121872.])

[40] # There is huge difference b/w 'y' and 'y_predict' , that means our model is not Linear or 'Less Linear'

[41] model.predict([[10.5]])

Major_Project1.ipynb

File Edit View Insert Runtime Tools Help Last saved at 2:10 PM

Comment Share Settings S

+ Code + Text

RAM Disk Editing

[40] # There is huge difference b/w 'y' and 'y_predict' , that means our model is not Linear or 'Less Linear'

[41] model.predict([[10.5]])

predicting the salary of person having 10.5 years of experience

array([125016.80457395])

[16] #which is accurate

[42] # For Verification

we take equation of the straight line

y = mx + c

From excel --> m = 9449.962 , c = 25792.2

[44] m = model.coef_ # slope

m

array([9449.96232146])

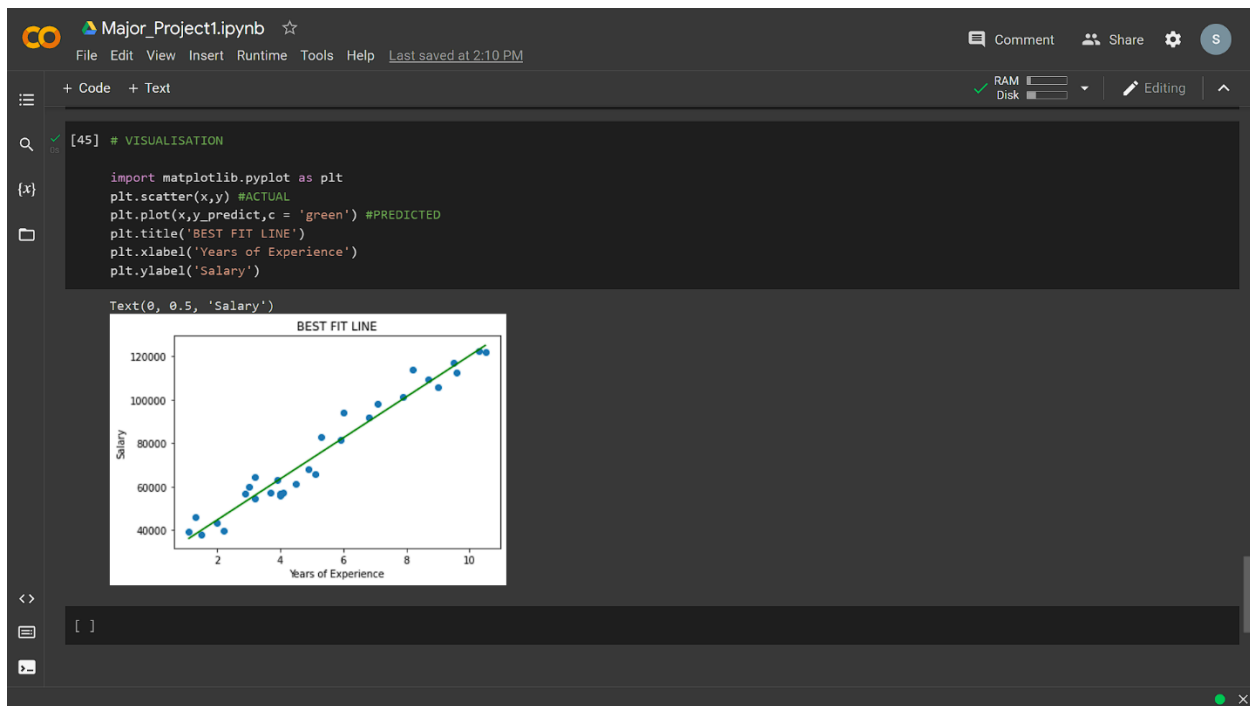
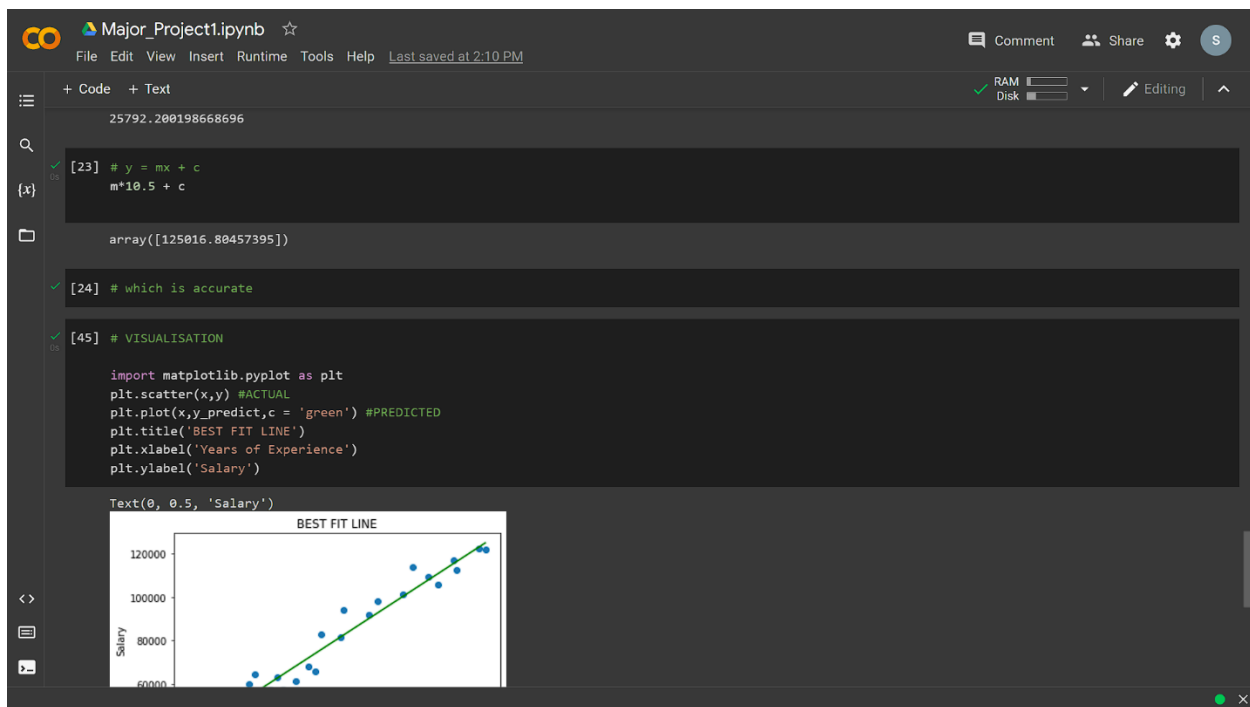
[43] c = model.intercept_ # y-intercept

c

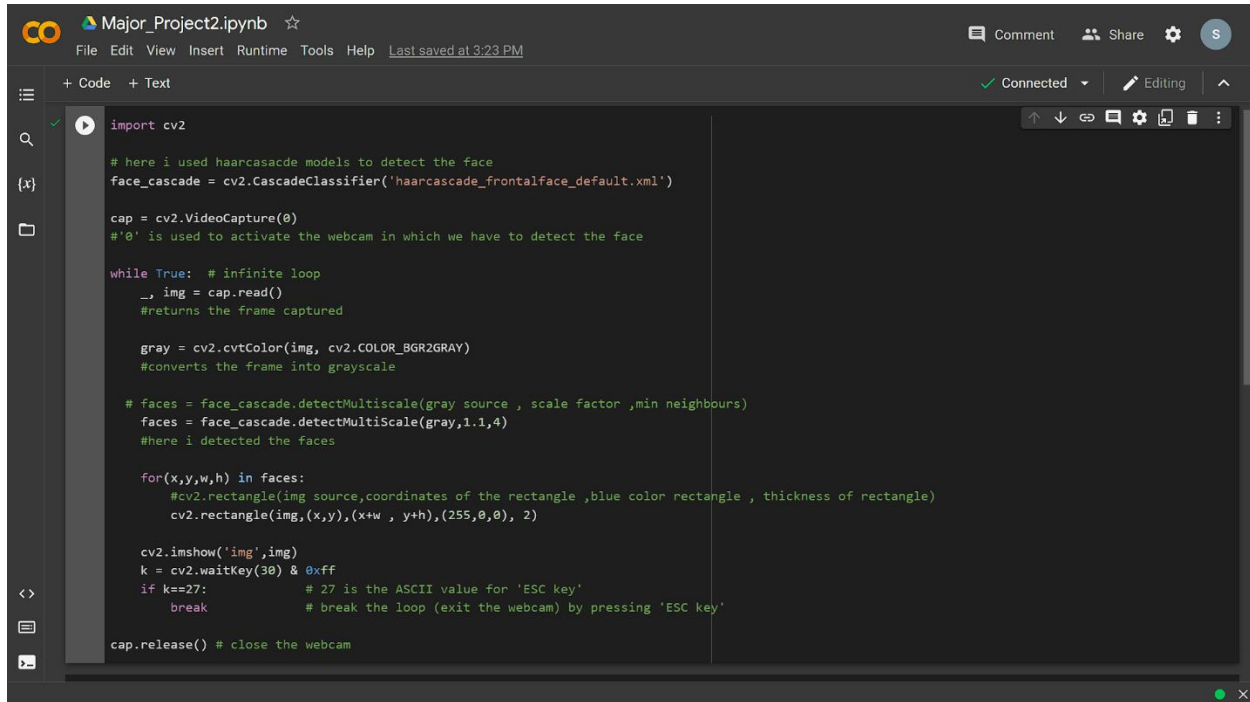
25792.200198668696

[23] # y = mx + c

m*10.5 + c



Major Project 2 Screenshots :



The screenshot displays a Jupyter Notebook titled "Major_Project2.ipynb" in a dark-themed editor. The interface includes a top menu bar with options like File, Edit, View, Insert, Runtime, Tools, and Help. Below the menu, there are tabs for "Code" and "Text", and a status bar indicating "Connected" and "Editing". The main area contains a Python script for face detection using OpenCV. The script imports cv2, initializes a face cascade classifier, captures video from a webcam, converts frames to grayscale, and uses the classifier to detect faces. It also includes a loop to display the detected faces with bounding boxes and a key press to exit the loop.

```
import cv2

# here i used haarcascade models to detect the face
face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

cap = cv2.VideoCapture(0)
# '0' is used to activate the webcam in which we have to detect the face

while True: # infinite loop
    _, img = cap.read()
    # returns the frame captured

    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    # converts the frame into grayscale

    # faces = face_cascade.detectMultiscale(gray, scale factor, min neighbours)
    faces = face_cascade.detectMultiScale(gray, 1.1, 4)
    # here i detected the faces

    for (x, y, w, h) in faces:
        # cv2.rectangle(img, source, coordinates of the rectangle, blue color rectangle, thickness of rectangle)
        cv2.rectangle(img, (x, y), (x+w, y+h), (255, 0, 0), 2)

    cv2.imshow('img', img)
    k = cv2.waitKey(30) & 0xff
    if k == 27: # 27 is the ASCII value for 'ESC key'
        break # break the loop (exit the webcam) by pressing 'ESC key'

cap.release() # close the webcam
```

My Github Link :

<https://github.com/Kaivalya16/Rinex.git>

My Rinex Google Drive Link :

https://drive.google.com/drive/folders/1IV7jSs4ClfcYuzeUT6dLRTPWqPKQSOQg?usp=share_link