VALIA C.I. COLLEGE OF COMMERCE & VALIA LC COLLEGE OF ARTS
CES ROAD D.N NAGAR

(Affiliated to University Of Mumbai)

Mumbai-Maharashtra-400053

DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

This is to certify that the Journal entitled **Deep Learning** is bonafied work of **GOVIND SAINI** bearing Roll No: **07** submitted in partial fulfillment of the requirements for the award of degree of **BACHELOR OF SCIENCE in INFORMATION TECHNOLOGY** from University of Mumbai.

Date: **Internal Guide**

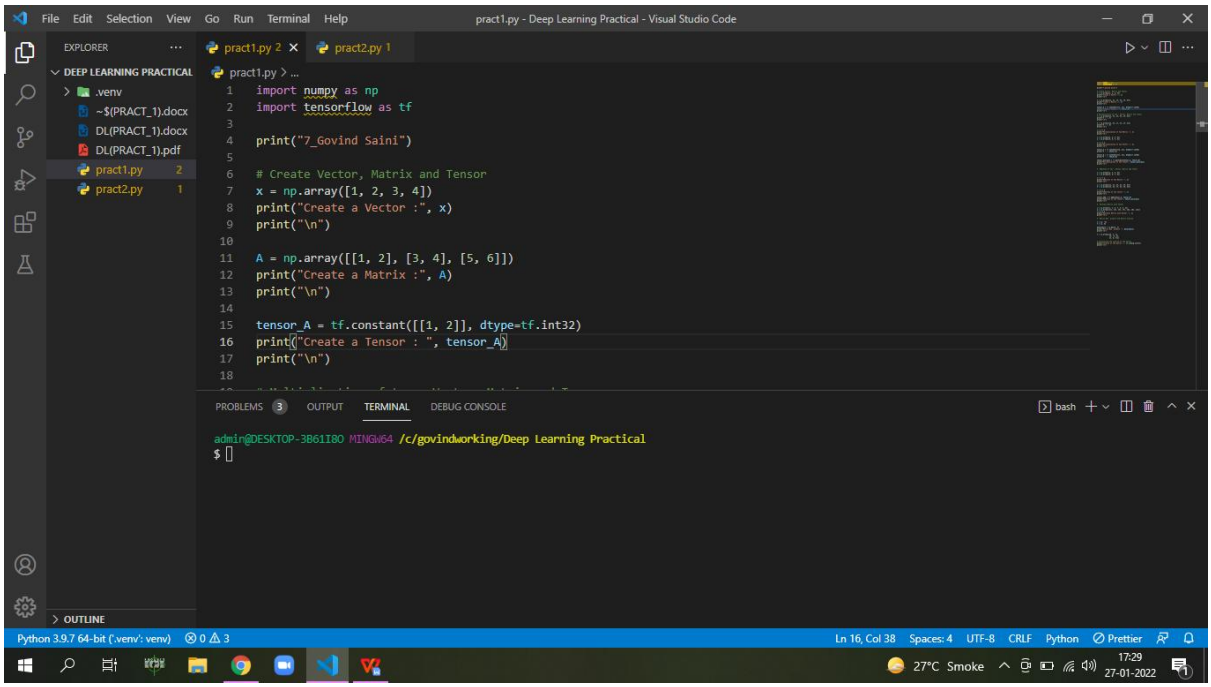
INDEX

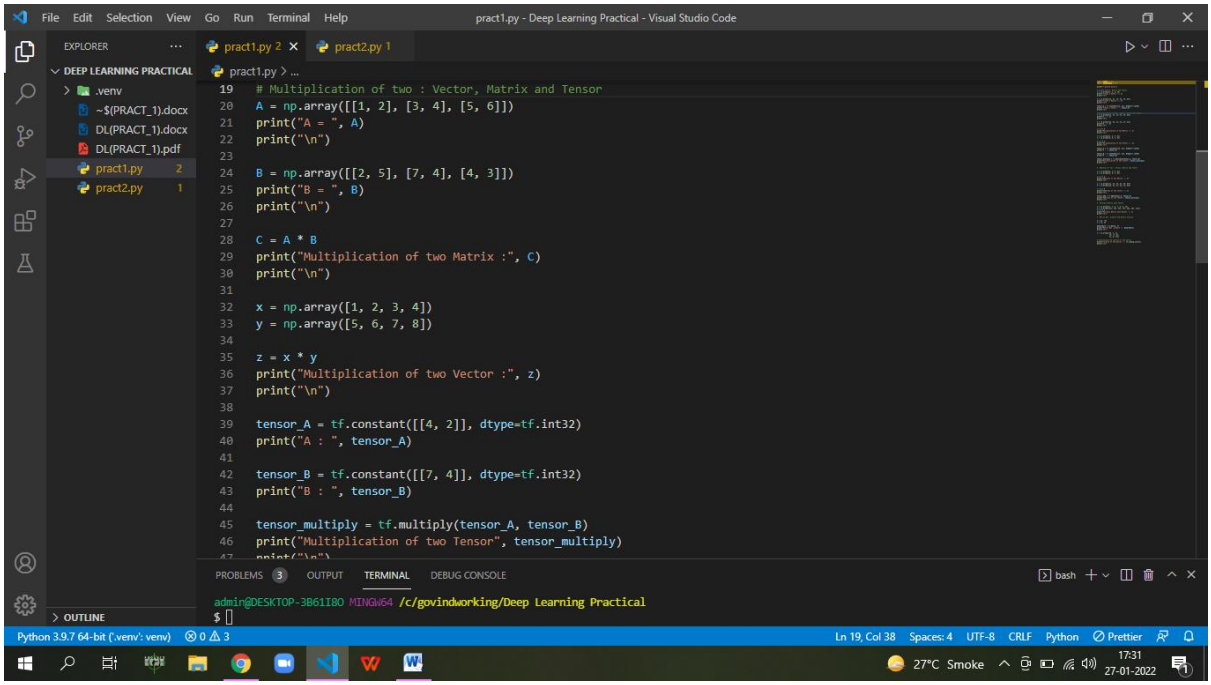
PRACTICAL NO : 1

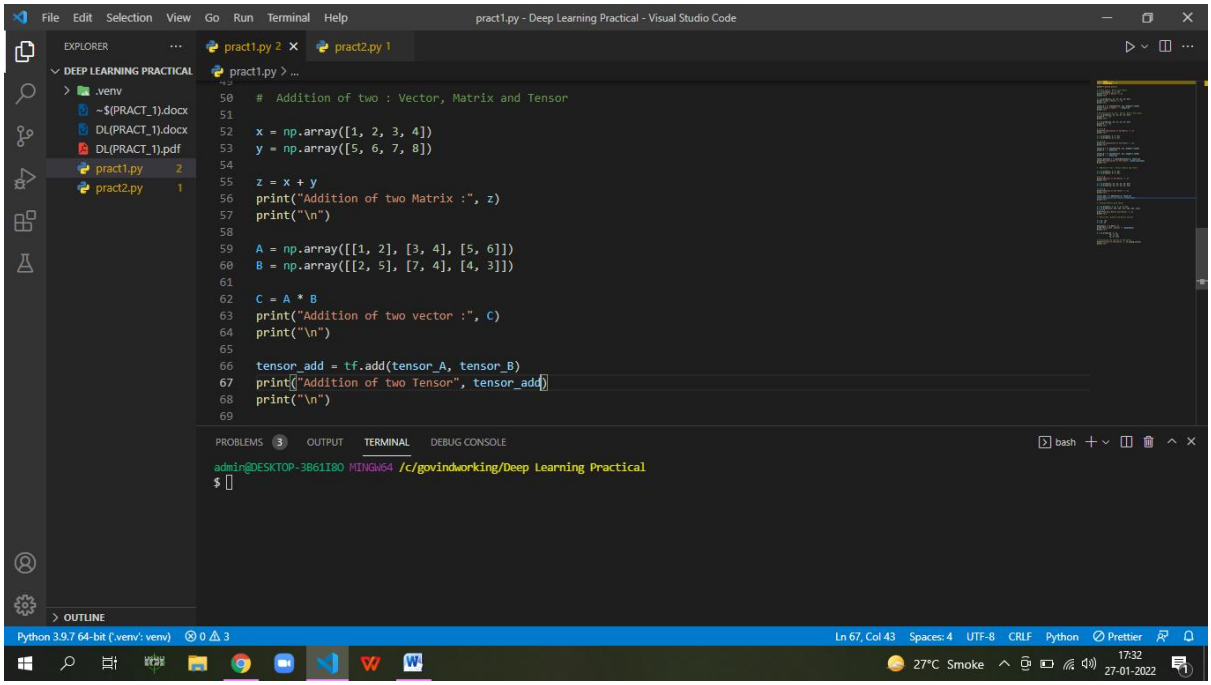
Aim : Write a Program to demonstrate following operations

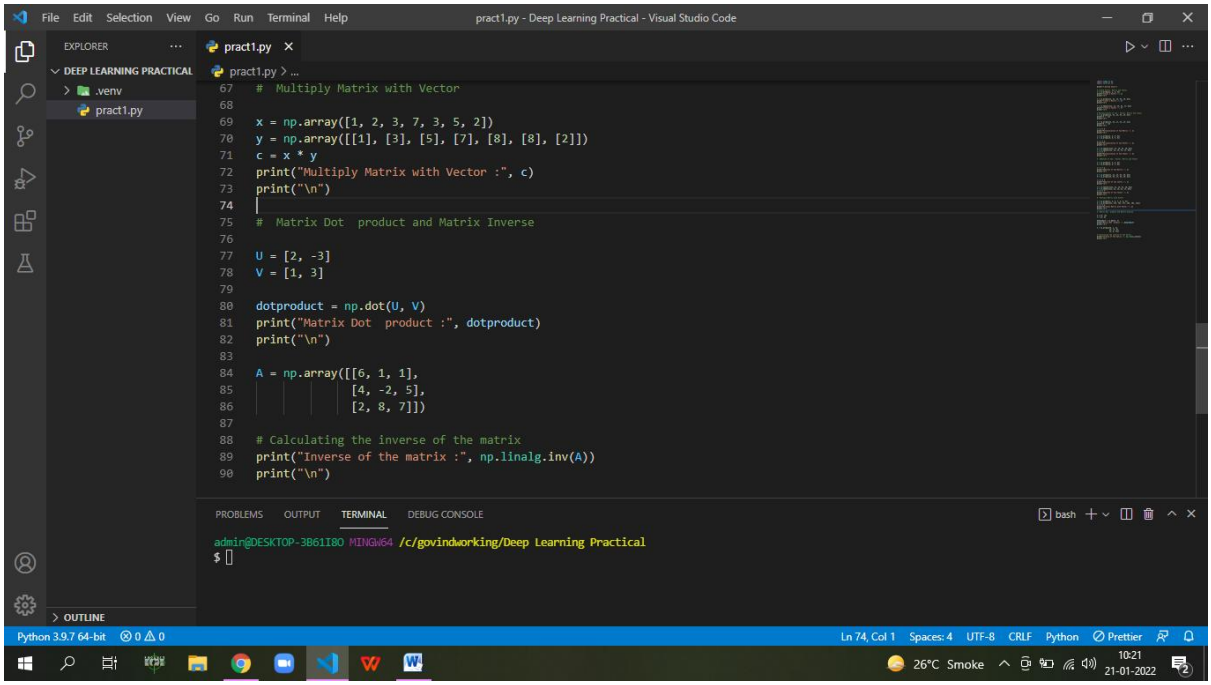
1) Create Vector, Matrix and Tensor
2) Multiplication of two : Vector, Matrix and Tensor
3) Addition of two : Vector, Matrix and Tensor
4) Multiply Matrix with Vector
5) Matrix Dot product and Matrix Inverse

Code :

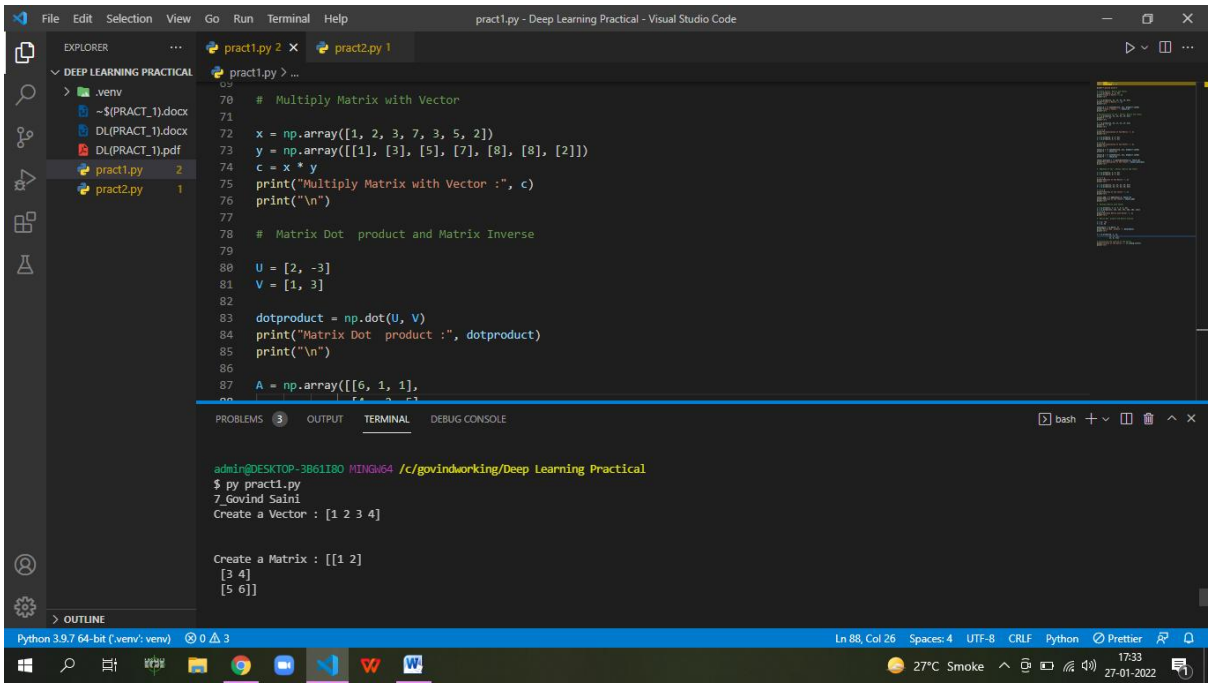


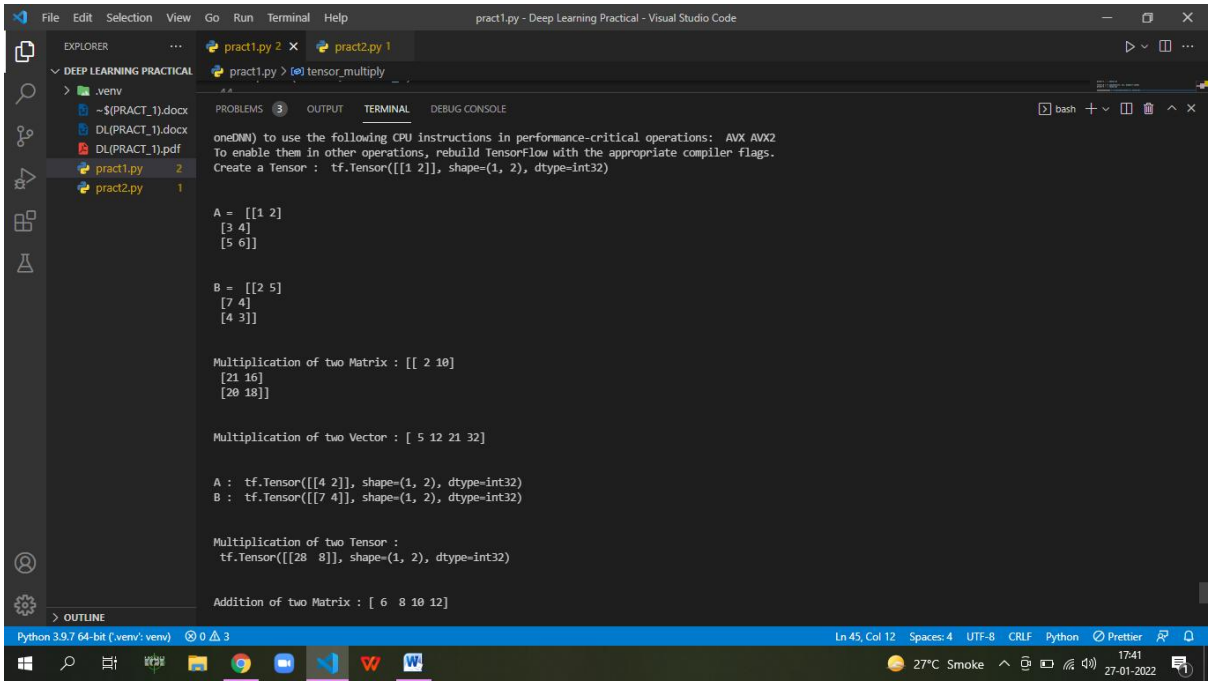

File Edit Selection View Go Run Terminal Help
pract1.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
.venv
~$(PRACT_1).docx
DL(PRACT_1).docx
DL(PRACT_1).pdf
pract1.py 2
pract2.py 1
pract1.py 2
pract2.py 1
pract1.py > ...
# Multiplication of two : Vector, Matrix and Tensor
A = np.array([[1, 2], [3, 4], [5, 6]])
print("A = ", A)
print("\n")
B = np.array([[2, 5], [7, 4], [4, 3]])
print("B = ", B)
print("\n")
C = A * B
print("Multiplication of two Matrix :", C)
print("\n")
x = np.array([1, 2, 3, 4])
y = np.array([5, 6, 7, 8])
z = x * y
print("Multiplication of two Vector :", z)
print("\n")
tensor_A = tf.constant([[4, 2]], dtype=tf.int32)
print("A : ", tensor_A)
tensor_B = tf.constant([[7, 4]], dtype=tf.int32)
print("B : ", tensor_B)
tensor_multiply = tf.multiply(tensor_A, tensor_B)
print("Multiplication of two Tensor", tensor_multiply)
PROBLEMS 3 OUTPUT TERMINAL DEBUG CONSOLE
bash
admin@DESKTOP-3B61I8O MINGW64 /c/govindworking/Deep Learning Practical
$
OUTLINE
Python 3.9.7 64-bit ('.venv': venv) 0 3
Ln 19, Col 38 Spaces: 4 UTF-8 CRLF Python Prettier
27°C Smoke
17:31
27-01-2022

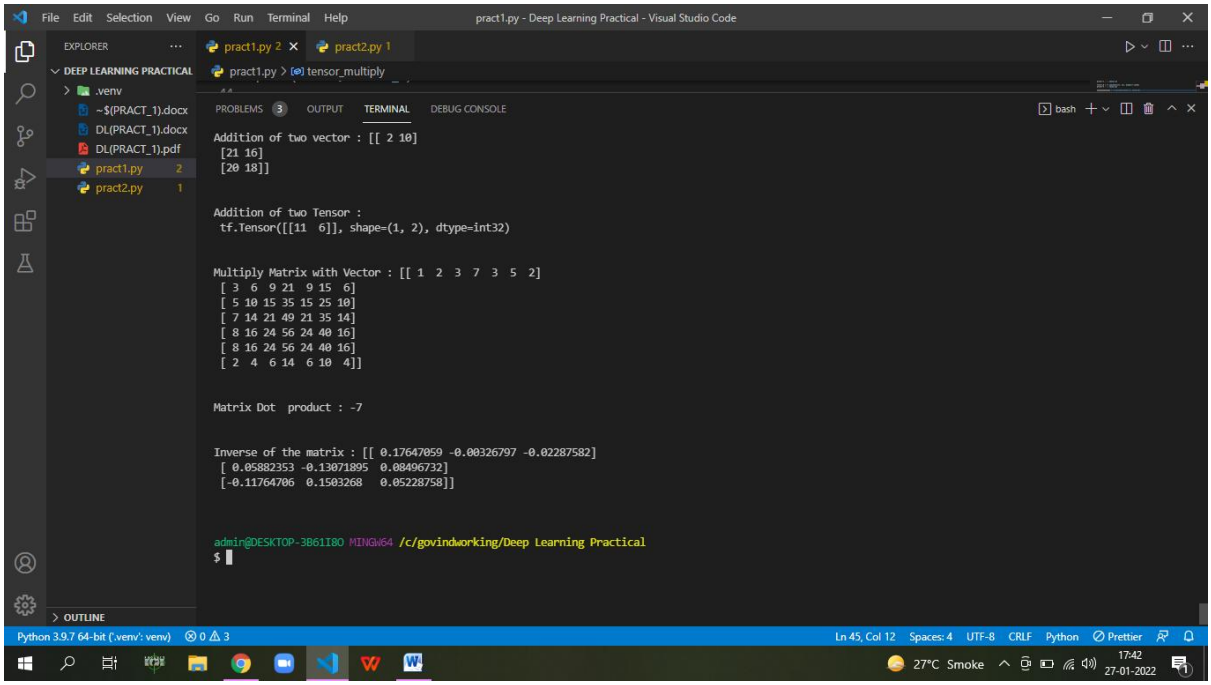

File Edit Selection View Go Run Terminal Help
pract1.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
.venv
~$(PRACT_1).docx
DL(PRACT_1).docx
DL(PRACT_1).pdf
pract1.py 2
pract2.py 1
pract1.py 2
pract2.py 1
pract1.py > ...
# Addition of two : Vector, Matrix and Tensor
x = np.array([1, 2, 3, 4])
y = np.array([5, 6, 7, 8])
z = x + y
print("Addition of two Matrix :", z)
print("\n")
A = np.array([[1, 2], [3, 4], [5, 6]])
B = np.array([[2, 5], [7, 4], [4, 3]])
C = A * B
print("Addition of two vector :", C)
print("\n")
tensor_add = tf.add(tensor_A, tensor_B)
print("Addition of two Tensor", tensor_add)
print("\n")
PROBLEMS 3 OUTPUT TERMINAL DEBUG CONSOLE
bash
admin@DESKTOP-3B61I8O MINGW64 /c/govindworking/Deep Learning Practical
$
OUTLINE
Python 3.9.7 64-bit ('.venv': venv) 0 3
Ln 67, Col 43 Spaces: 4 UTF-8 CRLF Python Prettier
27°C Smoke
17:32
27-01-2022


File Edit Selection View Go Run Terminal Help
pract1.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
.venv
pract1.py
pract1.py
# Multiply Matrix with Vector
x = np.array([1, 2, 3, 7, 3, 5, 2])
y = np.array([[1], [3], [5], [7], [8], [8], [2]])
c = x * y
print("Multiply Matrix with Vector :", c)
print("\n")
# Matrix Dot product and Matrix Inverse
U = [2, -3]
V = [1, 3]
dotproduct = np.dot(U, V)
print("Matrix Dot product :", dotproduct)
print("\n")
A = np.array([[6, 1, 1],
[4, -2, 5],
[2, 8, 7]])
# Calculating the inverse of the matrix
print("Inverse of the matrix :", np.linalg.inv(A))
print("\n")
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
bash
admin@DESKTOP-3B61I8O MINGW64 /c/govindworking/Deep Learning Practical
$
OUTLINE
Python 3.9.7 64-bit
Ln 74, Col 1 Spaces: 4 UTF-8 CRLF Python Prettier
26°C Smoke
10:21
21-01-2022

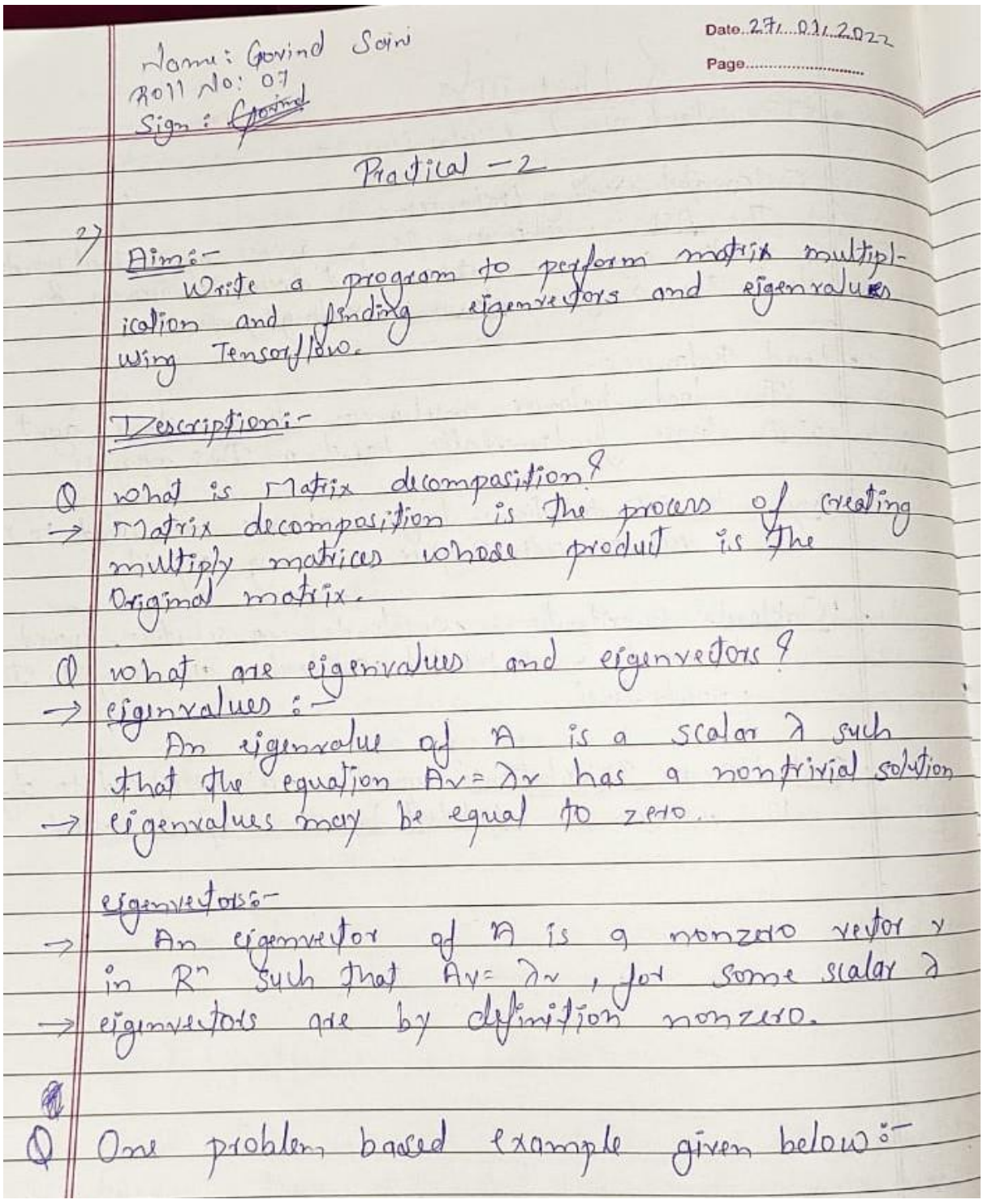
Output :


File Edit Selection View Go Run Terminal Help
pract1.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
.venv
~$(PRACT_1).docx
DL(PRACT_1).docx
DL(PRACT_1).pdf
pract1.py 2
pract2.py 1
pract1.py 2
pract2.py 1
# Multiply Matrix with Vector
x = np.array([1, 2, 3, 7, 3, 5, 2])
y = np.array([[1], [3], [5], [7], [8], [8], [2]])
c = x * y
print("Multiply Matrix with Vector :", c)
print("\n")
# Matrix Dot product and Matrix Inverse
U = [2, -3]
V = [1, 3]
dotproduct = np.dot(U, V)
print("Matrix Dot product :", dotproduct)
print("\n")
A = np.array([[6, 1, 1],
PROBLEMS 3 OUTPUT TERMINAL DEBUG CONSOLE
bash
admin@DESKTOP-3B61I8O MINGW64 /c/govindworking/Deep Learning Practical
$ py pract1.py
7_Govind Saini
Create a Vector : [1 2 3 4]
Create a Matrix : [[1 2]
[3 4]
[5 6]]
OUTLINE
Python 3.9.7 64-bit ('.venv': venv)
Ln 88, Col 26 Spaces: 4 UTF-8 CRLF Python Prettier
27°C Smoke
17:33
27-01-2022


File Edit Selection View Go Run Terminal Help
pract1.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
.venv
~$(PRACT_1).docx
DL(PRACT_1).docx
DL(PRACT_1).pdf
pract1.py 2
pract2.py 1
pract1.py 2
pract2.py 1
pract1.py > tensor_multiply
PROBLEMS 3 OUTPUT TERMINAL DEBUG CONSOLE
bash
oneDNN) to use the following CPU instructions in performance-critical operations: AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
Create a Tensor : tf.Tensor([[1 2]], shape=(1, 2), dtype=int32)
A = [[1 2]
[3 4]
[5 6]]
B = [[2 5]
[7 4]
[4 3]]
Multiplication of two Matrix : [[2 10]
[21 16]
[20 18]]
Multiplication of two Vector : [5 12 21 32]
A : tf.Tensor([[4 2]], shape=(1, 2), dtype=int32)
B : tf.Tensor([[7 4]], shape=(1, 2), dtype=int32)
Multiplication of two Tensor :
tf.Tensor([[28 8]], shape=(1, 2), dtype=int32)
Addition of two Matrix : [6 8 10 12]
OUTLINE
Python 3.9.7 64-bit ('.venv': venv)
Ln 45, Col 12 Spaces: 4 UTF-8 CRLF Python Prettier
27°C Smoke
17:41
27-01-2022


File Edit Selection View Go Run Terminal Help
pract1.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
.venv
~$(PRACT_1).docx
DL(PRACT_1).docx
DL(PRACT_1).pdf
pract1.py 2
pract2.py 1
pract1.py 2
pract2.py 1
pract1.py > tensor_multiply
PROBLEMS 3 OUTPUT TERMINAL DEBUG CONSOLE
bash
Addition of two vector : [[2 10]
[21 16]
[20 18]]
Addition of two Tensor :
tf.Tensor([[11 6]], shape=(1, 2), dtype=int32)
Multiply Matrix with Vector : [[1 2 3 7 3 5 2]
[3 6 9 21 9 15 6]
[5 10 15 35 15 25 10]
[7 14 21 49 21 35 14]
[8 16 24 56 24 40 16]
[8 16 24 56 24 40 16]
[2 4 6 14 6 10 4]]
Matrix Dot product : -7
Inverse of the matrix : [[0.17647059 -0.00326797 -0.02287582]
[0.05882353 -0.13071895 0.08496732]
[-0.11764706 0.1503268 0.05228758]]
admin@DESKTOP-3B61I8O MINGW64 /c/govindworking/Deep Learning Practical
$
OUTLINE
Python 3.9.7 64-bit ('.venv': venv)
Ln 45, Col 12 Spaces: 4 UTF-8 CRLF Python Prettier
27°C Smoke
17:42
27-01-2022

Practical No : 2



Name: Govind Saini
Roll No: 07
Sign: Govind

Date 27/01/2022
Page

Practical - 2

Aim:-
Write a program to perform matrix multiplication and finding eigenvectors and eigenvalues using Tensorflow.

Description:-

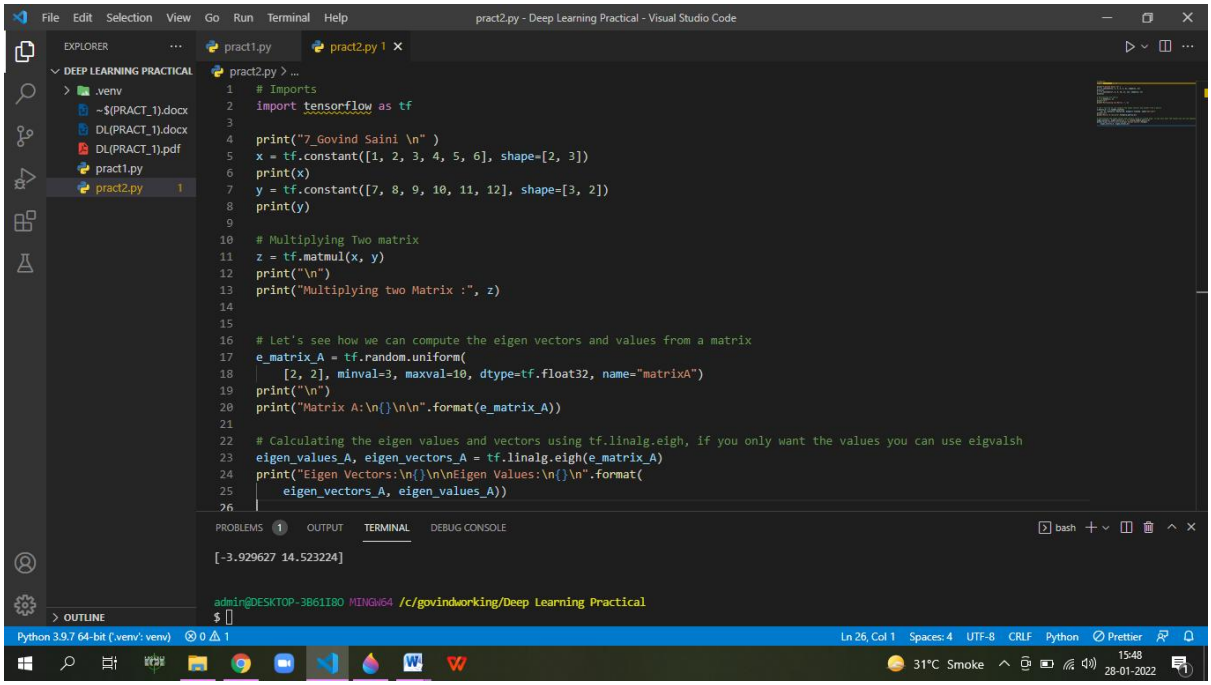
Q what is matrix decomposition?
→ matrix decomposition is the process of creating multiply matrices whose product is the original matrix.

Q what are eigenvalues and eigenvectors?
→ eigenvalues :-
An eigenvalue of A is a scalar λ such that the equation $Av = \lambda v$ has a nontrivial solution
→ eigenvalues may be equal to zero.

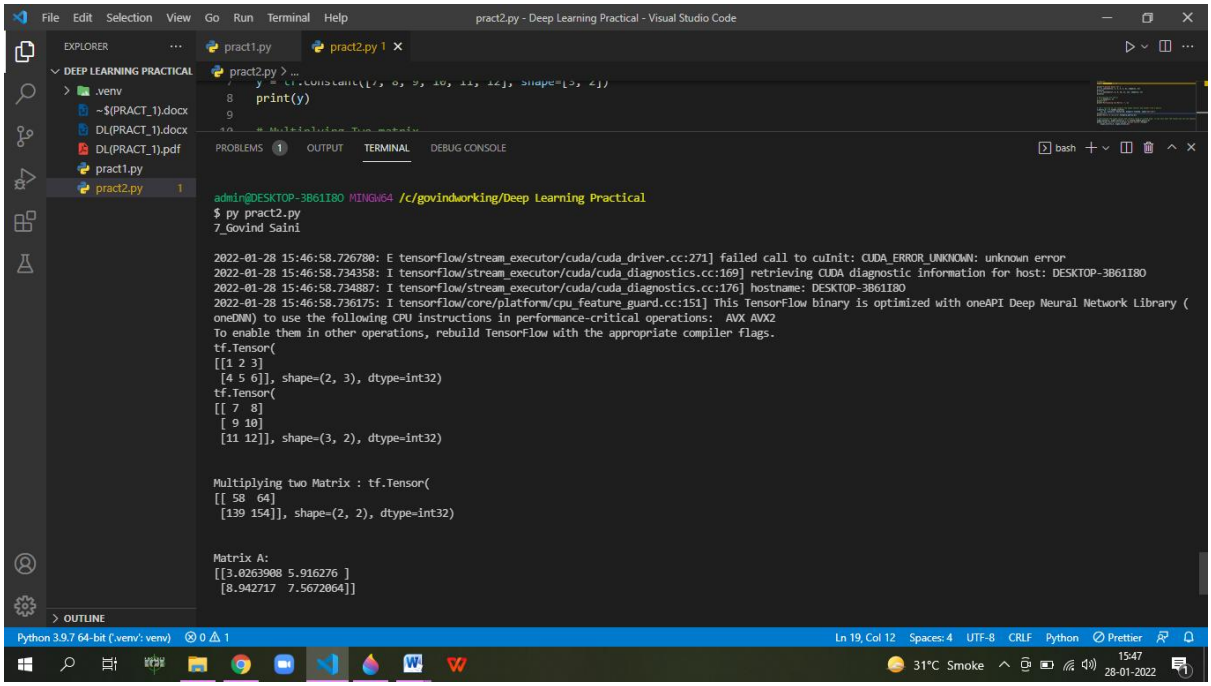
eigenvectors:-
→ An eigenvector of A is a nonzero vector v in R^n such that $Av = \lambda v$, for some scalar λ
→ eigenvectors are by definition nonzero.

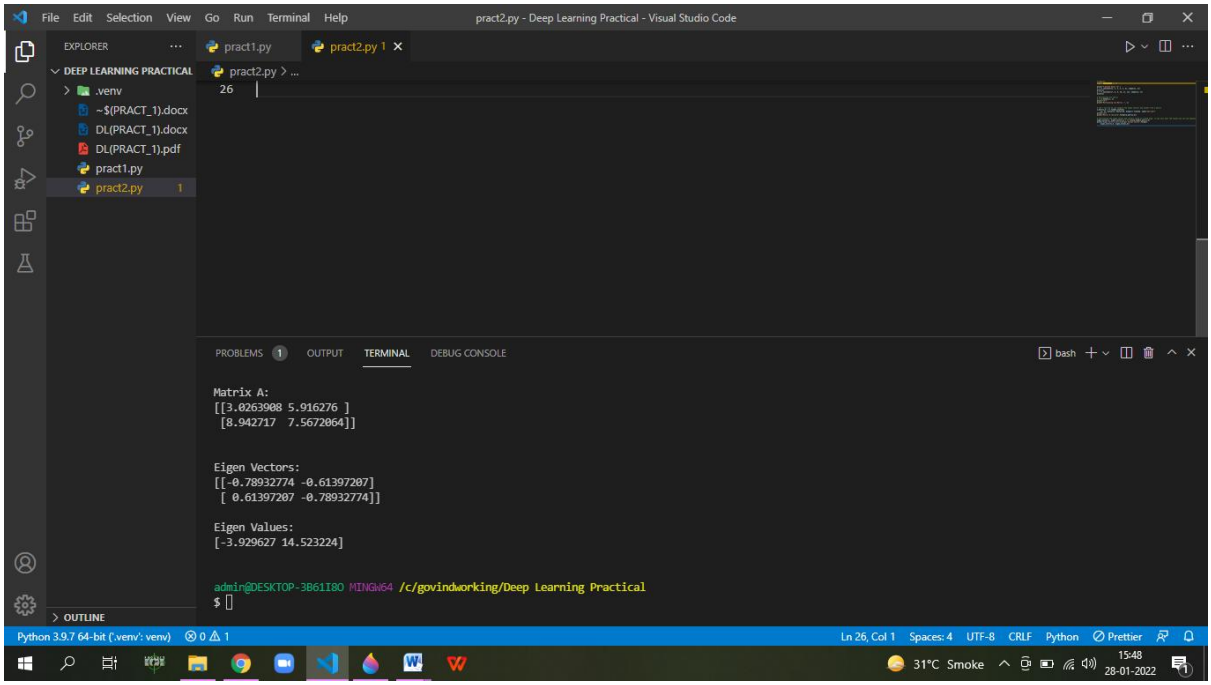
Q One problem based example given below:-

Code :


File Edit Selection View Go Run Terminal Help
pract2.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
.venv
~$(PRACT_1).docx
DL(PRACT_1).docx
DL(PRACT_1).pdf
pract1.py
pract2.py
pract1.py
pract2.py 1
pract2.py > ...
# Imports
import tensorflow as tf
print("7_Govind Saini \n")
x = tf.constant([1, 2, 3, 4, 5, 6], shape=[2, 3])
print(x)
y = tf.constant([7, 8, 9, 10, 11, 12], shape=[3, 2])
print(y)
# Multiplying Two matrix
z = tf.matmul(x, y)
print("\n")
print("Multiplying two Matrix :", z)
# Let's see how we can compute the eigen vectors and values from a matrix
e_matrix_A = tf.random.uniform(
[2, 2], minval=3, maxval=10, dtype=tf.float32, name="matrixA")
print("\n")
print("Matrix A:\n{}\n\n".format(e_matrix_A))
# Calculating the eigen values and vectors using tf.linalg.eigh, if you only want the values you can use eigvalsh
eigen_values_A, eigen_vectors_A = tf.linalg.eigh(e_matrix_A)
print("Eigen Vectors:\n{}\n\nEigen Values:\n{}\n".format(
eigen_vectors_A, eigen_values_A))
PROBLEMS 1 OUTPUT TERMINAL DEBUG CONSOLE
bash
[-3.929627 14.523224]
admin@DESKTOP-3B61I8O MINGW64 /c/govindworking/Deep Learning Practical
$
OUTLINE
Python 3.9.7 64-bit ('.venv': venv) 0 1
Ln 26, Col 1 Spaces: 4 UTF-8 CRLF Python Prettier
31°C Smoke
15:48
28-01-2022

Output :


File Edit Selection View Go Run Terminal Help
pract2.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
.venv
~$(PRACT_1).docx
DL(PRACT_1).docx
DL(PRACT_1).pdf
pract1.py
pract2.py
pract1.py
pract2.py 1
pract2.py > ...
print(y)
PROBLEMS 1 OUTPUT TERMINAL DEBUG CONSOLE
bash
admin@DESKTOP-3B61I8O MINGW64 /c/govindworking/Deep Learning Practical
$ py pract2.py
7_Govind Saini
2022-01-28 15:46:58.726780: E tensorflow/stream_executor/cuda/cuda_driver.cc:271] failed call to cuInit: CUDA_ERROR_UNKNOWN: unknown error
2022-01-28 15:46:58.734358: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:169] retrieving CUDA diagnostic information for host: DESKTOP-3B61I8O
2022-01-28 15:46:58.734887: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:176] hostname: DESKTOP-3B61I8O
2022-01-28 15:46:58.736175: I tensorflow/core/platform/cpu_feature_guard.cc:151] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (
oneDNN) to use the following CPU instructions in performance-critical operations: AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
tf.Tensor(
[[1 2 3]
[4 5 6]], shape=(2, 3), dtype=int32)
tf.Tensor(
[[7 8]
[9 10]
[11 12]], shape=(3, 2), dtype=int32)
Multiplying two Matrix : tf.Tensor(
[[58 64]
[139 154]], shape=(2, 2), dtype=int32)
Matrix A:
[[3.0263908 5.916276]
[8.942717 7.5672064]]
OUTLINE
Python 3.9.7 64-bit ('.venv': venv) 0 1
Ln 19, Col 12 Spaces: 4 UTF-8 CRLF Python Prettier
31°C Smoke
15:47
28-01-2022


File Edit Selection View Go Run Terminal Help
pract2.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
.venv
~$(PRACT_1).docx
DL(PRACT_1).docx
DL(PRACT_1).pdf
pract1.py
pract2.py
pract1.py
pract2.py 1
pract2.py > ...
26
PROBLEMS 1
OUTPUT
TERMINAL
DEBUG CONSOLE
bash
Matrix A:
[[3.0263908 5.916276]
[8.942717 7.5672064]]
Eigen Vectors:
[[-0.78932774 -0.61397207]
[0.61397207 -0.78932774]]
Eigen Values:
[-3.929627 14.523224]
admin@DESKTOP-3B61I8O MINGW64 /c/govindworking/Deep Learning Practical
$
OUTLINE
Python 3.9.7 64-bit ('.venv': venv)
Ln 26, Col 1
Spaces: 4
UTF-8
CRLF
Python
Prettier
31°C Smoke
15:48
28-01-2022

Practical No : 3

Name : Govind Saini
Roll No : 07
Sign : Govind

Date 5/12/2022
Page

Practical - 3

3)

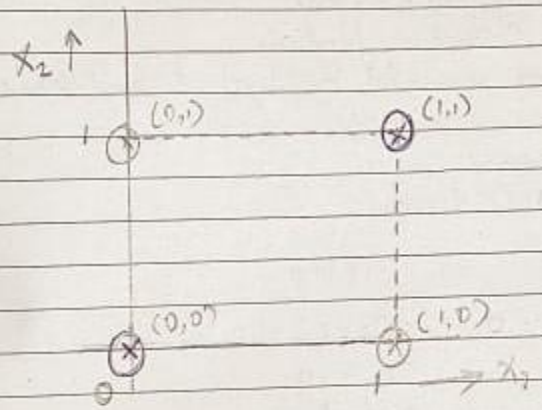
Aim :-

Solving XOR problem using deep forward network.

Description :-

XOR problem :-

Single layer perception, no hidden neurons
Cannot classify input pattern - Not linear Separable
The Exclusive OR (XOR) problem



X_1	X_2	y
0	0	0
0	1	1
1	0	1
1	1	0

⊗ → 0 pattern
⊗ → 1 pattern

Our Algorithm - regardless of how it work must correctly output the XOR value each of 4 points. will b modelling this as a classification problem, so class1 wou represent an XOR value of 1, while class0 would represen a value of 0.

Govind

Prehics

Name: Govind Saini
Roll No: 07
Sign: Govind

Date: / /
Page:

Single layer perceptron:-
Let's model the problem using a single layer perceptron.

Input Data :-
The data we'll train our model on is the value we saw for the XOR function

Data	Target
[0, 0]	0
[0, 1]	1
[1, 0]	1
[1, 1]	0

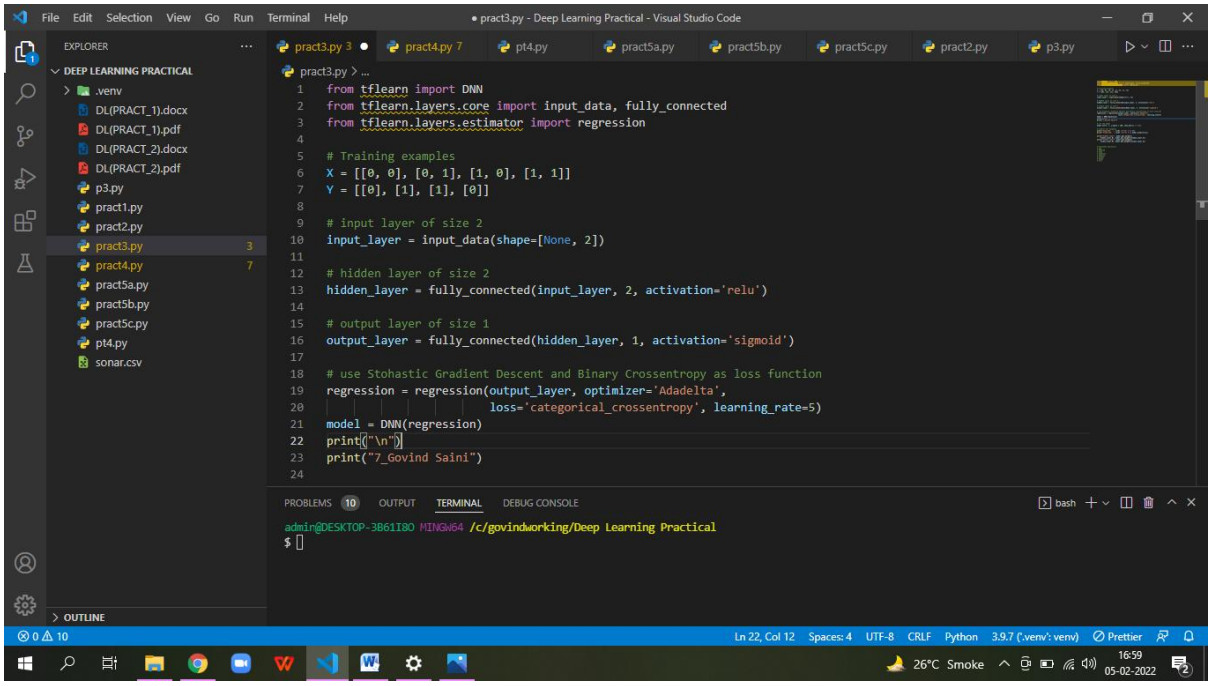
The Data :-
We next create our training data. This data is the same of each kind of logic gate, since they all take in two boolean value or input

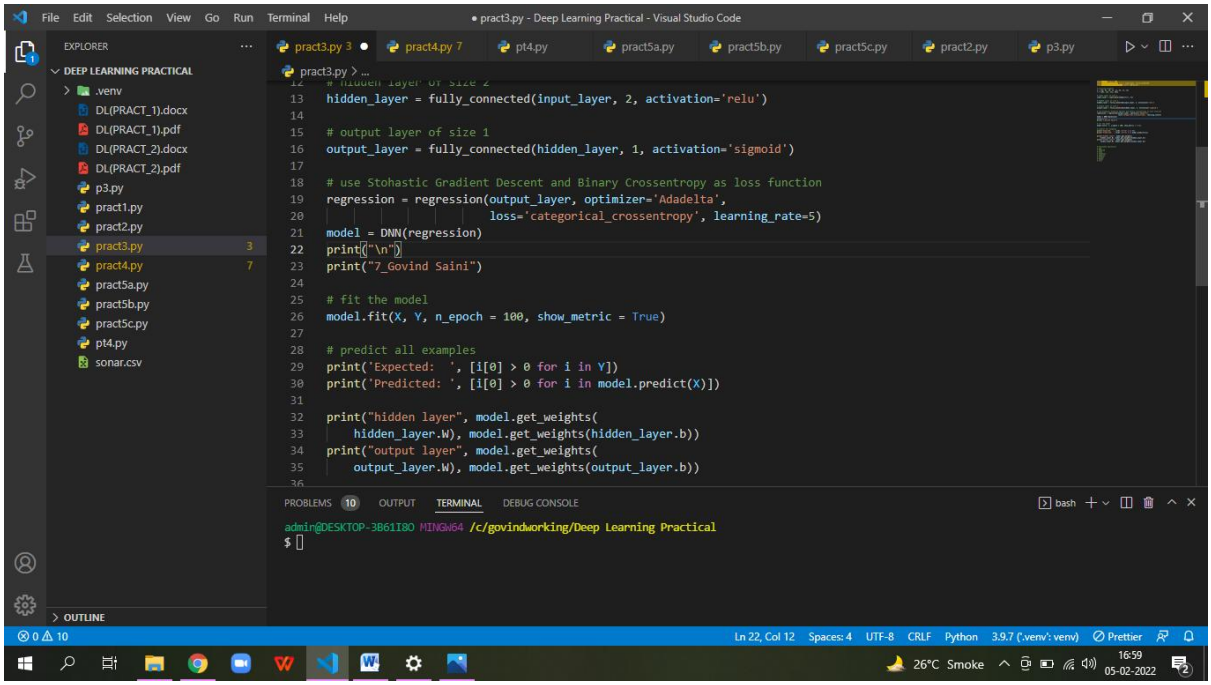
Training function :-
Here, we cycle through the data identifiantly keeping track how many consecutive datapoint we correctly classified if we managed to classify everything in one stretch, we terminate our algorithms

perception class :-
To bring everything together, we create a simple perceptron class with the function we just discussed. we have some instance Variable like the training data, target, number of input node and learning rate.

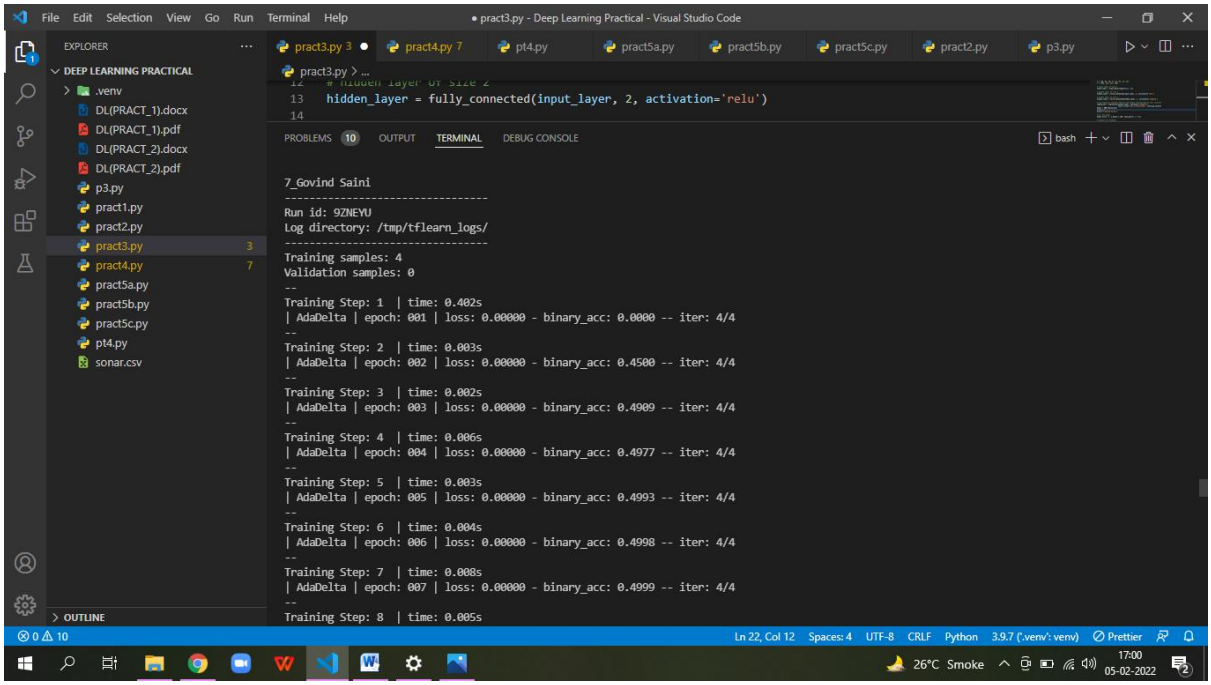
Govind

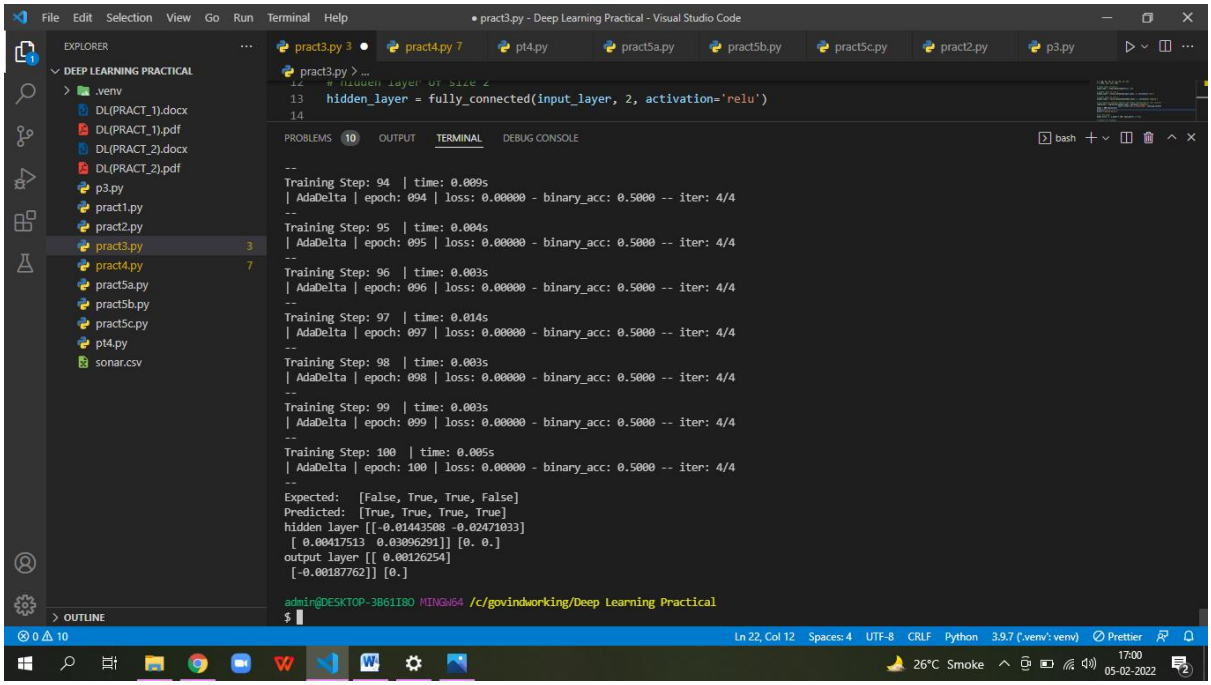
Code :


File Edit Selection View Go Run Terminal Help
• pract3.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
.venv
DL(PRACT_1).docx
DL(PRACT_1).pdf
DL(PRACT_2).docx
DL(PRACT_2).pdf
p3.py
pract1.py
pract2.py
pract3.py
pract4.py
pract5a.py
pract5b.py
pract5c.py
pt4.py
sonar.csv
pract3.py 3
pract4.py 7
pt4.py
pract5a.py
pract5b.py
pract5c.py
pract2.py
p3.py
from tflearn import DNN
from tflearn.layers.core import input_data, fully_connected
from tflearn.layers.estimator import regression
# Training examples
X = [[0, 0], [0, 1], [1, 0], [1, 1]]
Y = [[0], [1], [1], [0]]
# input layer of size 2
input_layer = input_data(shape=[None, 2])
# hidden layer of size 2
hidden_layer = fully_connected(input_layer, 2, activation='relu')
# output layer of size 1
output_layer = fully_connected(hidden_layer, 1, activation='sigmoid')
# use Stohastic Gradient Descent and Binary Crossentropy as loss function
regression = regression(output_layer, optimizer='Adadelta',
loss='categorical_crossentropy', learning_rate=5)
model = DNN(regression)
print("\n")
print("7_Govind Saini")
PROBLEMS 10 OUTPUT TERMINAL DEBUG CONSOLE
bash
admin@DESKTOP-3B61I8O MINGW64 /c/govindworking/Deep Learning Practical
$
OUTLINE
Ln 22, Col 12 Spaces: 4 UTF-8 CRLF Python 3.9.7 ('.venv': venv) Prettier
26°C Smoke
16:59
05-02-2022


File Edit Selection View Go Run Terminal Help
• pract3.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
.venv
DL(PRACT_1).docx
DL(PRACT_1).pdf
DL(PRACT_2).docx
DL(PRACT_2).pdf
p3.py
pract1.py
pract2.py
pract3.py
pract4.py
pract5a.py
pract5b.py
pract5c.py
pt4.py
sonar.csv
pract3.py 3
pract4.py 7
pt4.py
pract5a.py
pract5b.py
pract5c.py
pract2.py
p3.py
hidden_layer = fully_connected(input_layer, 2, activation='relu')
# output layer of size 1
output_layer = fully_connected(hidden_layer, 1, activation='sigmoid')
# use Stohastic Gradient Descent and Binary Crossentropy as loss function
regression = regression(output_layer, optimizer='Adadelta',
loss='categorical_crossentropy', learning_rate=5)
model = DNN(regression)
print("\n")
print("7_Govind Saini")
# fit the model
model.fit(X, Y, n_epoch = 100, show_metric = True)
# predict all examples
print('Expected: ', [i[0] > 0 for i in Y])
print('Predicted: ', [i[0] > 0 for i in model.predict(X)])
print("hidden layer", model.get_weights(
hidden_layer.W), model.get_weights(hidden_layer.b))
print("output layer", model.get_weights(
output_layer.W), model.get_weights(output_layer.b))
PROBLEMS 10 OUTPUT TERMINAL DEBUG CONSOLE
bash
admin@DESKTOP-3B61I8O MINGW64 /c/govindworking/Deep Learning Practical
$
OUTLINE
Ln 22, Col 12 Spaces: 4 UTF-8 CRLF Python 3.9.7 ('.venv': venv) Prettier
26°C Smoke
16:59
05-02-2022

Output :


File Edit Selection View Go Run Terminal Help
• pract3.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
.venv
DL(PRACT_1).docx
DL(PRACT_1).pdf
DL(PRACT_2).docx
DL(PRACT_2).pdf
p3.py
pract1.py
pract2.py
pract3.py 3
pract4.py 7
pract5a.py
pract5b.py
pract5c.py
pt4.py
sonar.csv
OUTLINE
pract3.py 3
pract4.py 7
pt4.py
pract5a.py
pract5b.py
pract5c.py
pract2.py
p3.py
pract3.py > ...
hidden_layer = fully_connected(input_layer, 2, activation='relu')
PROBLEMS 10
OUTPUT
TERMINAL
DEBUG CONSOLE
bash
7_Govind Saini
Run id: 9ZNEYU
Log directory: /tmp/tflearn_logs/
Training samples: 4
Validation samples: 0
Training Step: 1 | time: 0.402s
| AdaDelta | epoch: 001 | loss: 0.00000 - binary_acc: 0.0000 -- iter: 4/4
Training Step: 2 | time: 0.003s
| AdaDelta | epoch: 002 | loss: 0.00000 - binary_acc: 0.4500 -- iter: 4/4
Training Step: 3 | time: 0.002s
| AdaDelta | epoch: 003 | loss: 0.00000 - binary_acc: 0.4909 -- iter: 4/4
Training Step: 4 | time: 0.006s
| AdaDelta | epoch: 004 | loss: 0.00000 - binary_acc: 0.4977 -- iter: 4/4
Training Step: 5 | time: 0.003s
| AdaDelta | epoch: 005 | loss: 0.00000 - binary_acc: 0.4993 -- iter: 4/4
Training Step: 6 | time: 0.004s
| AdaDelta | epoch: 006 | loss: 0.00000 - binary_acc: 0.4998 -- iter: 4/4
Training Step: 7 | time: 0.008s
| AdaDelta | epoch: 007 | loss: 0.00000 - binary_acc: 0.4999 -- iter: 4/4
Training Step: 8 | time: 0.005s
Ln 22, Col 12 Spaces: 4 UTF-8 CRLF Python 3.9.7 ('.venv': venv) Prettier
26°C Smoke
17:00
05-02-2022


File Edit Selection View Go Run Terminal Help
• pract3.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
.venv
DL(PRACT_1).docx
DL(PRACT_1).pdf
DL(PRACT_2).docx
DL(PRACT_2).pdf
p3.py
pract1.py
pract2.py
pract3.py 3
pract4.py 7
pract5a.py
pract5b.py
pract5c.py
pt4.py
sonar.csv
OUTLINE
pract3.py 3
pract4.py 7
pt4.py
pract5a.py
pract5b.py
pract5c.py
pract2.py
p3.py
pract3.py > ...
hidden_layer = fully_connected(input_layer, 2, activation='relu')
PROBLEMS 10
OUTPUT
TERMINAL
DEBUG CONSOLE
bash
Training Step: 94 | time: 0.009s
| AdaDelta | epoch: 094 | loss: 0.00000 - binary_acc: 0.5000 -- iter: 4/4
Training Step: 95 | time: 0.004s
| AdaDelta | epoch: 095 | loss: 0.00000 - binary_acc: 0.5000 -- iter: 4/4
Training Step: 96 | time: 0.003s
| AdaDelta | epoch: 096 | loss: 0.00000 - binary_acc: 0.5000 -- iter: 4/4
Training Step: 97 | time: 0.014s
| AdaDelta | epoch: 097 | loss: 0.00000 - binary_acc: 0.5000 -- iter: 4/4
Training Step: 98 | time: 0.003s
| AdaDelta | epoch: 098 | loss: 0.00000 - binary_acc: 0.5000 -- iter: 4/4
Training Step: 99 | time: 0.003s
| AdaDelta | epoch: 099 | loss: 0.00000 - binary_acc: 0.5000 -- iter: 4/4
Training Step: 100 | time: 0.005s
| AdaDelta | epoch: 100 | loss: 0.00000 - binary_acc: 0.5000 -- iter: 4/4
Expected: [False, True, True, False]
Predicted: [True, True, True, True]
hidden layer [[-0.01443508 -0.02471033]
[0.00417513 0.03096291]] [0. 0.]
output layer [[0.00126254]
[-0.00187762]] [0.]
admin@DESKTOP-3B61I8O MINGW64 /c/govindworking/Deep Learning Practical
$
Ln 22, Col 12 Spaces: 4 UTF-8 CRLF Python 3.9.7 ('.venv': venv) Prettier
26°C Smoke
17:00
05-02-2022

Practical No : 4

Name: Govind Saini
Roll No: 07.
Sign: Govind

Date 20.1.2.1.2020
Page.....

Practical - 4

4)

Aim:-
Implement deep neural network for performing binary classification task.

Description:-

Describe the problem statement:

the given dataset comprises of the file transmitted sonar signal is a frequency-modulate chirp, rising in frequency.
The data set contains signal obtained from a variety of different aspect angles, spanning 90 degrees for the cylinder and 180 degrees for the rock.

Explain the activation function used in hidden and Output layer with equation

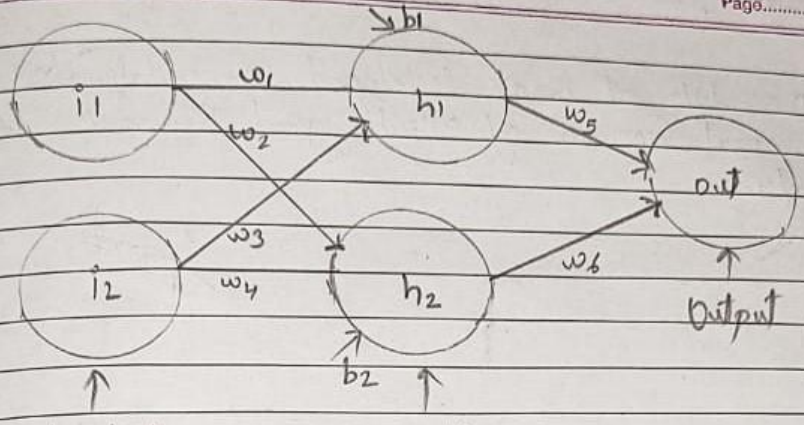
Activation function:-
Activation function decides, whether a neuron should be activated or not by calculating weighted sum and further adding bias with it.
The purpose of the activation function is to introduce non-linearity into the output of a neuron.

Variant	Equation
Linear function	i.e. $y = ax$
Sigmoid function	i.e. $A = 1/(1+e^{-x})$
Tanh function	i.e. $\tan(x) = 2 * \text{Sigmoid}(2x) - 1$

Govind

Name : Govind Saini
Roll No: 07
Sign: Govind

Date...../...../.....
Page.....



Dataset Description:-

Title :- Sonar, Mines vs Rock.

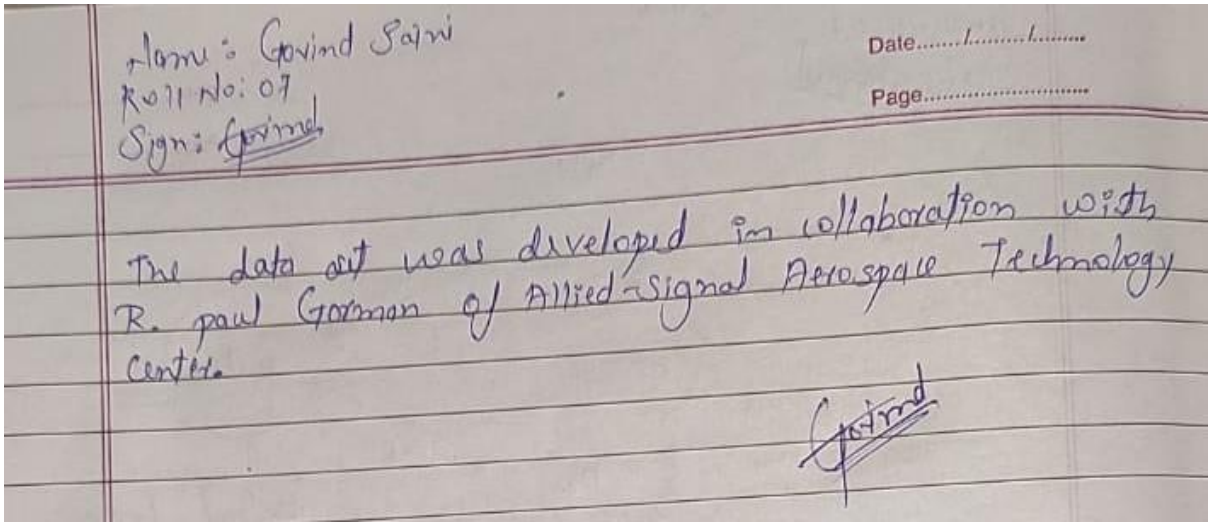
This data set can be used in a number of different way to test learning speed, quality of ultimate learning ability to generate, or combinations of these factor. It is a binary classification problem that requires a model to differentiate rocks from metal cylinders.

Field information

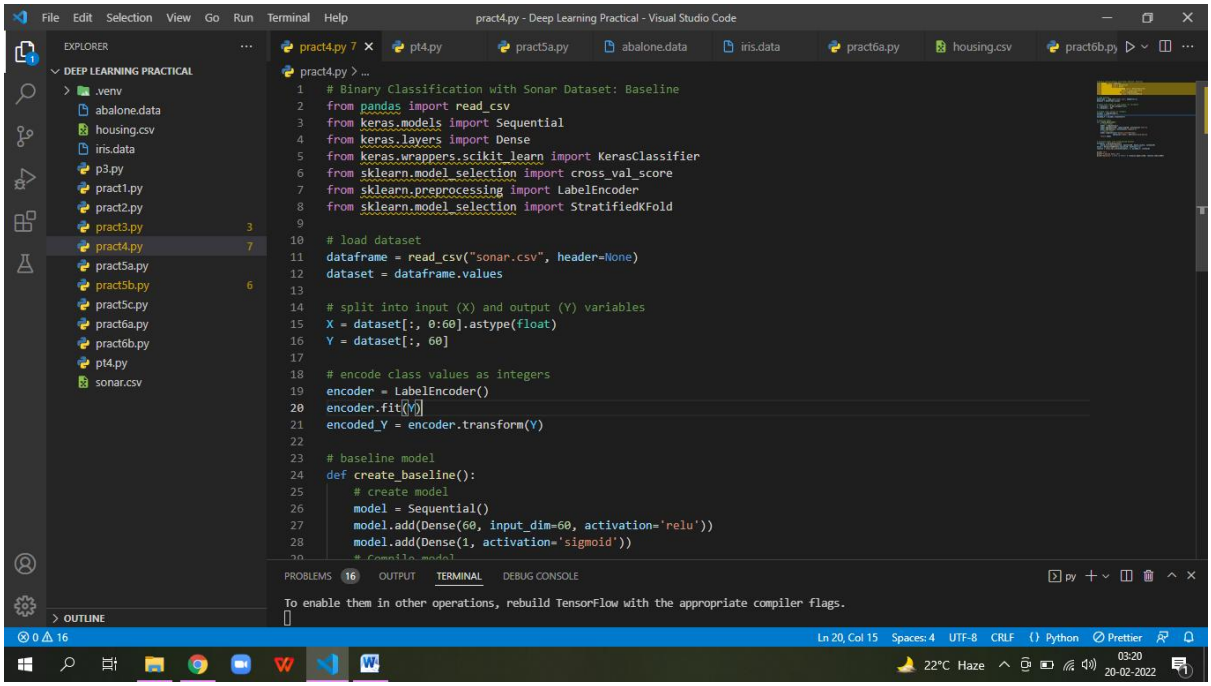
field name	Order	Type (format)
attribute_1	1	number (default)
attribute_2	2	number (default)
:		
class	61	string (default)

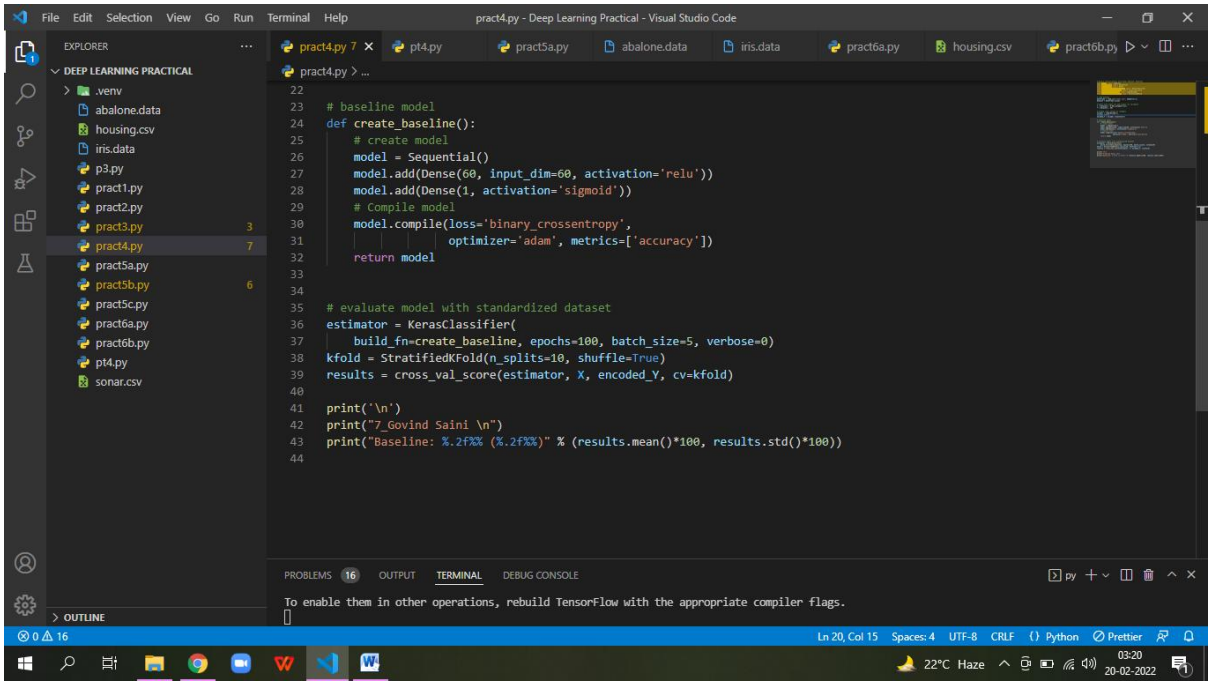
Dataset source or citation:-

The dataset was contributed to the benchmark co
by Terry Sejnowski, now at the Salk Institute and
University of California at San Diego. Govind

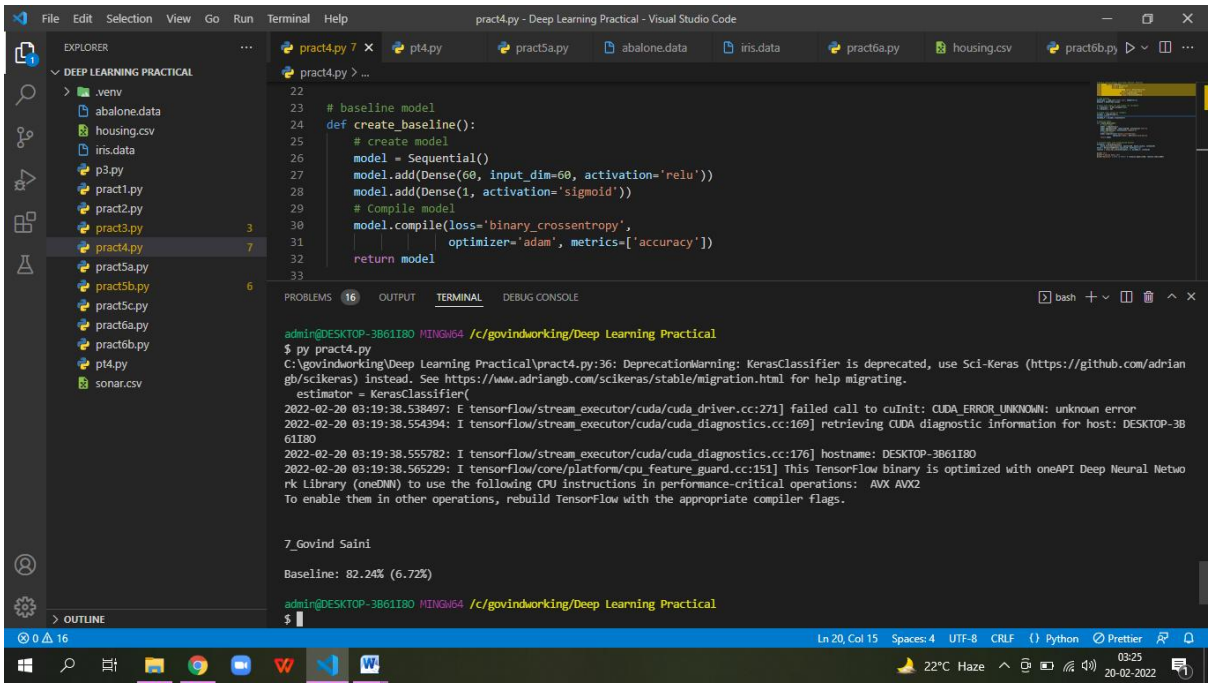

Name: Govind Saini
Roll No: 07
Sign: Govind
Date...../...../.....
Page.....
The data set was developed in collaboration with R. paul Gorman of Allied-Signal Aerospace Technology Center.
Govind

Code :

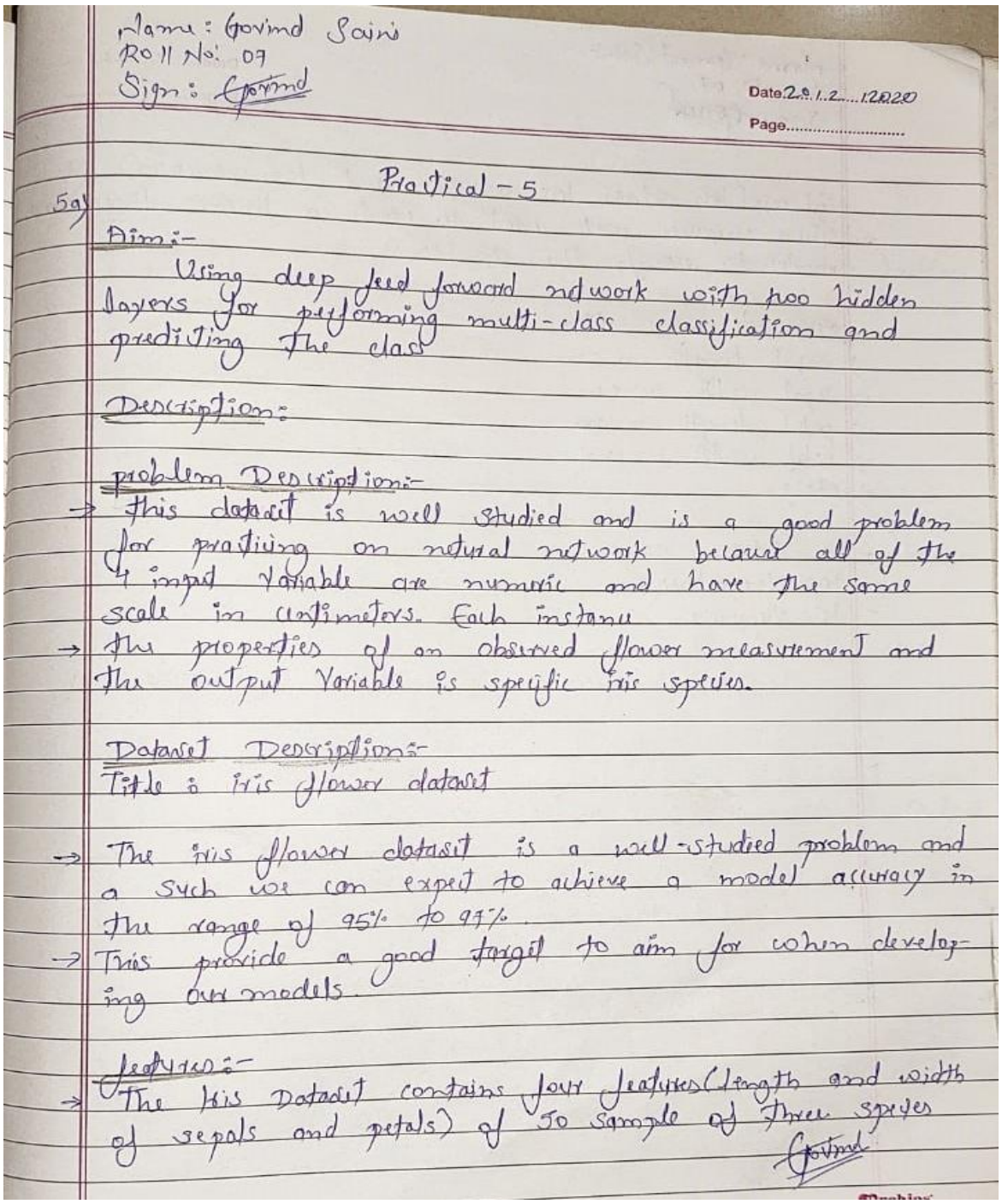

File Edit Selection View Go Run Terminal Help
pract4.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
.venv
abalone.data
housing.csv
iris.data
p3.py
pract1.py
pract2.py
pract3.py
pract4.py
pract5a.py
pract5b.py
pract5c.py
pract6a.py
pract6b.py
pt4.py
sonar.csv
OUTLINE
pract4.py 7
pt4.py
pract5a.py
abalone.data
iris.data
pract6a.py
housing.csv
pract6b.py
# Binary Classification with Sonar Dataset: Baseline
from pandas import read_csv
from keras.models import Sequential
from keras.layers import Dense
from keras.wrappers.scikit_learn import KerasClassifier
from sklearn.model_selection import cross_val_score
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import StratifiedKFold
# load dataset
dataframe = read_csv("sonar.csv", header=None)
dataset = dataframe.values
# split into input (X) and output (Y) variables
X = dataset[:, 0:60].astype(float)
Y = dataset[:, 60]
# encode class values as integers
encoder = LabelEncoder()
encoder.fit(Y)
encoded_Y = encoder.transform(Y)
# baseline model
def create_baseline():
# create model
model = Sequential()
model.add(Dense(60, input_dim=60, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
PROBLEMS 16 OUTPUT TERMINAL DEBUG CONSOLE
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
Ln 20, Col 15 Spaces: 4 UTF-8 CRLF Python Prettier
22°C Haze
03:20
20-02-2022


File Edit Selection View Go Run Terminal Help
pract4.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
.venv
abalone.data
housing.csv
iris.data
p3.py
pract1.py
pract2.py
pract3.py
pract4.py
pract5a.py
pract5b.py
pract5c.py
pract6a.py
pract6b.py
pt4.py
sonar.csv
OUTLINE
pract4.py 7
pt4.py
pract5a.py
abalone.data
iris.data
pract6a.py
housing.csv
pract6b.py
pract4.py > ...
# baseline model
def create_baseline():
# create model
model = Sequential()
model.add(Dense(60, input_dim=60, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
# Compile model
model.compile(loss='binary_crossentropy',
optimizer='adam', metrics=['accuracy'])
return model
# evaluate model with standardized dataset
estimator = KerasClassifier(
build_fn=create_baseline, epochs=100, batch_size=5, verbose=0)
kfold = StratifiedKFold(n_splits=10, shuffle=True)
results = cross_val_score(estimator, X, encoded_Y, cv=kfold)
print('\n')
print("7_Govind Saini \n")
print("Baseline: %.2f%% (%.2f%%)" % (results.mean()*100, results.std()*100))
PROBLEMS 16
OUTPUT
TERMINAL
DEBUG CONSOLE
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
Ln 20, Col 15 Spaces: 4 UTF-8 CRLF Python Prettier
22°C Haze
03:20
20-02-2022

Output :


File Edit Selection View Go Run Terminal Help
pract4.py - Deep Learning Practical - Visual Studio Code
DEEP LEARNING PRACTICAL
# baseline model
def create_baseline():
# create model
model = Sequential()
model.add(Dense(60, input_dim=60, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
# Compile model
model.compile(loss='binary_crossentropy',
optimizer='adam', metrics=['accuracy'])
return model
PROBLEMS 16
OUTPUT
TERMINAL
DEBUG CONSOLE
admin@DESKTOP-3B61I8O MINGW64 /c/govindworking/Deep Learning Practical
$ py pract4.py
C:\govindworking\Deep Learning Practical\pract4.py:36: DeprecationWarning: KerasClassifier is deprecated, use Sci-Keras (https://github.com/adriangb/scikeras) instead. See https://www.adriangb.com/scikeras/stable/migration.html for help migrating.
estimator = KerasClassifier(
2022-02-20 03:19:38.538497: E tensorflow/stream_executor/cuda/cuda_driver.cc:271] failed call to cuInit: CUDA_ERROR_UNKNOWN: unknown error
2022-02-20 03:19:38.554394: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:169] retrieving CUDA diagnostic information for host: DESKTOP-3B61I8O
2022-02-20 03:19:38.555782: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:176] hostname: DESKTOP-3B61I8O
2022-02-20 03:19:38.565229: I tensorflow/core/platform/cpu_feature_guard.cc:151] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (oneDNN) to use the following CPU instructions in performance-critical operations: AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
7_Govind Saini
Baseline: 82.24% (6.72%)
admin@DESKTOP-3B61I8O MINGW64 /c/govindworking/Deep Learning Practical
$
Ln 20, Col 15 Spaces: 4 UTF-8 CRLF Python Prettier
22°C Haze
03:25
20-02-2022

Practical No : 5



Name : Govind Saini
ROll No: 07
Sign : Govind

Date 2.8.1.2.....12020
Page.....

Practical - 5

5a)

Aim :-

Using deep feed forward network with two hidden layers for performing multi-class classification and predicting the class

Description :

problem Description :-

→ this dataset is well studied and is a good problem for practising on neural network because all of the 4 input variable are numeric and have the same scale in centimeters. Each instance
→ the properties of an observed flower measurement and the output variable is specific iris species.

Dataset Description :-

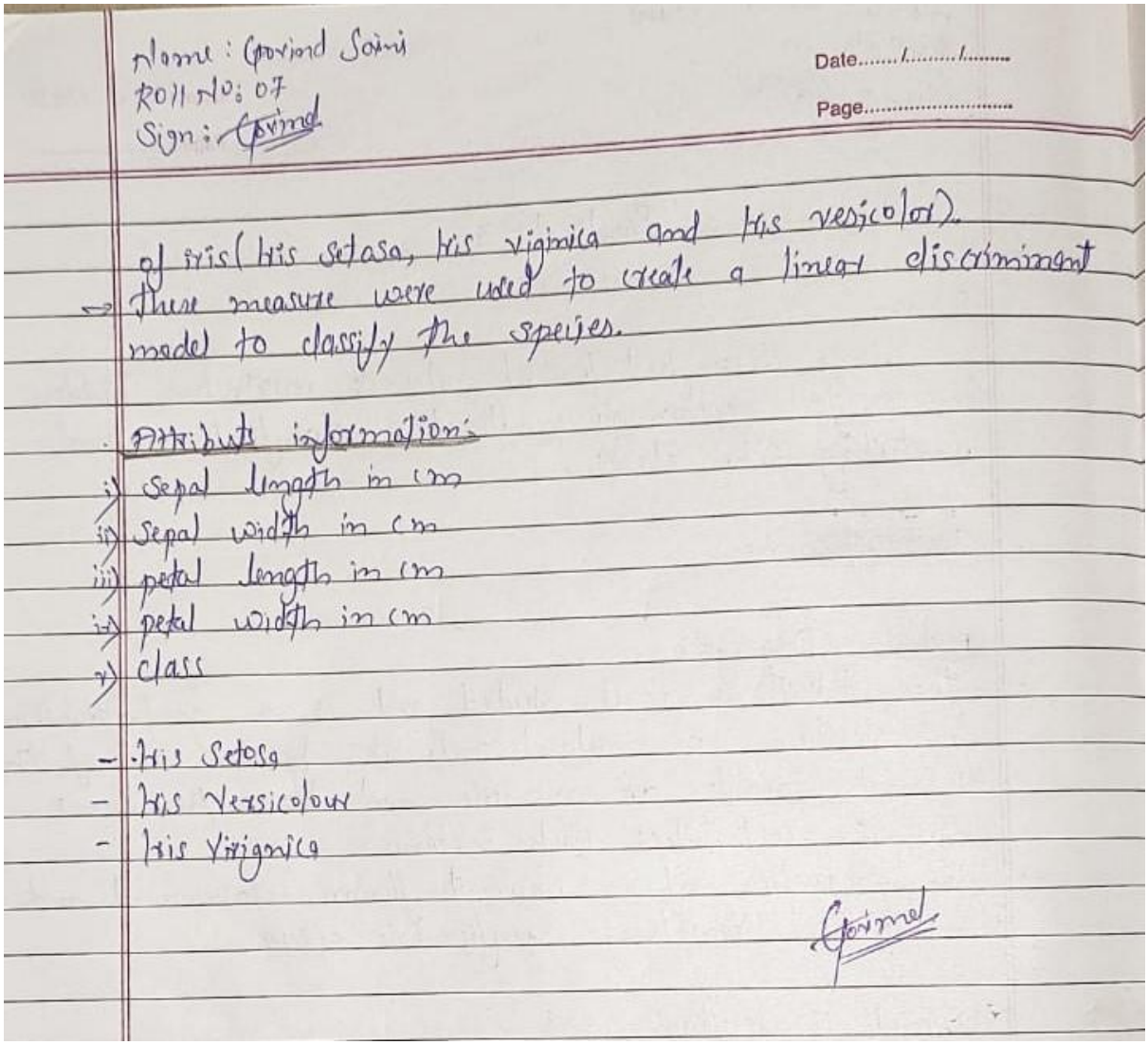
Title : iris flower dataset

→ The iris flower dataset is a well-studied problem and a such we can expect to achieve a model accuracy in the range of 95% to 97%.
→ This provide a good target to aim for when developing our models.

features :-

→ The Iris Dataset contains four features (length and width of sepals and petals) of 50 sample of three species

Govind



Name : Govind Saini
Roll No: 07
Sign : Govind

Date...../...../.....
Page.....

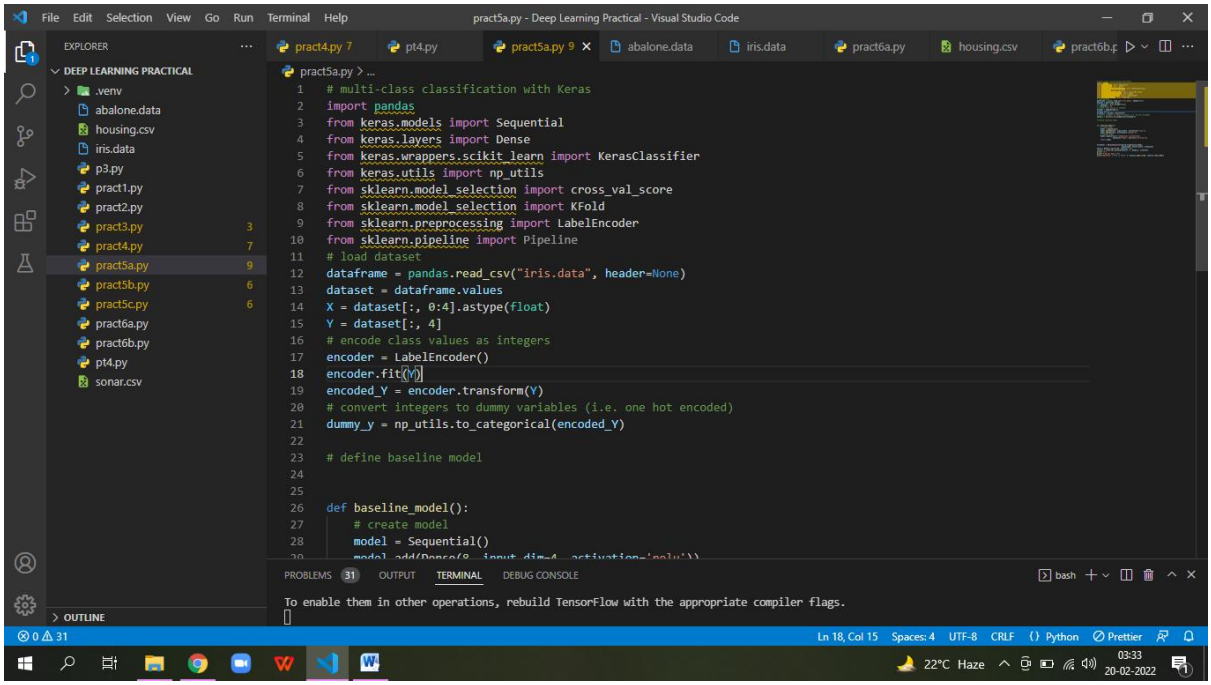
of iris (his setasa, his viginica and his vesicolor).
⇒ These measure were used to create a linear discriminant model to classify the species.

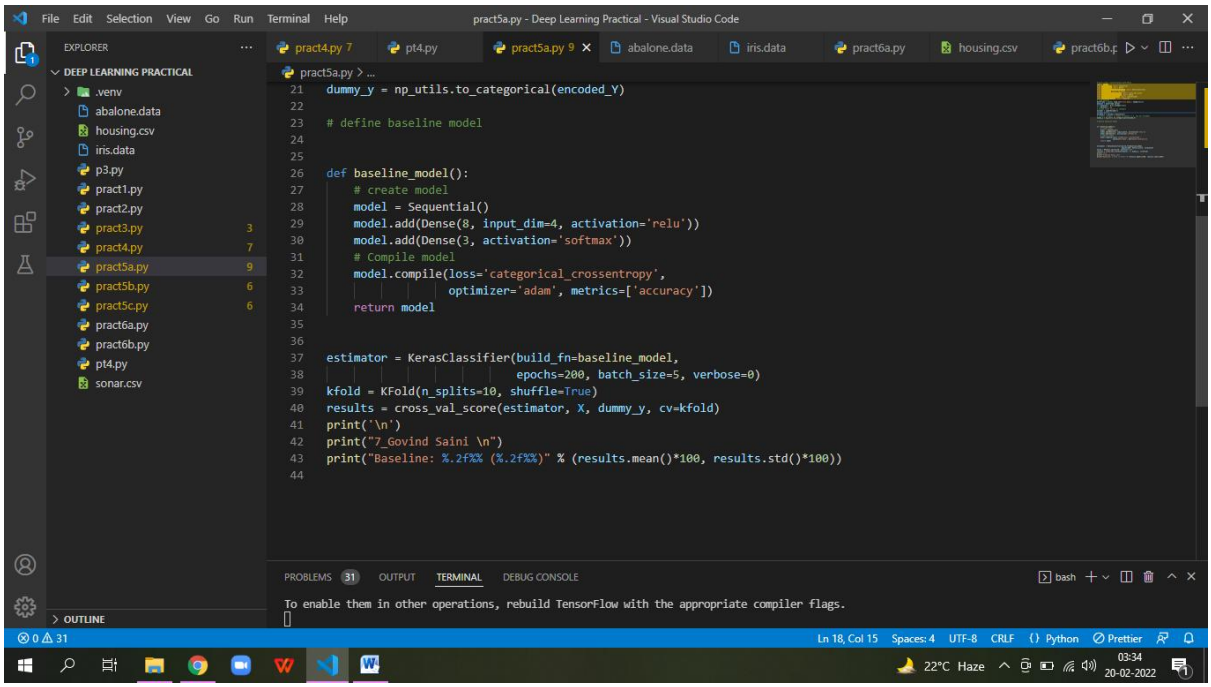
Attributs informations
i) Sepal length in cm
ii) Sepal width in cm
iii) petal length in cm
iv) petal width in cm
v) class

- His Setosa
- His Versicolour
- His Virignica

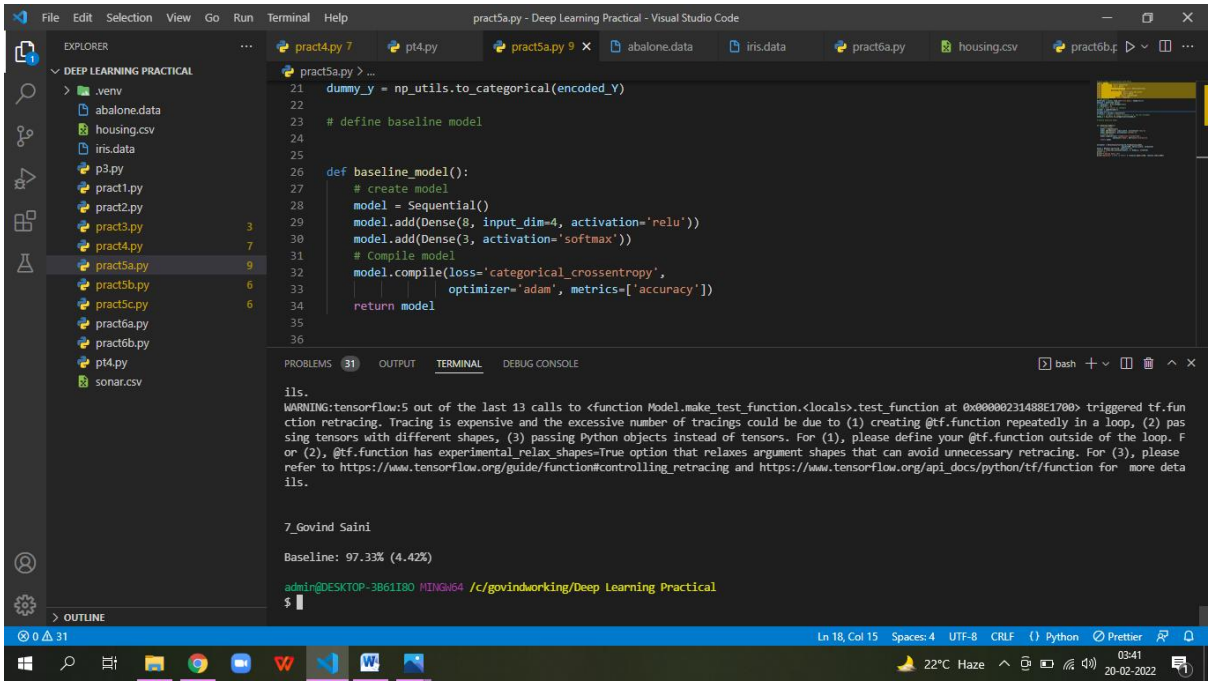
Govind

Code :


File Edit Selection View Go Run Terminal Help
pract5a.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
.venv
abalone.data
housing.csv
iris.data
p3.py
pract1.py
pract2.py
pract3.py
pract4.py
pract5a.py
pract5b.py
pract5c.py
pract6a.py
pract6b.py
pt4.py
sonar.csv
pract4.py 7
pt4.py
pract5a.py 9
abalone.data
iris.data
pract6a.py
housing.csv
pract5a.py > ...
# multi-class classification with Keras
import pandas
from keras.models import Sequential
from keras.layers import Dense
from keras.wrappers.scikit_learn import KerasClassifier
from keras.utils import np_utils
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import KFold
from sklearn.preprocessing import LabelEncoder
from sklearn.pipeline import Pipeline
# load dataset
dataframe = pandas.read_csv("iris.data", header=None)
dataset = dataframe.values
X = dataset[:, 0:4].astype(float)
Y = dataset[:, 4]
# encode class values as integers
encoder = LabelEncoder()
encoder.fit(Y)
encoded_Y = encoder.transform(Y)
# convert integers to dummy variables (i.e. one hot encoded)
dummy_y = np_utils.to_categorical(encoded_Y)
# define baseline model
def baseline_model():
# create model
model = Sequential()
PROBLEMS 31 OUTPUT TERMINAL DEBUG CONSOLE
bash
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
OUTLINE
Ln 18, Col 15 Spaces: 4 UTF-8 CRLF Python Prettier
22°C Haze
03:33
20-02-2022


File Edit Selection View Go Run Terminal Help
pract5a.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
.venv
abalone.data
housing.csv
iris.data
p3.py
pract1.py
pract2.py
pract3.py
pract4.py
pract5a.py
pract5b.py
pract5c.py
pract6a.py
pract6b.py
pt4.py
sonar.csv
pract4.py 7
pt4.py
pract5a.py 9
abalone.data
iris.data
pract6a.py
housing.csv
pract5a.py > ...
dummy_y = np_utils.to_categorical(encoded_Y)
# define baseline model
def baseline_model():
# create model
model = Sequential()
model.add(Dense(8, input_dim=4, activation='relu'))
model.add(Dense(3, activation='softmax'))
# Compile model
model.compile(loss='categorical_crossentropy',
optimizer='adam', metrics=['accuracy'])
return model
estimator = KerasClassifier(build_fn=baseline_model,
epochs=200, batch_size=5, verbose=0)
kfold = KFold(n_splits=10, shuffle=True)
results = cross_val_score(estimator, X, dummy_y, cv=kfold)
print('\n')
print("7_Govind Saini \n")
print("Baseline: %.2f%% (%.2f%%)" % (results.mean()*100, results.std()*100))
PROBLEMS 31 OUTPUT TERMINAL DEBUG CONSOLE
bash
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
OUTLINE
Ln 18, Col 15 Spaces: 4 UTF-8 CRLF Python Prettier
22°C Haze
03:34
20-02-2022

Output :


File Edit Selection View Go Run Terminal Help
pract5a.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
.venv
abalone.data
housing.csv
iris.data
p3.py
pract1.py
pract2.py
pract3.py 3
pract4.py 7
pract5a.py 9
pract5b.py 6
pract5c.py 6
pract6a.py
pract6b.py
pt4.py
sonar.csv
OUTLINE
pract4.py 7
pt4.py
pract5a.py 9
abalone.data
iris.data
pract6a.py
housing.csv
pract5a.py > ...
dummy_y = np_utils.to_categorical(encoded_Y)
# define baseline model
def baseline_model():
# create model
model = Sequential()
model.add(Dense(8, input_dim=4, activation='relu'))
model.add(Dense(3, activation='softmax'))
# Compile model
model.compile(loss='categorical_crossentropy',
optimizer='adam', metrics=['accuracy'])
return model
PROBLEMS 31
OUTPUT
TERMINAL
DEBUG CONSOLE
bash
ils.
WARNING:tensorflow:5 out of the last 13 calls to <function Model.make_test_function.<locals>.test_function at 0x00000231488E1700> triggered tf.function retracing. Tracing is expensive and the excessive number of tracings could be due to (1) creating @tf.function repeatedly in a loop, (2) passing tensors with different shapes, (3) passing Python objects instead of tensors. For (1), please define your @tf.function outside of the loop. For (2), @tf.function has experimental_relax_shapes=True option that relaxes argument shapes that can avoid unnecessary retracing. For (3), please refer to https://www.tensorflow.org/guide/function#controlling_retracing and https://www.tensorflow.org/api_docs/python/tf/function for more details.
7_Govind Saini
Baseline: 97.33% (4.42%)
admin@DESKTOP-3B61I8O MINGW64 /c/govindworking/Deep Learning Practical
$
Ln 18, Col 15 Spaces: 4 UTF-8 CRLF Python Prettier
22°C Haze
03:41
20-02-2022

Name: Govind Saini
Roll No: 07
Sign:- Govind

Date. 20.12.....1.2020
Page.....

5b)

Aim:-

Using a deep feed forward network with two hidden layer for performing classification and predicting the probability of class.

Description:-

Problem Description:-

When using neural network model such as regular deep feedforward net and convolution nets for classification task over some set of class label, one wonder whether if is possible to interpret the output

i.e. $y = [0.02, 0.05, 0.93]$

as the probability of some input being in a class equal to the respectively component value.

Algorithms (flowchart):-

One way to aggregate the result of each individual neural net model is to use a softmax as the ensemble output to give a final probability

In order to automatically determine the optimal weighting of the final softwave averaging.

Deriving the softmax function for multinomial classification problems starting from simple logistic regression

Using the softmax activation function in the output layer of a deep neural net to represent a categorical distribution over class label, and obtaining the probabilities of each input element belonging to a label.

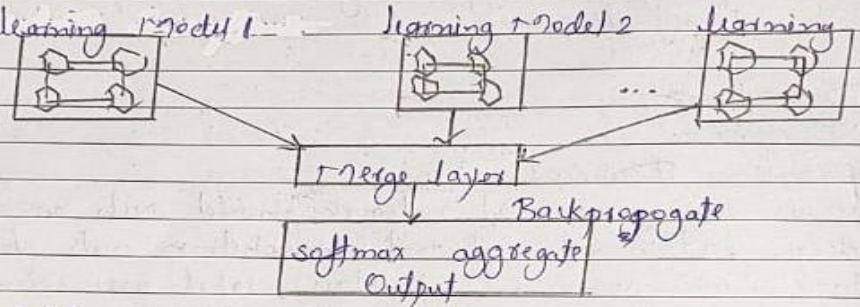
Govind

Archies

Name: Govind Saini
Roll No: 07
Sign: Govind

Date...../...../.....
Page.....

Building a robust ensemble neural net classifier with softmax output aggregation using the Keras functional API



Dataset Description:-

sklearn.datasets.load_wine

load and return the wine dataset (classification)

Now, new in version 0.18.

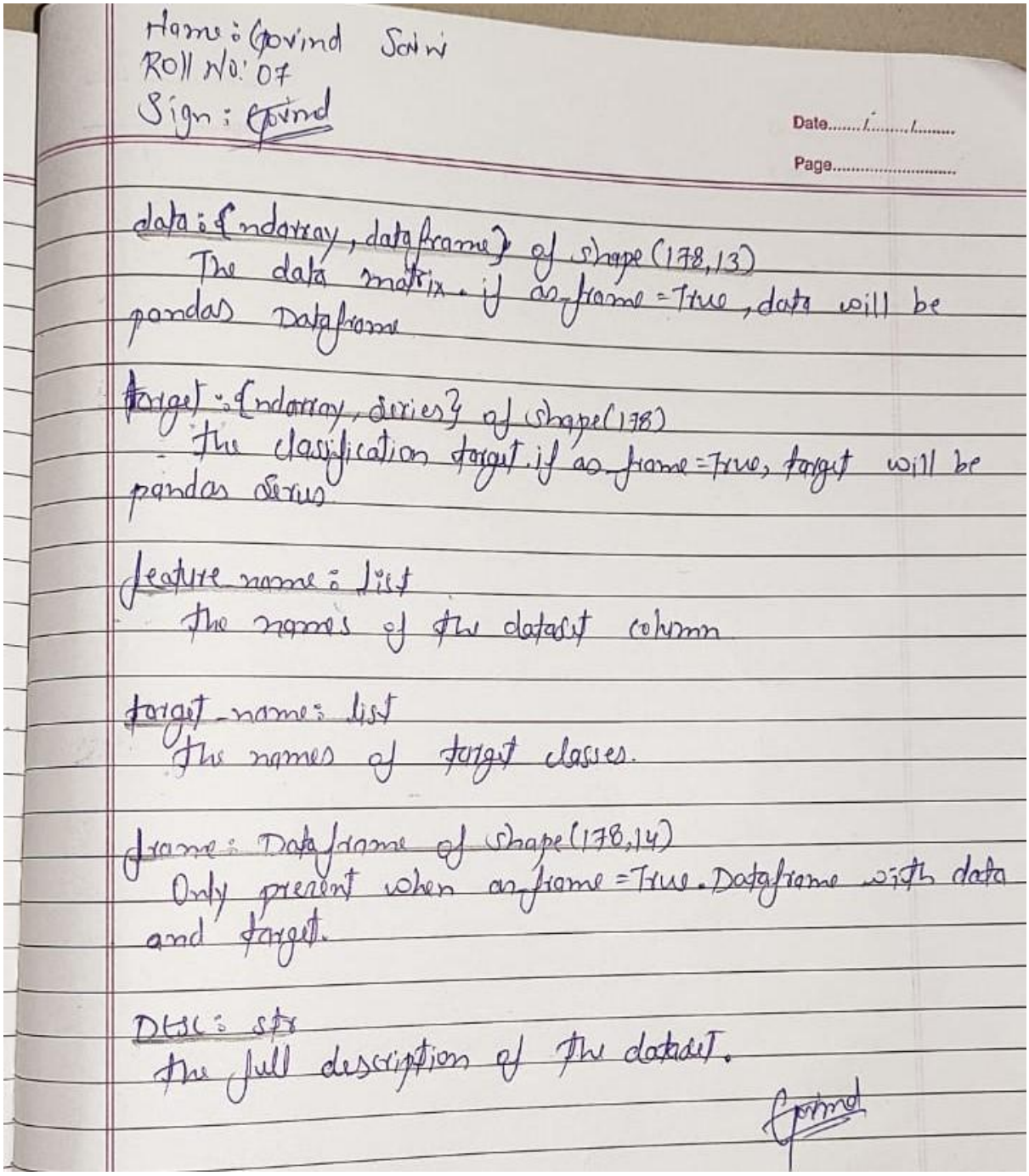
The wine dataset is a classic and very easy multi-class classification dataset

classes	3
Sample per class	[59, 71, 48]
Sample total	178
Dimensionality	13
features	real, positive

Data:

Dictionary-like Object, with the following attributes.

Govind



Name : Govind Saini
Roll No. 07
Sign : Govind

Date...../...../.....
Page.....

data : {ndarray, dataframe} of shape (178, 13)
The data matrix. if as_frame = True, data will be pandas Dataframe

target : {ndarray, series} of shape (178)
The classification target. if as_frame = True, target will be pandas Series

feature_names : list
The names of the dataset column

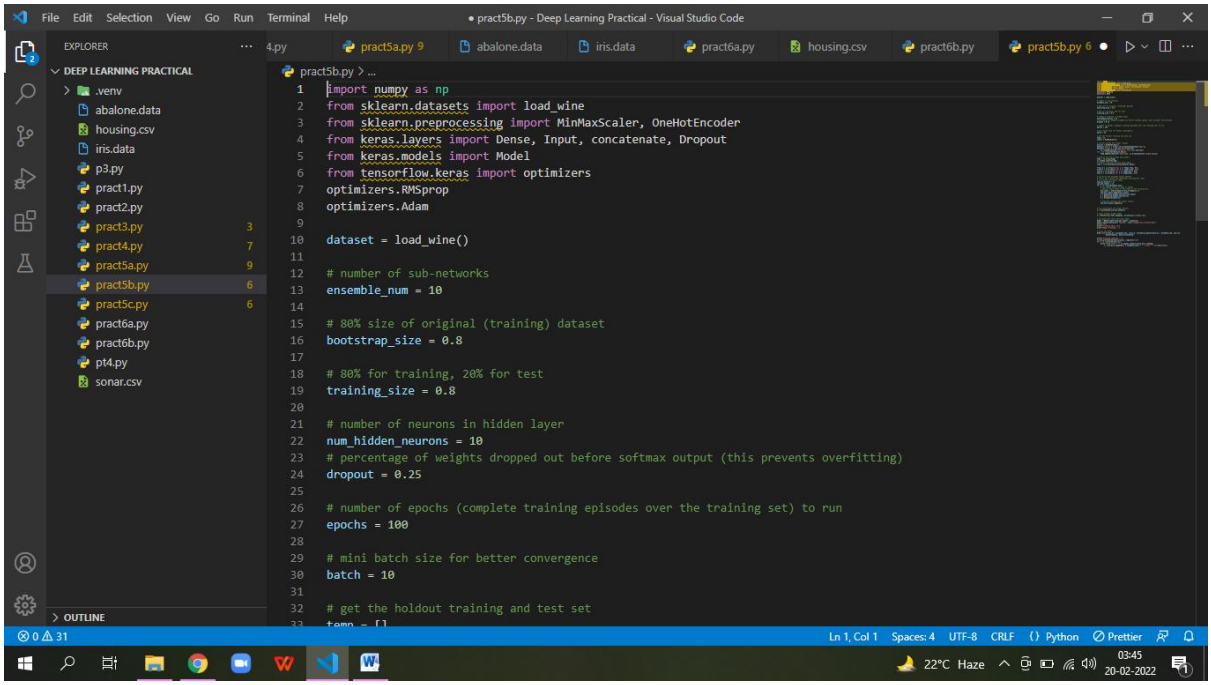
target_names : list
The names of target classes.

frame : Dataframe of shape (178, 14)
Only present when as_frame = True. Dataframe with data and target.

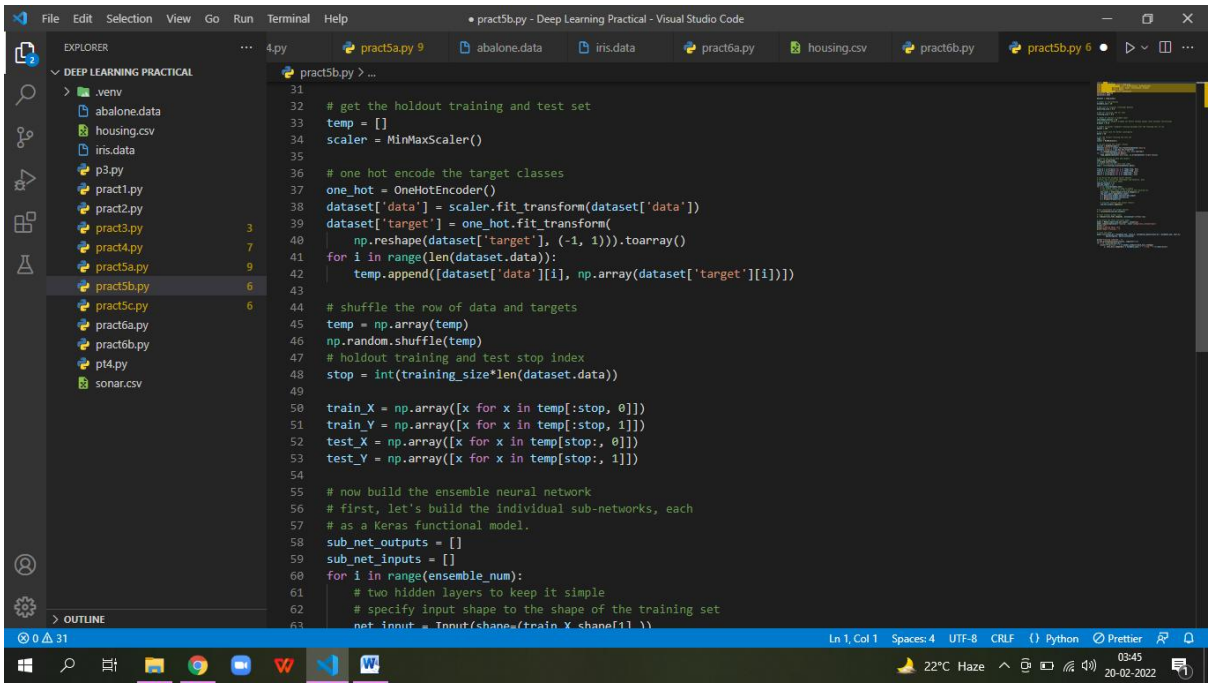
DESC : str
The full description of the dataset.

Govind

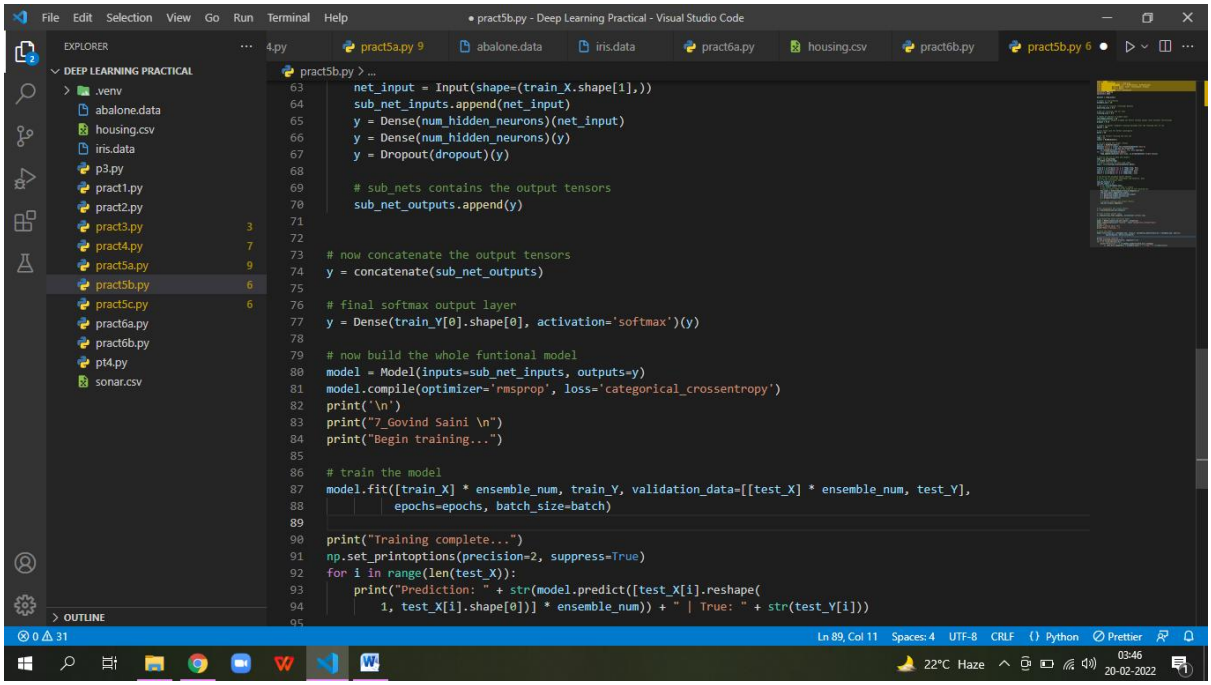
Code :



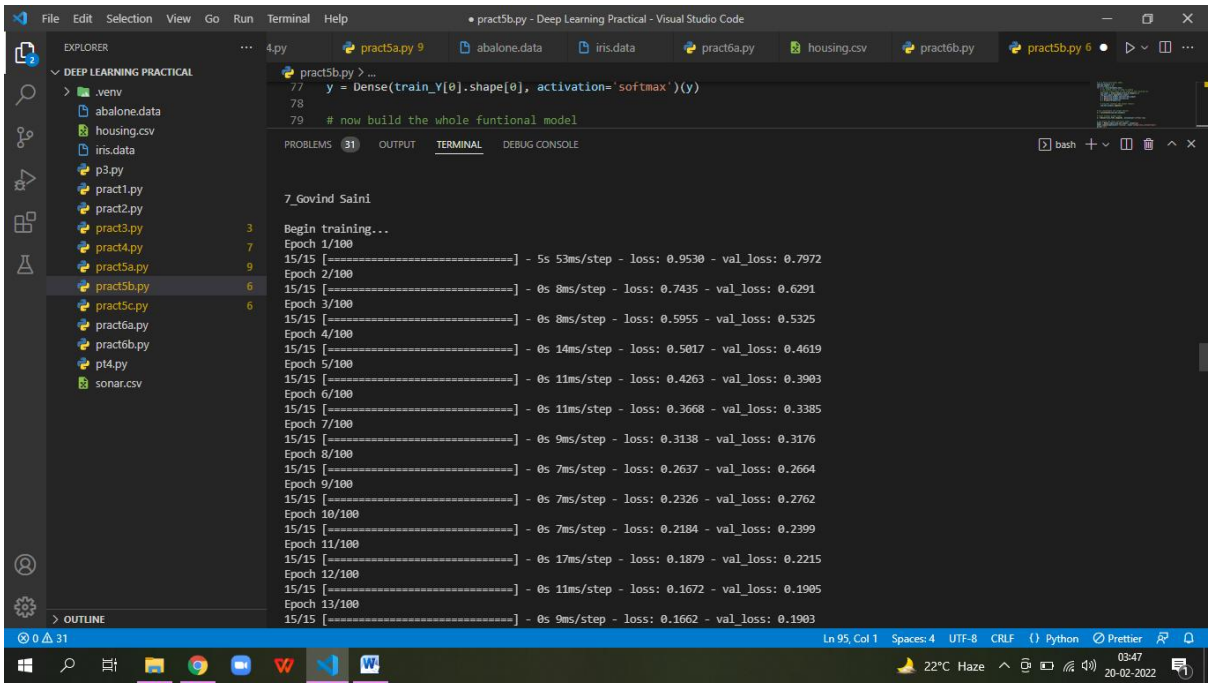
```python
import numpy as np
from sklearn.datasets import load_wine
from sklearn.preprocessing import MinMaxScaler, OneHotEncoder
from keras.layers import Dense, Input, concatenate, Dropout
from keras.models import Model
from tensorflow.keras import optimizers
optimizers.RMSprop
optimizers.Adam

dataset = load_wine()

# number of sub-networks
ensemble_num = 10

# 80% size of original (training) dataset
bootstrap_size = 0.8

# 80% for training, 20% for test
training_size = 0.8

# number of neurons in hidden layer
num_hidden_neurons = 10
# percentage of weights dropped out before softmax output (this prevents overfitting)
dropout = 0.25

# number of epochs (complete training episodes over the training set) to run
epochs = 100

# mini batch size for better convergence
batch = 10

# get the holdout training and test set
```

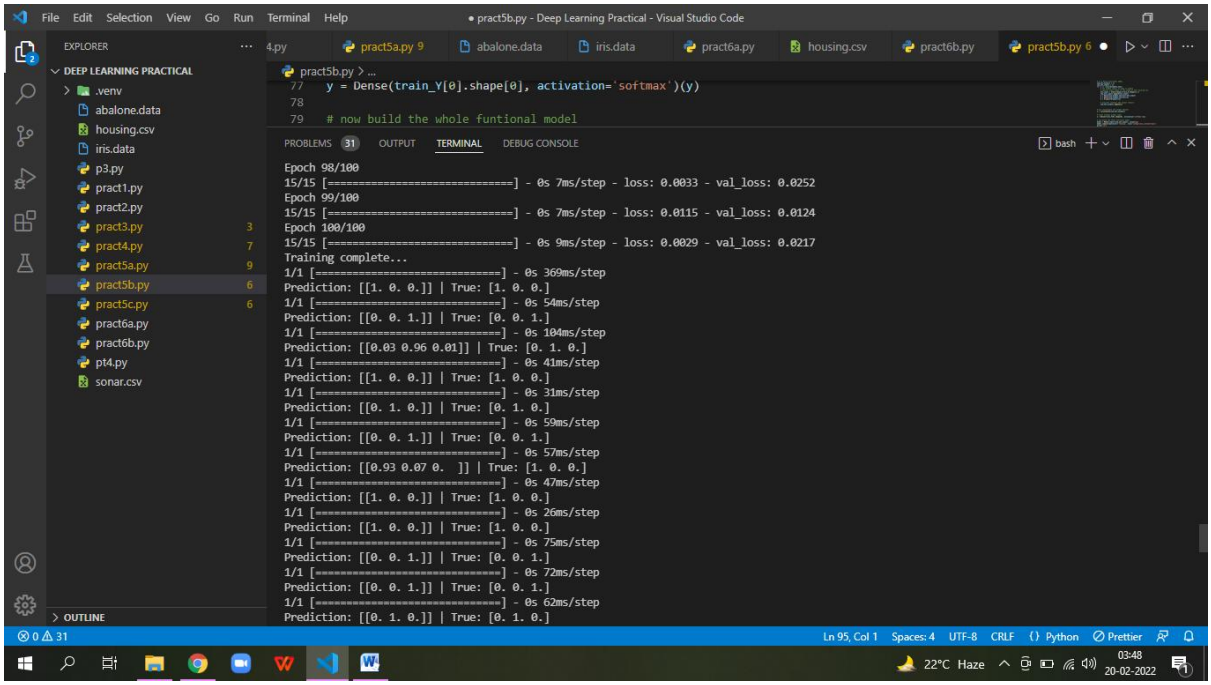


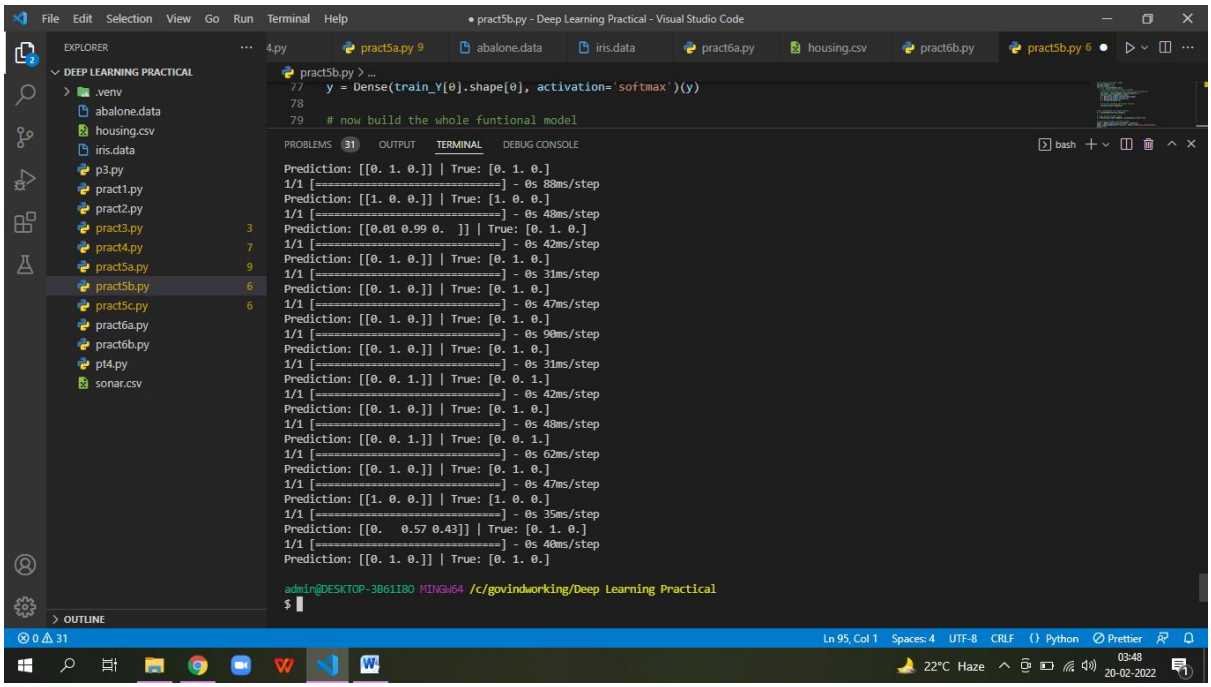
```python
# get the holdout training and test set
temp = []
scaler = MinMaxScaler()

# one hot encode the target classes
one_hot = OneHotEncoder()
dataset['data'] = scaler.fit_transform(dataset['data'])
dataset['target'] = one_hot.fit_transform(
    np.reshape(dataset['target'], (-1, 1))).toarray()
for i in range(len(dataset.data)):
    temp.append([dataset['data'][i], np.array(dataset['target'][i])])

# shuffle the row of data and targets
temp = np.array(temp)
np.random.shuffle(temp)
# holdout training and test stop index
stop = int(training_size*len(dataset.data))

train_X = np.array([x for x in temp[:stop, 0]])
train_Y = np.array([x for x in temp[:stop, 1]])
test_X = np.array([x for x in temp[stop:, 0]])
test_Y = np.array([x for x in temp[stop:, 1]])

# now build the ensemble neural network
# first, let's build the individual sub-networks, each
# as a Keras functional model.
sub_net_outputs = []
sub_net_inputs = []
for i in range(ensemble_num):
    # two hidden layers to keep it simple
    # specify input shape to the shape of the training set
    net_input = Input(shape=(train_X.shape[1],))
```


File Edit Selection View Go Run Terminal Help
• pract5b.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
.venv
abalone.data
housing.csv
iris.data
p3.py
pract1.py
pract2.py
pract3.py
pract4.py
pract5a.py
pract5b.py
pract5c.py
pract6a.py
pract6b.py
pt4.py
sonar.csv
OUTLINE
pract5a.py 9
abalone.data
iris.data
pract6a.py
housing.csv
pract6b.py
pract5b.py 6
pract5b.py > ...
net_input = Input(shape=(train_X.shape[1],))
sub_net_inputs.append(net_input)
y = Dense(num_hidden_neurons)(net_input)
y = Dense(num_hidden_neurons)(y)
y = Dropout(dropout)(y)
# sub_nets contains the output tensors
sub_net_outputs.append(y)
# now concatenate the output tensors
y = concatenate(sub_net_outputs)
# final softmax output layer
y = Dense(train_Y[0].shape[0], activation='softmax')(y)
# now build the whole funtional model
model = Model(inputs=sub_net_inputs, outputs=y)
model.compile(optimizer='rmsprop', loss='categorical_crossentropy')
print('\n')
print("7_Govind Saini \n")
print("Begin training...")
# train the model
model.fit([train_X] * ensemble_num, train_Y, validation_data=[[test_X] * ensemble_num, test_Y],
epochs=epochs, batch_size=batch)
print("Training complete...")
np.set_printoptions(precision=2, suppress=True)
for i in range(len(test_X)):
print("Prediction: " + str(model.predict([test_X[i].reshape(
1, test_X[i].shape[0])] * ensemble_num)) + " | True: " + str(test_Y[i]))
Ln 89, Col 11 Spaces: 4 UTF-8 CRLF Python Prettier
22°C Haze
03:46
20-02-2022

Output :


File Edit Selection View Go Run Terminal Help
• pract5b.py - Deep Learning Practical - Visual Studio Code
PROBLEMS 31 OUTPUT TERMINAL DEBUG CONSOLE
bash
7_Govind Saini
Begin training...
Epoch 1/100
15/15 [==============================] - 5s 53ms/step - loss: 0.9530 - val_loss: 0.7972
Epoch 2/100
15/15 [==============================] - 0s 8ms/step - loss: 0.7435 - val_loss: 0.6291
Epoch 3/100
15/15 [==============================] - 0s 8ms/step - loss: 0.5955 - val_loss: 0.5325
Epoch 4/100
15/15 [==============================] - 0s 14ms/step - loss: 0.5017 - val_loss: 0.4619
Epoch 5/100
15/15 [==============================] - 0s 11ms/step - loss: 0.4263 - val_loss: 0.3903
Epoch 6/100
15/15 [==============================] - 0s 11ms/step - loss: 0.3668 - val_loss: 0.3385
Epoch 7/100
15/15 [==============================] - 0s 9ms/step - loss: 0.3138 - val_loss: 0.3176
Epoch 8/100
15/15 [==============================] - 0s 7ms/step - loss: 0.2637 - val_loss: 0.2664
Epoch 9/100
15/15 [==============================] - 0s 7ms/step - loss: 0.2326 - val_loss: 0.2762
Epoch 10/100
15/15 [==============================] - 0s 7ms/step - loss: 0.2184 - val_loss: 0.2399
Epoch 11/100
15/15 [==============================] - 0s 17ms/step - loss: 0.1879 - val_loss: 0.2215
Epoch 12/100
15/15 [==============================] - 0s 11ms/step - loss: 0.1672 - val_loss: 0.1905
Epoch 13/100
15/15 [==============================] - 0s 9ms/step - loss: 0.1662 - val_loss: 0.1903
Ln 95, Col 1 Spaces: 4 UTF-8 CRLF Python Prettier
22°C Haze
03:47
20-02-2022


File Edit Selection View Go Run Terminal Help
• pract5b.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
.venv
abalone.data
housing.csv
iris.data
p3.py
pract1.py
pract2.py
pract3.py 3
pract4.py 7
pract5a.py 9
pract5b.py 6
pract5c.py 6
pract6a.py
pract6b.py
pt4.py
sonar.csv
OUTLINE
pract5a.py 9
abalone.data
iris.data
pract6a.py
housing.csv
pract6b.py
pract5b.py 6
pract5b.py > ...
77 y = Dense(train_Y[0].shape[0], activation='softmax')(y)
78
79 # now build the whole funtional model
PROBLEMS 31 OUTPUT TERMINAL DEBUG CONSOLE
bash
Epoch 98/100
15/15 [==========] - 0s 7ms/step - loss: 0.0033 - val_loss: 0.0252
Epoch 99/100
15/15 [==========] - 0s 7ms/step - loss: 0.0115 - val_loss: 0.0124
Epoch 100/100
15/15 [==========] - 0s 9ms/step - loss: 0.0029 - val_loss: 0.0217
Training complete...
1/1 [==========] - 0s 369ms/step
Prediction: [[1. 0. 0.]] | True: [1. 0. 0.]
1/1 [==========] - 0s 54ms/step
Prediction: [[0. 0. 1.]] | True: [0. 0. 1.]
1/1 [==========] - 0s 104ms/step
Prediction: [[0.03 0.96 0.01]] | True: [0. 1. 0.]
1/1 [==========] - 0s 41ms/step
Prediction: [[1. 0. 0.]] | True: [1. 0. 0.]
1/1 [==========] - 0s 31ms/step
Prediction: [[0. 1. 0.]] | True: [0. 1. 0.]
1/1 [==========] - 0s 59ms/step
Prediction: [[0. 0. 1.]] | True: [0. 0. 1.]
1/1 [==========] - 0s 57ms/step
Prediction: [[0.93 0.07 0.]] | True: [1. 0. 0.]
1/1 [==========] - 0s 47ms/step
Prediction: [[1. 0. 0.]] | True: [1. 0. 0.]
1/1 [==========] - 0s 26ms/step
Prediction: [[1. 0. 0.]] | True: [1. 0. 0.]
1/1 [==========] - 0s 75ms/step
Prediction: [[0. 0. 1.]] | True: [0. 0. 1.]
1/1 [==========] - 0s 72ms/step
Prediction: [[0. 0. 1.]] | True: [0. 0. 1.]
1/1 [==========] - 0s 62ms/step
Prediction: [[0. 1. 0.]] | True: [0. 1. 0.]
⊗ 0 ⚠ 31
Ln 95, Col 1 Spaces: 4 UTF-8 CRLF Python Prettier
22°C Haze
03:48
20-02-2022


File Edit Selection View Go Run Terminal Help
• pract5b.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
.venv
abalone.data
housing.csv
iris.data
p3.py
pract1.py
pract2.py
pract3.py 3
pract4.py 7
pract5a.py 9
pract5b.py 6
pract5c.py 6
pract6a.py
pract6b.py
pt4.py
sonar.csv
OUTLINE
pract5b.py > ...
77 y = Dense(train_Y[0].shape[0], activation='softmax')(y)
78
79 # now build the whole funtional model
PROBLEMS 31 OUTPUT TERMINAL DEBUG CONSOLE
bash
Prediction: [[0. 1. 0.]] | True: [0. 1. 0.]
1/1 [==========] - 0s 88ms/step
Prediction: [[1. 0. 0.]] | True: [1. 0. 0.]
1/1 [==========] - 0s 48ms/step
Prediction: [[0.01 0.99 0.]] | True: [0. 1. 0.]
1/1 [==========] - 0s 42ms/step
Prediction: [[0. 1. 0.]] | True: [0. 1. 0.]
1/1 [==========] - 0s 31ms/step
Prediction: [[0. 1. 0.]] | True: [0. 1. 0.]
1/1 [==========] - 0s 47ms/step
Prediction: [[0. 1. 0.]] | True: [0. 1. 0.]
1/1 [==========] - 0s 90ms/step
Prediction: [[0. 1. 0.]] | True: [0. 1. 0.]
1/1 [==========] - 0s 31ms/step
Prediction: [[0. 0. 1.]] | True: [0. 0. 1.]
1/1 [==========] - 0s 42ms/step
Prediction: [[0. 1. 0.]] | True: [0. 1. 0.]
1/1 [==========] - 0s 48ms/step
Prediction: [[0. 0. 1.]] | True: [0. 0. 1.]
1/1 [==========] - 0s 62ms/step
Prediction: [[0. 1. 0.]] | True: [0. 1. 0.]
1/1 [==========] - 0s 47ms/step
Prediction: [[1. 0. 0.]] | True: [1. 0. 0.]
1/1 [==========] - 0s 35ms/step
Prediction: [[0. 0.57 0.43]] | True: [0. 1. 0.]
1/1 [==========] - 0s 40ms/step
Prediction: [[0. 1. 0.]] | True: [0. 1. 0.]
admin@DESKTOP-3B61I8O MINGW64 /c/govindworking/Deep Learning Practical
$
⊗ 0 ⚠ 31
Ln 95, Col 1 Spaces: 4 UTF-8 CRLF Python Prettier
22°C Haze
03:48
20-02-2022

Name:- Govind Saini
Roll No:07
Sign: Govind

Date 20.1.2...1.2020
Page.....

5)

Aim:-
Using a deep feed forward network with two hidden layer for performing multiple regression and predicting values.

Description:-

Problem Description:-
→ The dataset describe 13 numerical properties of house in Boston suburbs and is concerned with modling the price of house in those suburbs in thousand of dollars.
→ As a input attribute include things like crime rate, proportion of nonretail because business acres, chemical concentrations and more.

Data Description:
Title : Boston house price dataset.

The Boston housing Dataset is a derived from information collected by US. Census Servile concerning housing in the area of Boston MA. the following describe the dataset columns.

CRIM - per capita crime rate by town
ZN - proportion of residential land zoned for lots over 25,000 sq.ft.
INDUS - proportion of non-retail business acres per town.

Govind

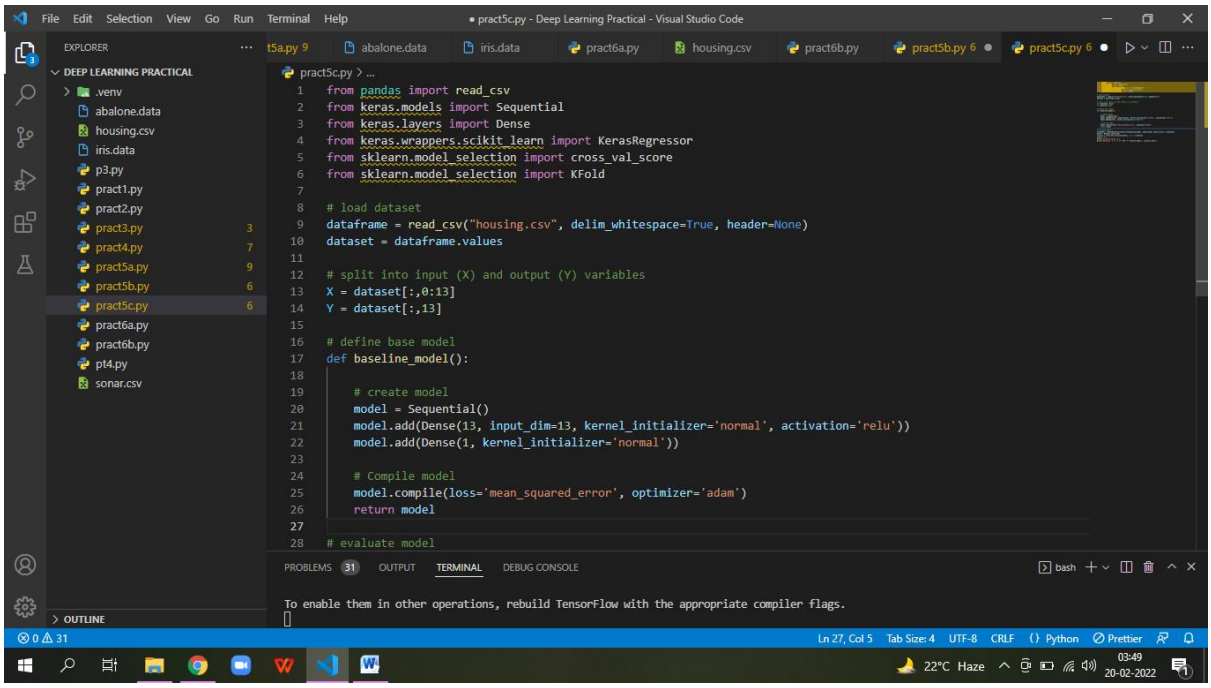
Name:- Govind Saini
ROll No:- 07
Sign:- Govind

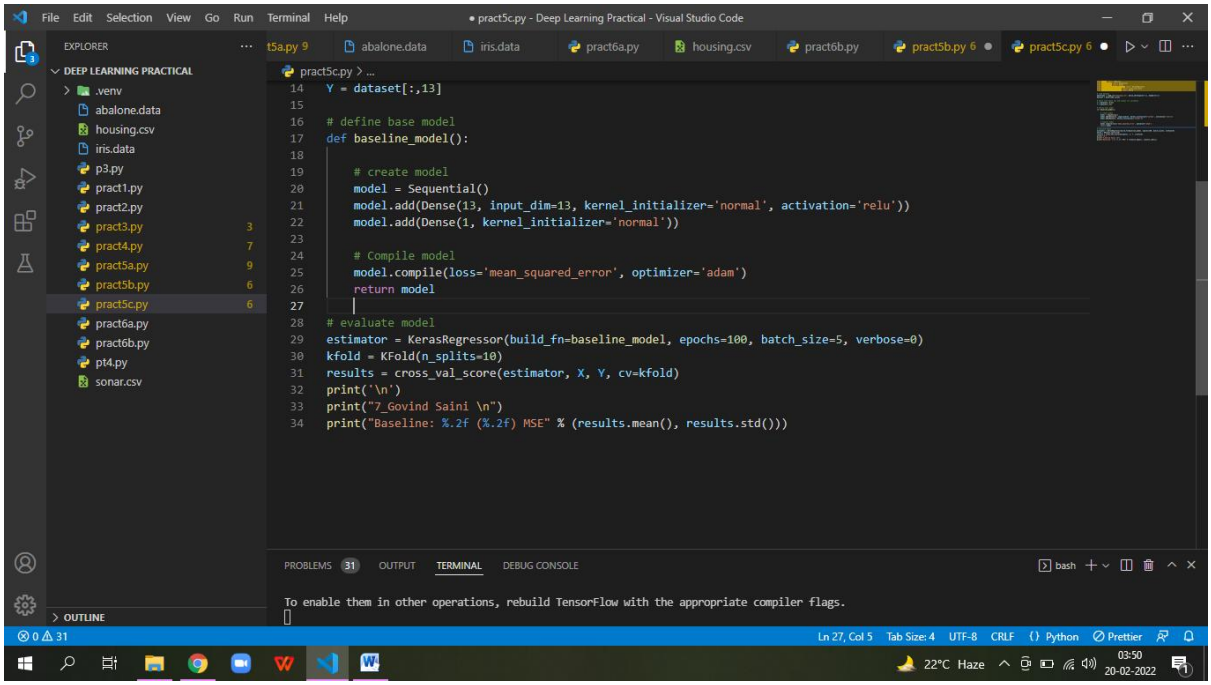
Date...../...../.....
Page.....

CHAS - charles River dummy Variable
Nox - nitric oxide concentration (parts per 10 million)
RM - average number of rooms per dwelling
AGE - proportion of owner-occupied units built prior to 1940
DIS - weighted distances to five Boston employee centers
RAD - index of accessibility to radial highways
TAX - full-value property rate per $10,000.
PTRATIO - pupil-teacher ratio by town
B - 1000(Bk - 0.63)^2 where Bk is the proportion of blak by town
LSTAT - % lower status of the population
MEDV - Median value of owner-occupied home in $1000's.

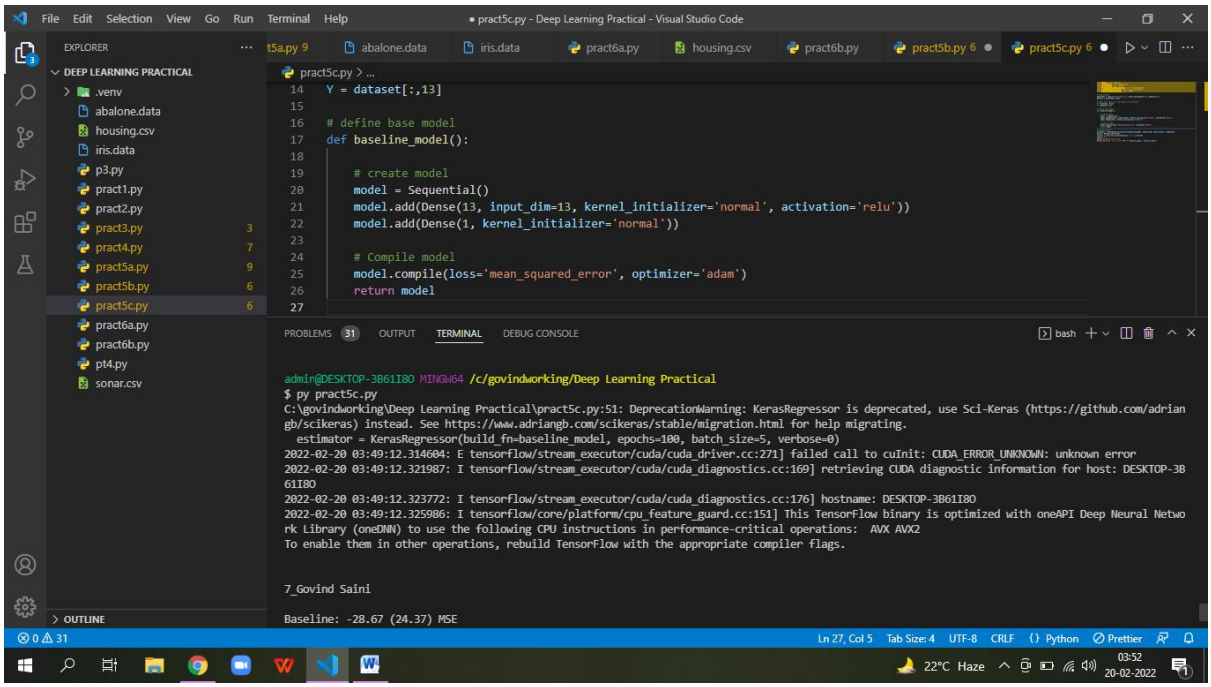
Govind

Code :




File Edit Selection View Go Run Terminal Help
• pract5c.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
.venv
abalone.data
housing.csv
iris.data
p3.py
pract1.py
pract2.py
pract3.py 3
pract4.py 7
pract5a.py 9
pract5b.py 6
pract5c.py 6
pract6a.py
pract6b.py
pt4.py
sonar.csv
t5a.py 9
abalone.data
iris.data
pract6a.py
housing.csv
pract6b.py
pract5b.py 6
pract5c.py 6
pract5c.py > ...
14 Y = dataset[:,13]
15
16 # define base model
17 def baseline_model():
18
19 # create model
20 model = Sequential()
21 model.add(Dense(13, input_dim=13, kernel_initializer='normal', activation='relu'))
22 model.add(Dense(1, kernel_initializer='normal'))
23
24 # Compile model
25 model.compile(loss='mean_squared_error', optimizer='adam')
26 return model
27
28 # evaluate model
29 estimator = KerasRegressor(build_fn=baseline_model, epochs=100, batch_size=5, verbose=0)
30 kfold = KFold(n_splits=10)
31 results = cross_val_score(estimator, X, Y, cv=kfold)
32 print('\n')
33 print("7_Govind Saini \n")
34 print("Baseline: %.2f (%.2f) MSE" % (results.mean(), results.std()))
PROBLEMS 31 OUTPUT TERMINAL DEBUG CONSOLE
bash
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
OUTLINE
Ln 27, Col 5 Tab Size: 4 UTF-8 CRLF Python Prettier
22°C Haze
03:50
20-02-2022

Output :


File Edit Selection View Go Run Terminal Help
• pract5c.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
.venv
abalone.data
housing.csv
iris.data
p3.py
pract1.py
pract2.py
pract3.py 3
pract4.py 7
pract5a.py 9
pract5b.py 6
pract5c.py 6
pract6a.py
pract6b.py
pt4.py
sonar.csv
t5a.py 9
abalone.data
iris.data
pract6a.py
housing.csv
pract6b.py
pract5b.py 6
pract5c.py 6
pract5c.py > ...
14 Y = dataset[:,13]
15
16 # define base model
17 def baseline_model():
18
19 # create model
20 model = Sequential()
21 model.add(Dense(13, input_dim=13, kernel_initializer='normal', activation='relu'))
22 model.add(Dense(1, kernel_initializer='normal'))
23
24 # Compile model
25 model.compile(loss='mean_squared_error', optimizer='adam')
26 return model
27
PROBLEMS 31 OUTPUT TERMINAL DEBUG CONSOLE
bash
admin@DESKTOP-3B61I8O MINGW64 /c/govindworking/Deep Learning Practical
$ py pract5c.py
C:\govindworking\Deep Learning Practical\pract5c.py:51: DeprecationWarning: KerasRegressor is deprecated, use Sci-Keras (https://github.com/adriangb/scikeras) instead. See https://www.adriangb.com/scikeras/stable/migration.html for help migrating.
estimator = KerasRegressor(build_fn=baseline_model, epochs=100, batch_size=5, verbose=0)
2022-02-20 03:49:12.314604: E tensorflow/stream_executor/cuda/cuda_driver.cc:271] failed call to cuInit: CUDA_ERROR_UNKNOWN: unknown error
2022-02-20 03:49:12.321987: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:169] retrieving CUDA diagnostic information for host: DESKTOP-3B61I8O
2022-02-20 03:49:12.323772: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:176] hostname: DESKTOP-3B61I8O
2022-02-20 03:49:12.325986: I tensorflow/core/platform/cpu_feature_guard.cc:151] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (oneDNN) to use the following CPU instructions in performance-critical operations: AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
7_Govind Saini
Baseline: -28.67 (24.37) MSE
OUTLINE
Ln 27, Col 5 Tab Size: 4 UTF-8 CRLF Python Prettier
22°C Haze
03:52
20-02-2022

Practical No : 6

Name: Govind Saini
Roll No: 07
Sign: Govind
Date...../...../.....
Page.....

Practical - 6

6a)

Aim:-
Evaluating feed forward deep Network for regression using k-fold cross validation.

Description:-

Problem Statement :
Unfortunately, there is no single method that works best for all kinds of problem statement. Often, a custom cross validation technique based on a feature, or combination of features, could be created if that gives the most stable cross validation score while making submission in hackathons.

Algorithm :-

- Pick a number of fold - k Usually, k is 5 or 10 but you can choose any number which is less than the dataset length
- Split the dataset into k equal (if possible) parts (they are called folds)
- Choose k-1 folds as the training set the remaining fold will be the test set
- Train the model on the training set. On each iteration of cross-validation, you must train a new model independently of the model trained on the iteration.

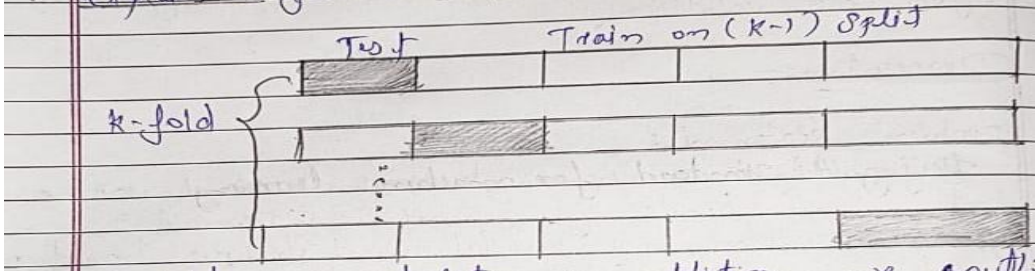
Govind

Name: Govind Saini
Roll No: 07
Sig: Govind

Date...../...../.....
Page.....

- Validate on the test set
- Save the result of the validation

Explain k-fold validation and its merit :



- k-fold cross-validation in addition, we outline an overview of the different matrix used to evaluate the chosen classifiers.
- In general, the criteria of comparison between algorithm can be done by measuring accuracy, speed, scalability, interpretability and robustness the ability to construct a model efficiency when the classifier is applied to the large set of data.

Dataset Description :-

Title :- Ames-iowa housing.csv

The Ames housing dataset examine features of houses sold in Ames during the 2006-10 time/ran
The goal is to use the traing data to predict

Govind

Archies

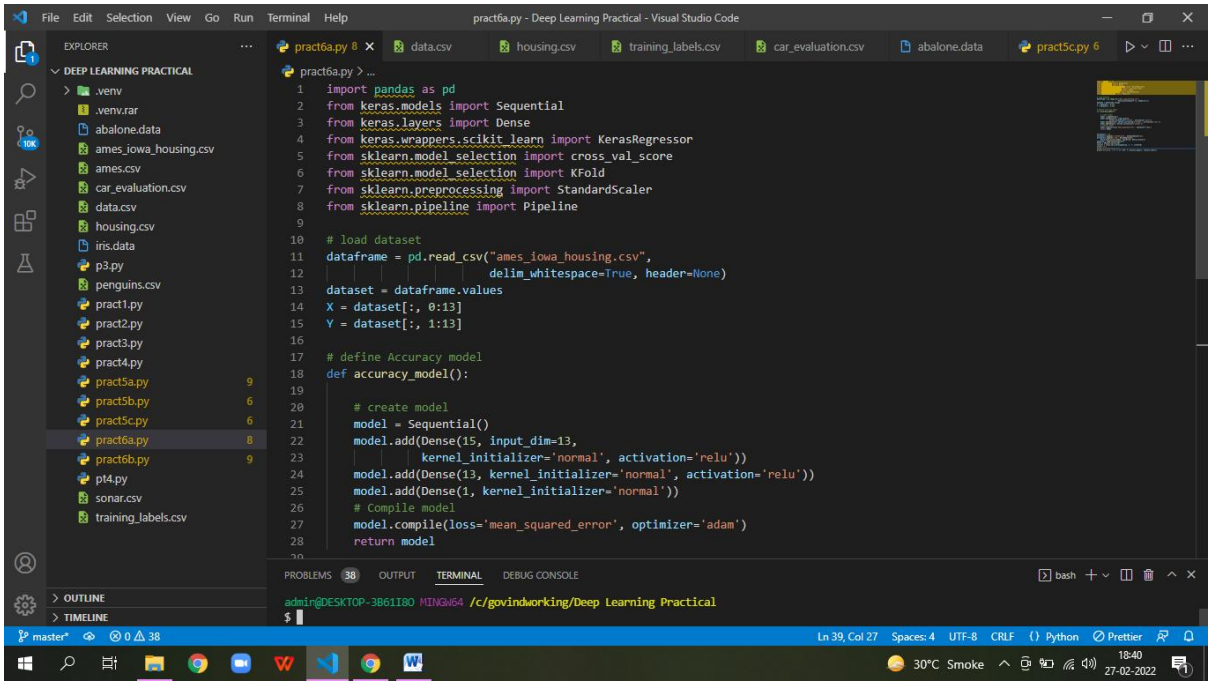
Name: Govind Saini
Roll No: 07
Sign: Govind

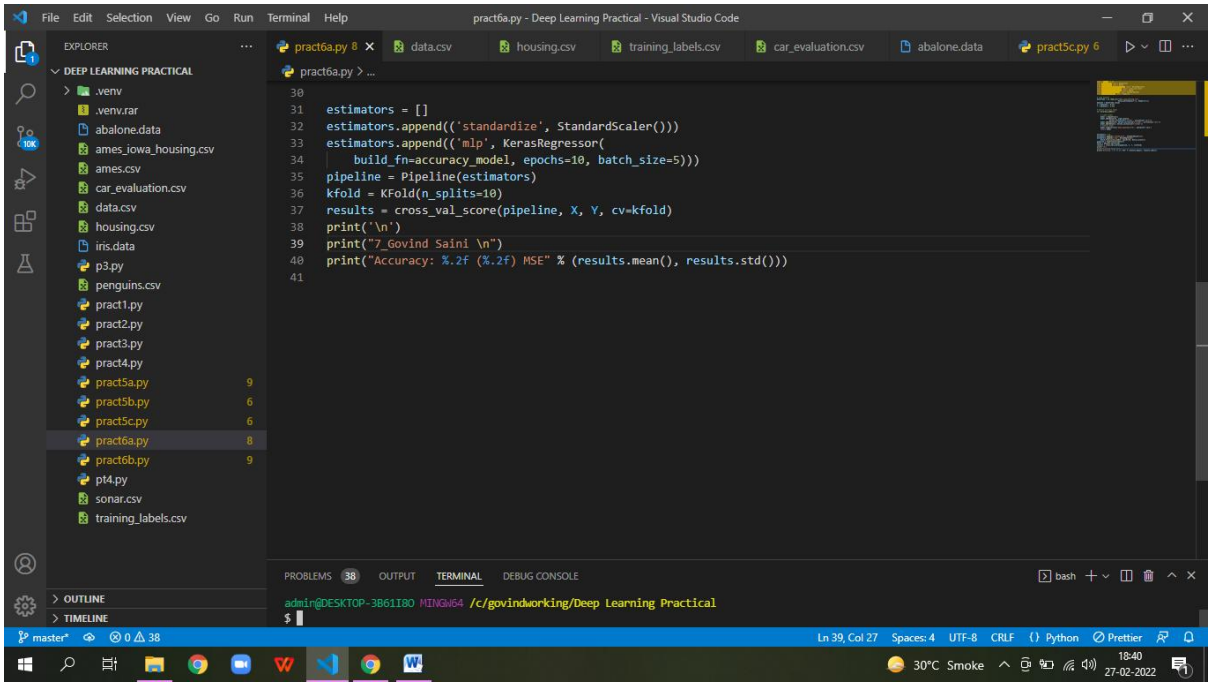
Date...../...../.....
Page.....

The sale prices of the house in the testing data.
The following describe the dataset columns
CRIM, ZN, INDUS, CHAS, NOR, RM, AGE, DIS, RAD, TAX,
PTRATIO, B, LSTAT, MEDV.

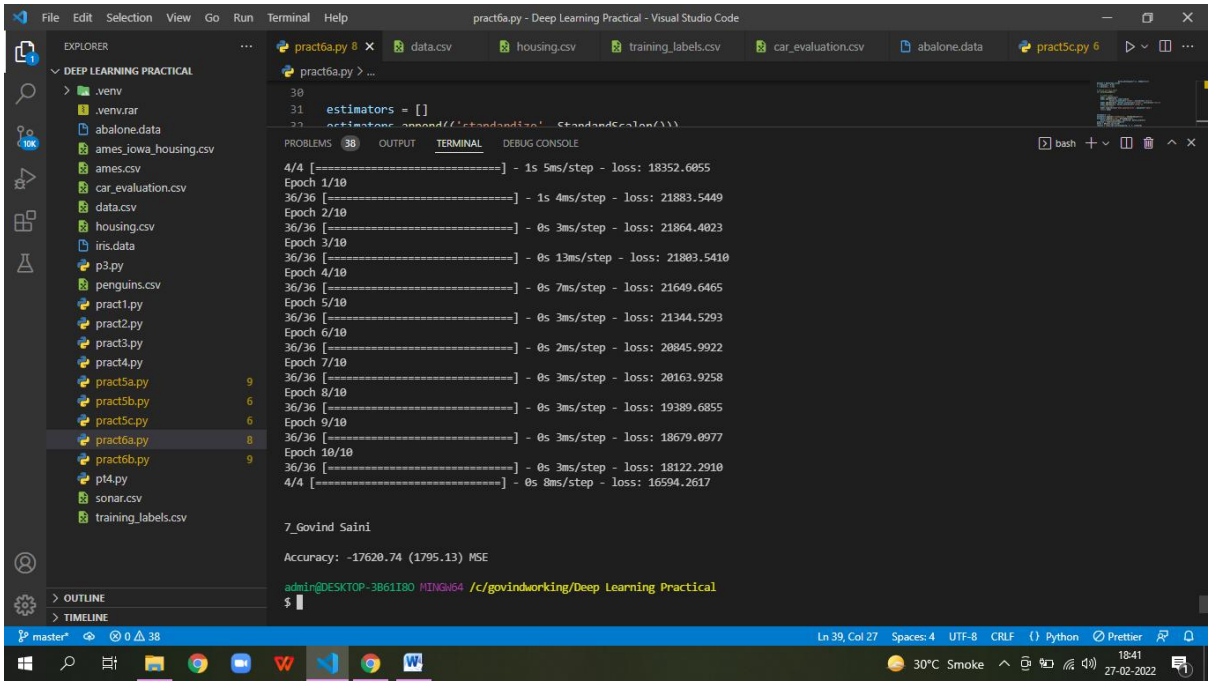
Govind

Code :


File Edit Selection View Go Run Terminal Help
pract6a.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
pract6a.py 8
data.csv
housing.csv
training_labels.csv
car_evaluation.csv
abalone.data
pract5c.py 6
import pandas as pd
from keras.models import Sequential
from keras.layers import Dense
from keras.wrappers.scikit_learn import KerasRegressor
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import KFold
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline
# load dataset
dataframe = pd.read_csv("ames_iowa_housing.csv",
delim_whitespace=True, header=None)
dataset = dataframe.values
X = dataset[:, 0:13]
Y = dataset[:, 1:13]
# define Accuracy model
def accuracy_model():
# create model
model = Sequential()
model.add(Dense(15, input_dim=13,
kernel_initializer='normal', activation='relu'))
model.add(Dense(13, kernel_initializer='normal', activation='relu'))
model.add(Dense(1, kernel_initializer='normal'))
# Compile model
model.compile(loss='mean_squared_error', optimizer='adam')
return model
PROBLEMS 38
OUTPUT
TERMINAL
DEBUG CONSOLE
admin@DESKTOP-3B61I8O MINGW64 /c/govindworking/Deep Learning Practical
$
OUTLINE
TIMELINE
master*
Ln 39, Col 27
Spaces: 4
UTF-8
CRLF
Python
Prettier
30°C Smoke
18:40
27-02-2022

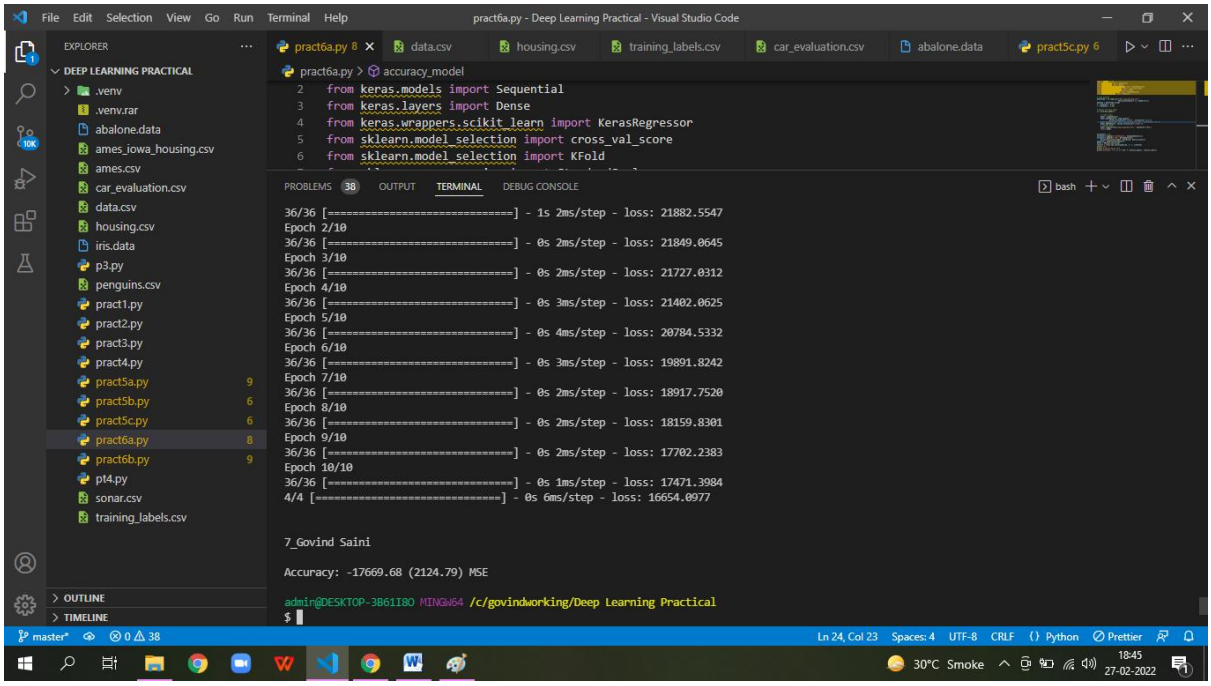

File Edit Selection View Go Run Terminal Help
pract6a.py - Deep Learning Practical - Visual Studio Code
estimators = []
estimators.append(('standardize', StandardScaler()))
estimators.append(('mlp', KerasRegressor(
build_fn=accuracy_model, epochs=10, batch_size=5)))
pipeline = Pipeline(estimators)
kfold = KFold(n_splits=10)
results = cross_val_score(pipeline, X, Y, cv=kfold)
print('\n')
print("7_Govind Saini \n")
print("Accuracy: %.2f (%.2f) MSE" % (results.mean(), results.std()))
admin@DESKTOP-3B61I8O MINGW64 /c/govindworking/Deep Learning Practical
$
30°C Smoke
18:40
27-02-2022

Output :



(After changing neuron)

```
20  # create model
21  model = Sequential()
22  model.add(Dense(15, input_dim=13,
23                  kernel_initializer='normal', activation='relu'))
24  model.add(Dense(20, kernel_initializer='normal', activation='relu'))
25  model.add(Dense(1, kernel_initializer='normal'))
26  # Compile model
27  model.compile(loss='mean_squared_error', optimizer='adam')
28  return model
29
```



Name: Govind Saini
Roll No: 07
Sign: Govind

Date...../...../.....
Page.....

6b) Aim:-
Evaluating feed forward deep network for multi-class classification using k-fold validation.

Description:-

problem Description:-
This is a multi-class classification problem, meaning that there are more than two classes to be predicted In fact there are three flower species.
This is an important type of problem on which to practice with neural network because the three class value require specialized handling.

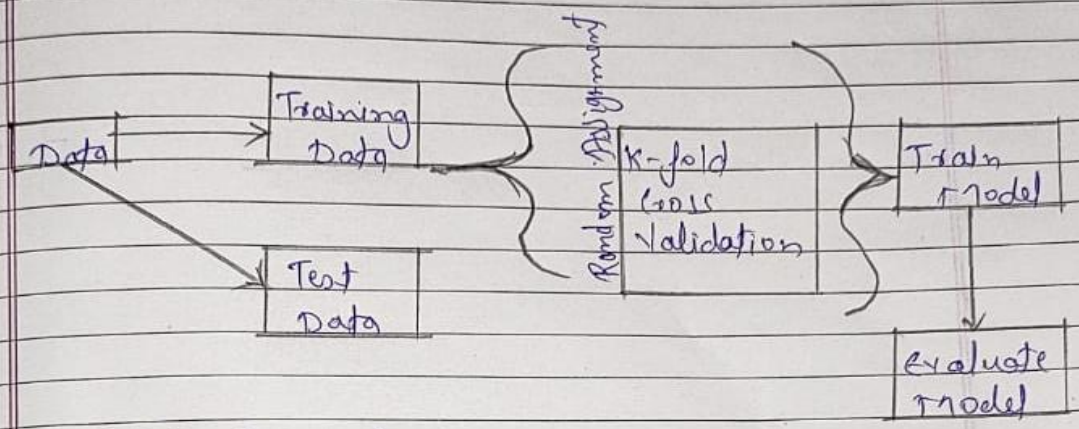
Let previous we three k-fold validation and this
→ K-fold Cross Validation:
It is type of predictive analytics and involve the following step

→ Original data are split randomly into k subsample (eg, k=5) which become the training data
→ model are estimated using k-1 subsample for each fold with the k^{th} subsample serving as the validation sample - process repeat until every subsample has served as the validation data and model result can be averaged across folds.

Govind

Name: Govind Saini
Roll No: 07
Sign: Govind

Date...../...../.....
Page.....

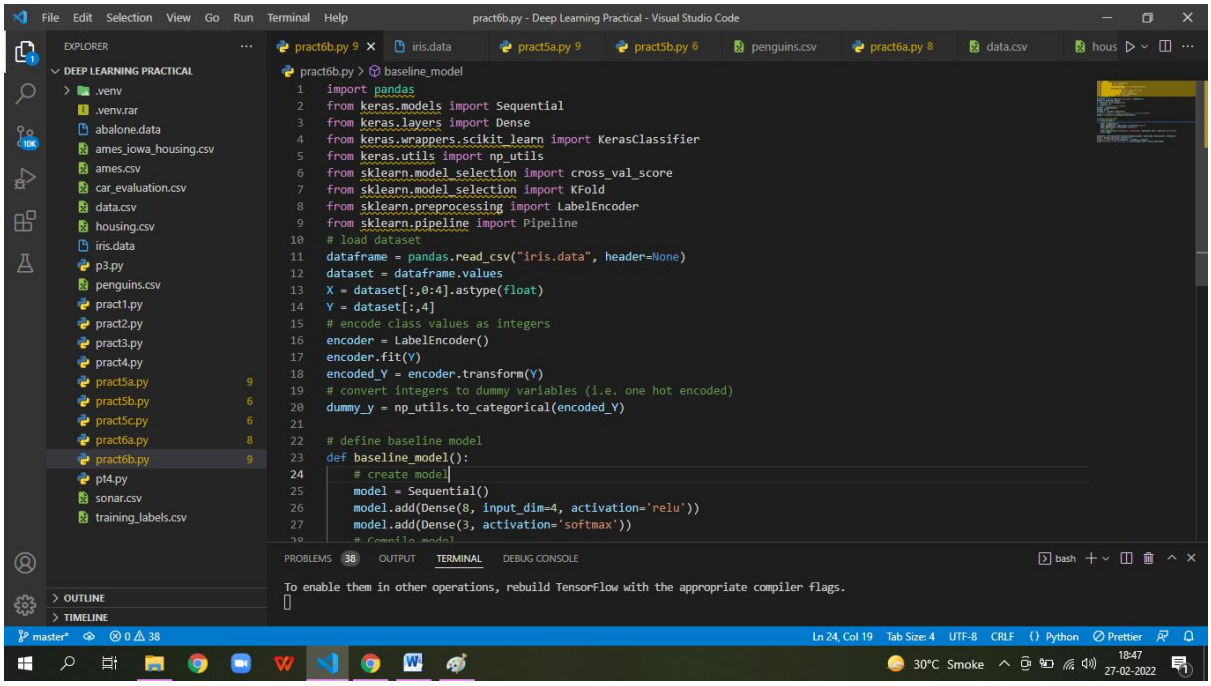


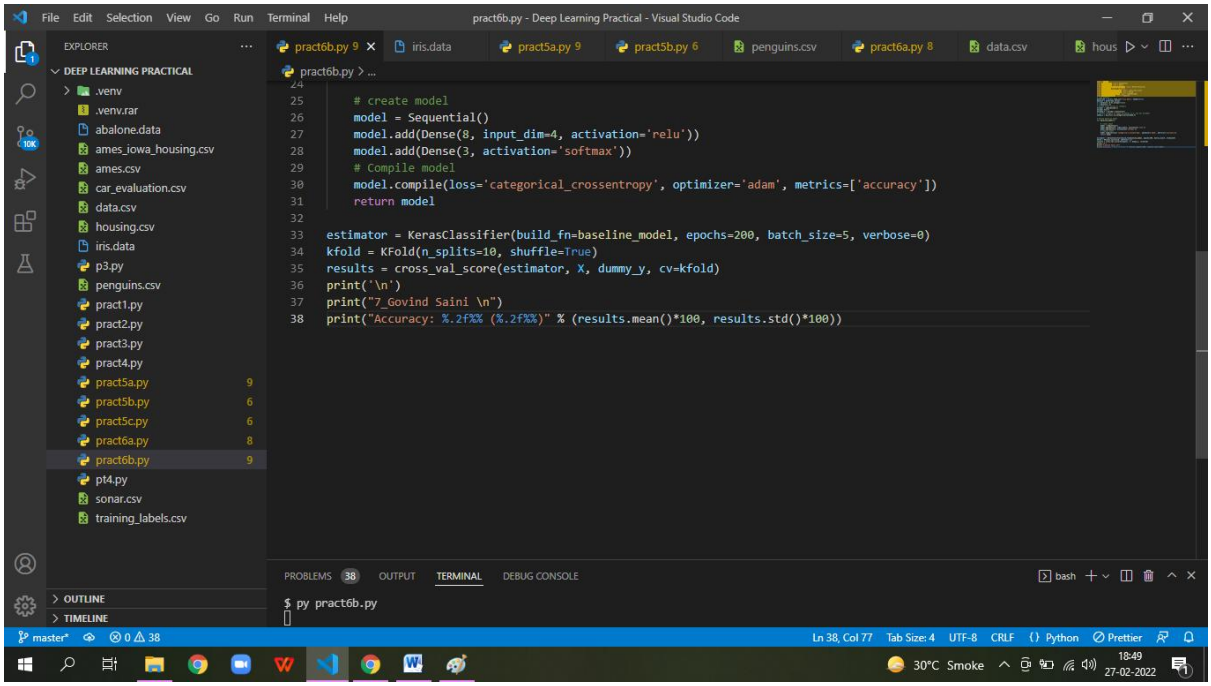
Dataset Description:-

- The iris flower dataset is a well-studied problem and a such we can expect to achieve a model accuracy in the range of 95% to 97%.
- This provide a good target to aim for when developing our model.
- The Iris Dataset contain four features of 50 sample of their species of Iris (Iris setosa, Iris viginic and Iris Versicolor).
- The measure were used to create a linear discriminant model to classify the species.

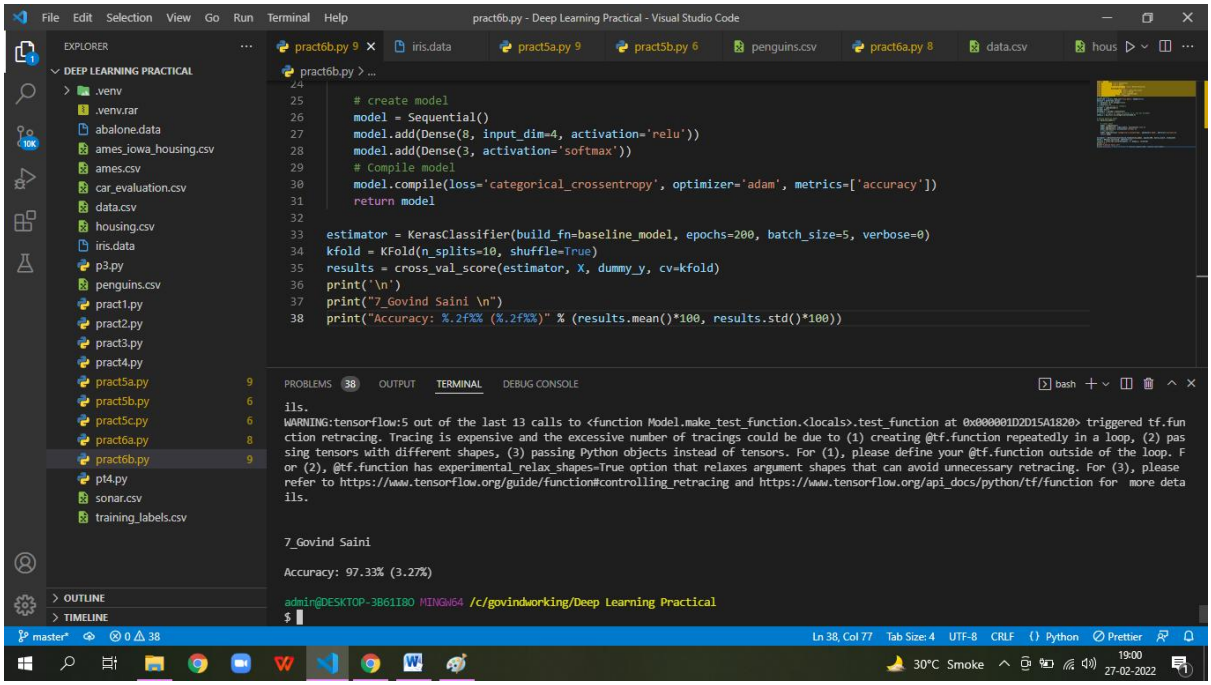
Govind

Code :


File Edit Selection View Go Run Terminal Help
pract6b.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
pract6b.py 9
iris.data
pract5a.py 9
pract5b.py 6
penguins.csv
pract6a.py 8
data.csv
pract6b.py > baseline_model
import pandas
from keras.models import Sequential
from keras.layers import Dense
from keras.wrappers.scikit_learn import KerasClassifier
from keras.utils import np_utils
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import KFold
from sklearn.preprocessing import LabelEncoder
from sklearn.pipeline import Pipeline
# load dataset
dataframe = pandas.read_csv("iris.data", header=None)
dataset = dataframe.values
X = dataset[:,0:4].astype(float)
Y = dataset[:,4]
# encode class values as integers
encoder = LabelEncoder()
encoder.fit(Y)
encoded_Y = encoder.transform(Y)
# convert integers to dummy variables (i.e. one hot encoded)
dummy_y = np_utils.to_categorical(encoded_Y)
# define baseline model
def baseline_model():
# create model
model = Sequential()
model.add(Dense(8, input_dim=4, activation='relu'))
model.add(Dense(3, activation='softmax'))
PROBLEMS 38 OUTPUT TERMINAL DEBUG CONSOLE
bash
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
OUTLINE
TIMELINE
master* 0 38
Ln 24, Col 19 Tab Size: 4 UTF-8 CRLF Python Prettier
30°C Smoke
18:47
27-02-2022


File Edit Selection View Go Run Terminal Help
pract6b.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
pract6b.py 9
iris.data
pract5a.py 9
pract5b.py 6
penguins.csv
pract6a.py 8
data.csv
pract6b.py > ...
# create model
model = Sequential()
model.add(Dense(8, input_dim=4, activation='relu'))
model.add(Dense(3, activation='softmax'))
# Compile model
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
return model
estimator = KerasClassifier(build_fn=baseline_model, epochs=200, batch_size=5, verbose=0)
kfold = KFold(n_splits=10, shuffle=True)
results = cross_val_score(estimator, X, dummy_y, cv=kfold)
print('\n')
print("7_Govind Saini \n")
print("Accuracy: %.2f%% (%.2f%%)" % (results.mean()*100, results.std()*100))
PROBLEMS 38 OUTPUT TERMINAL DEBUG CONSOLE
bash
$ py pract6b.py
OUTLINE
TIMELINE
master* 0 38
Ln 38, Col 77 Tab Size: 4 UTF-8 CRLF Python Prettier
30°C Smoke
18:49
27-02-2022

Output:


File Edit Selection View Go Run Terminal Help
pract6b.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
.venv
.venv.rar
abalone.data
ames_iowa_housing.csv
ames.csv
car_evaluation.csv
data.csv
housing.csv
iris.data
p3.py
penguins.csv
pract1.py
pract2.py
pract3.py
pract4.py
pract5a.py
pract5b.py
pract5c.py
pract6a.py
pract6b.py
pt4.py
sonar.csv
training_labels.csv
OUTLINE
TIMELINE
pract6b.py 9
iris.data
pract5a.py 9
pract5b.py 6
penguins.csv
pract6a.py 8
data.csv
pract6b.py > ...
# create model
model = Sequential()
model.add(Dense(8, input_dim=4, activation='relu'))
model.add(Dense(3, activation='softmax'))
# Compile model
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
return model
estimator = KerasClassifier(build_fn=baseline_model, epochs=200, batch_size=5, verbose=0)
kfold = KFold(n_splits=10, shuffle=True)
results = cross_val_score(estimator, X, dummy_y, cv=kfold)
print('\n')
print("7_Govind Saini \n")
print("Accuracy: %.2f%% (%.2f%%)" % (results.mean()*100, results.std()*100))
PROBLEMS 38
OUTPUT
TERMINAL
DEBUG CONSOLE
bash
ils.
WARNING:tensorflow:5 out of the last 13 calls to <function Model.make_test_function.<locals>.test_function at 0x000001D2D15A1820> triggered tf.function retracing. Tracing is expensive and the excessive number of tracings could be due to (1) creating @tf.function repeatedly in a loop, (2) passing tensors with different shapes, (3) passing Python objects instead of tensors. For (1), please define your @tf.function outside of the loop. For (2), @tf.function has experimental_relax_shapes=True option that relaxes argument shapes that can avoid unnecessary retracing. For (3), please refer to https://www.tensorflow.org/guide/function#controlling_retracing and https://www.tensorflow.org/api_docs/python/tf/function for more details.
7_Govind Saini
Accuracy: 97.33% (3.27%)
admin@DESKTOP-3B61I8O MINGW64 /c/govindworking/Deep Learning Practical
$
master*
Ln 38, Col 77
Tab Size: 4
UTF-8
CRLF
Python
Prettier
30°C Smoke
19:00
27-02-2022

Practical No : 7

Name: Govind Saini
Roll No: 07
Sign: Govind

Date: 5.1.3.1.2022
Page:

Practical - 7

7)

Aim:-

Implementation of convolution neural network to predict number from number images.

Description:-

- Convolution neural networks (CNN) - the concept behind recent breakthrough and developments in deep learning
- CNN have broken the mold and ascended the throne to become the state-of-the art computer vision technique.
- Among the different types of neural network long short term memory (LSTM), artificial neural network (ANN) etc., CNN are easily most popular.

Dataset Description:-

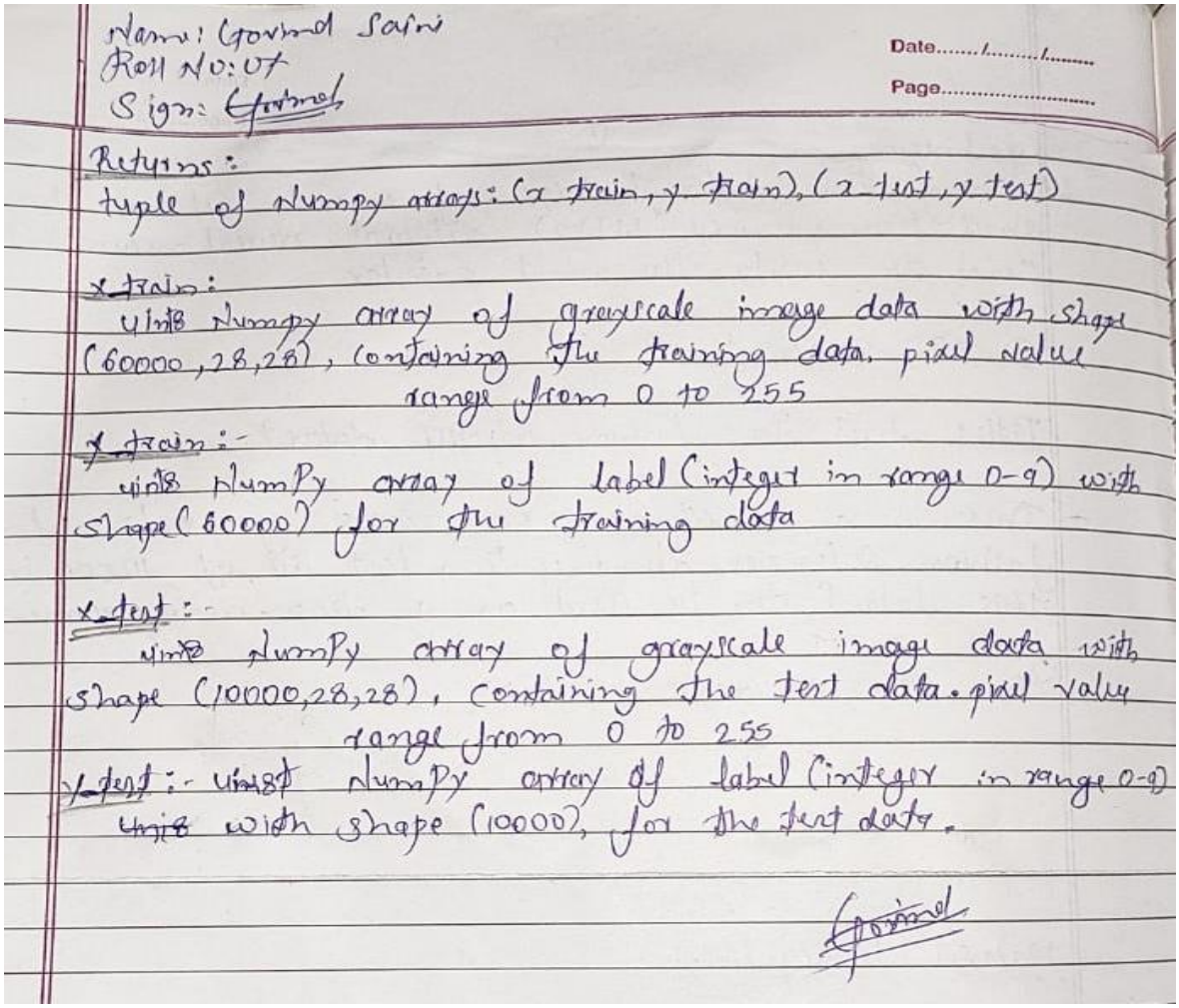
Title: ~~mini~~ mnist dataset

This is a dataset of 60,000 28x28 grayscales images of the 10 digits, along with a test ~~data~~ set of 10,000 images

Args	
path	path where to cache the dataset locally (relative to ~/.keras/datasets).

Govind

Orchies



Name: Govind Saini
Roll No: 07
Sign: Govind

Date...../...../.....
Page.....

Returns :
tuple of numpy arrays: (x_train, y_train), (x_test, y_test)

x_train :
uint8 NumPy array of grayscale image data with shape (60000, 28, 28), containing the training data. pixel value range from 0 to 255

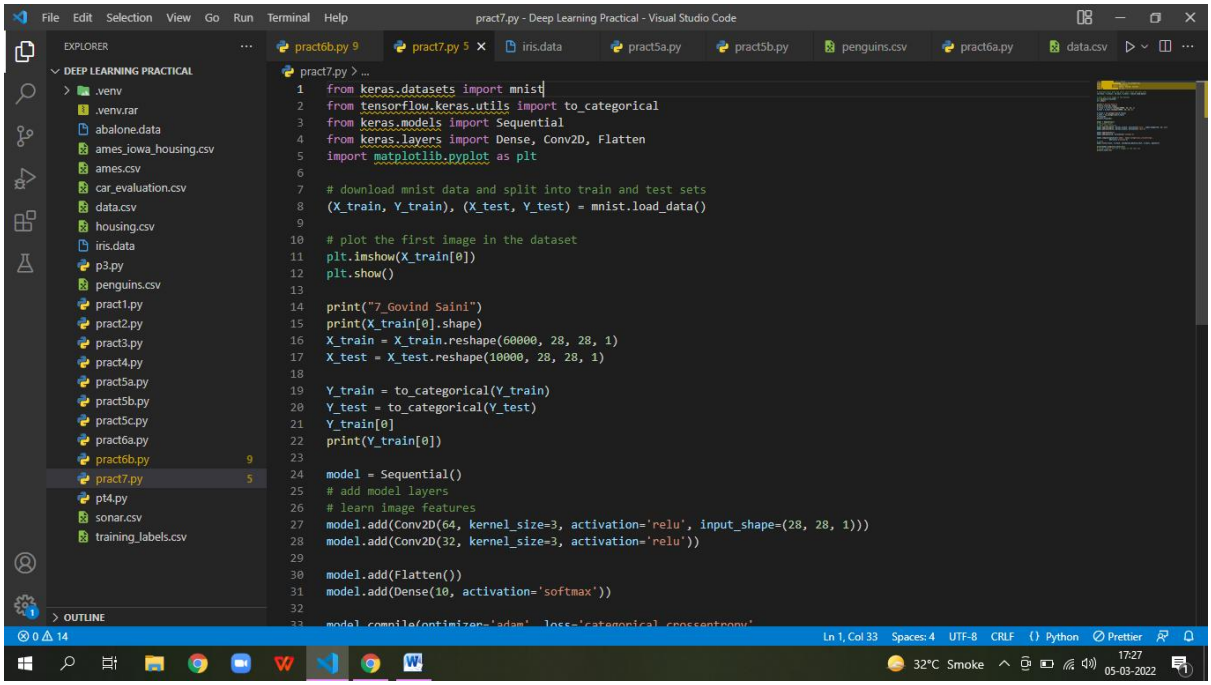
y_train :-
uint8 NumPy array of label (integer in range 0-9) with shape (60000) for the training data

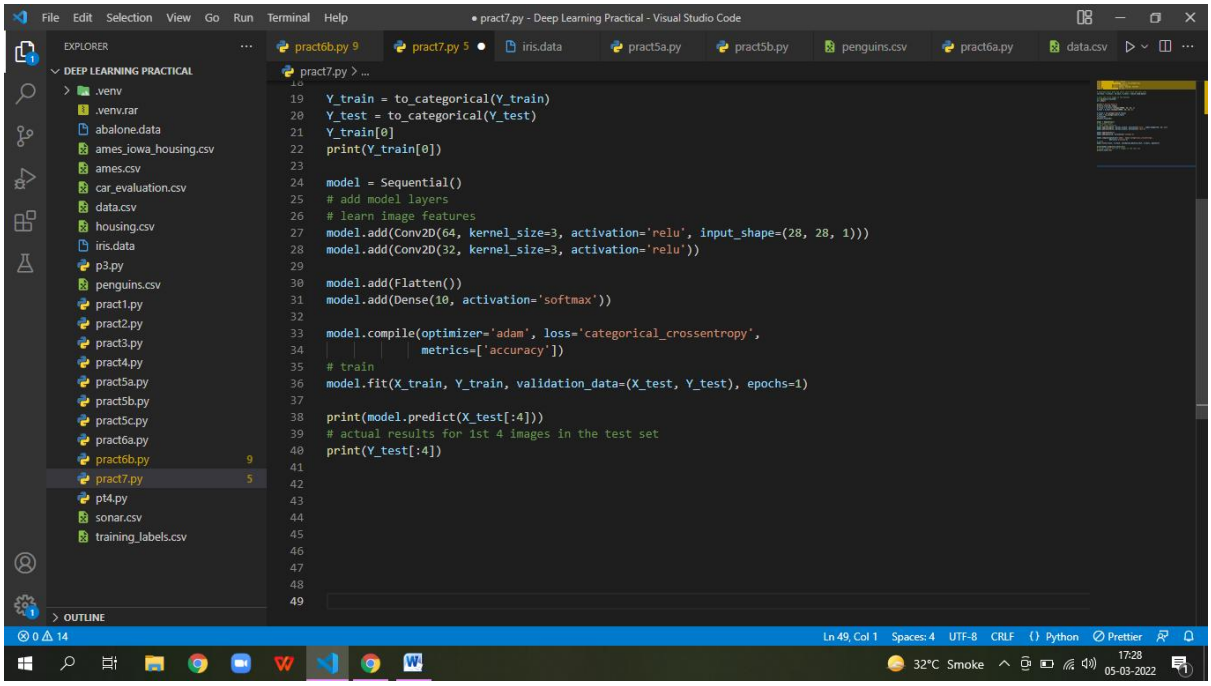
x_test :-
uint8 NumPy array of grayscale image data with shape (10000, 28, 28), containing the test data. pixel value range from 0 to 255

y_test :- uint8 NumPy array of label (integer in range 0-9) with shape (10000), for the test data.

Govind

Code :




File Edit Selection View Go Run Terminal Help
• pract7.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
.venv
.venv.rar
abalone.data
ames_iowa_housing.csv
ames.csv
car_evaluation.csv
data.csv
housing.csv
iris.data
p3.py
penguins.csv
pract1.py
pract2.py
pract3.py
pract4.py
pract5a.py
pract5b.py
pract5c.py
pract6a.py
pract6b.py 9
pract7.py 5
pt4.py
sonar.csv
training_labels.csv
OUTLINE
pract6b.py 9
pract7.py 5
iris.data
pract5a.py
pract5b.py
penguins.csv
pract6a.py
data.csv
pract7.py > ...
Y_train = to_categorical(Y_train)
Y_test = to_categorical(Y_test)
Y_train[0]
print(Y_train[0])
model = Sequential()
# add model layers
# learn image features
model.add(Conv2D(64, kernel_size=3, activation='relu', input_shape=(28, 28, 1)))
model.add(Conv2D(32, kernel_size=3, activation='relu'))
model.add(Flatten())
model.add(Dense(10, activation='softmax'))
model.compile(optimizer='adam', loss='categorical_crossentropy',
metrics=['accuracy'])
# train
model.fit(X_train, Y_train, validation_data=(X_test, Y_test), epochs=1)
print(model.predict(X_test[:4]))
# actual results for 1st 4 images in the test set
print(Y_test[:4])
Ln 49, Col 1 Spaces: 4 UTF-8 CRLF Python Prettier
32°C Smoke
17:28
05-03-2022

Output :

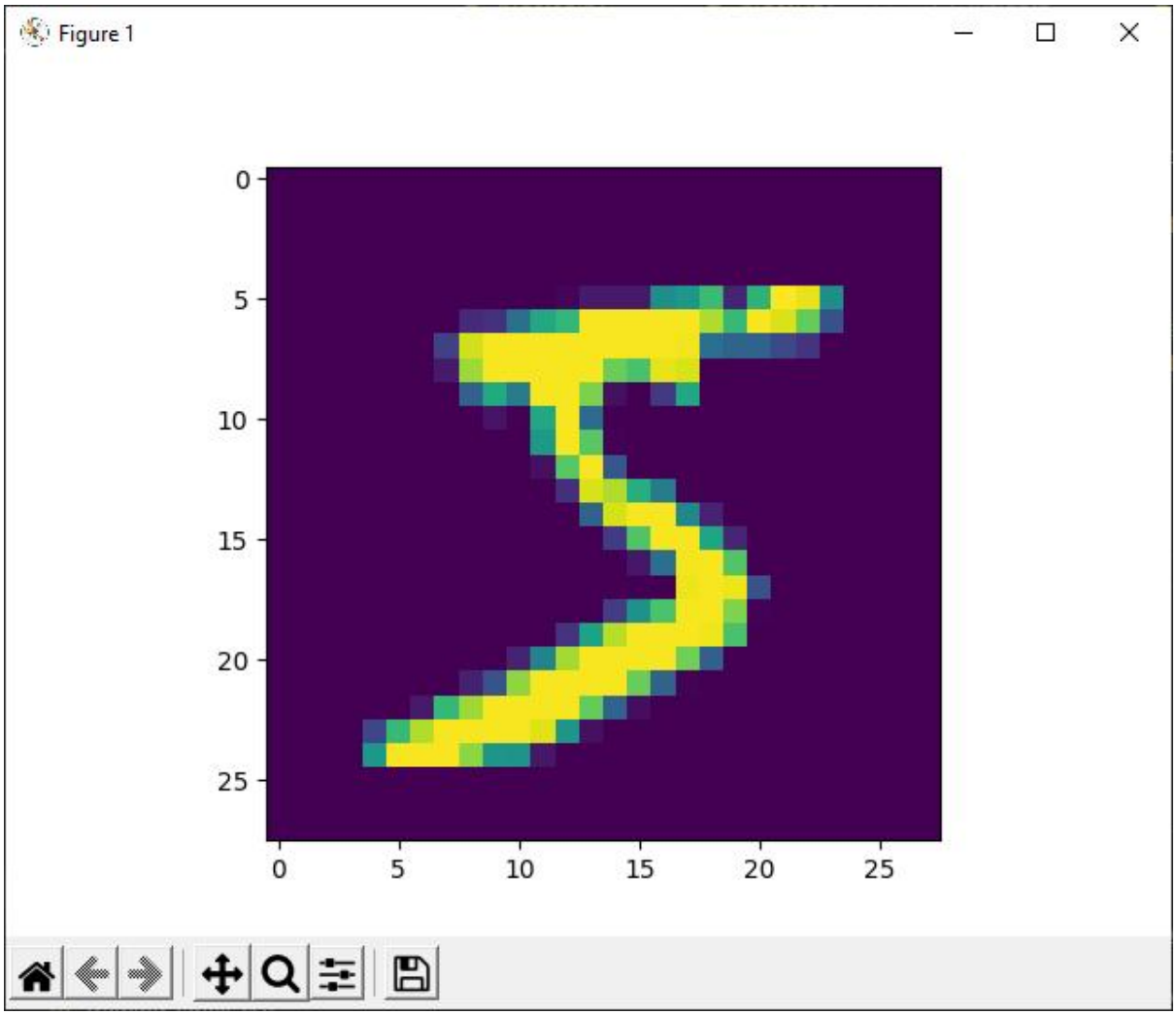
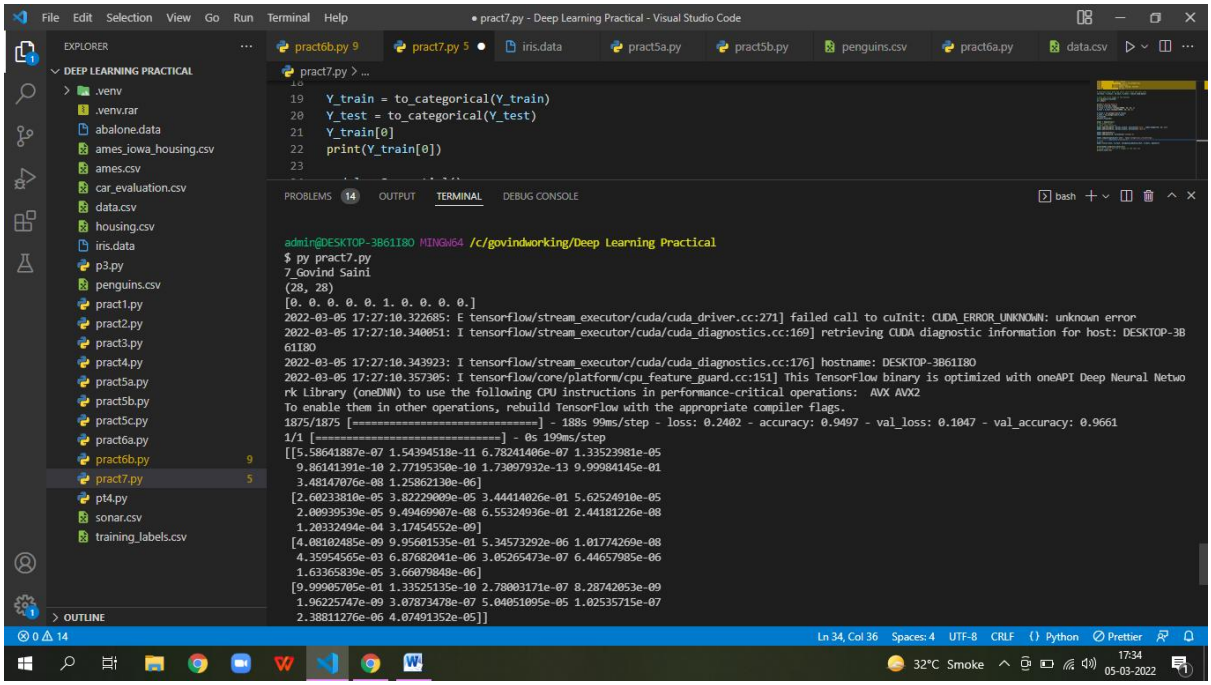
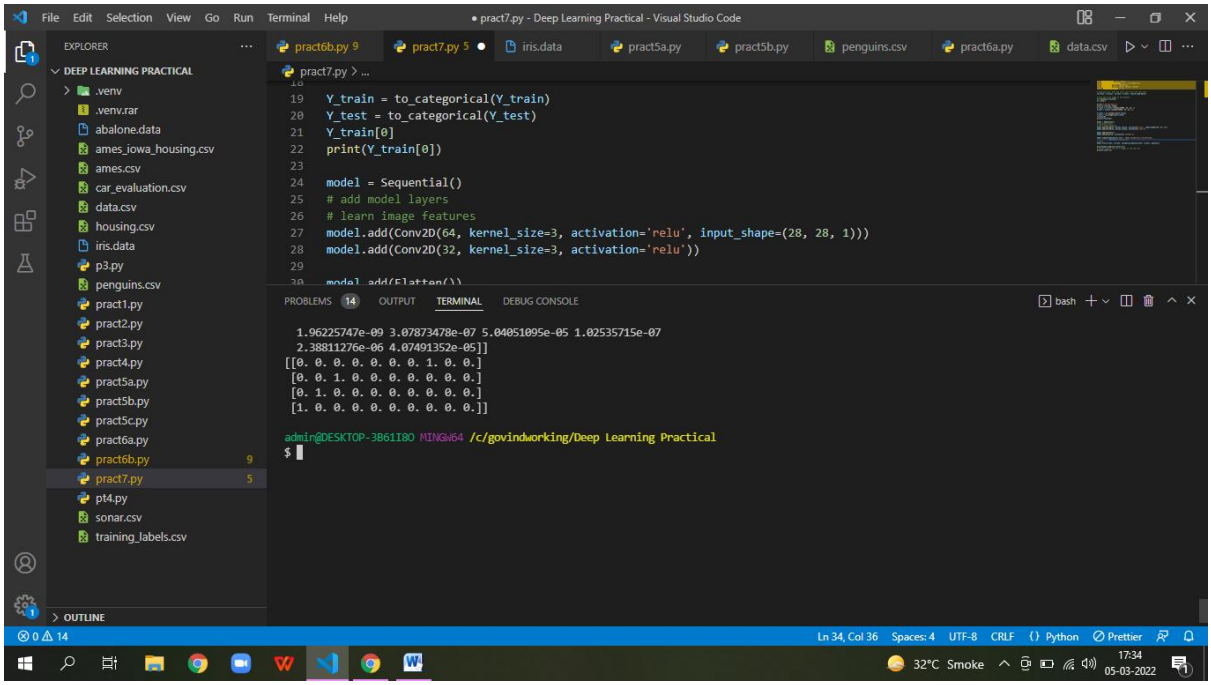


Figure 1
0
5
10
15
20
25
0
5
10
15
20
25


File Edit Selection View Go Run Terminal Help
• pract7.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
pract6b.py 9
pract7.py 5
iris.data
pract5a.py
pract5b.py
penguins.csv
pract6a.py
data.csv
Y_train = to_categorical(Y_train)
Y_test = to_categorical(Y_test)
Y_train[0]
print(Y_train[0])
PROBLEMS 14 OUTPUT TERMINAL DEBUG CONSOLE
admin@DESKTOP-3B61I8O MINGW64 /c/govindworking/Deep Learning Practical
$ py pract7.py
7_Govind Saini
(28, 28)
[0. 0. 0. 0. 0. 1. 0. 0. 0. 0.]
2022-03-05 17:27:10.322685: E tensorflow/stream_executor/cuda/cuda_driver.cc:271] failed call to cuInit: CUDA_ERROR_UNKNOWN: unknown error
2022-03-05 17:27:10.340051: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:169] retrieving CUDA diagnostic information for host: DESKTOP-3B61I8O
2022-03-05 17:27:10.343923: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:176] hostname: DESKTOP-3B61I8O
2022-03-05 17:27:10.357305: I tensorflow/core/platform/cpu_feature_guard.cc:151] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (oneDNN) to use the following CPU instructions in performance-critical operations: AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
1875/1875 [==========] - 188s 99ms/step - loss: 0.2402 - accuracy: 0.9497 - val_loss: 0.1047 - val_accuracy: 0.9661
1/1 [==========] - 0s 199ms/step
[[5.58641887e-07 1.54394518e-11 6.78241406e-07 1.33523981e-05
9.86141391e-10 2.77195350e-10 1.73097932e-13 9.99984145e-01
3.48147076e-08 1.25862130e-06]
[2.60233810e-05 3.82229009e-05 3.44414026e-01 5.62524910e-05
2.00939539e-05 9.49469907e-08 6.55324936e-01 2.44181226e-08
1.20332494e-04 3.17454552e-09]
[4.08102485e-09 9.95601535e-01 5.34573292e-06 1.01774269e-08
4.35954565e-03 6.87682041e-06 3.05265473e-07 6.44657985e-06
1.63365839e-05 3.66079848e-06]
[9.99905705e-01 1.33525135e-10 2.78003171e-07 8.28742053e-09
1.96225747e-09 3.07873478e-07 5.04051095e-05 1.02535715e-07
2.38811276e-06 4.07491352e-05]]
OUTLINE
Ln 34, Col 36 Spaces: 4 UTF-8 CRLF Python Prettier
32°C Smoke
17:34
05-03-2022


File Edit Selection View Go Run Terminal Help
• pract7.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
pract6b.py 9
pract7.py 5
iris.data
pract5a.py
pract5b.py
penguins.csv
pract6a.py
data.csv
Y_train = to_categorical(Y_train)
Y_test = to_categorical(Y_test)
Y_train[0]
print(Y_train[0])
model = Sequential()
# add model layers
# learn image features
model.add(Conv2D(64, kernel_size=3, activation='relu', input_shape=(28, 28, 1)))
model.add(Conv2D(32, kernel_size=3, activation='relu'))
PROBLEMS 14 OUTPUT TERMINAL DEBUG CONSOLE
1.96225747e-09 3.07873478e-07 5.04051095e-05 1.02535715e-07
2.38811276e-06 4.07491352e-05]]
[[0. 0. 0. 0. 0. 0. 0. 1. 0. 0.]
[0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
[0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
[1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
admin@DESKTOP-3B61I8O MINGW64 /c/govindworking/Deep Learning Practical
$
OUTLINE
Ln 34, Col 36 Spaces: 4 UTF-8 CRLF Python Prettier
32°C Smoke
17:34
05-03-2022

Practical No : 8

Name: Govind Saini
Roll No: 07
Sign: Govind

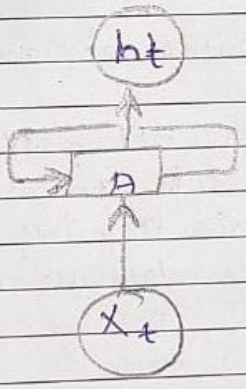
Date 11/03/2022
Page

Practical - 8

8) Aim:-
Demonstrate recurrent neural network that learns to perform sequence analysis for stock price.

Description:-

Recurrent net Neural network implement the same concept using machine; they have loop and allow information to persist where tradition neural network can't
Let's us a few illustration to demonstrate how a RNN work

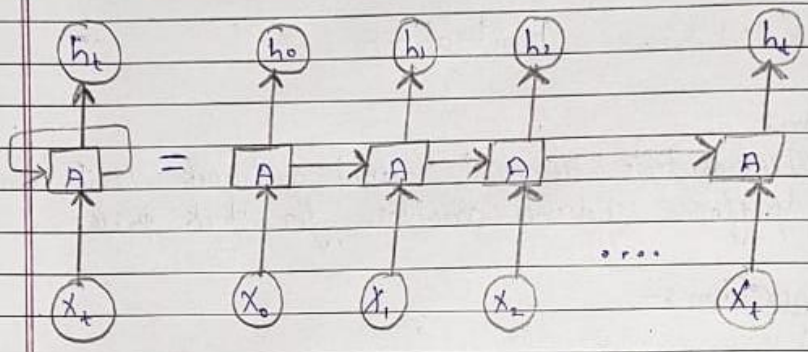


when A takes the input X_t, then H_t will be the Output.
A RNN is like multiple copies of the same network that passes the message to a successor.
Now let's think a little bit about what happen if we unroll previous loop

Govind

Name: Govind Saini
Roll No: 07
Sign: Govind

Date...../...../.....
Page.....



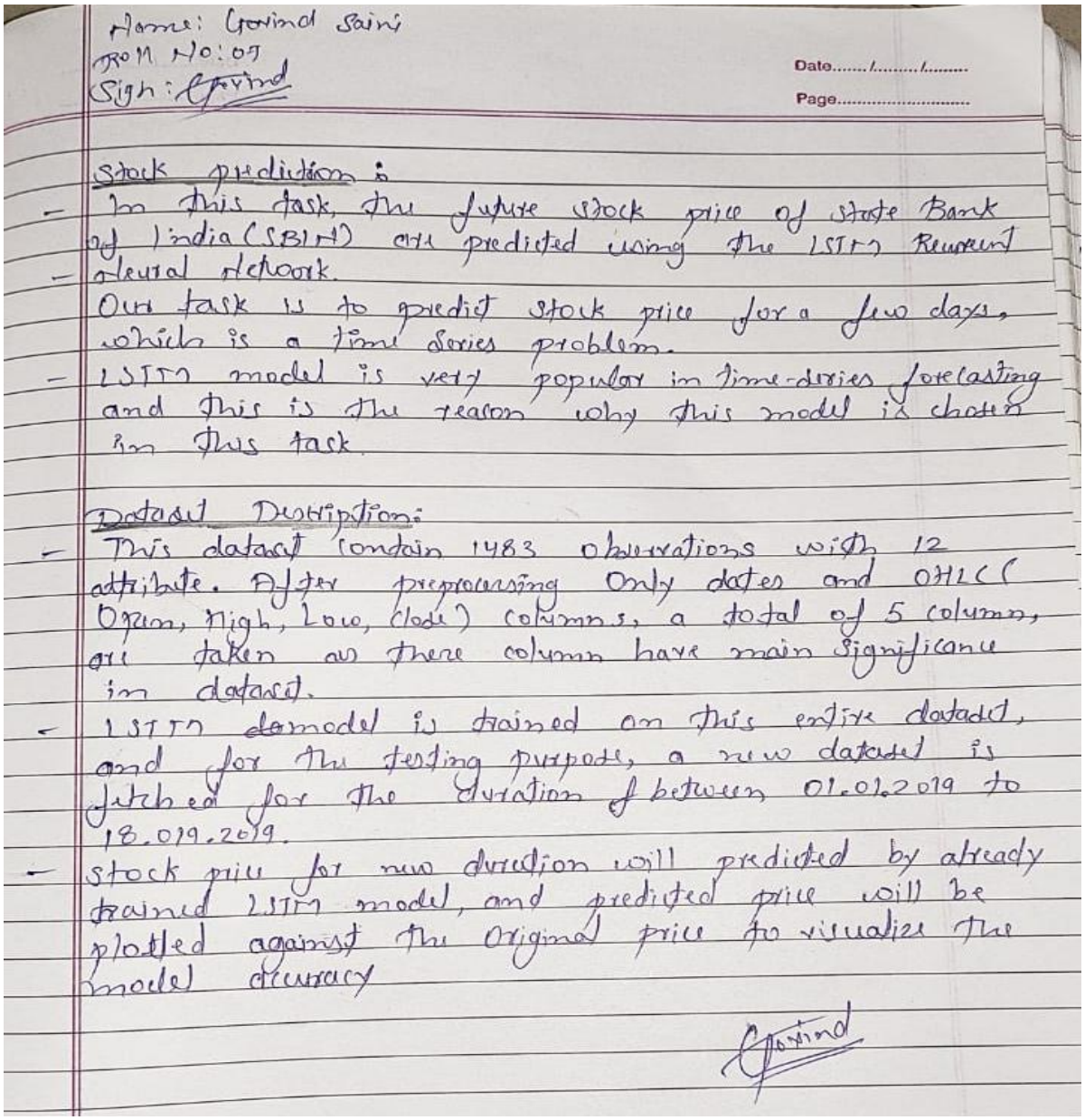
Here comes the vanishing gradient problem of the RNN, where it can not handle large sequence. Long short-term memory (LSTM) are designed to handle long-term dependence.

library are used :-

NSEPy :-
It is a library to extract historical and realtime data from NSE's website.
This library aims to keep the API very simple python is a great tool for data analysis along with the scipy stack and the main objective of NSEpy is to provide analysis ready data-dirves for use with scipy stack

LSTM Recurrent Neural Network :
- long-short-Term Memory RNN belong to the family of deep learning algorithm
It is recurrent network because of the feedback connections in its architecture.

Govind



Name: Govind Saini
ROM No: 07
Sign: Govind

Date...../...../.....
Page.....

Stock prediction :

- In this task, the future stock price of State Bank of India (SBIN) are predicted using the LSTM Reurrent
- Neural Network.

Our task is to predict stock price for a few days, which is a time series problem.

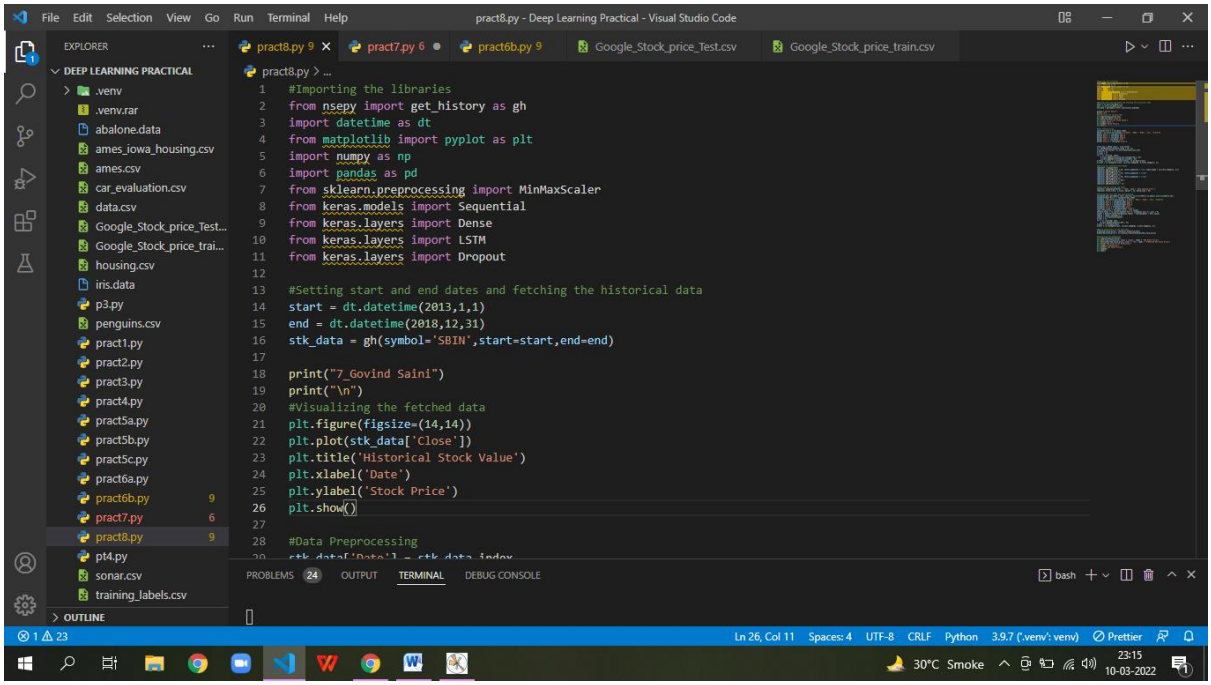
- LSTM model is very popular in time-series forecasting and this is the reason why this model is chosen in this task.

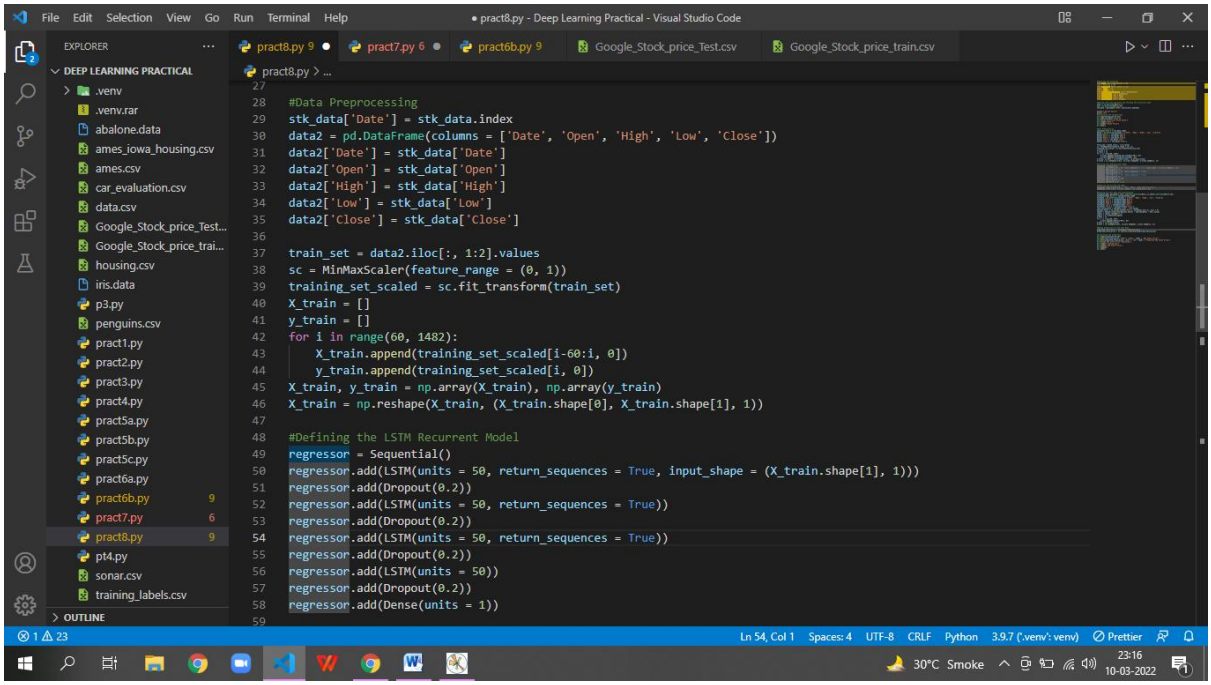
Dataset Description:

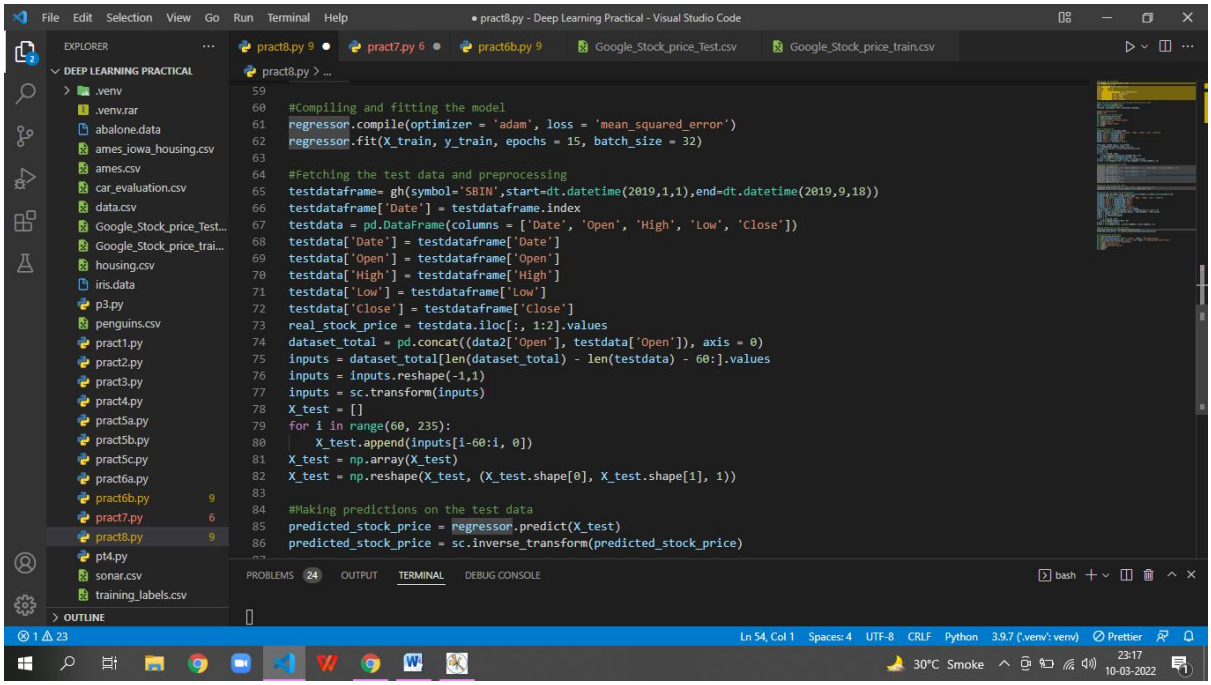
- This dataset contain 1483 observations with 12 attribute. After preprocessing only dates and OHLC (Open, High, Low, Close) columns, a total of 5 column, are taken as there column have main significance in dataset.
- LSTM ~~de~~model is trained on this entire dataset, and for the testing purpose, a new dataset is fetched for the duration of between 01.01.2019 to 18.019.2019.
- Stock price for new duration will predicted by already trained LSTM model, and predicted price will be plotted against the original price to visualize the model accuracy

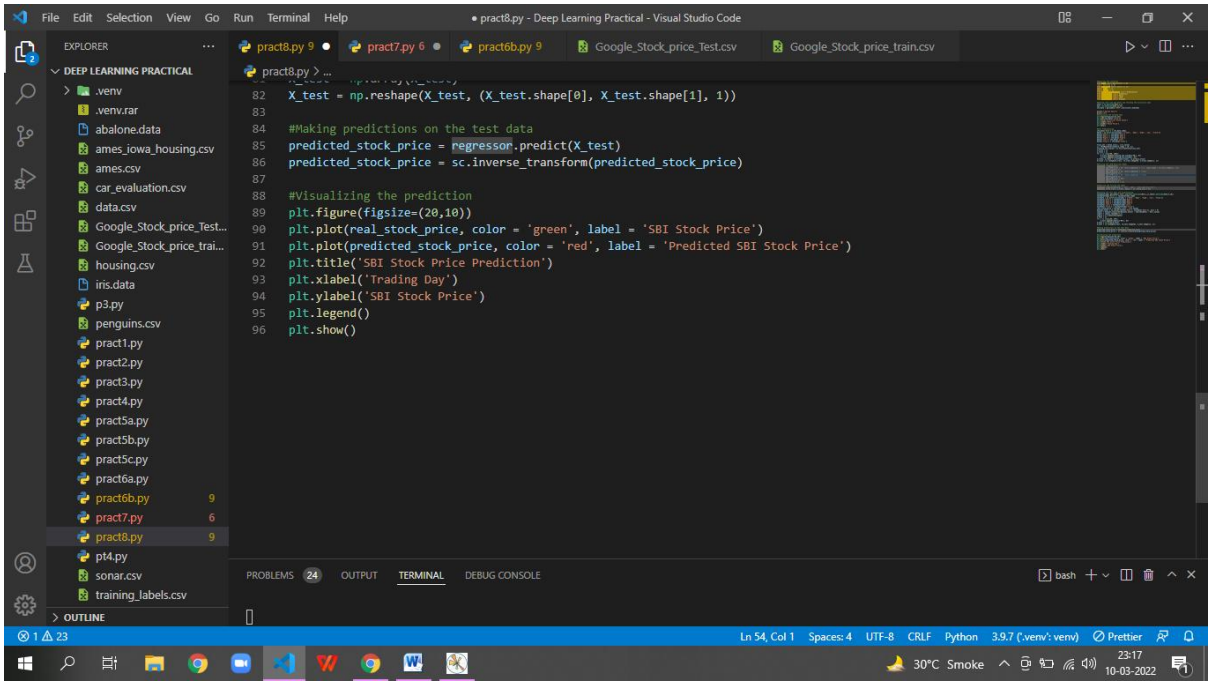
Govind

Code :


File Edit Selection View Go Run Terminal Help
pract8.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
pract8.py 9
pract7.py 6
pract6b.py 9
Google_Stock_price_Test.csv
Google_Stock_price_train.csv
#Importing the libraries
from nsepy import get_history as gh
import datetime as dt
from matplotlib import pyplot as plt
import numpy as np
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM
from keras.layers import Dropout
#Setting start and end dates and fetching the historical data
start = dt.datetime(2013,1,1)
end = dt.datetime(2018,12,31)
stk_data = gh(symbol='SBIN',start=start,end=end)
print("7_Govind Saini")
print("\n")
#Visualizing the fetched data
plt.figure(figsize=(14,14))
plt.plot(stk_data['Close'])
plt.title('Historical Stock Value')
plt.xlabel('Date')
plt.ylabel('Stock Price')
plt.show()
#Data Preprocessing
PROBLEMS 24 OUTPUT TERMINAL DEBUG CONSOLE
Ln 26, Col 11 Spaces: 4 UTF-8 CRLF Python 3.9.7 ('.venv': venv) Prettier
30°C Smoke
23:15
10-03-2022


File Edit Selection View Go Run Terminal Help
• pract8.py - Deep Learning Practical - Visual Studio Code
#Data Preprocessing
stk_data['Date'] = stk_data.index
data2 = pd.DataFrame(columns = ['Date', 'Open', 'High', 'Low', 'Close'])
data2['Date'] = stk_data['Date']
data2['Open'] = stk_data['Open']
data2['High'] = stk_data['High']
data2['Low'] = stk_data['Low']
data2['Close'] = stk_data['Close']
train_set = data2.iloc[:, 1:2].values
sc = MinMaxScaler(feature_range = (0, 1))
training_set_scaled = sc.fit_transform(train_set)
X_train = []
y_train = []
for i in range(60, 1482):
X_train.append(training_set_scaled[i-60:i, 0])
y_train.append(training_set_scaled[i, 0])
X_train, y_train = np.array(X_train), np.array(y_train)
X_train = np.reshape(X_train, (X_train.shape[0], X_train.shape[1], 1))
#Defining the LSTM Recurrent Model
regressor = Sequential()
regressor.add(LSTM(units = 50, return_sequences = True, input_shape = (X_train.shape[1], 1)))
regressor.add(Dropout(0.2))
regressor.add(LSTM(units = 50, return_sequences = True))
regressor.add(Dropout(0.2))
regressor.add(LSTM(units = 50, return_sequences = True))
regressor.add(Dropout(0.2))
regressor.add(LSTM(units = 50))
regressor.add(Dropout(0.2))
regressor.add(Dense(units = 1))
Ln 54, Col 1 Spaces: 4 UTF-8 CRLF Python 3.9.7 ('.venv': venv) Prettier
30°C Smoke
23:16
10-03-2022


File Edit Selection View Go Run Terminal Help
• pract8.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
pract8.py 9
pract7.py 6
pract6b.py 9
Google_Stock_price_Test.csv
Google_Stock_price_train.csv
pract8.py > ...
59
60 #Compiling and fitting the model
61 regressor.compile(optimizer = 'adam', loss = 'mean_squared_error')
62 regressor.fit(X_train, y_train, epochs = 15, batch_size = 32)
63
64 #Fetching the test data and preprocessing
65 testdataframe= gh(symbol='SBIN',start=dt.datetime(2019,1,1),end=dt.datetime(2019,9,18))
66 testdataframe['Date'] = testdataframe.index
67 testdata = pd.DataFrame(columns = ['Date', 'Open', 'High', 'Low', 'Close'])
68 testdata['Date'] = testdataframe['Date']
69 testdata['Open'] = testdataframe['Open']
70 testdata['High'] = testdataframe['High']
71 testdata['Low'] = testdataframe['Low']
72 testdata['Close'] = testdataframe['Close']
73 real_stock_price = testdata.iloc[:, 1:2].values
74 dataset_total = pd.concat((data2['Open'], testdata['Open']), axis = 0)
75 inputs = dataset_total[len(dataset_total) - len(testdata) - 60:].values
76 inputs = inputs.reshape(-1,1)
77 inputs = sc.transform(inputs)
78 X_test = []
79 for i in range(60, 235):
80 X_test.append(inputs[i-60:i, 0])
81 X_test = np.array(X_test)
82 X_test = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))
83
84 #Making predictions on the test data
85 predicted_stock_price = regressor.predict(X_test)
86 predicted_stock_price = sc.inverse_transform(predicted_stock_price)
PROBLEMS 24 OUTPUT TERMINAL DEBUG CONSOLE
bash
OUTLINE
Ln 54, Col 1 Spaces: 4 UTF-8 CRLF Python 3.9.7 ('.venv': venv) Prettier
30°C Smoke
23:17
10-03-2022


File Edit Selection View Go Run Terminal Help
• pract8.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
pract8.py 9
pract7.py 6
pract6b.py 9
Google_Stock_price_Test.csv
Google_Stock_price_train.csv
pract8.py > ...
82 X_test = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))
83
84 #Making predictions on the test data
85 predicted_stock_price = regressor.predict(X_test)
86 predicted_stock_price = sc.inverse_transform(predicted_stock_price)
87
88 #Visualizing the prediction
89 plt.figure(figsize=(20,10))
90 plt.plot(real_stock_price, color = 'green', label = 'SBI Stock Price')
91 plt.plot(predicted_stock_price, color = 'red', label = 'Predicted SBI Stock Price')
92 plt.title('SBI Stock Price Prediction')
93 plt.xlabel('Trading Day')
94 plt.ylabel('SBI Stock Price')
95 plt.legend()
96 plt.show()
PROBLEMS 24 OUTPUT TERMINAL DEBUG CONSOLE
bash
OUTLINE
Ln 54, Col 1 Spaces: 4 UTF-8 CRLF Python 3.9.7 ('.venv': venv) Prettier
30°C Smoke
23:17
10-03-2022

Output :

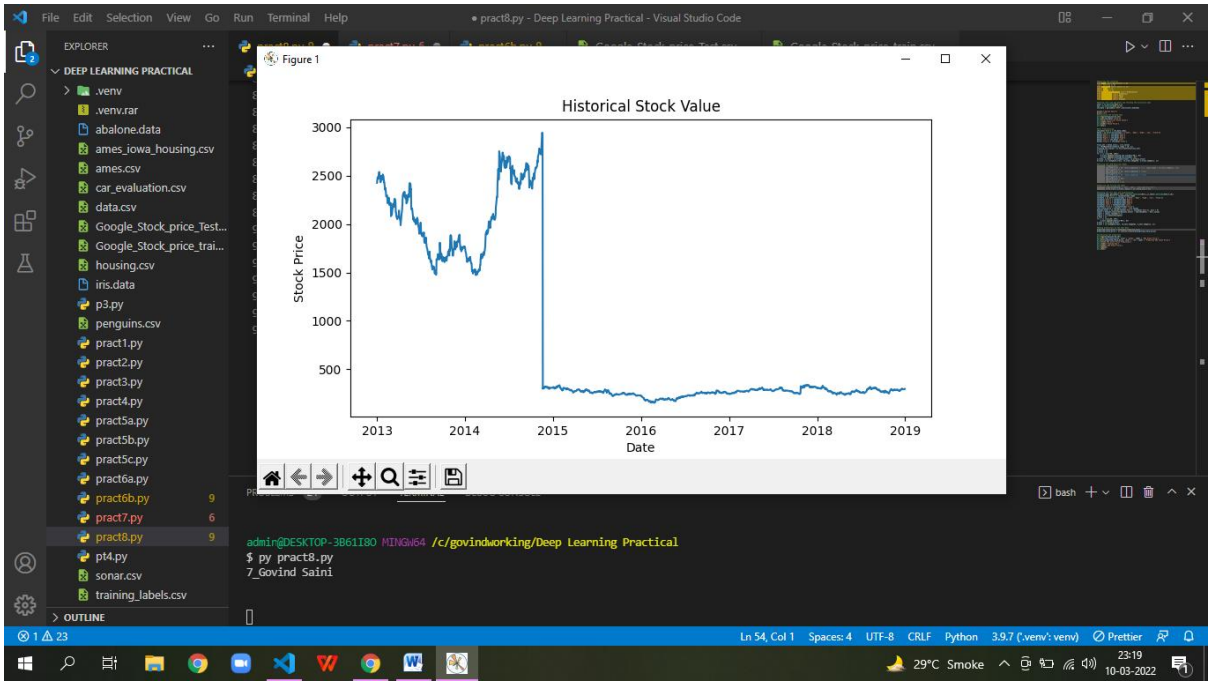


Figure 1
Historical Stock Value
Stock Price
Date
3000
2500
2000
1500
1000
500
2013
2014
2015
2016
2017
2018
2019
admin@DESKTOP-3B61I8O MINGW64 /c/govindworking/Deep Learning Practical
$ py pract8.py
7_Govind Saini

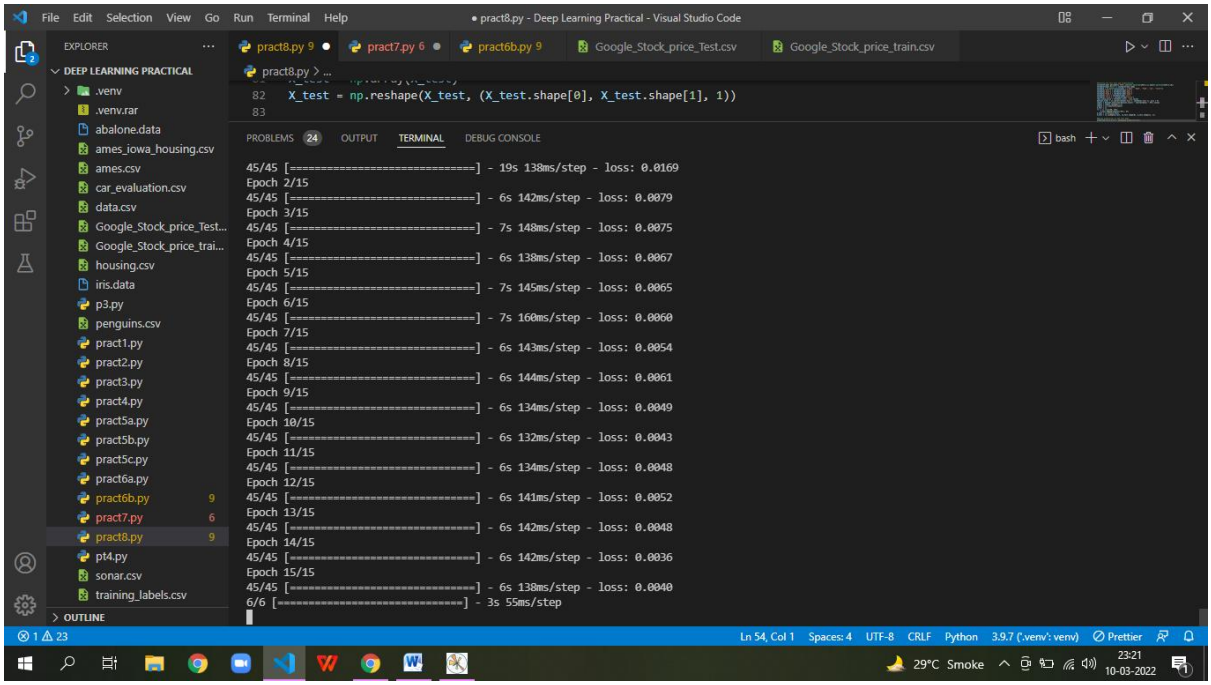

X_test = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))
45/45 [==========] - 19s 138ms/step - loss: 0.0169
Epoch 2/15
45/45 [==========] - 6s 142ms/step - loss: 0.0079
Epoch 3/15
45/45 [==========] - 7s 148ms/step - loss: 0.0075
Epoch 4/15
45/45 [==========] - 6s 138ms/step - loss: 0.0067
Epoch 5/15
45/45 [==========] - 7s 145ms/step - loss: 0.0065
Epoch 6/15
45/45 [==========] - 7s 160ms/step - loss: 0.0060
Epoch 7/15
45/45 [==========] - 6s 143ms/step - loss: 0.0054
Epoch 8/15
45/45 [==========] - 6s 144ms/step - loss: 0.0061
Epoch 9/15
45/45 [==========] - 6s 134ms/step - loss: 0.0049
Epoch 10/15
45/45 [==========] - 6s 132ms/step - loss: 0.0043
Epoch 11/15
45/45 [==========] - 6s 134ms/step - loss: 0.0048
Epoch 12/15
45/45 [==========] - 6s 141ms/step - loss: 0.0052
Epoch 13/15
45/45 [==========] - 6s 142ms/step - loss: 0.0048
Epoch 14/15
45/45 [==========] - 6s 142ms/step - loss: 0.0036
Epoch 15/15
45/45 [==========] - 6s 138ms/step - loss: 0.0040
6/6 [==========] - 3s 55ms/step


Figure 1
SBI Stock Price Prediction
SBI Stock Price
Predicted SBI Stock Price
SBI Stock Price
Trading Day
360
340
320
300
280
260
0
25
50
75
100
125
150
175
45/45 [==========] - 6s 142ms/step - loss: 0.0036
Epoch 15/15
45/45 [==========] - 6s 138ms/step - loss: 0.0040
6/6 [==========] - 3s 55ms/step

Practical No : 9

Name : Govind Saini
Roll No: 07
Sign:

Date 19/3/2022
Page

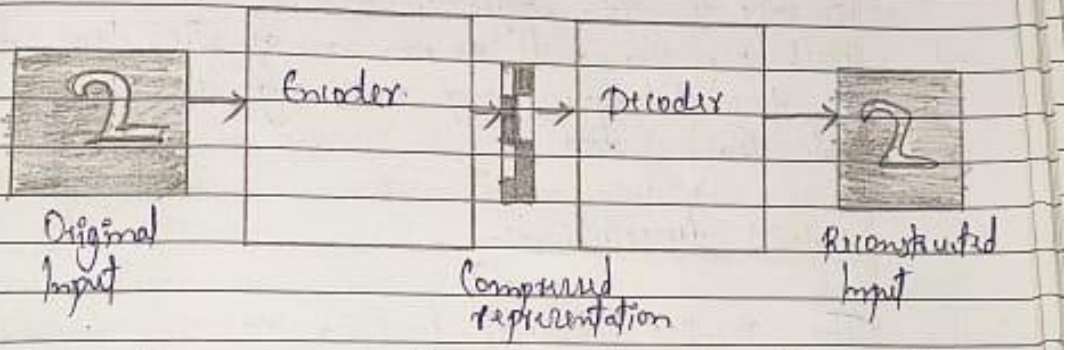
Practical - 9

9)

Aim:-

Performing encoding and decoding of images using deep autoencoder.

Description:-



Autoencoding is a data compress algorithm where the compression and decompression function

i) data-specific
ii) lossy
iii) learned automatically from example rather that engineered by a human.

Autoencoders are data-specific, which means that they will only be able to compress data similar to what they have been trained on.
This is different from say MPEG-2 Audio layer III (MP3) compression algorithm, which hold assumption about 'sound' in general, but not about specific types of sound.

Name: Govind Saini
Roll No: 07
Sign: Govind

Date: / /
Page:

Autoencoders are lossy, which means that the decompressed outputs will be degraded compared to the original input (similar to MP3 or JPEG compression). this differs from lossless arithmetic operator.

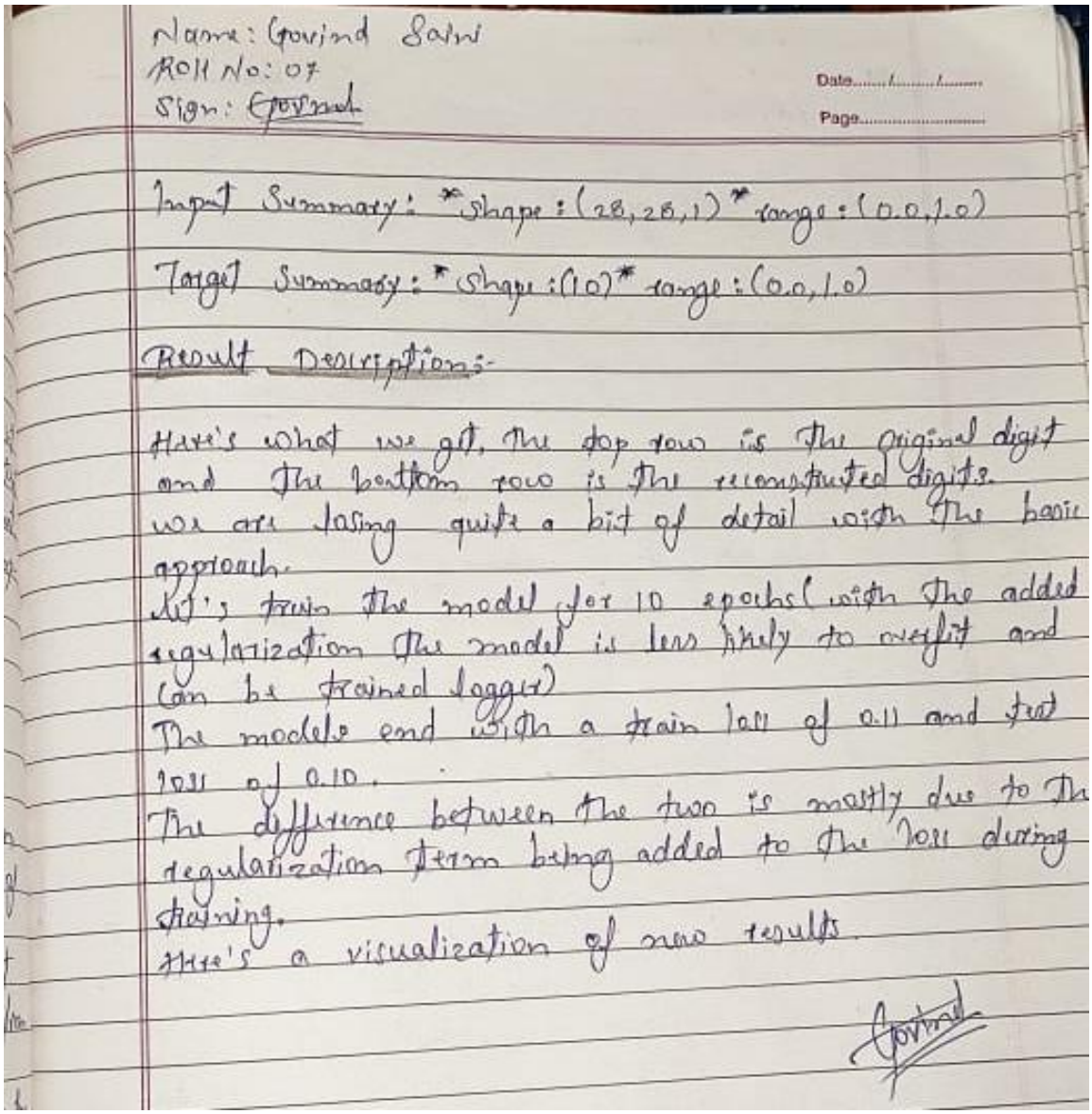
Autoencoders are learned automatically from data examples, which is useful property: it means that it is easy to train specialized instance of algorithm that will perform well on a specific type of input. It doesn't require any new engineering, just appropriate training data.

Dataset Description:-

The MNIST dataset contain 70,000 images of handwritten digit (zero to nine) that have been size-normalized and centered in a square grid of pixels.
Each image is a 28x28x1 array of floating-point numbers representing grayscale intensities ranging from 0 (black) to 1 (white).
The target data consist of One-hot binary vector of size 10, corresponding to the digit classification categories zero through nine. some example MNIST images are shown below:

Information: * name : MNIST * length : 70000

Govind


Name: Govind Saini
ROll No: 07
Sign: Govind
Date...../...../.....
Page.....

Input Summary: * Shape: (28,28,1) * range: (0.0,1.0)

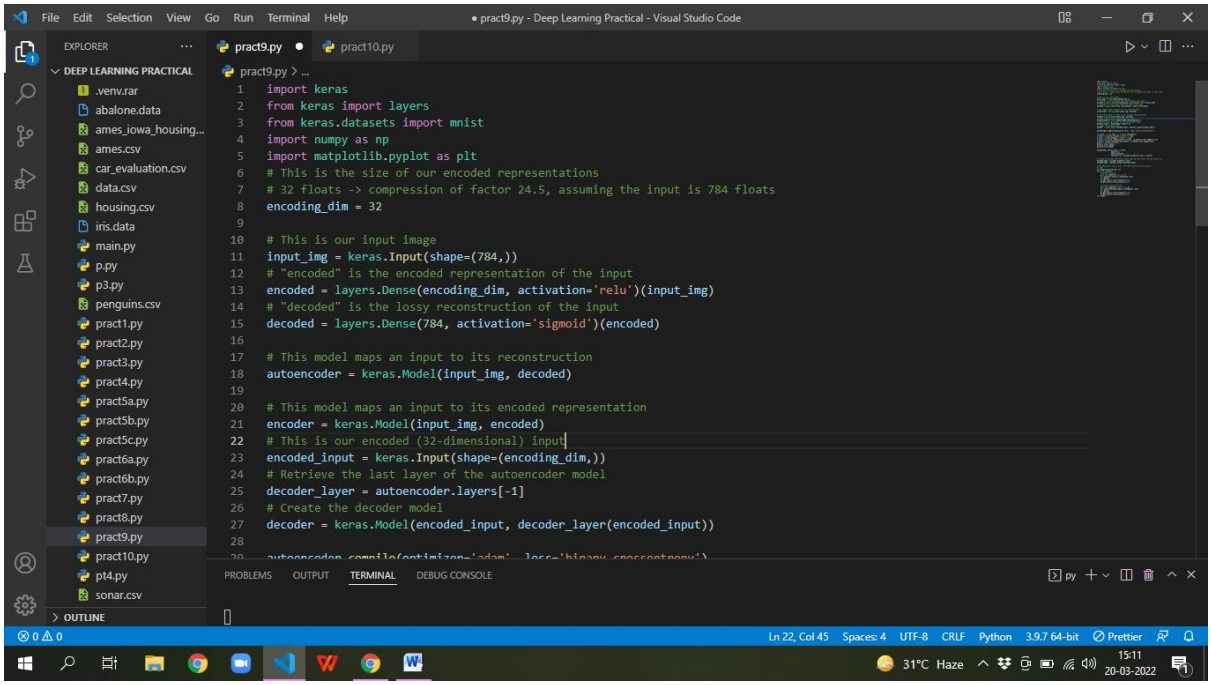
Target Summary: * Shape: (10) * range: (0.0,1.0)

Result Description:-

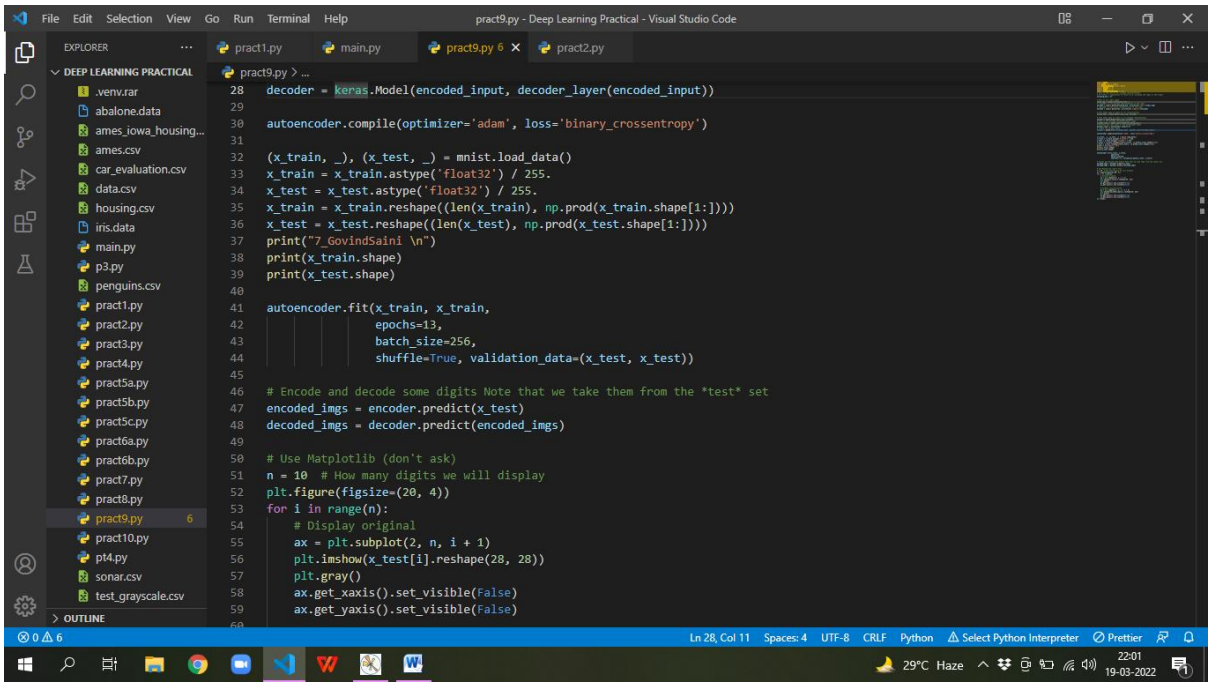
Here's what we got. The top row is the original digit and the bottom row is the reconstructed digits.
We are losing quite a bit of detail with the basic approach.
Let's train the model for 10 epochs (with the added regularization the model is less likely to overfit and can be trained longer)
The models end with a train loss of 0.11 and test loss of 0.10.
The difference between the two is mostly due to the regularization term being added to the loss during training.
Here's a visualization of new results

Govind

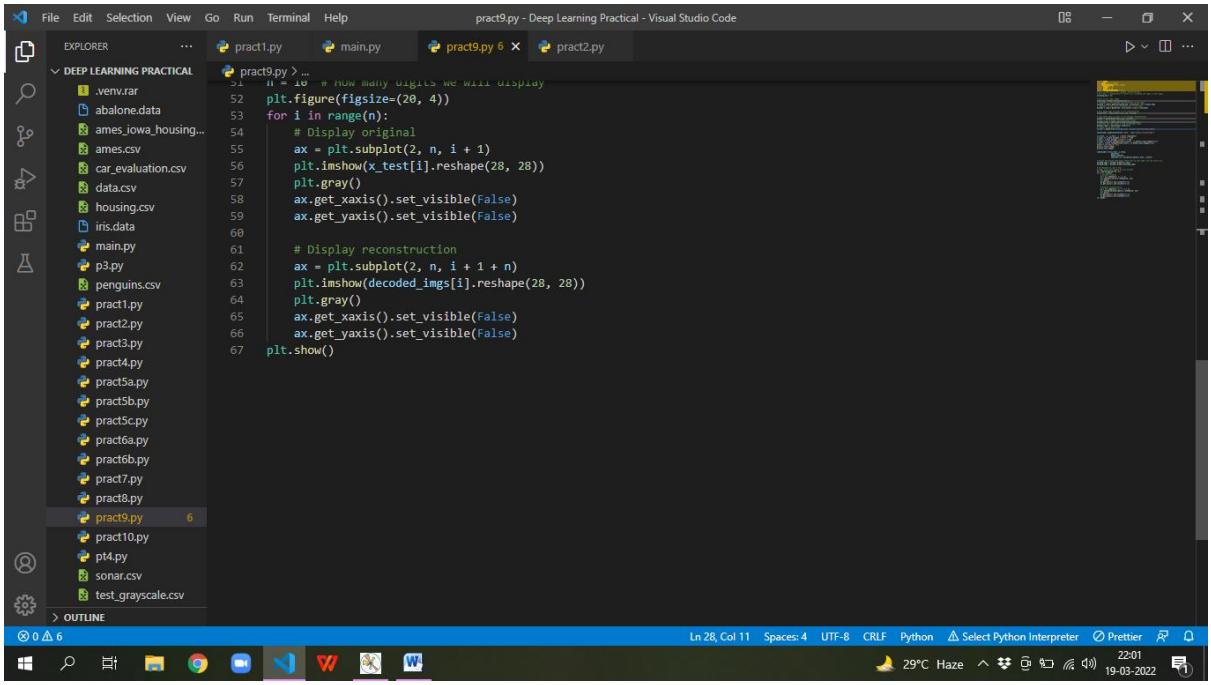
Code :



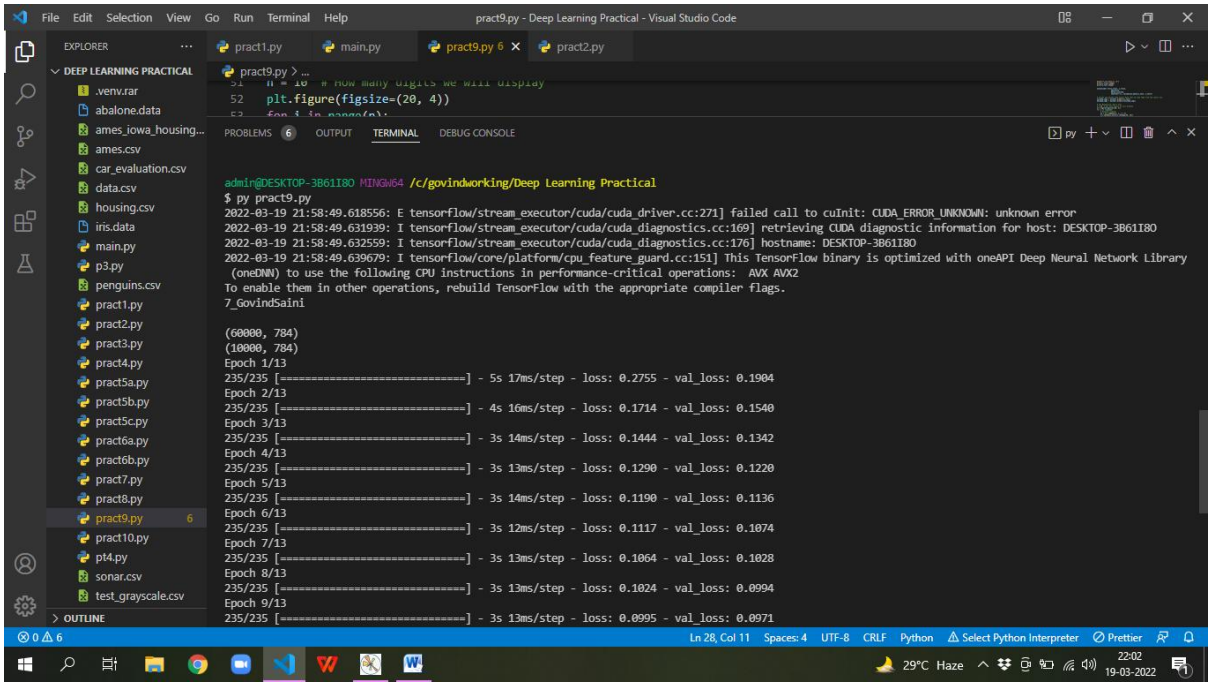
```python
import keras
from keras import layers
from keras.datasets import mnist
import numpy as np
import matplotlib.pyplot as plt
# This is the size of our encoded representations
# 32 floats -> compression of factor 24.5, assuming the input is 784 floats
encoding_dim = 32

# This is our input image
input_img = keras.Input(shape=(784,))
# "encoded" is the encoded representation of the input
encoded = layers.Dense(encoding_dim, activation='relu')(input_img)
# "decoded" is the lossy reconstruction of the input
decoded = layers.Dense(784, activation='sigmoid')(encoded)

# This model maps an input to its reconstruction
autoencoder = keras.Model(input_img, decoded)

# This model maps an input to its encoded representation
encoder = keras.Model(input_img, encoded)
# This is our encoded (32-dimensional) input
encoded_input = keras.Input(shape=(encoding_dim,))
# Retrieve the last layer of the autoencoder model
decoder_layer = autoencoder.layers[-1]
# Create the decoder model
decoder = keras.Model(encoded_input, decoder_layer(encoded_input))
```

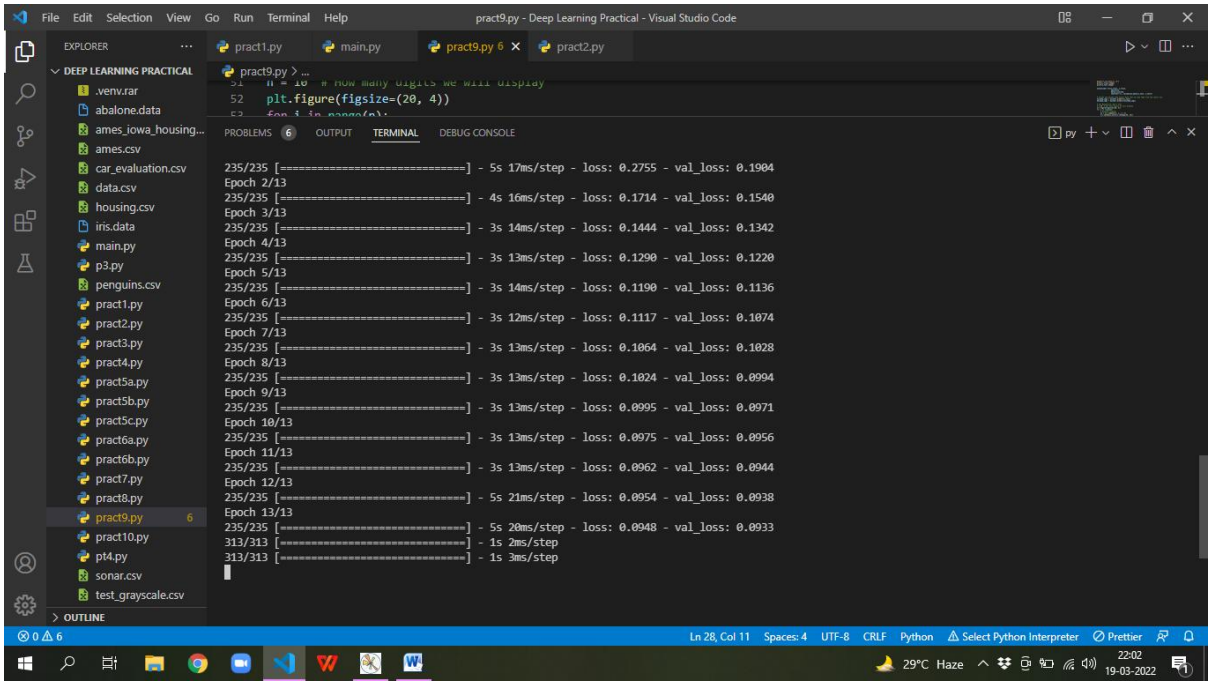


```python
decoder = keras.Model(encoded_input, decoder_layer(encoded_input))

autoencoder.compile(optimizer='adam', loss='binary_crossentropy')

(x_train, _), (x_test, _) = mnist.load_data()
x_train = x_train.astype('float32') / 255.
x_test = x_test.astype('float32') / 255.
x_train = x_train.reshape((len(x_train), np.prod(x_train.shape[1:])))
x_test = x_test.reshape((len(x_test), np.prod(x_test.shape[1:])))
print("7_GovindSaini \n")
print(x_train.shape)
print(x_test.shape)

autoencoder.fit(x_train, x_train,
                epochs=13,
                batch_size=256,
                shuffle=True, validation_data=(x_test, x_test))

# Encode and decode some digits Note that we take them from the *test* set
encoded_imgs = encoder.predict(x_test)
decoded_imgs = decoder.predict(encoded_imgs)

# Use Matplotlib (don't ask)
n = 10  # How many digits we will display
plt.figure(figsize=(20, 4))
for i in range(n):
    # Display original
    ax = plt.subplot(2, n, i + 1)
    plt.imshow(x_test[i].reshape(28, 28))
    plt.gray()
    ax.get_xaxis().set_visible(False)
    ax.get_yaxis().set_visible(False)
```


File Edit Selection View Go Run Terminal Help
pract9.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
pract1.py
main.py
pract9.py 6
pract2.py
pract9.py > ...
plt.figure(figsize=(20, 4))
for i in range(n):
# Display original
ax = plt.subplot(2, n, i + 1)
plt.imshow(x_test[i].reshape(28, 28))
plt.gray()
ax.get_xaxis().set_visible(False)
ax.get_yaxis().set_visible(False)
# Display reconstruction
ax = plt.subplot(2, n, i + 1 + n)
plt.imshow(decoded_imgs[i].reshape(28, 28))
plt.gray()
ax.get_xaxis().set_visible(False)
ax.get_yaxis().set_visible(False)
plt.show()
OUTLINE
Ln 28, Col 11 Spaces: 4 UTF-8 CRLF Python Select Python Interpreter Prettier
29°C Haze
22:01
19-03-2022

Ouput :


File Edit Selection View Go Run Terminal Help
pract9.py - Deep Learning Practical - Visual Studio Code
PROBLEMS 6 OUTPUT TERMINAL DEBUG CONSOLE
admin@DESKTOP-3B61I8O MINGW64 /c/govindworking/Deep Learning Practical
$ py pract9.py
2022-03-19 21:58:49.618556: E tensorflow/stream_executor/cuda/cuda_driver.cc:271] failed call to cuInit: CUDA_ERROR_UNKNOWN: unknown error
2022-03-19 21:58:49.631939: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:169] retrieving CUDA diagnostic information for host: DESKTOP-3B61I8O
2022-03-19 21:58:49.632559: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:176] hostname: DESKTOP-3B61I8O
2022-03-19 21:58:49.639679: I tensorflow/core/platform/cpu_feature_guard.cc:151] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (oneDNN) to use the following CPU instructions in performance-critical operations: AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
7_GovindSaini
(60000, 784)
(10000, 784)
Epoch 1/13
235/235 [==========] - 5s 17ms/step - loss: 0.2755 - val_loss: 0.1904
Epoch 2/13
235/235 [==========] - 4s 16ms/step - loss: 0.1714 - val_loss: 0.1540
Epoch 3/13
235/235 [==========] - 3s 14ms/step - loss: 0.1444 - val_loss: 0.1342
Epoch 4/13
235/235 [==========] - 3s 13ms/step - loss: 0.1290 - val_loss: 0.1220
Epoch 5/13
235/235 [==========] - 3s 14ms/step - loss: 0.1190 - val_loss: 0.1136
Epoch 6/13
235/235 [==========] - 3s 12ms/step - loss: 0.1117 - val_loss: 0.1074
Epoch 7/13
235/235 [==========] - 3s 13ms/step - loss: 0.1064 - val_loss: 0.1028
Epoch 8/13
235/235 [==========] - 3s 13ms/step - loss: 0.1024 - val_loss: 0.0994
Epoch 9/13
235/235 [==========] - 3s 13ms/step - loss: 0.0995 - val_loss: 0.0971
Ln 28, Col 11 Spaces: 4 UTF-8 CRLF Python Select Python Interpreter Prettier
29°C Haze
22:02
19-03-2022


pract9.py - Deep Learning Practical - Visual Studio Code
pract1.py
main.py
pract9.py 6
pract2.py
plt.figure(figsize=(20, 4))
235/235 [==========] - 5s 17ms/step - loss: 0.2755 - val_loss: 0.1904
Epoch 2/13
235/235 [==========] - 4s 16ms/step - loss: 0.1714 - val_loss: 0.1540
Epoch 3/13
235/235 [==========] - 3s 14ms/step - loss: 0.1444 - val_loss: 0.1342
Epoch 4/13
235/235 [==========] - 3s 13ms/step - loss: 0.1290 - val_loss: 0.1220
Epoch 5/13
235/235 [==========] - 3s 14ms/step - loss: 0.1190 - val_loss: 0.1136
Epoch 6/13
235/235 [==========] - 3s 12ms/step - loss: 0.1117 - val_loss: 0.1074
Epoch 7/13
235/235 [==========] - 3s 13ms/step - loss: 0.1064 - val_loss: 0.1028
Epoch 8/13
235/235 [==========] - 3s 13ms/step - loss: 0.1024 - val_loss: 0.0994
Epoch 9/13
235/235 [==========] - 3s 13ms/step - loss: 0.0995 - val_loss: 0.0971
Epoch 10/13
235/235 [==========] - 3s 13ms/step - loss: 0.0975 - val_loss: 0.0956
Epoch 11/13
235/235 [==========] - 3s 13ms/step - loss: 0.0962 - val_loss: 0.0944
Epoch 12/13
235/235 [==========] - 5s 21ms/step - loss: 0.0954 - val_loss: 0.0938
Epoch 13/13
235/235 [==========] - 5s 20ms/step - loss: 0.0948 - val_loss: 0.0933
313/313 [==========] - 1s 2ms/step
313/313 [==========] - 1s 3ms/step

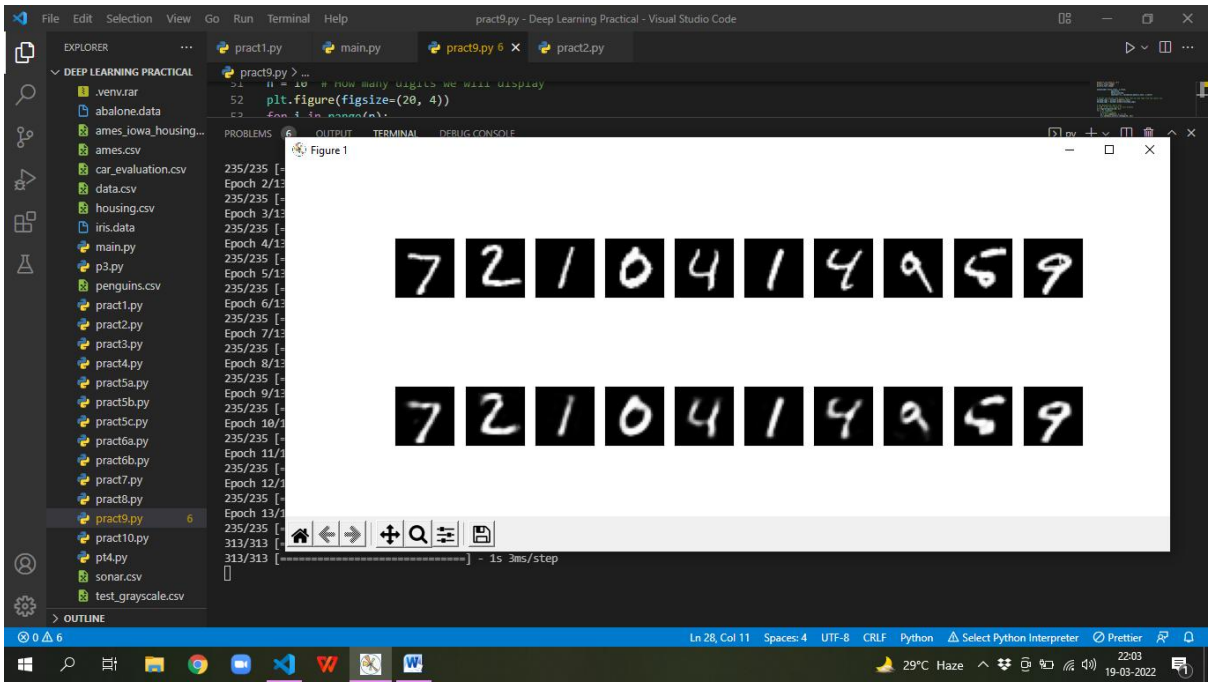


Figure 1
313/313 [==========] - 1s 3ms/step

Practical No : 10

Name: Govind Saini
Roll No: 07
Sign: Govind

Date 19/3/2022
Page

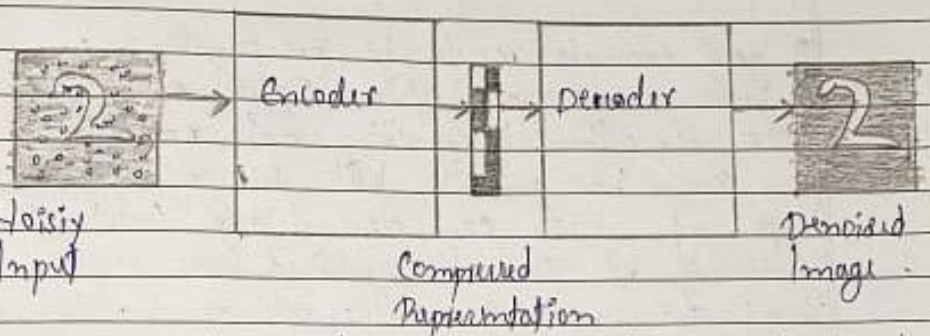
Practical - 10

10)

Aim:-

Denoising of images using autoencoder

Description:-



Denoising auto-encoders are an extension of simple autoencoder; however it's worth noting that denoising autoencoder were not originally meant to automatically denoise an image.

Instead, the denoising autoencoder produre was invented to help:

i) The hidden layer of the autoencoder learn more robust filters.

ii) Reduce the risk of overfitting in the autoencoder

iii) Prevent the autoencoder from learning a simple identity function.

Continued

Name: Govind Saini
ROll No: 07
Sign: Govind

Date: / /
Page:

Noise was stochastically added to the input data, and then the autoencoder was trained to recover the original, noiseperturbed signal.

From an image processing standpoint, we can train an autoencoder to perform automatic image pre-passing for us.

A great example would be pre-processing an image to improve the accuracy of an Optical character recognition (OCR) algorithm. If you've ever applied OCR before, you know how just a little bit of the wrong type of noise can dramatically hurt the performance of your OCR method.
Using denoising autoencoders, we can automatically preprocesses the image, improve the quality, and therefore increase the accuracy of the downstream OCR algorithms.

Dataset Description:
Title: MNIST

The MNIST dataset contain 70,000 image of handwritten digit (0 to 9) that have been size-normalized and centered in a square grid of pixels. Each image is a 28x28x1 array of floating-point numbers representing grayscale intensities ranging from 0 (black) to 1 (White).

Govind

Name: Govind Saini
Roll no: 07
Sign: Govind

Date: / /
Page:

The target data consists of one-hot binary vector of size 10, corresponding to the digit classification categories zero through nine. Some example MNIST images are shown below:

Information: * name: MNIST * length: 70000

Input Summary: * shape: (28, 28, 1) * range: (0.0, 1.0)

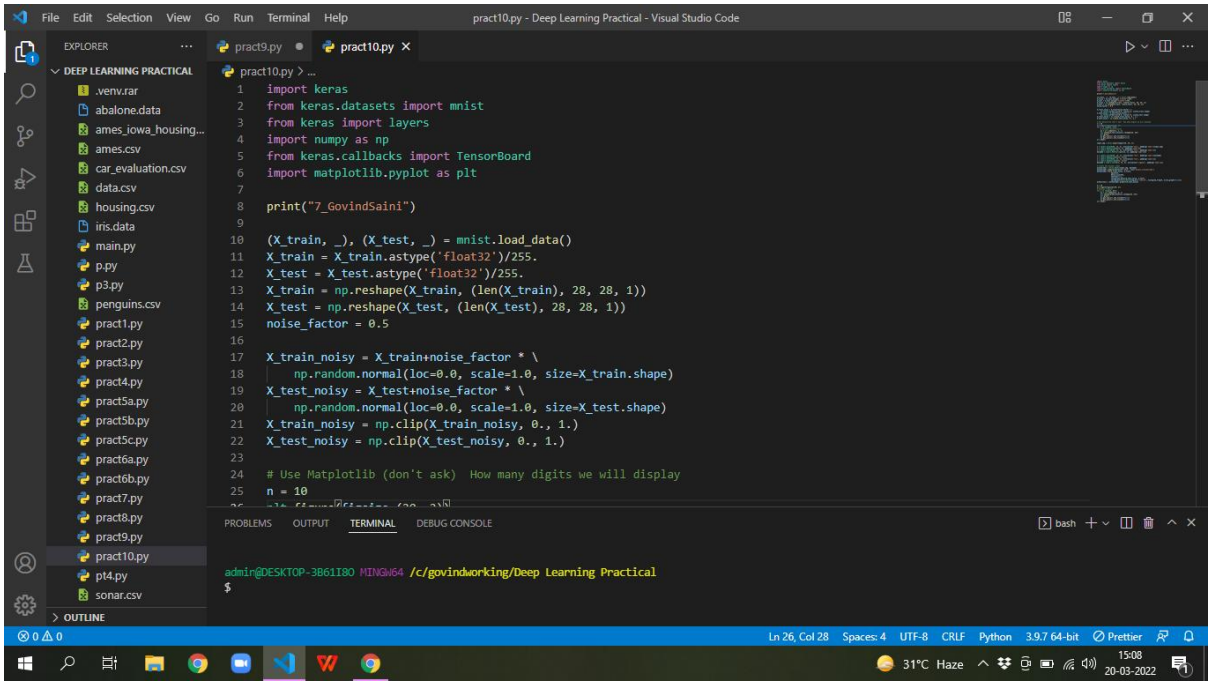
Target Summary: * Shape: (10) * range: (0.0, 1.0)

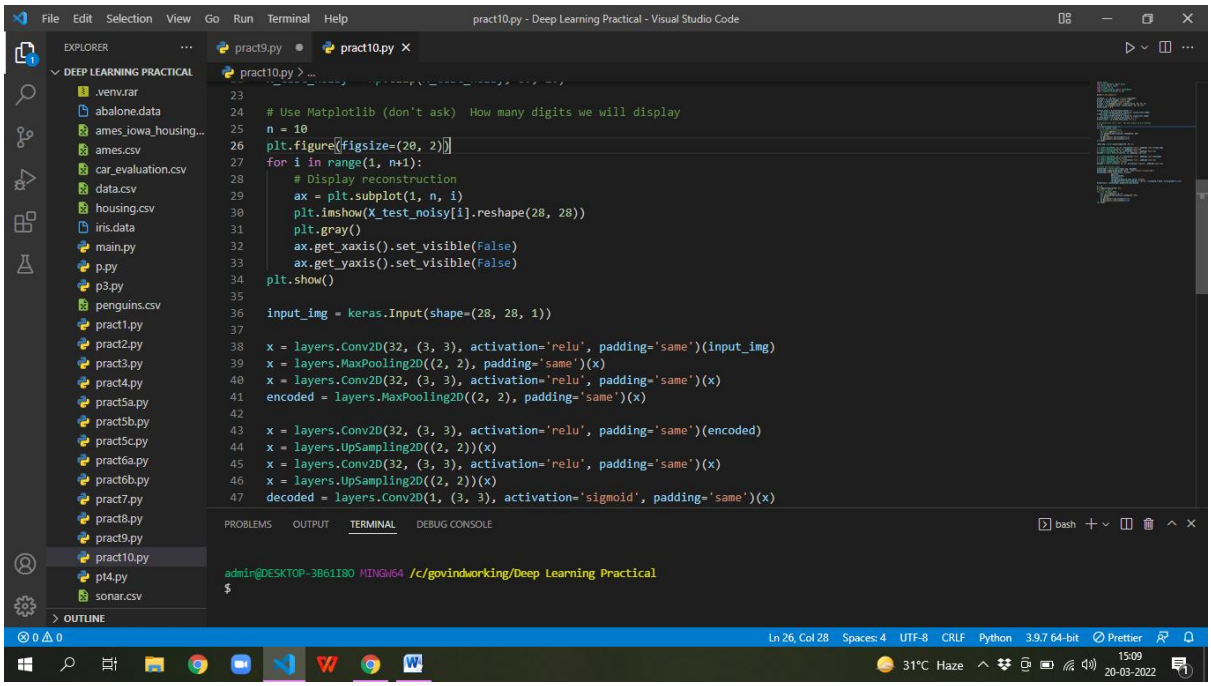
Result Description:-

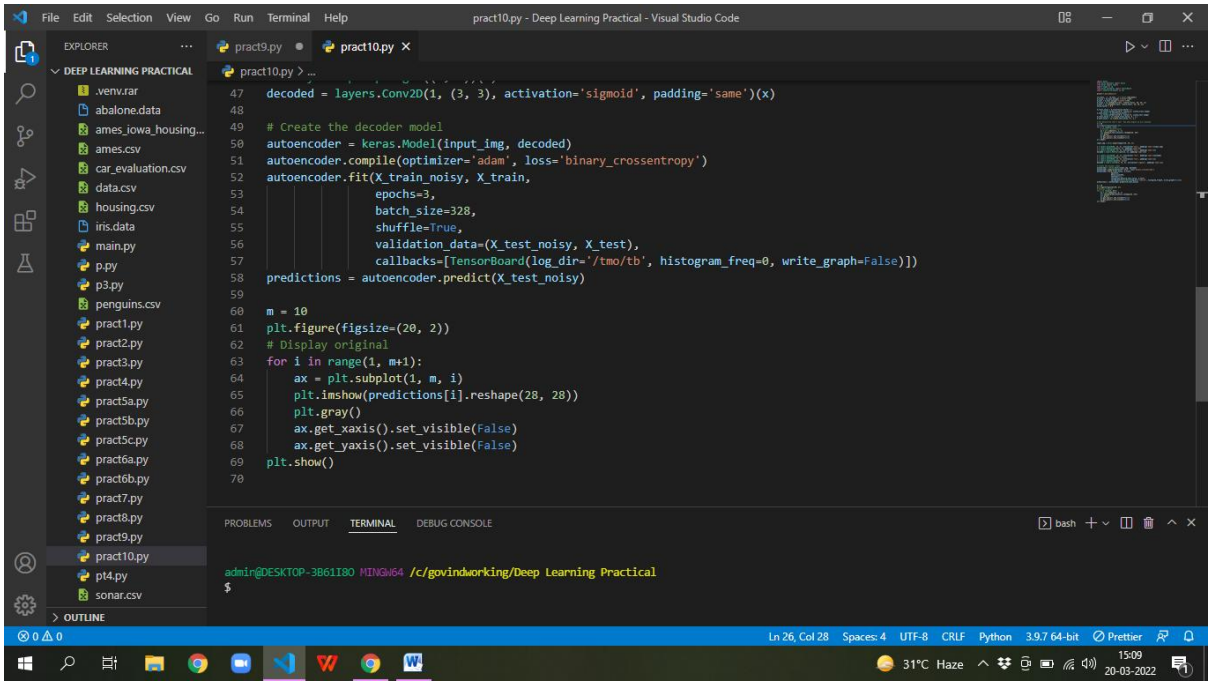
Now, let's ~~take~~ take a look at the results. Top, the noisy digits fed to the network, and bottom, the digits are reconstructed by the network.
On the top we have the original MNIST digit we added noise to while on the bottom we have the output of the denoising autoencoder - we can clearly see that the denoising autoencoder was able to recover the original digital from the image while removing the noise.
More advanced denoising autoencoder can be used to automatically pre-process images to facilitate better OCR accuracy.

Govind

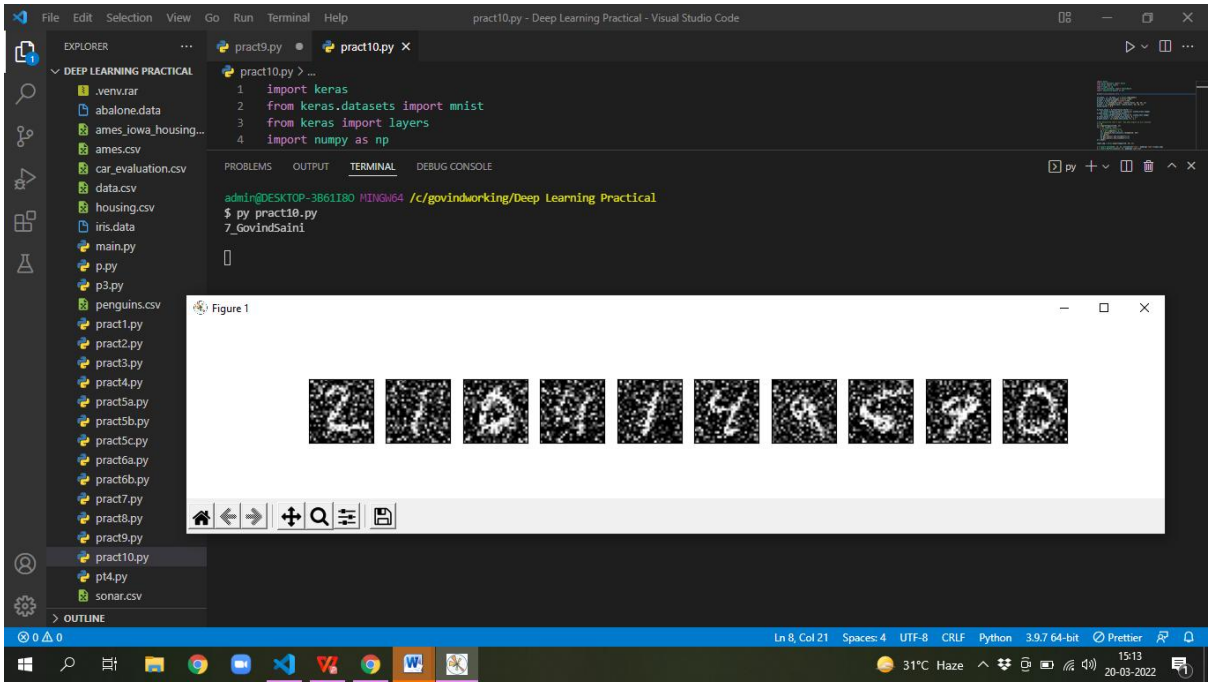
Code :

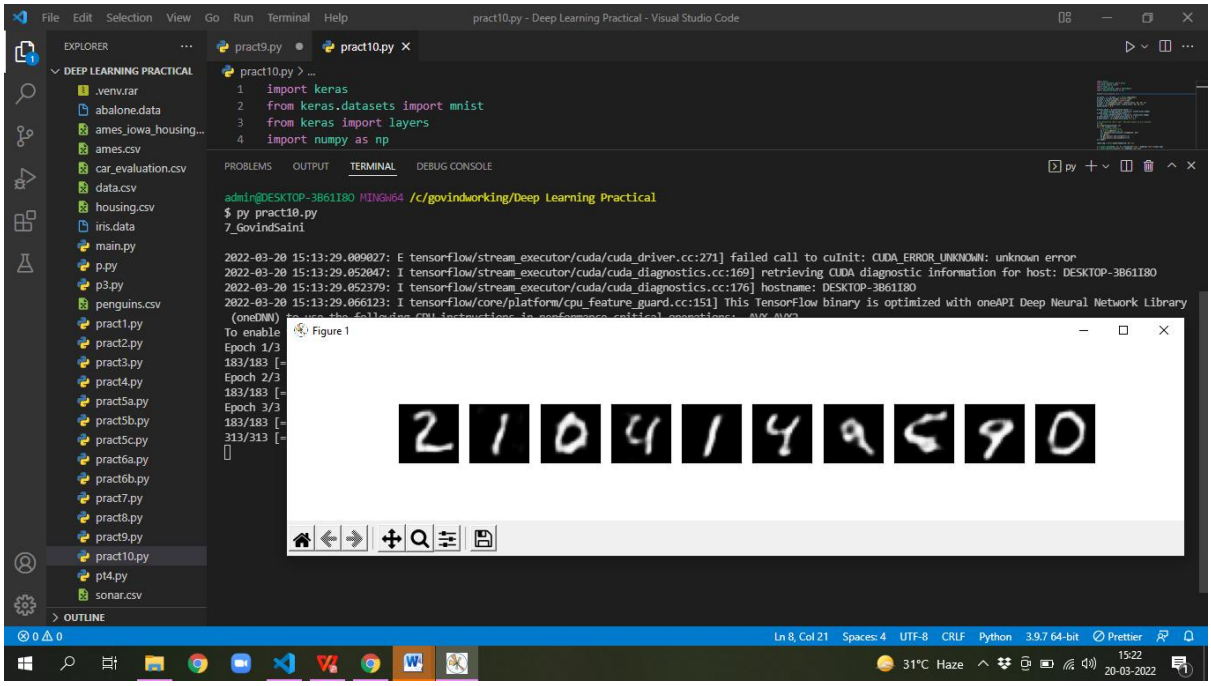

File Edit Selection View Go Run Terminal Help
pract10.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
pract9.py
pract10.py
import keras
from keras.datasets import mnist
from keras import layers
import numpy as np
from keras.callbacks import TensorBoard
import matplotlib.pyplot as plt
print("7_GovindSaini")
(X_train, _), (X_test, _) = mnist.load_data()
X_train = X_train.astype('float32')/255.
X_test = X_test.astype('float32')/255.
X_train = np.reshape(X_train, (len(X_train), 28, 28, 1))
X_test = np.reshape(X_test, (len(X_test), 28, 28, 1))
noise_factor = 0.5
X_train_noisy = X_train+noise_factor * \
np.random.normal(loc=0.0, scale=1.0, size=X_train.shape)
X_test_noisy = X_test+noise_factor * \
np.random.normal(loc=0.0, scale=1.0, size=X_test.shape)
X_train_noisy = np.clip(X_train_noisy, 0., 1.)
X_test_noisy = np.clip(X_test_noisy, 0., 1.)
# Use Matplotlib (don't ask) How many digits we will display
n = 10
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
admin@DESKTOP-3B61I8O MINGW64 /c/govindworking/Deep Learning Practical
$
Ln 26, Col 28 Spaces: 4 UTF-8 CRLF Python 3.9.7 64-bit Prettier
31°C Haze
15:08
20-03-2022


File Edit Selection View Go Run Terminal Help
pract10.py - Deep Learning Practical - Visual Studio Code
# Use Matplotlib (don't ask) How many digits we will display
n = 10
plt.figure(figsize=(20, 2))
for i in range(1, n+1):
# Display reconstruction
ax = plt.subplot(1, n, i)
plt.imshow(X_test_noisy[i].reshape(28, 28))
plt.gray()
ax.get_xaxis().set_visible(False)
ax.get_yaxis().set_visible(False)
plt.show()
input_img = keras.Input(shape=(28, 28, 1))
x = layers.Conv2D(32, (3, 3), activation='relu', padding='same')(input_img)
x = layers.MaxPooling2D((2, 2), padding='same')(x)
x = layers.Conv2D(32, (3, 3), activation='relu', padding='same')(x)
encoded = layers.MaxPooling2D((2, 2), padding='same')(x)
x = layers.Conv2D(32, (3, 3), activation='relu', padding='same')(encoded)
x = layers.UpSampling2D((2, 2))(x)
x = layers.Conv2D(32, (3, 3), activation='relu', padding='same')(x)
x = layers.UpSampling2D((2, 2))(x)
decoded = layers.Conv2D(1, (3, 3), activation='sigmoid', padding='same')(x)
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
admin@DESKTOP-3B61I8O MINGW64 /c/govindworking/Deep Learning Practical
$
Ln 26, Col 28 Spaces: 4 UTF-8 CRLF Python 3.9.7 64-bit Prettier
31°C Haze
15:09
20-03-2022


pract10.py - Deep Learning Practical - Visual Studio Code
DEEP LEARNING PRACTICAL
decoded = layers.Conv2D(1, (3, 3), activation='sigmoid', padding='same')(x)
# Create the decoder model
autoencoder = keras.Model(input_img, decoded)
autoencoder.compile(optimizer='adam', loss='binary_crossentropy')
autoencoder.fit(X_train_noisy, X_train,
epochs=3,
batch_size=328,
shuffle=True,
validation_data=(X_test_noisy, X_test),
callbacks=[TensorBoard(log_dir='/tmo/tb', histogram_freq=0, write_graph=False)])
predictions = autoencoder.predict(X_test_noisy)
m = 10
plt.figure(figsize=(20, 2))
# Display original
for i in range(1, m+1):
ax = plt.subplot(1, m, i)
plt.imshow(predictions[i].reshape(28, 28))
plt.gray()
ax.get_xaxis().set_visible(False)
ax.get_yaxis().set_visible(False)
plt.show()
admin@DESKTOP-3B61I8O MINGW64 /c/govindworking/Deep Learning Practical
$

Output :


pract10.py - Deep Learning Practical - Visual Studio Code
import keras
from keras.datasets import mnist
from keras import layers
import numpy as np
admin@DESKTOP-3B61I8O MINGW64 /c/govindworking/Deep Learning Practical
$ py pract10.py
7_GovindSaini
Figure 1


File Edit Selection View Go Run Terminal Help
pract10.py - Deep Learning Practical - Visual Studio Code
EXPLORER
DEEP LEARNING PRACTICAL
pract9.py
pract10.py
import keras
from keras.datasets import mnist
from keras import layers
import numpy as np
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
admin@DESKTOP-3B61I8O MINGW64 /c/govindworking/Deep Learning Practical
$ py pract10.py
7_GovindSaini
2022-03-20 15:13:29.009027: E tensorflow/stream_executor/cuda/cuda_driver.cc:271] failed call to cuInit: CUDA_ERROR_UNKNOWN: unknown error
2022-03-20 15:13:29.052047: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:169] retrieving CUDA diagnostic information for host: DESKTOP-3B61I8O
2022-03-20 15:13:29.052379: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:176] hostname: DESKTOP-3B61I8O
2022-03-20 15:13:29.066123: I tensorflow/core/platform/cpu_feature_guard.cc:151] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (oneDNN)
To enable
Epoch 1/3
183/183 [=
Epoch 2/3
183/183 [=
Epoch 3/3
183/183 [=
313/313 [=
Figure 1
OUTLINE
Ln 8, Col 21 Spaces: 4 UTF-8 CRLF Python 3.9.7 64-bit Prettier
31°C Haze
15:22
20-03-2022