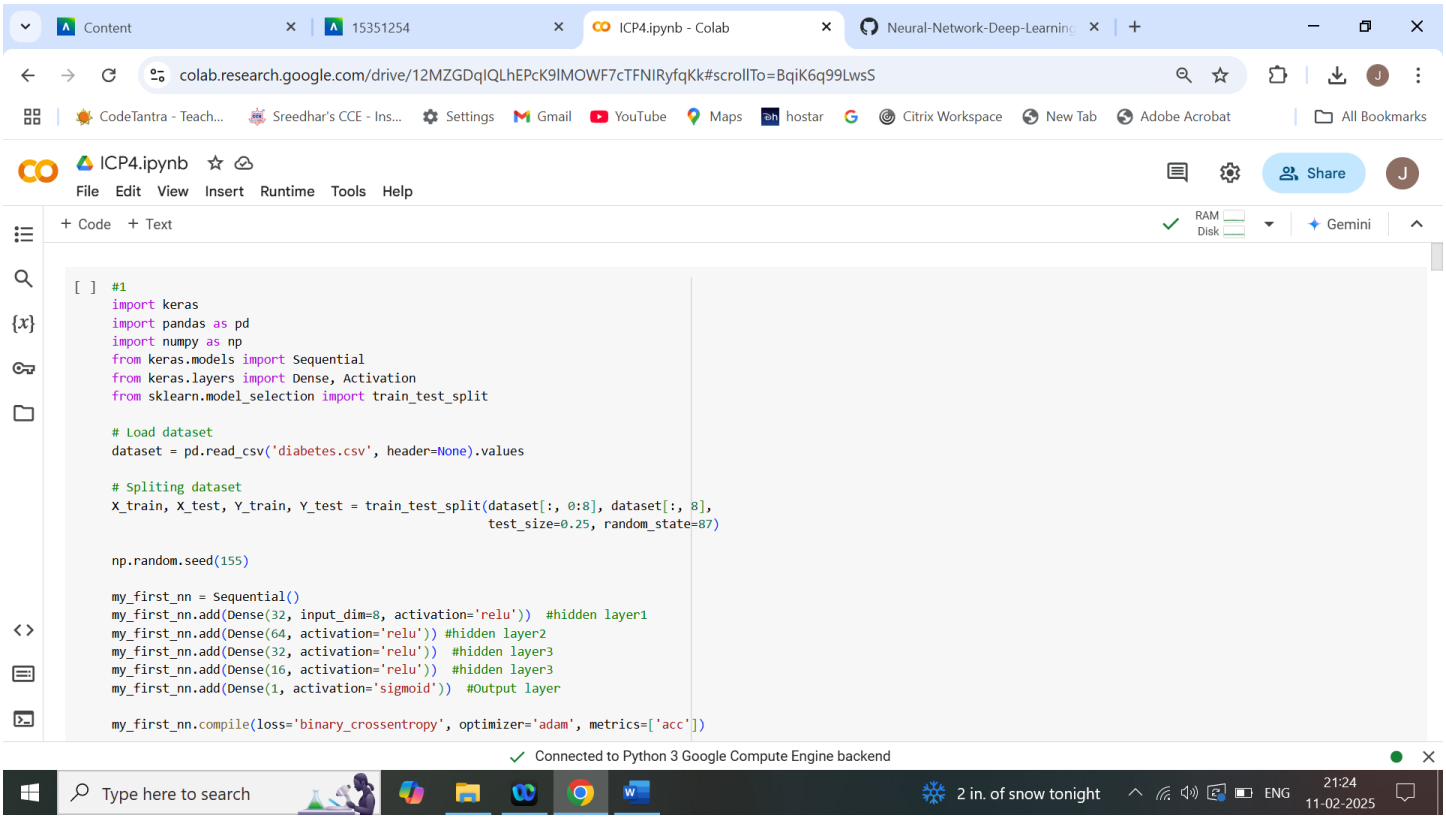
NNDL: ICP4

Jyothi Kiran Boddeda
700769023

GITHUB LINK : https://github.com/jyothikiranboddeda/Neural-Network-Deep-Learning.git

VIDEO LINK :
https://drive.google.com/file/d/1ifu6zSTop0harDN7IHqeI7e_WgKrPaGz/view?usp=sharing

1. Use the use case in the class: a. Add more Dense layers to the existing code and check how the accuracy changes



Content | 15351254 | ICP4.ipynb - Colab | Neural-Network-Deep-Learning

colab.research.google.com/drive/12MZGDqIQLhEPcK9lMOWF7cTFNIRyfqKk#scrollTo=BqiK6q99LwsS

CodeTantra - Teach... | Sreedhar's CCE - Ins... | Settings | Gmail | YouTube | Maps | hostar | Citrix Workspace | New Tab | Adobe Acrobat | All Bookmarks

ICP4.ipynb

File Edit View Insert Runtime Tools Help

Share

+ Code + Text

RAM Disk | Gemini

```
[ ] # Spliting dataset
X_train, X_test, Y_train, Y_test = train_test_split(dataset[:, 0:8], dataset[:, 8],
                                                    test_size=0.25, random_state=87)

    np.random.seed(155)

    my_first_nn = Sequential()
    my_first_nn.add(Dense(32, input_dim=8, activation='relu'))  #hidden layer1
    my_first_nn.add(Dense(64, activation='relu')) #hidden layer2
    my_first_nn.add(Dense(32, activation='relu'))  #hidden layer3
    my_first_nn.add(Dense(16, activation='relu'))  #hidden layer3
    my_first_nn.add(Dense(1, activation='sigmoid'))  #Output layer

    my_first_nn.compile(loss='binary_crossentropy', optimizer='adam', metrics=['acc'])

    my_first_nn_fitted = my_first_nn.fit(X_train, Y_train, epochs=100, initial_epoch=0, validation_data=(X_test, Y_test))

    print(my_first_nn.summary())

    # Evaluating model performance
    test_loss, test_acc = my_first_nn.evaluate(X_test, Y_test)
    print(f"Test Accuracy: {test_acc:.4f}")
```

```
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 6ms/step - acc: 0.7825 - loss: 0.4649 - val_acc: 0.6823 - val_loss: 0.6881
Epoch 84/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 6ms/step - acc: 0.7650 - loss: 0.4688 - val_acc: 0.6979 - val_loss: 0.6029
```

Connected to Python 3 Google Compute Engine backend

Type here to search | ENG 21:24 11-02-2025

Content | 15351254 | ICP4.ipynb - Colab | Neural-Network-Deep-Learning

colab.research.google.com/drive/12MZGDqIQLhEPcK9lMOWF7cTFNIRyfqKk#scrollTo=BqiK6q99LwsS

CodeTantra - Teach... | Sreedhar's CCE - Ins... | Settings | Gmail | YouTube | Maps | hostar | Citrix Workspace | New Tab | Adobe Acrobat | All Bookmarks

ICP4.ipynb

File Edit View Insert Runtime Tools Help

Share

+ Code + Text

RAM Disk | Gemini

```
[ ] Epoch 98/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step - acc: 0.7796 - loss: 0.4390 - val_acc: 0.6927 - val_loss: 0.6219
Epoch 99/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 6ms/step - acc: 0.7893 - loss: 0.4387 - val_acc: 0.6979 - val_loss: 0.6468
Epoch 100/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 6ms/step - acc: 0.7789 - loss: 0.4518 - val_acc: 0.6927 - val_loss: 0.6410
Model: "sequential_2"
```

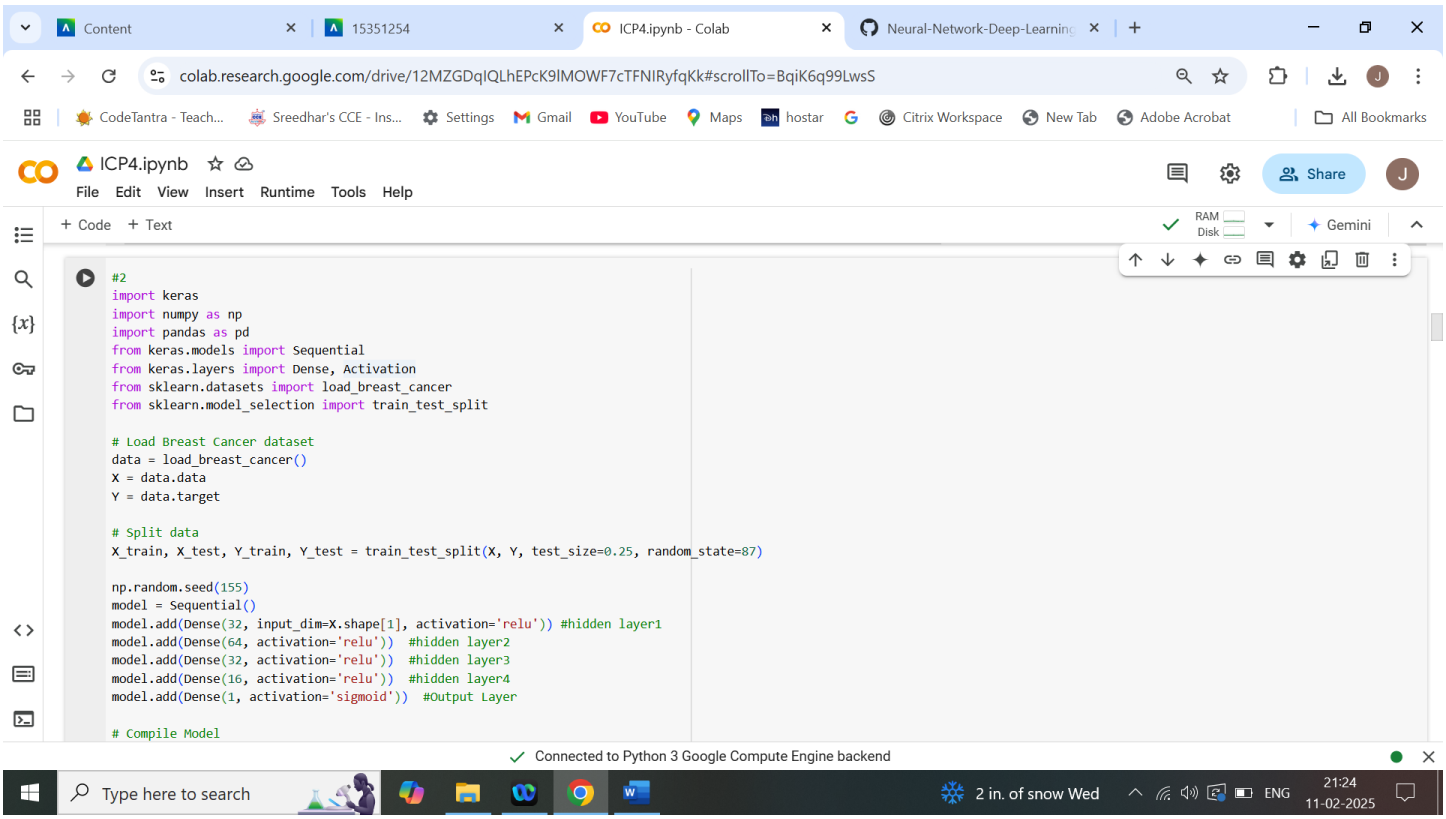
Layer (type)	Output Shape	Param #
dense_4 (Dense)	(None, 32)	288
dense_5 (Dense)	(None, 64)	2,112
dense_6 (Dense)	(None, 32)	2,080
dense_7 (Dense)	(None, 16)	528
dense_8 (Dense)	(None, 1)	17

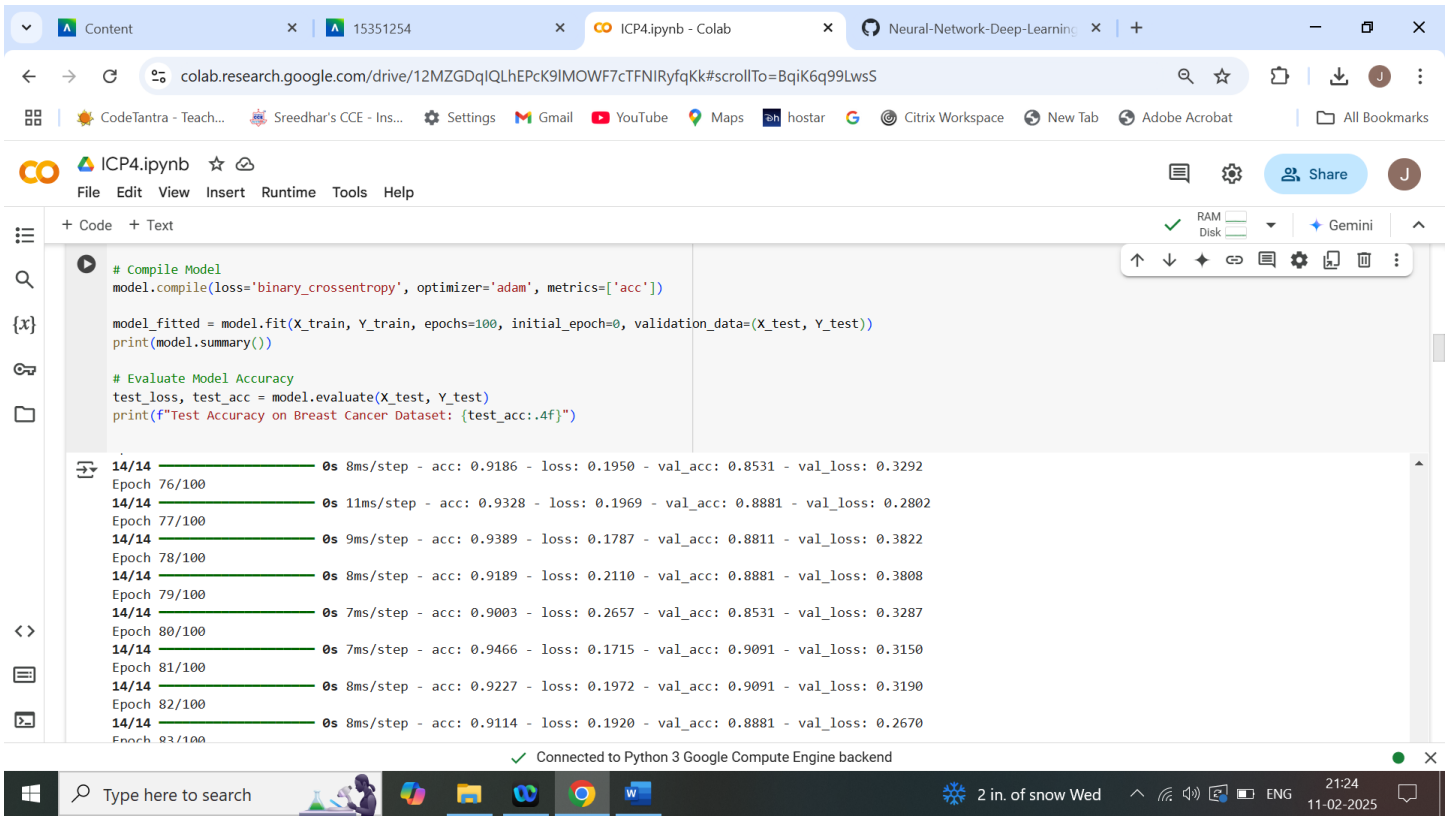
```
Total params: 15,077 (58.90 KB)
Trainable params: 5,025 (19.63 KB)
Non-trainable params: 0 (0.00 B)
Optimizer params: 10,052 (39.27 KB)
None
6/6 ━━━━━━━━━━━━━━━━━━━━ 0s 6ms/step - acc: 0.6839 - loss: 0.6653
Test Accuracy: 0.6927
```

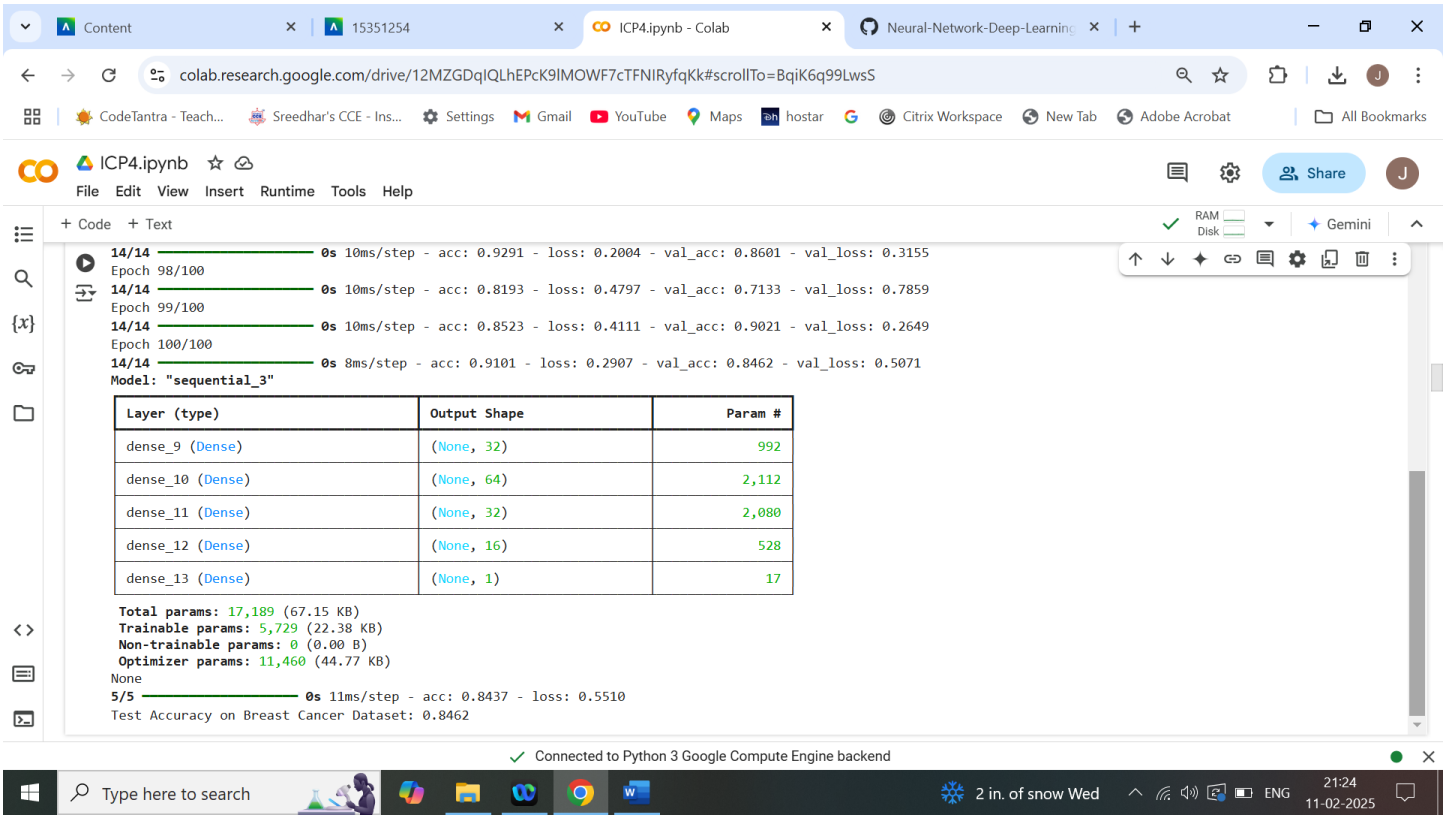
Connected to Python 3 Google Compute Engine backend

Type here to search | 2 in. of snow Wed | ENG 21:24 11-02-2025

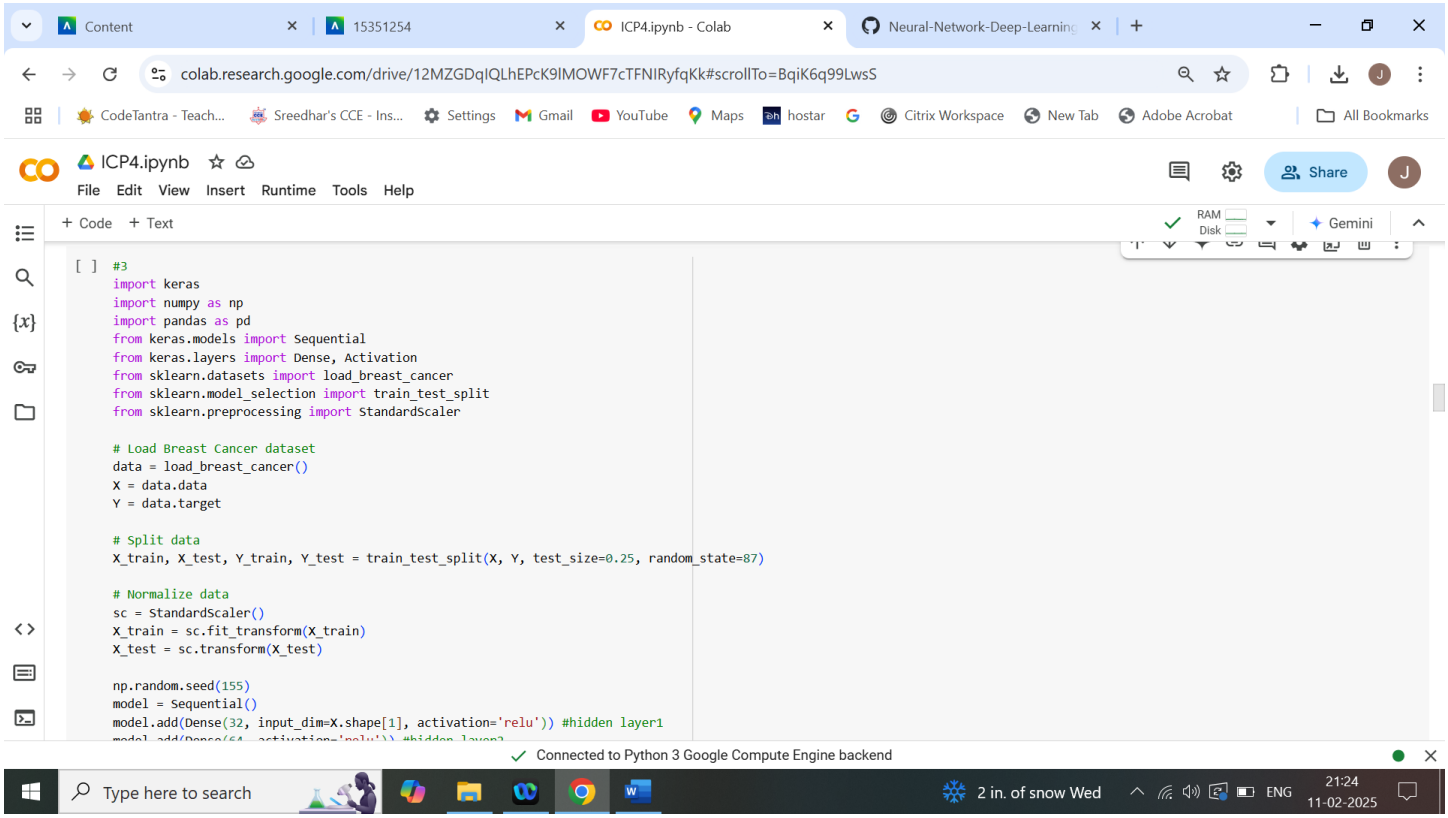
2. Change the data source to Breast Cancer dataset * available in the source code folder and make required changes. Report accuracy of the model.

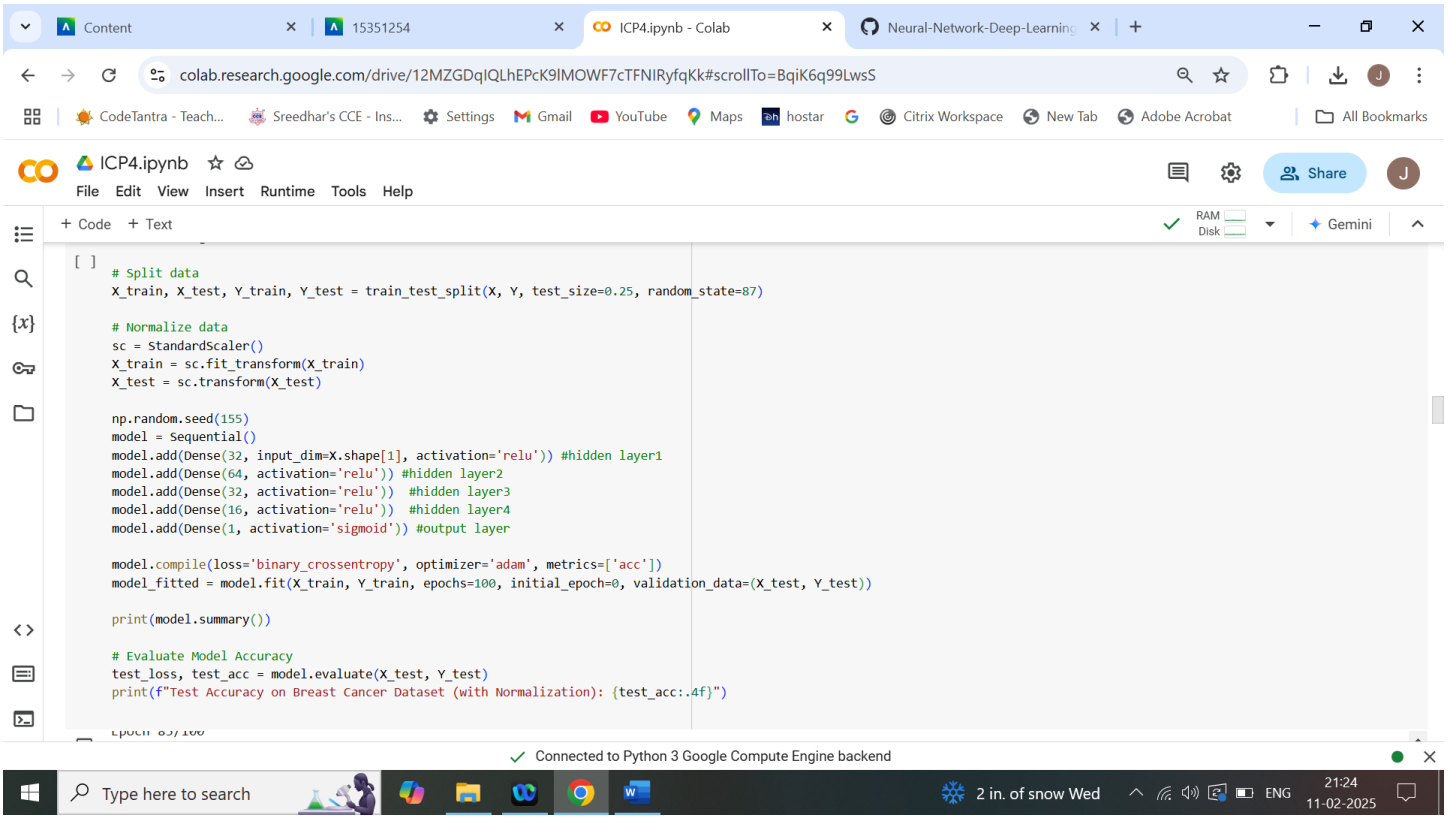


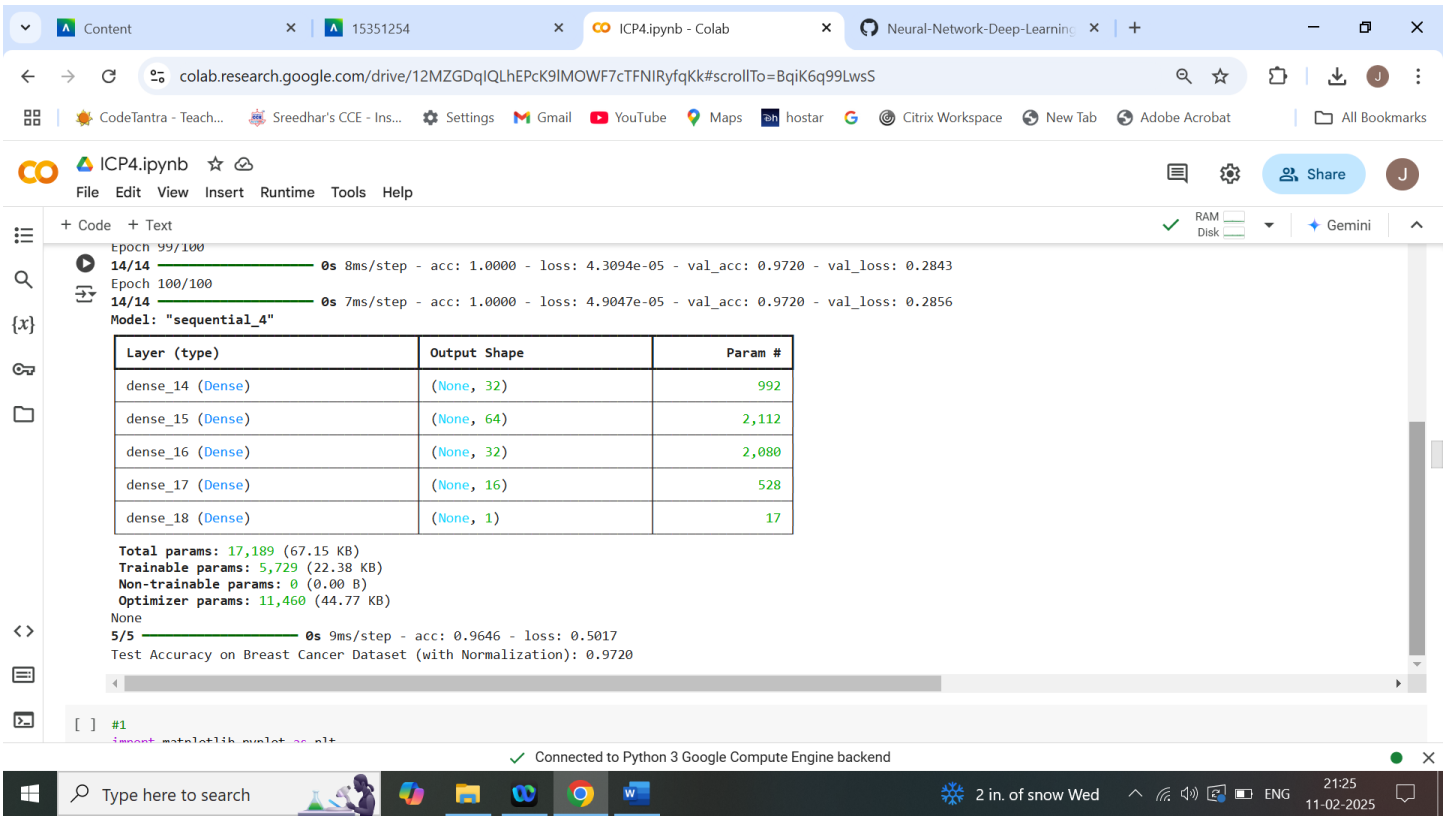

Content
15351254
ICP4.ipynb - Colab
Neural-Network-Deep-Learning
colab.research.google.com/drive/12MZGDqIQLhEPcK9IMOWF7cTFNIRyfqKk#scrollTo=BqiK6q99LwsS
ICP4.ipynb
File Edit View Insert Runtime Tools Help
Share
+ Code + Text
RAM Disk
Gemini
# Compile Model
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['acc'])
model_fitted = model.fit(X_train, Y_train, epochs=100, initial_epoch=0, validation_data=(X_test, Y_test))
print(model.summary())
# Evaluate Model Accuracy
test_loss, test_acc = model.evaluate(X_test, Y_test)
print(f"Test Accuracy on Breast Cancer Dataset: {test_acc:.4f}")
14/14 0s 8ms/step - acc: 0.9186 - loss: 0.1950 - val_acc: 0.8531 - val_loss: 0.3292
Epoch 76/100
14/14 0s 11ms/step - acc: 0.9328 - loss: 0.1969 - val_acc: 0.8881 - val_loss: 0.2802
Epoch 77/100
14/14 0s 9ms/step - acc: 0.9389 - loss: 0.1787 - val_acc: 0.8811 - val_loss: 0.3822
Epoch 78/100
14/14 0s 8ms/step - acc: 0.9189 - loss: 0.2110 - val_acc: 0.8881 - val_loss: 0.3808
Epoch 79/100
14/14 0s 7ms/step - acc: 0.9003 - loss: 0.2657 - val_acc: 0.8531 - val_loss: 0.3287
Epoch 80/100
14/14 0s 7ms/step - acc: 0.9466 - loss: 0.1715 - val_acc: 0.9091 - val_loss: 0.3150
Epoch 81/100
14/14 0s 8ms/step - acc: 0.9227 - loss: 0.1972 - val_acc: 0.9091 - val_loss: 0.3190
Epoch 82/100
14/14 0s 8ms/step - acc: 0.9114 - loss: 0.1920 - val_acc: 0.8881 - val_loss: 0.2670
Epoch 83/100
Connected to Python 3 Google Compute Engine backend
Type here to search
2 in. of snow Wed
ENG
21:24
11-02-2025


Content
15351254
ICP4.ipynb - Colab
Neural-Network-Deep-Learning
colab.research.google.com/drive/12MZGDqIQLhEPcK9IMOWF7cTFNIRyfqKk#scrollTo=BqiK6q99LwsS
ICP4.ipynb
File Edit View Insert Runtime Tools Help
Share
+ Code + Text
RAM Disk
Gemini
14/14 0s 10ms/step - acc: 0.9291 - loss: 0.2004 - val_acc: 0.8601 - val_loss: 0.3155
Epoch 98/100
14/14 0s 10ms/step - acc: 0.8193 - loss: 0.4797 - val_acc: 0.7133 - val_loss: 0.7859
Epoch 99/100
14/14 0s 10ms/step - acc: 0.8523 - loss: 0.4111 - val_acc: 0.9021 - val_loss: 0.2649
Epoch 100/100
14/14 0s 8ms/step - acc: 0.9101 - loss: 0.2907 - val_acc: 0.8462 - val_loss: 0.5071
Model: "sequential_3"
Layer (type) Output Shape Param #
dense_9 (Dense) (None, 32) 992
dense_10 (Dense) (None, 64) 2,112
dense_11 (Dense) (None, 32) 2,080
dense_12 (Dense) (None, 16) 528
dense_13 (Dense) (None, 1) 17
Total params: 17,189 (67.15 KB)
Trainable params: 5,729 (22.38 KB)
Non-trainable params: 0 (0.00 B)
Optimizer params: 11,460 (44.77 KB)
None
5/5 0s 11ms/step - acc: 0.8437 - loss: 0.5510
Test Accuracy on Breast Cancer Dataset: 0.8462
Connected to Python 3 Google Compute Engine backend
Type here to search
2 in. of snow Wed
ENG
21:24
11-02-2025

3. Normalize the data before feeding the data to the model and check how the normalization change your accuracy (code given below).
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()

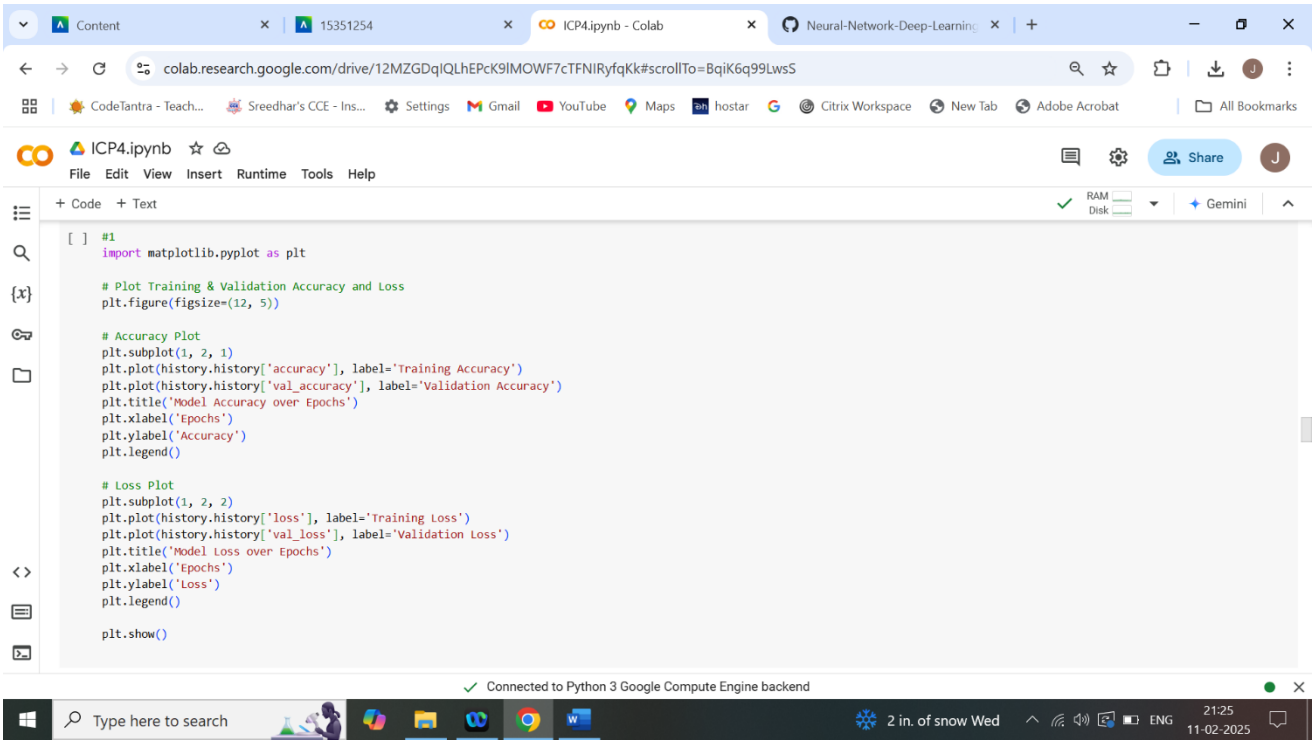



Content
15351254
ICP4.ipynb - Colab
Neural-Network-Deep-Learning
colab.research.google.com/drive/12MZGDqIQLhEPcK9IMOWF7cTFNIRyfqKk#scrollTo=BqiK6q99LwsS
ICP4.ipynb
File Edit View Insert Runtime Tools Help
Share
+ Code + Text
RAM Disk
Gemini
[ ]
Split data
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.25, random_state=87)
# Normalize data
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
np.random.seed(155)
model = Sequential()
model.add(Dense(32, input_dim=X.shape[1], activation='relu')) #hidden layer1
model.add(Dense(64, activation='relu')) #hidden layer2
model.add(Dense(32, activation='relu')) #hidden layer3
model.add(Dense(16, activation='relu')) #hidden layer4
model.add(Dense(1, activation='sigmoid')) #output layer
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['acc'])
model_fitted = model.fit(X_train, Y_train, epochs=100, initial_epoch=0, validation_data=(X_test, Y_test))
print(model.summary())
# Evaluate Model Accuracy
test_loss, test_acc = model.evaluate(X_test, Y_test)
print(f"Test Accuracy on Breast Cancer Dataset (with Normalization): {test_acc:.4f}")
Connected to Python 3 Google Compute Engine backend
Type here to search
2 in. of snow Wed
ENG
21:24
11-02-2025

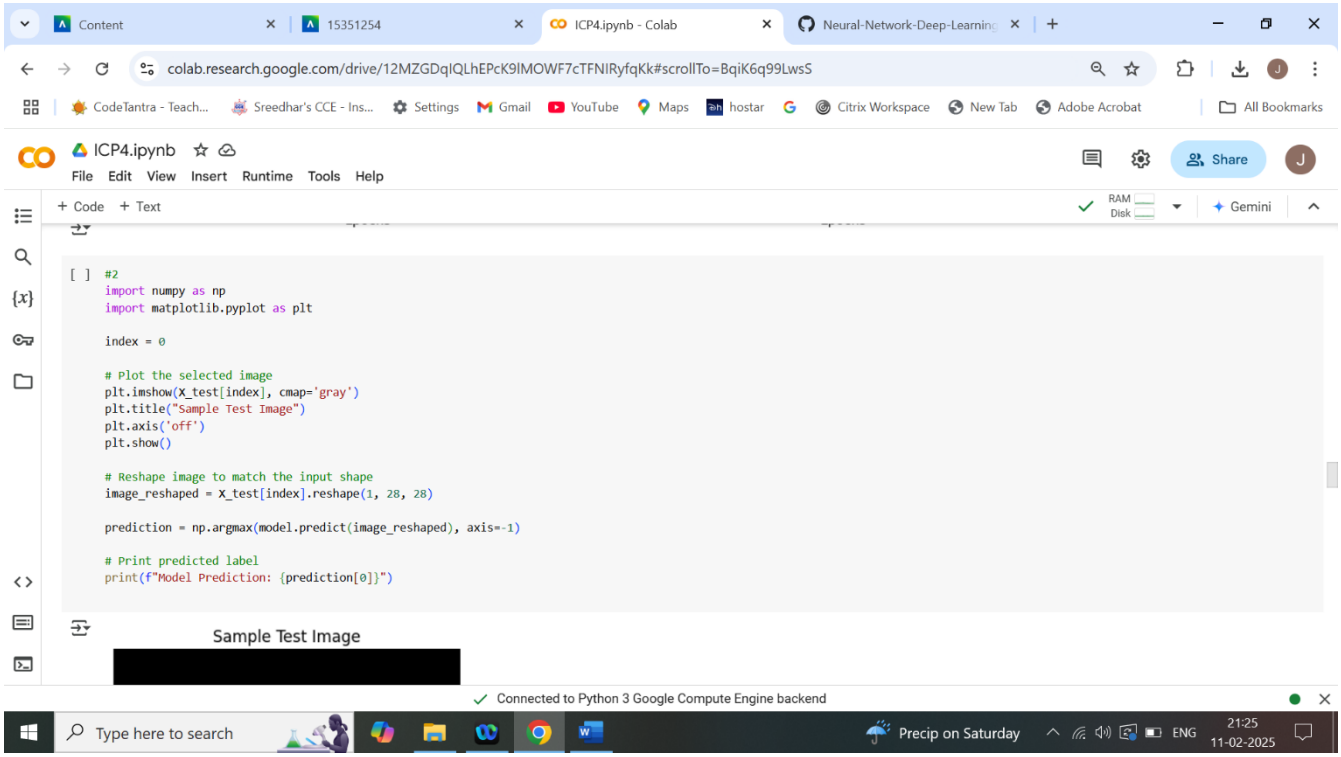

Content
15351254
ICP4.ipynb - Colab
Neural-Network-Deep-Learning
colab.research.google.com/drive/12MZGDqIQLhEPcK9IMOWF7cTFNIRyfqKk#scrollTo=BqiK6q99LwsS
ICP4.ipynb
File Edit View Insert Runtime Tools Help
Share
+ Code + Text
RAM Disk
Gemini
Epoch 99/100
14/14 0s 8ms/step - acc: 1.0000 - loss: 4.3094e-05 - val_acc: 0.9720 - val_loss: 0.2843
Epoch 100/100
14/14 0s 7ms/step - acc: 1.0000 - loss: 4.9047e-05 - val_acc: 0.9720 - val_loss: 0.2856
Model: "sequential_4"
Layer (type) Output Shape Param #
dense_14 (Dense) (None, 32) 992
dense_15 (Dense) (None, 64) 2,112
dense_16 (Dense) (None, 32) 2,080
dense_17 (Dense) (None, 16) 528
dense_18 (Dense) (None, 1) 17
Total params: 17,189 (67.15 KB)
Trainable params: 5,729 (22.38 KB)
Non-trainable params: 0 (0.00 B)
Optimizer params: 11,460 (44.77 KB)
None
5/5 0s 9ms/step - acc: 0.9646 - loss: 0.5017
Test Accuracy on Breast Cancer Dataset (with Normalization): 0.9720
[ ]
#1
Connected to Python 3 Google Compute Engine backend
Type here to search
2 in. of snow Wed
ENG
21:25
11-02-2025

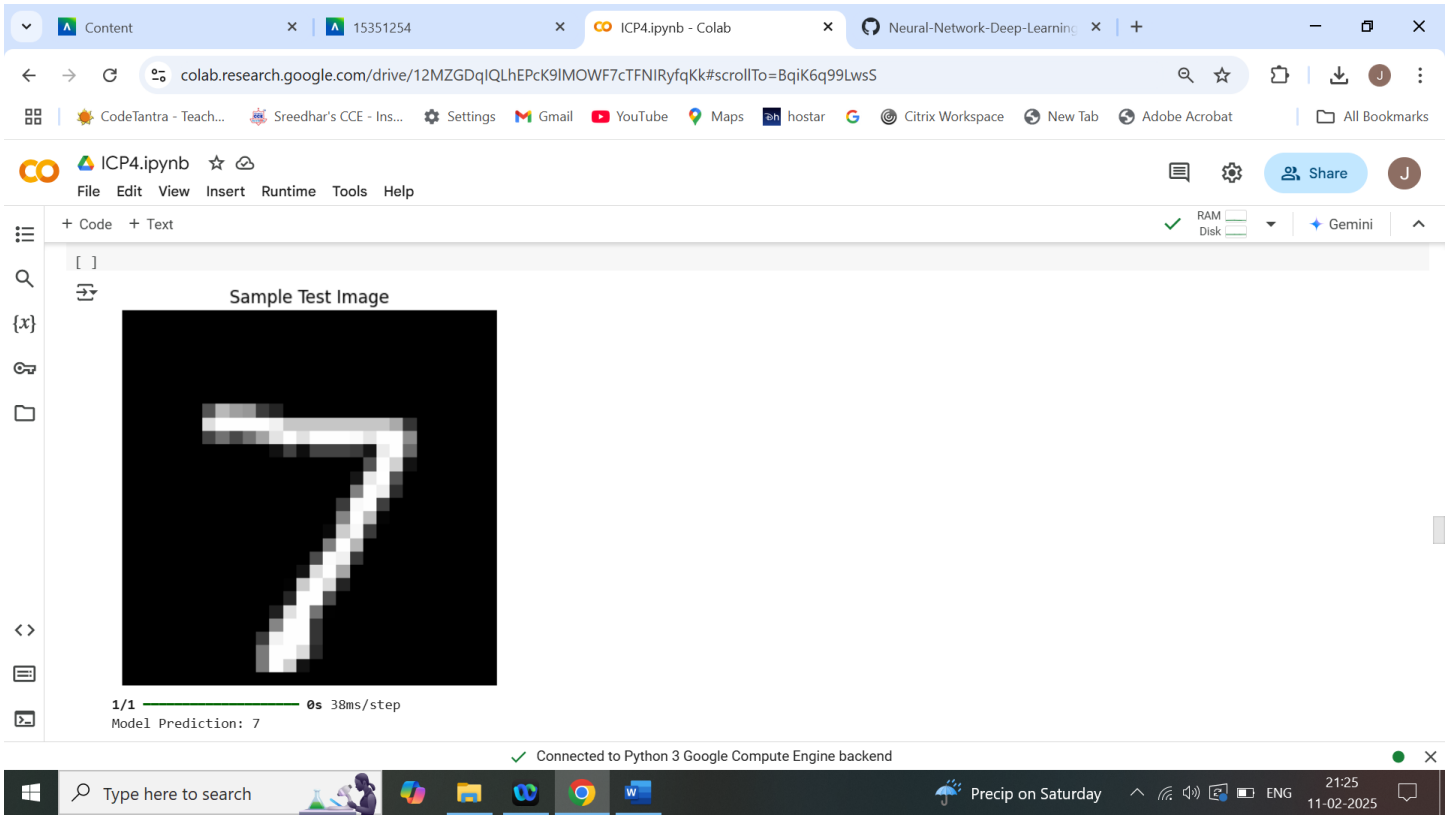
Use Image Classification on the hand written digits data set (mnist)

1. Plot the loss and accuracy for both training data and validation data using the history object in the source code.

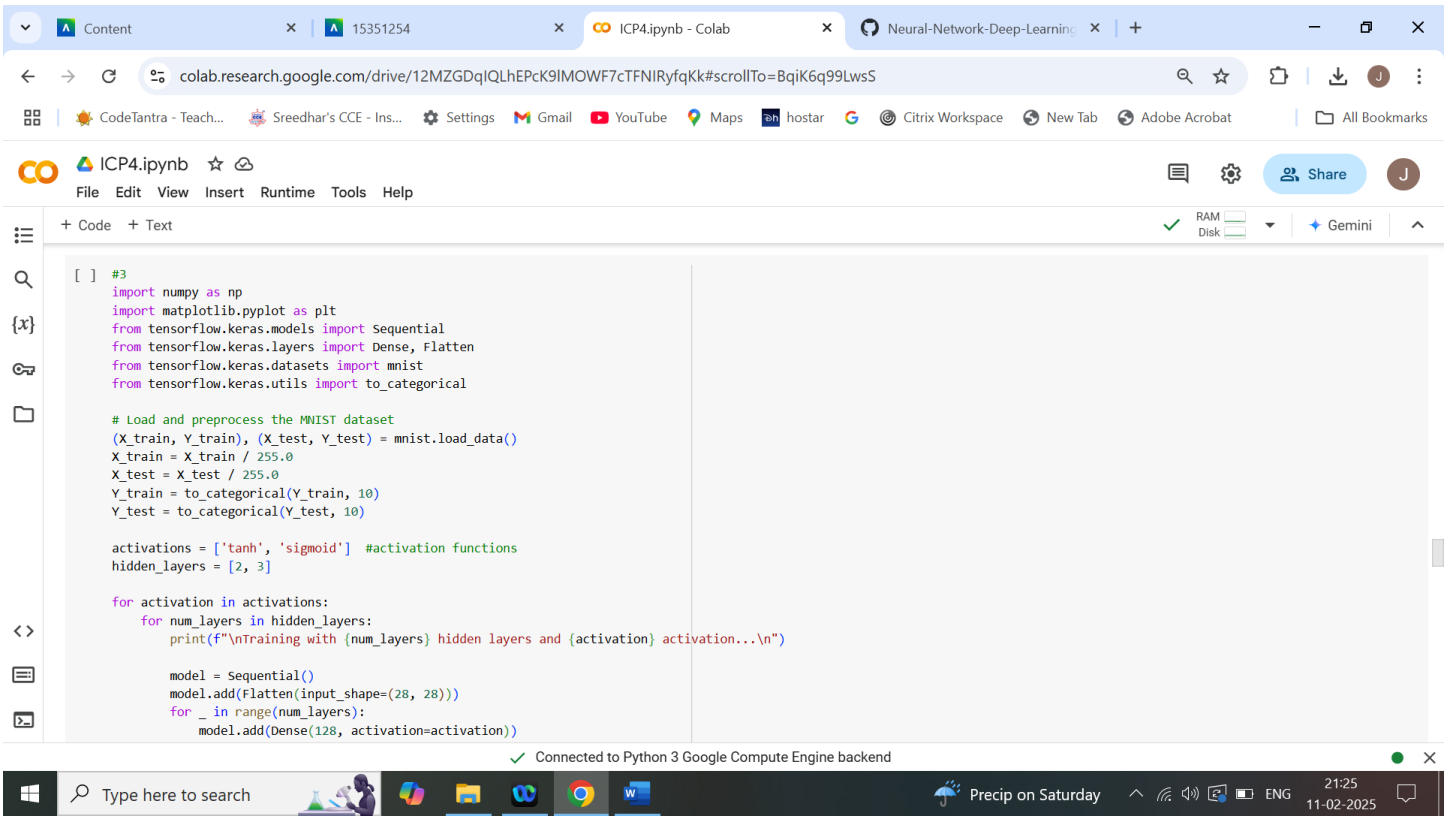


2. Plot one of the images in the test data, and then do inferencing to check what is the prediction of the model on that single image.

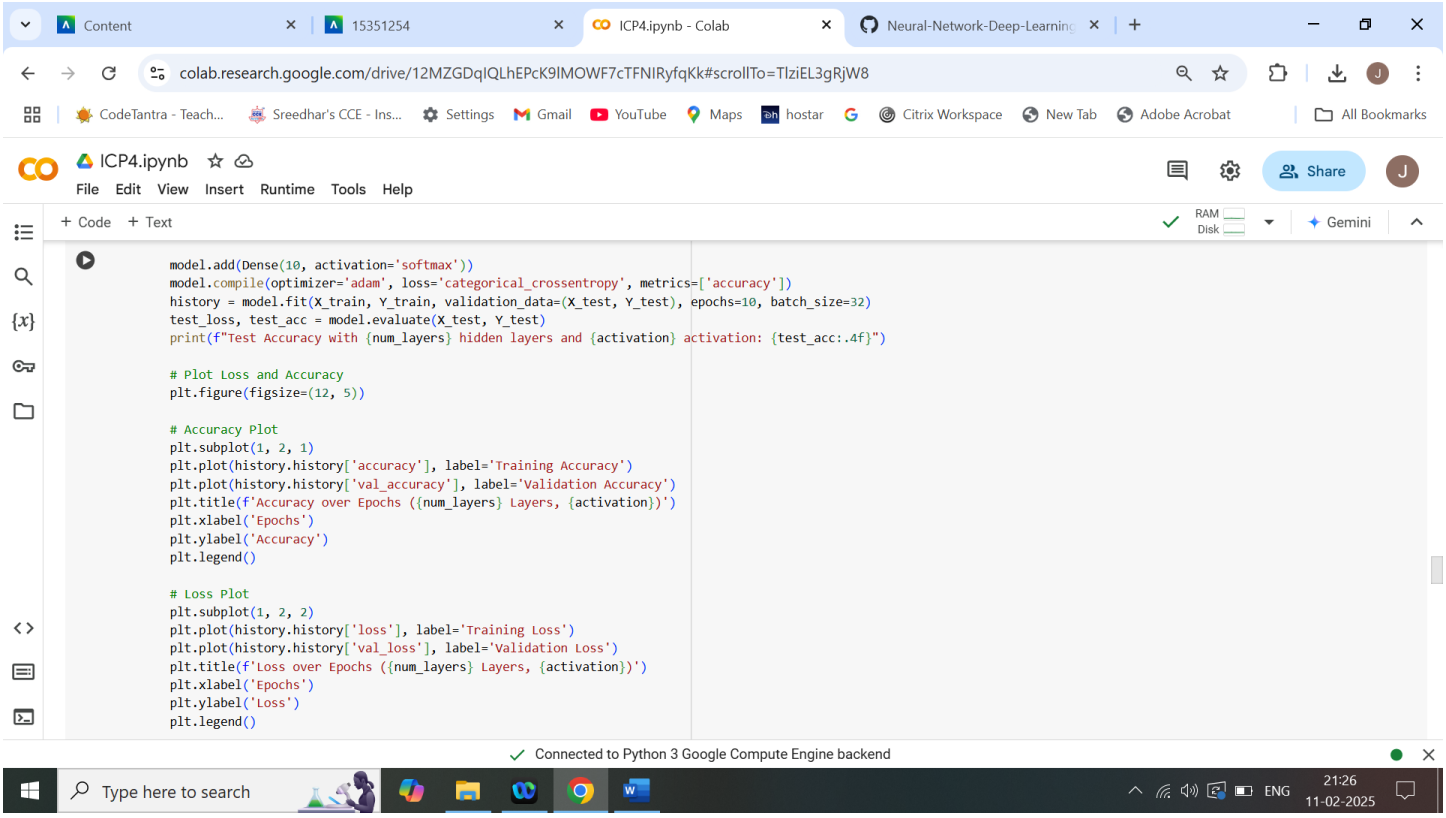




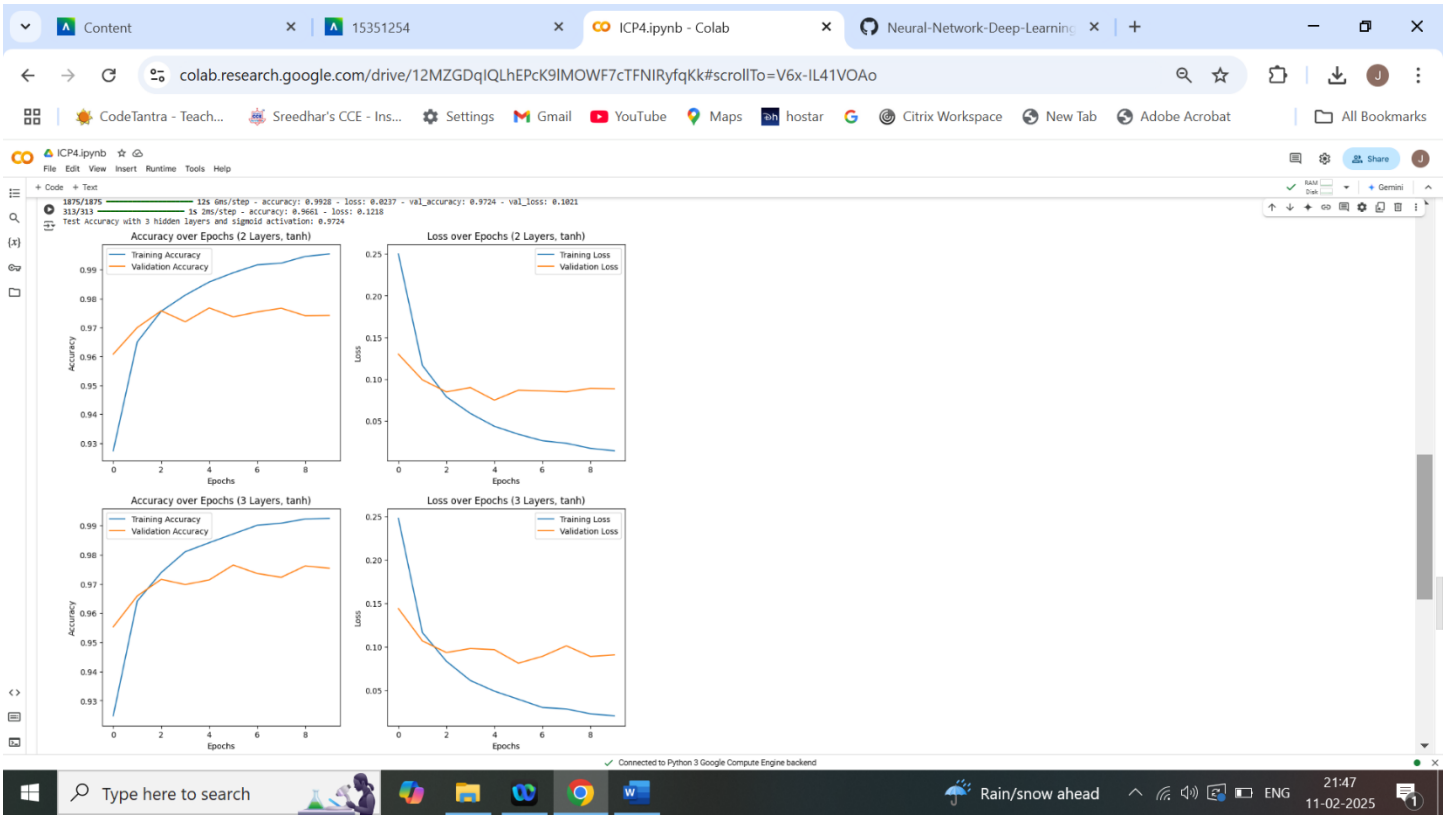
3. We had used 2 hidden layers and Relu activation. Try to change the number of hidden layer and the activation to tanh or sigmoid and see what happens

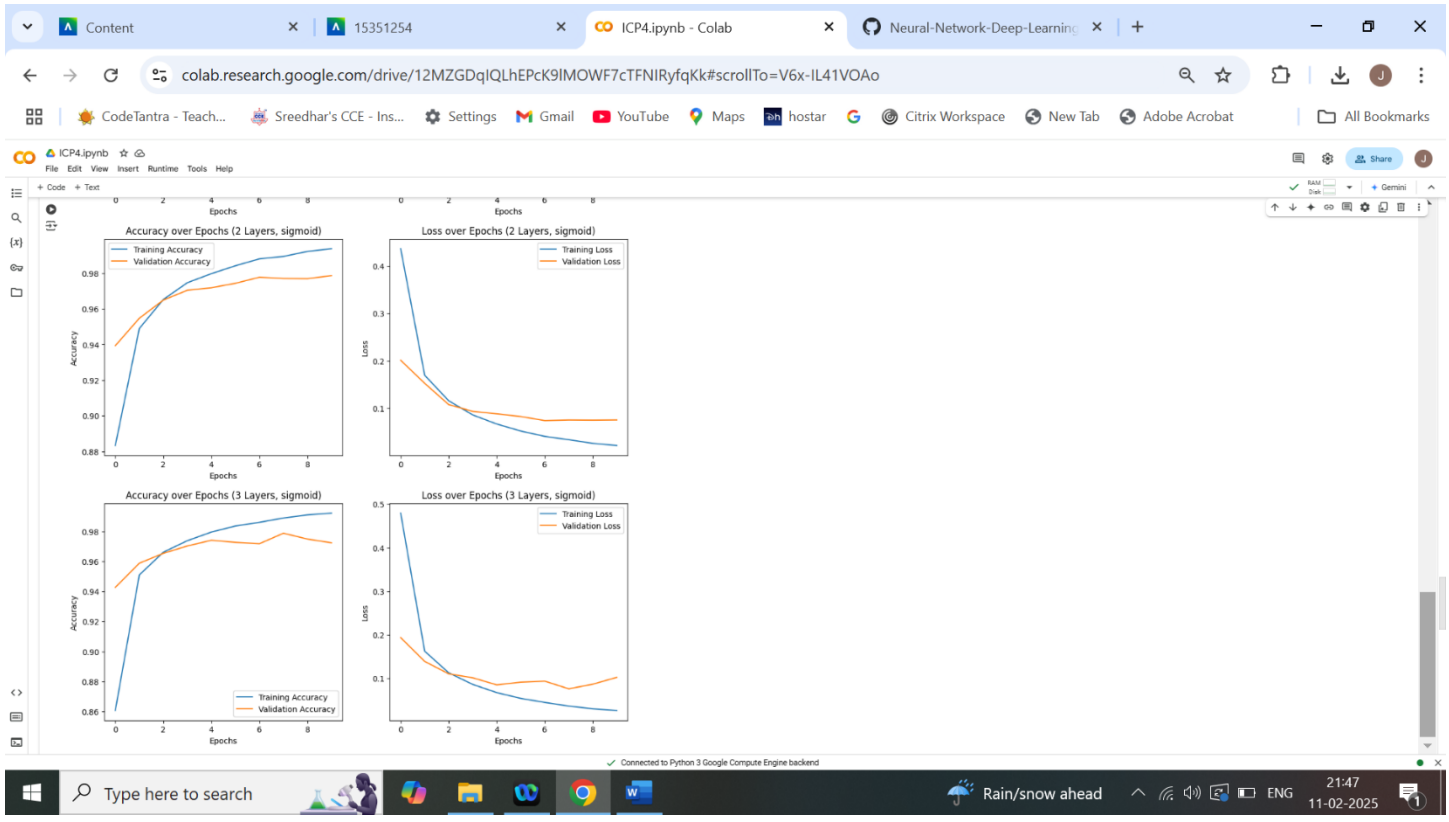


```
#3
import numpy as np
import matplotlib.pyplot as plt
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Flatten
from tensorflow.keras.datasets import mnist
from tensorflow.keras.utils import to_categorical

# Load and preprocess the MNIST dataset
(X_train, Y_train), (X_test, Y_test) = mnist.load_data()
X_train = X_train / 255.0
X_test = X_test / 255.0
Y_train = to_categorical(Y_train, 10)
Y_test = to_categorical(Y_test, 10)

activations = ['tanh', 'sigmoid']  #activation functions
hidden_layers = [2, 3]

for activation in activations:
    for num_layers in hidden_layers:
        print(f"\nTraining with {num_layers} hidden layers and {activation} activation...\n")

        model = Sequential()
        model.add(Flatten(input_shape=(28, 28)))
        for _ in range(num_layers):
            model.add(Dense(128, activation=activation))
```

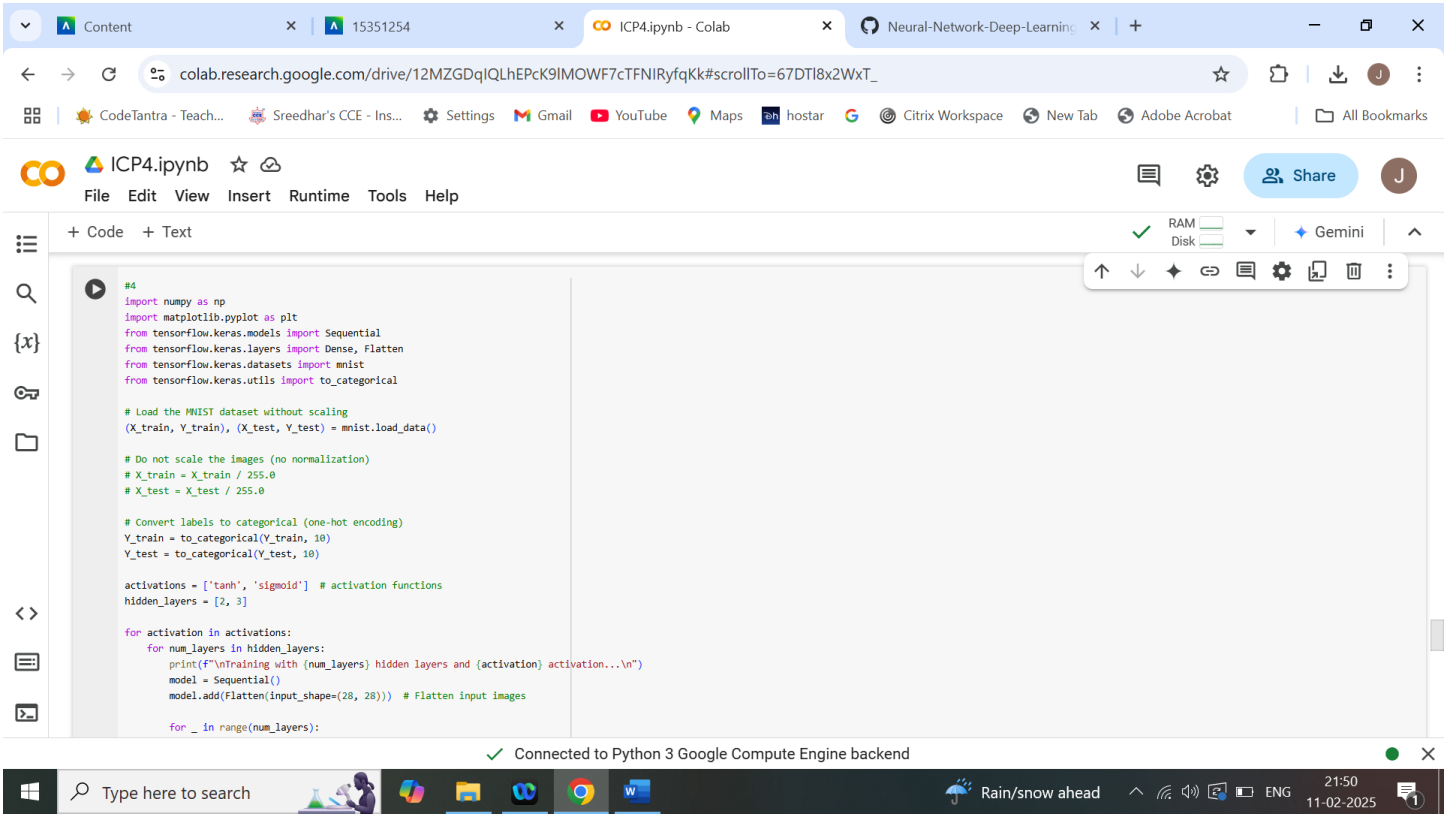


```
        model.add(Dense(10, activation='softmax'))
        model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
        history = model.fit(X_train, Y_train, validation_data=(X_test, Y_test), epochs=10, batch_size=32)
        test_loss, test_acc = model.evaluate(X_test, Y_test)
        print(f"Test Accuracy with {num_layers} hidden layers and {activation} activation: {test_acc:.4f}")

        # Plot Loss and Accuracy
        plt.figure(figsize=(12, 5))

        # Accuracy Plot
        plt.subplot(1, 2, 1)
        plt.plot(history.history['accuracy'], label='Training Accuracy')
        plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
        plt.title(f'Accuracy over Epochs ({num_layers} Layers, {activation})')
        plt.xlabel('Epochs')
        plt.ylabel('Accuracy')
        plt.legend()

        # Loss Plot
        plt.subplot(1, 2, 2)
        plt.plot(history.history['loss'], label='Training Loss')
        plt.plot(history.history['val_loss'], label='Validation Loss')
        plt.title(f'Loss over Epochs ({num_layers} Layers, {activation})')
        plt.xlabel('Epochs')
        plt.ylabel('Loss')
        plt.legend()
```

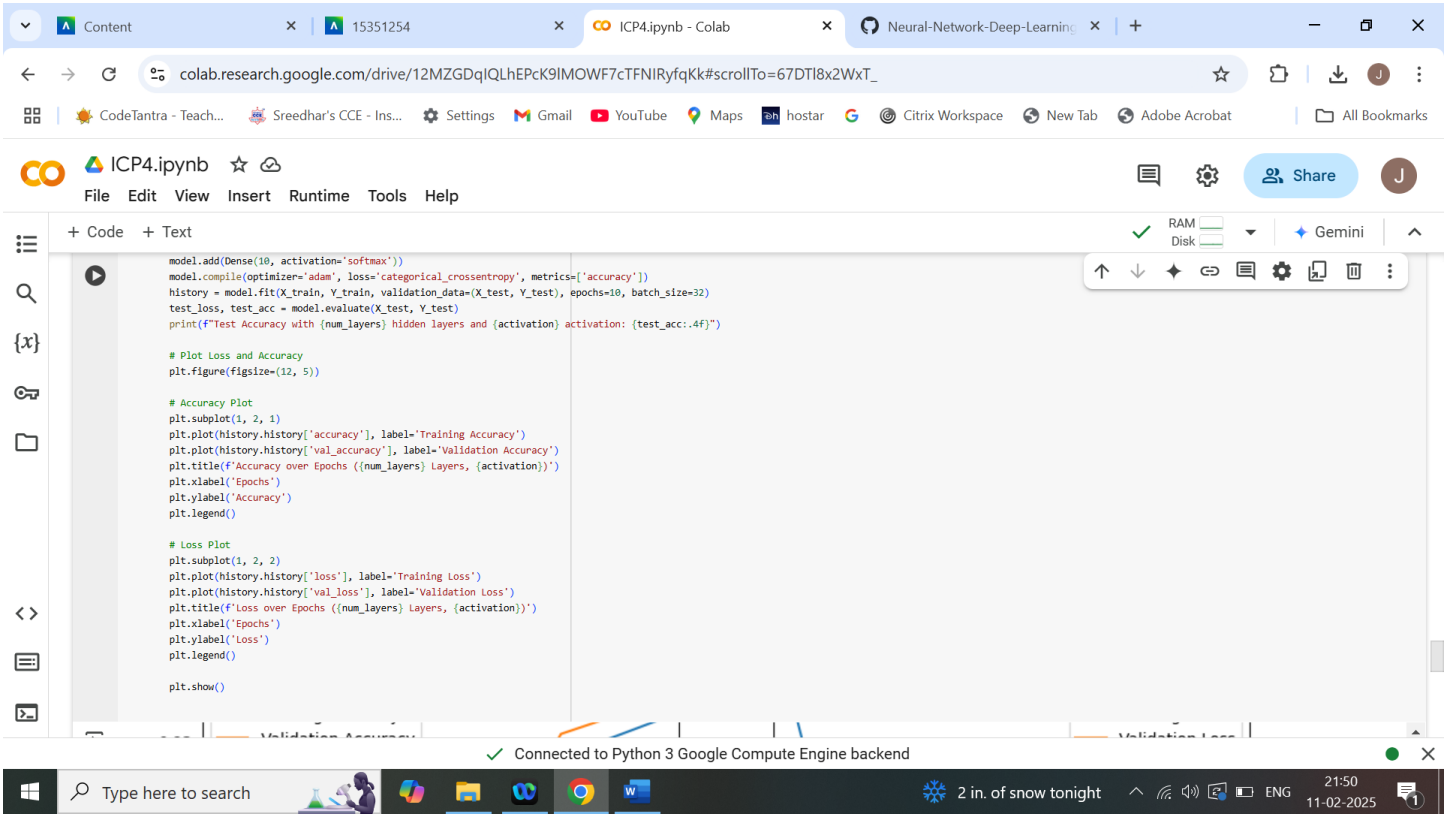

Content
15351254
ICP4.ipynb - Colab
Neural-Network-Deep-Learning
colab.research.google.com/drive/12MZGDqIQLhEPcK9IMOWF7cTFNIRyfqKk#scrollTo=V6x-IL41VOAo
CodeTantra - Teach...
Sreedhar's CCE - Ins...
Settings
Gmail
YouTube
Maps
hostar
Citrix Workspace
New Tab
Adobe Acrobat
All Bookmarks
ICP4.ipynb
File Edit View Insert Runtime Tools Help
Share
+ Code + Text
Gemini
1875/1875 12s 6ms/step - accuracy: 0.9928 - loss: 0.0237 - val_accuracy: 0.9724 - val_loss: 0.1021
313/313 1s 2ms/step - accuracy: 0.9661 - loss: 0.1218
Test Accuracy with 3 hidden layers and sigmoid activation: 0.9724
Accuracy over Epochs (2 Layers, tanh)
Training Accuracy
Validation Accuracy
Accuracy
Epochs
Loss over Epochs (2 Layers, tanh)
Training Loss
Validation Loss
Loss
Epochs
Accuracy over Epochs (3 Layers, tanh)
Training Accuracy
Validation Accuracy
Accuracy
Epochs
Loss over Epochs (3 Layers, tanh)
Training Loss
Validation Loss
Loss
Epochs
Connected to Python 3 Google Compute Engine backend
Type here to search
Rain/snow ahead
ENG
21:47
11-02-2025


Content
15351254
ICP4.ipynb - Colab
Neural-Network-Deep-Learning
colab.research.google.com/drive/12MZGDqIQLhEPcK9IMOWF7cTFNIRyfqKk#scrollTo=V6x-IL41VOAo
CodeTantra - Teach...
Sreedhar's CCE - Ins...
Settings
Gmail
YouTube
Maps
hostar
Citrix Workspace
New Tab
Adobe Acrobat
All Bookmarks
ICP4.ipynb
File Edit View Insert Runtime Tools Help
Share
+ Code + Text
Gemini
Epochs
Accuracy over Epochs (2 Layers, sigmoid)
Training Accuracy
Validation Accuracy
Accuracy
Epochs
Loss over Epochs (2 Layers, sigmoid)
Training Loss
Validation Loss
Loss
Epochs
Accuracy over Epochs (3 Layers, sigmoid)
Training Accuracy
Validation Accuracy
Accuracy
Epochs
Loss over Epochs (3 Layers, sigmoid)
Training Loss
Validation Loss
Loss
Epochs
Connected to Python 3 Google Compute Engine backend
Type here to search
Rain/snow ahead
ENG
21:47
11-02-2025

4. Run the same code without scaling the images and check the performance?



```
#4
import numpy as np
import matplotlib.pyplot as plt
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Flatten
from tensorflow.keras.datasets import mnist
from tensorflow.keras.utils import to_categorical

# Load the MNIST dataset without scaling
(X_train, Y_train), (X_test, Y_test) = mnist.load_data()

# Do not scale the images (no normalization)
# X_train = X_train / 255.0
# X_test = X_test / 255.0

# Convert labels to categorical (one-hot encoding)
Y_train = to_categorical(Y_train, 10)
Y_test = to_categorical(Y_test, 10)

activations = ['tanh', 'sigmoid']  # activation functions
hidden_layers = [2, 3]

for activation in activations:
    for num_layers in hidden_layers:
        print(f"\nTraining with {num_layers} hidden layers and {activation} activation...\n")
        model = Sequential()
        model.add(Flatten(input_shape=(28, 28)))  # Flatten input images

        for _ in range(num_layers):
```



```
        model.add(Dense(10, activation='softmax'))
        model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
        history = model.fit(X_train, Y_train, validation_data=(X_test, Y_test), epochs=10, batch_size=32)
        test_loss, test_acc = model.evaluate(X_test, Y_test)
        print(f"Test Accuracy with {num_layers} hidden layers and {activation} activation: {test_acc:.4f}")

        # Plot Loss and Accuracy
        plt.figure(figsize=(12, 5))

        # Accuracy Plot
        plt.subplot(1, 2, 1)
        plt.plot(history.history['accuracy'], label='Training Accuracy')
        plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
        plt.title(f'Accuracy over Epochs ({num_layers} Layers, {activation})')
        plt.xlabel('Epochs')
        plt.ylabel('Accuracy')
        plt.legend()

        # Loss Plot
        plt.subplot(1, 2, 2)
        plt.plot(history.history['loss'], label='Training Loss')
        plt.plot(history.history['val_loss'], label='Validation Loss')
        plt.title(f'Loss over Epochs ({num_layers} Layers, {activation})')
        plt.xlabel('Epochs')
        plt.ylabel('Loss')
        plt.legend()

        plt.show()
```

Content | 15351254 | ICP4.ipynb - Colab | Neural-Network-Deep-Learning

colab.research.google.com/drive/12MZGDqIQLhEPcK9IMOWF7cTFNIRyfqKk#scrollTo=67DTI8x2WxT_

CodeTantra - Teach... | Sreedhar's CCE - Ins... | Settings | Gmail | YouTube | Maps | hostar | Citrix Workspace | New Tab | Adobe Acrobat | All Bookmarks

ICP4.ipynb

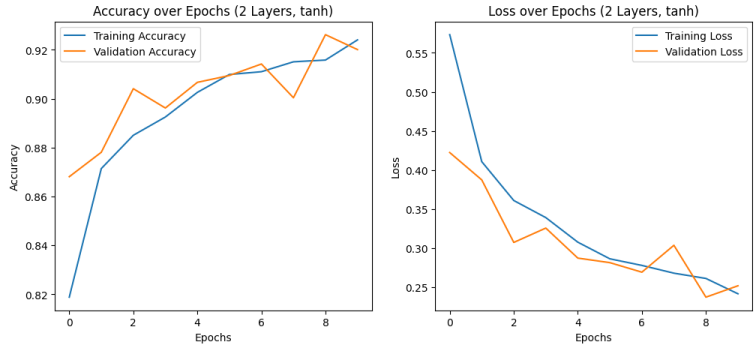
File Edit View Insert Runtime Tools Help

Share

+ Code + Text

Gemini

```
313/313 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.9080 - loss: 0.2909
Test Accuracy with 2 hidden layers and tanh activation: 0.9201
```




```
Training with 3 hidden layers and tanh activation...

Epoch 1/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 11s 5ms/step - accuracy: 0.7728 - loss: 0.7083 - val_accuracy: 0.8663 - val_loss: 0.4142
Epoch 2/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 10s 5ms/step - accuracy: 0.8575 - loss: 0.4376 - val_accuracy: 0.8808 - val_loss: 0.3780
Epoch 3/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 10s 5ms/step - accuracy: 0.8847 - loss: 0.3568 - val_accuracy: 0.9012 - val_loss: 0.3016
Epoch 4/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 9s 5ms/step - accuracy: 0.9010 - loss: 0.3061 - val_accuracy: 0.8971 - val_loss: 0.3178
```

Connected to Python 3 Google Compute Engine backend

Type here to search | Precip on Saturday | ENG | 21:55 11-02-2025

Content | 15351254 | ICP4.ipynb - Colab | Neural-Network-Deep-Learning

colab.research.google.com/drive/12MZGDqIQLhEPcK9IMOWF7cTFNIRyfqKk#scrollTo=67DTI8x2WxT_

CodeTantra - Teach... | Sreedhar's CCE - Ins... | Settings | Gmail | YouTube | Maps | hostar | Citrix Workspace | New Tab | Adobe Acrobat | All Bookmarks

ICP4.ipynb

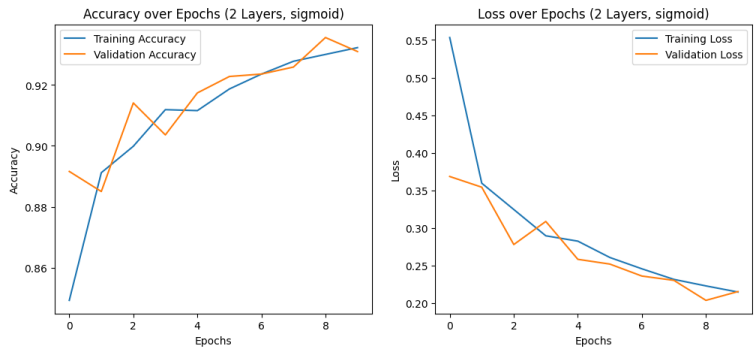
File Edit View Insert Runtime Tools Help

Share

+ Code + Text

Gemini

```
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 18s 9ms/step - accuracy: 0.9341 - loss: 0.2109 - val_accuracy: 0.9310 - val_loss: 0.2154
313/313 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.9203 - loss: 0.2442
Test Accuracy with 2 hidden layers and sigmoid activation: 0.9310
```




```
Training with 3 hidden layers and sigmoid activation...

Epoch 1/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 12s 6ms/step - accuracy: 0.7082 - loss: 0.9730 - val_accuracy: 0.8686 - val_loss: 0.4076
Epoch 2/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 10s 5ms/step - accuracy: 0.8797 - loss: 0.3880 - val_accuracy: 0.8916 - val_loss: 0.3527
Epoch 3/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 10s 5ms/step - accuracy: 0.8917 - loss: 0.3492 - val_accuracy: 0.9086 - val_loss: 0.2926
Epoch 4/10
```

Connected to Python 3 Google Compute Engine backend

Type here to search | Precip on Saturday | ENG | 21:55 11-02-2025

Content | 15351254 | ICP4.ipynb - Colab | Neural-Network-Deep-Learning

colab.research.google.com/drive/12MZGDqIQLhEPcK9IMOWF7cTFNIRyfqKk#scrollTo=67DTI8x2WxT_

CodeTantra - Teach... | Sreedhar's CCE - Ins... | Settings | Gmail | YouTube | Maps | hostar | Citrix Workspace | New Tab | Adobe Acrobat | All Bookmarks

ICP4.ipynb

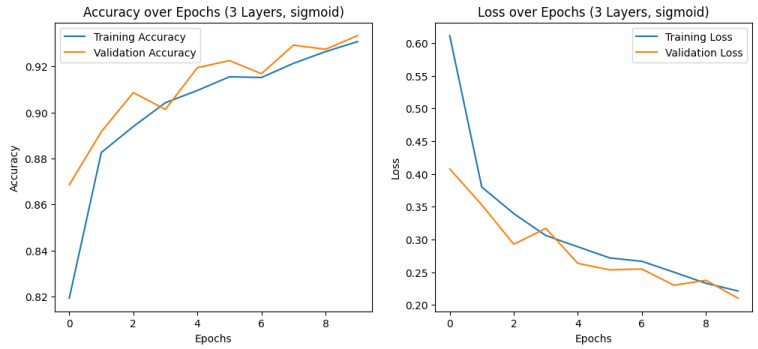
File Edit View Insert Runtime Tools Help

Share

+ Code + Text

Gemini

```
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 9s 5ms/step - accuracy: 0.9201 - loss: 0.2514 - val_accuracy: 0.9292 - val_loss: 0.2300
Epoch 9/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 12s 5ms/step - accuracy: 0.9296 - loss: 0.2214 - val_accuracy: 0.9274 - val_loss: 0.2376
Epoch 10/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 10s 5ms/step - accuracy: 0.9295 - loss: 0.2268 - val_accuracy: 0.9333 - val_loss: 0.2103
313/313 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.9272 - loss: 0.2272
Test Accuracy with 3 hidden layers and sigmoid activation: 0.9333
```



Connected to Python 3 Google Compute Engine backend

Type here to search | ENG 21:55 11-02-2025

Content | 15351254 | ICP4.ipynb - Colab | Neural-Network-Deep-Learning

colab.research.google.com/drive/12MZGDqIQLhEPcK9IMOWF7cTFNIRyfqKk#scrollTo=67DTI8x2WxT_

CodeTantra - Teach... | Sreedhar's CCE - Ins... | Settings | Gmail | YouTube | Maps | hostar | Citrix Workspace | New Tab | Adobe Acrobat | All Bookmarks

ICP4.ipynb

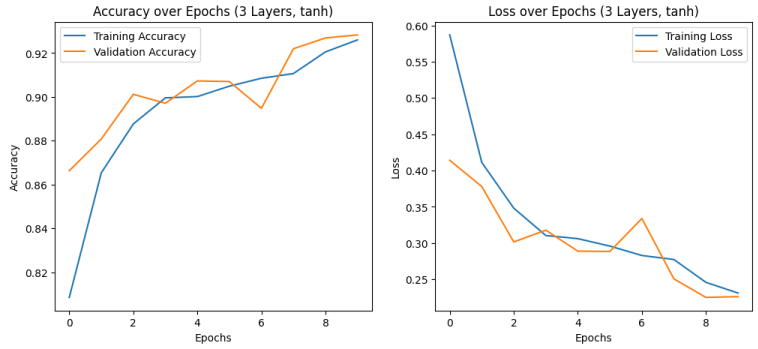
File Edit View Insert Runtime Tools Help

Share

+ Code + Text

Gemini

```
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 24s 9ms/step - accuracy: 0.9252 - loss: 0.2325 - val_accuracy: 0.9283 - val_loss: 0.2263
313/313 ━━━━━━━━━━━━━━━━━━━━ 1s 4ms/step - accuracy: 0.9201 - loss: 0.2512
Test Accuracy with 3 hidden layers and tanh activation: 0.9283
```




```
Training with 2 hidden layers and sigmoid activation...

Epoch 1/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 10s 4ms/step - accuracy: 0.7677 - loss: 0.8677 - val_accuracy: 0.8916 - val_loss: 0.3685
Epoch 2/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 11s 5ms/step - accuracy: 0.8882 - loss: 0.3736 - val_accuracy: 0.8850 - val_loss: 0.3542
Epoch 3/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 10s 5ms/step - accuracy: 0.8955 - loss: 0.3380 - val_accuracy: 0.9141 - val_loss: 0.2780
Epoch 4/10
```

Connected to Python 3 Google Compute Engine backend

Type here to search | Precip on Saturday | ENG 21:55 11-02-2025