**Spring 2024: CS5720 Neural Networks & Deep Learning - ICP-6**

**Assignment- 6**

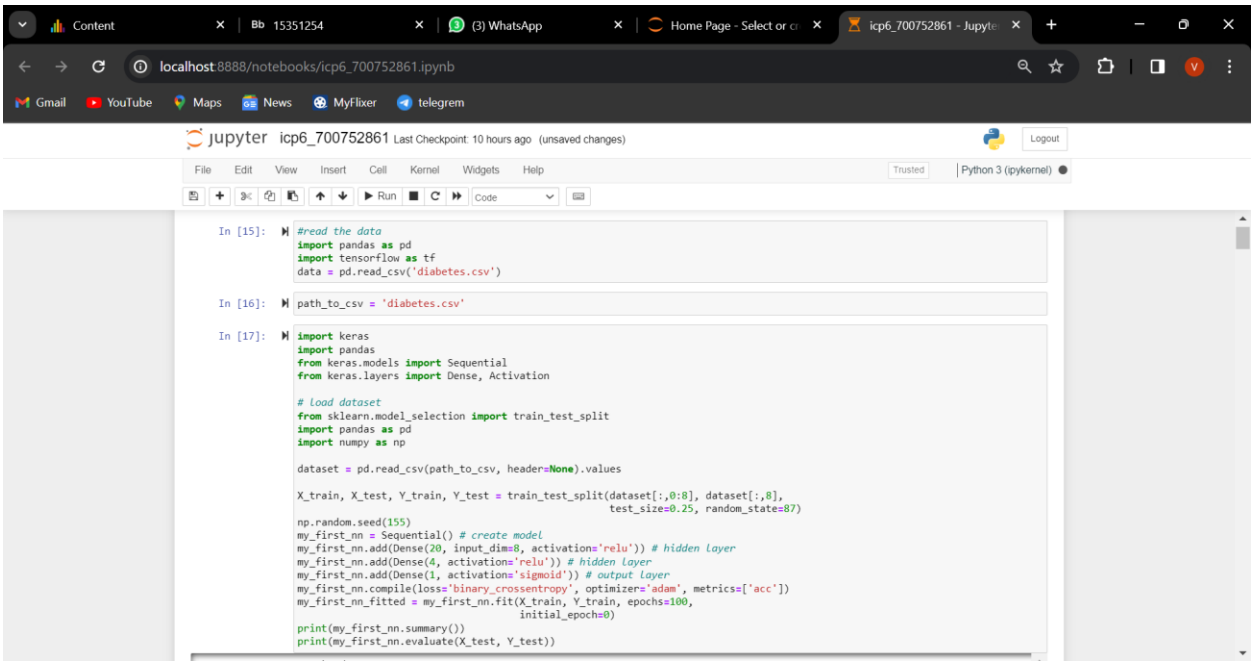
Name: Pushkara Naga Sai Sri Vyshnavi Chakka

STUDENT ID:700752861

GitHub link: https://github.com/PushkaraChakka/Assignment_6_icp6

Video link: https://drive.google.com/file/d/1a7-jlXj2-s6uflwhS4jtA9-RDNMJjaYy/view?usp=sharing

Use Case Description:
Predicting the diabetes disease
Programming elements:
Keras Basics
In class programming:
1. Use the use case in the class:
    a. Add more Dense layers to the existing code and check how the accuracy changes.
2. Change the data source to Breast Cancer dataset * available in the source code folder and make required changes. Report accuracy of the model.
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()
Breast Cancer dataset is designated to predict if a patient has Malignant (M) or Benign = B cancer



```
Epoch 1/100
18/18 [==============================] - 4s 10ms/step - loss: 2.4807 - acc: 0.4167
Epoch 2/100
18/18 [==============================] - 0s 6ms/step - loss: 0.8022 - acc: 0.6580
Epoch 3/100
18/18 [==============================] - 0s 6ms/step - loss: 0.7181 - acc: 0.6597
Epoch 4/100
18/18 [==============================] - 0s 8ms/step - loss: 0.6936 - acc: 0.6615
Epoch 5/100
18/18 [==============================] - 0s 6ms/step - loss: 0.6811 - acc: 0.6615
Epoch 6/100
18/18 [==============================] - 0s 6ms/step - loss: 0.6705 - acc: 0.6632
Epoch 7/100
18/18 [==============================] - 0s 7ms/step - loss: 0.6606 - acc: 0.6632
Epoch 8/100
18/18 [==============================] - 0s 9ms/step - loss: 0.6519 - acc: 0.6632
Epoch 9/100
18/18 [==============================] - 0s 9ms/step - loss: 0.6416 - acc: 0.6632
Epoch 10/100
```

In [7]:
```
data = pd.read_csv('breastcancer.csv')
```

In [8]:
```
path_to_csv = 'sample_data/breastcancer.csv'
```


```
Epoch 10/100
```

In [7]:
```
data = pd.read_csv('breastcancer.csv')
```

In [8]:
```
path_to_csv = 'sample_data/breastcancer.csv'
```


In [9]:
```
import keras
import pandas as pd
import numpy as np
from keras.models import Sequential
from keras.layers import Dense, Activation
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split

# load dataset
cancer_data = load_breast_cancer()
X_train, X_test, Y_train, Y_test = train_test_split(cancer_data.data, cancer_data.target,
                                                    test_size=0.25, random_state=87)

np.random.seed(155)
my_nn = Sequential() # create model
my_nn.add(Dense(20, input_dim=30, activation='relu')) # hidden layer 1
my_nn.add(Dense(1, activation='sigmoid')) # output layer
my_nn.compile(loss='binary_crossentropy', optimizer='adam', metrics=['acc'])
my_nn_fitted = my_nn.fit(X_train, Y_train, epochs=100,
                         initial_epoch=0)
print(my_nn.summary())
print(my_nn.evaluate(X_test, Y_test))
```

```
Epoch 1/100
14/14 [==============================] - 1s 2ms/step - loss: 67.7023 - acc: 0.3803
Epoch 2/100
14/14 [==============================] - 0s 2ms/step - loss: 44.1342 - acc: 0.3803
Epoch 3/100
14/14 [==============================] - 0s 2ms/step - loss: 23.8069 - acc: 0.3803
Epoch 4/100
14/14 [==============================] - 0s 2ms/step - loss: 5.5289 - acc: 0.5235
Epoch 5/100
14/14 [==============================] - 0s 3ms/step - loss: 1.3891 - acc: 0.7746
Epoch 6/100
14/14 [==============================] - 0s 2ms/step - loss: 0.4451 - acc: 0.8427
Epoch 7/100
14/14 [==============================] - 0s 2ms/step - loss: 0.3363 - acc: 0.8920
Epoch 8/100
14/14 [==============================] - 0s 3ms/step - loss: 0.2905 - acc: 0.9155
Epoch 9/100
14/14 [==============================] - 0s 2ms/step - loss: 0.2775 - acc: 0.9155
Epoch 10/100
```

Content | 15351254 | (3) WhatsApp | Home Page - Select or cr | icp6_700752861 - Jupyte

localhost:8888/notebooks/icp6_700752861.ipynb

Gmail YouTube Maps News MyFlixer telegrem

jupyter icp6_700752861 Last Checkpoint: 10 hours ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3 (ipykernel)

In [22]:
```
#read the data
data = pd.read_csv('breastcancer.csv')
```

In [23]:
```
path_to_csv = 'breastcancer.csv'
```

In [24]:
```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
```

In [25]:
```
import keras
import pandas as pd
import numpy as np
from keras.models import Sequential
from keras.layers import Dense, Activation
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split

# load dataset
cancer_data = load_breast_cancer()
X_train, X_test, Y_train, Y_test = train_test_split(cancer_data.data, cancer_data.target,
                                                    test_size=0.25, random_state=87)
np.random.seed(155)
my_nn = Sequential() # create model
my_nn.add(Dense(20, input_dim=30, activation='relu')) # hidden layer 1
my_nn.add(Dense(1, activation='sigmoid')) # output layer
my_nn.compile(loss='binary_crossentropy', optimizer='adam', metrics=['acc'])
my_nn_fitted = my_nn.fit(X_train, Y_train, epochs=100,
                         initial_epoch=0)
print(my_nn.summary())
print(my_nn.evaluate(X_test, Y_test))
```

Content | 15351254 | (3) WhatsApp | Home Page - Select or cr | icp6_700752861 - Jupyte

localhost:8888/notebooks/icp6_700752861.ipynb

Gmail YouTube Maps News MyFlixer telegrem

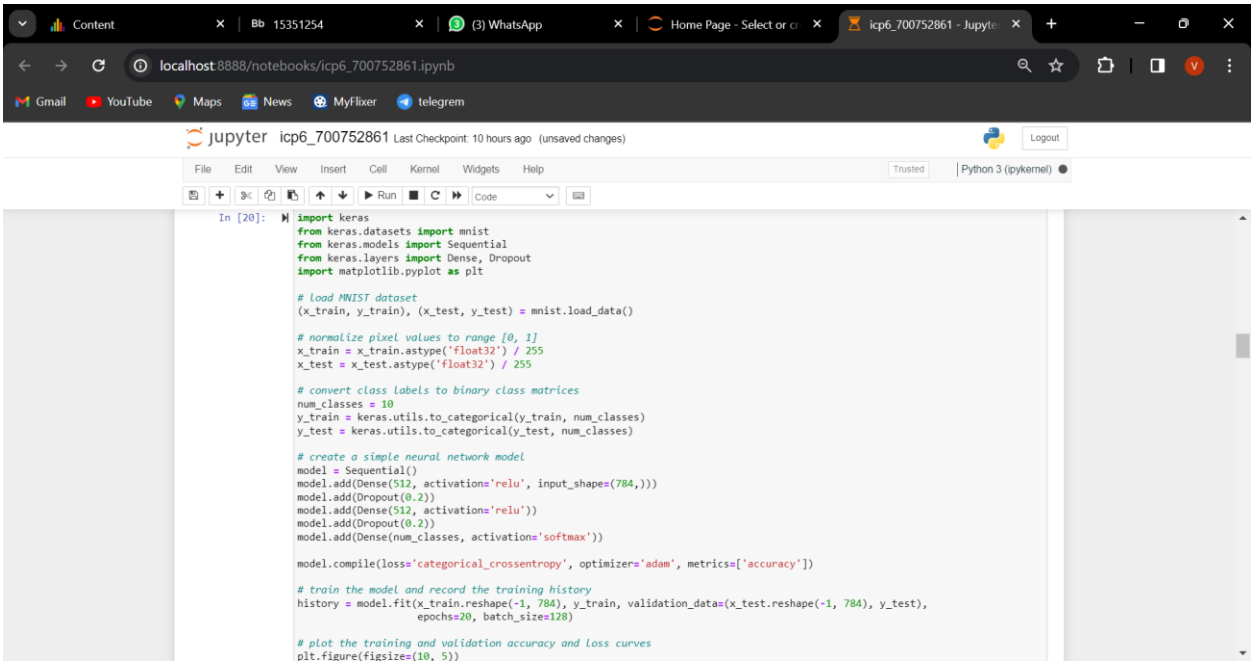
jupyter icp6_700752861 Last Checkpoint: 10 hours ago (unsaved changes)

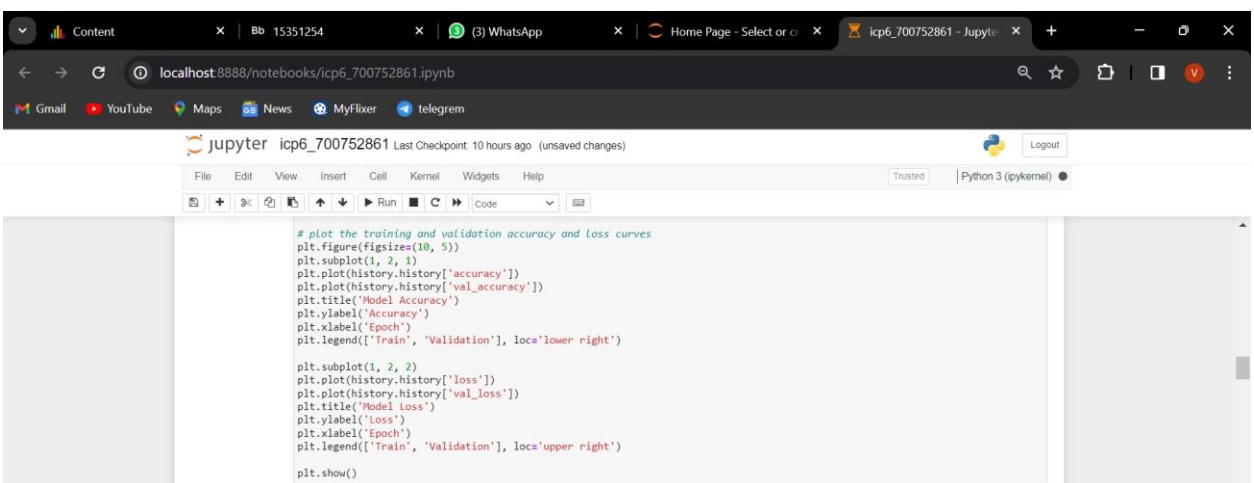
File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3 (ipykernel)

```
Epoch 1/100
14/14 [==============================] - 1s 3ms/step - loss: 14.6296 - acc: 0.3122
Epoch 2/100
14/14 [==============================] - 0s 3ms/step - loss: 8.9664 - acc: 0.2277
Epoch 3/100
14/14 [==============================] - 0s 3ms/step - loss: 4.9536 - acc: 0.3052
Epoch 4/100
14/14 [==============================] - 0s 3ms/step - loss: 1.7886 - acc: 0.5469
Epoch 5/100
14/14 [==============================] - 0s 2ms/step - loss: 0.6428 - acc: 0.8122
Epoch 6/100
14/14 [==============================] - 0s 3ms/step - loss: 0.5085 - acc: 0.8685
Epoch 7/100
14/14 [==============================] - 0s 2ms/step - loss: 0.4529 - acc: 0.8850
Epoch 8/100
14/14 [==============================] - 0s 2ms/step - loss: 0.4083 - acc: 0.9061
Epoch 9/100
14/14 [==============================] - 0s 2ms/step - loss: 0.4040 - acc: 0.8944
Epoch 10/100
```

In class programming:
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.
4. Run the same code without scaling the images and check the performance?



Content | 15351254 | (3) WhatsApp | Home Page - Select or cr | icp6_700752861 - Jupyte

localhost:8888/notebooks/icp6_700752861.ipynb

Gmail YouTube Maps News MyFlixer telegrem

jupyter icp6_700752861 Last Checkpoint: 10 hours ago (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

Run Code

```
Epoch 1/20
469/469 [==============================] - 9s 17ms/step - loss: 0.2481 - accuracy: 0.9267 - val_loss: 0.1136 - val_accuracy:
0.9629
Epoch 2/20
469/469 [==============================] - 7s 15ms/step - loss: 0.1030 - accuracy: 0.9679 - val_loss: 0.0805 - val_accuracy:
0.9749
Epoch 3/20
469/469 [==============================] - 7s 14ms/step - loss: 0.0711 - accuracy: 0.9779 - val_loss: 0.0823 - val_accuracy:
0.9742
Epoch 4/20
469/469 [==============================] - 8s 16ms/step - loss: 0.0543 - accuracy: 0.9830 - val_loss: 0.0742 - val_accuracy:
0.9778
Epoch 5/20
469/469 [==============================] - 7s 15ms/step - loss: 0.0463 - accuracy: 0.9848 - val_loss: 0.0623 - val_accuracy:
0.9809
Epoch 6/20
469/469 [==============================] - 8s 16ms/step - loss: 0.0401 - accuracy: 0.9870 - val_loss: 0.0680 - val_accuracy:
0.9801
Epoch 7/20
469/469 [==============================] - 7s 14ms/step - loss: 0.0354 - accuracy: 0.9886 - val_loss: 0.0739 - val_accuracy:
0.9788
Epoch 8/20
469/469 [==============================] - 7s 14ms/step - loss: 0.0290 - accuracy: 0.9905 - val_loss: 0.0652 - val_accuracy:
0.9812
Epoch 9/20
469/469 [==============================] - 7s 14ms/step - loss: 0.0279 - accuracy: 0.9909 - val_loss: 0.0768 - val_accuracy:
0.9786
Epoch 10/20
469/469 [==============================] - 7s 14ms/step - loss: 0.0269 - accuracy: 0.9909 - val_loss: 0.0669 - val_accuracy:
0.9824
Epoch 11/20
469/469 [==============================] - 7s 14ms/step - loss: 0.0229 - accuracy: 0.9920 - val_loss: 0.0622 - val_accuracy:
0.9842
```

Content | 15351254 | (3) WhatsApp | Home Page - Select or cr | icp6_700752861 - Jupyte

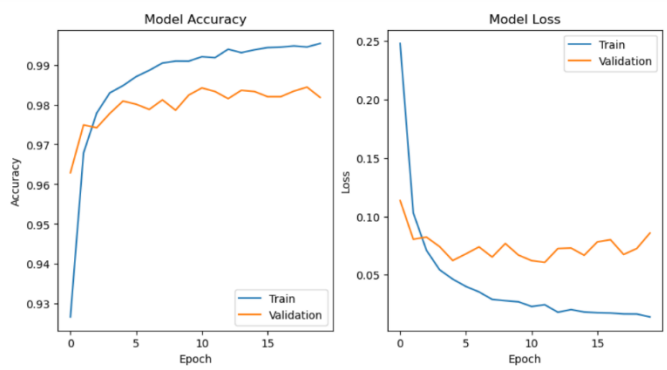
localhost:8888/notebooks/icp6_700752861.ipynb

Gmail YouTube Maps News MyFlixer telegrem

jupyter icp6_700752861 Last Checkpoint: 10 hours ago (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

Run Code



Content | 15351254 | (5) WhatsApp | Home Page - Select or cr | icp6_700752861 - Jupyte

localhost:8888/notebooks/icp6_700752861.ipynb

Gmail | YouTube | Maps | News | MyFlixer | telegrem

jupyter icp6_700752861 Last Checkpoint: 10 hours ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help | Trusted | Python 3 (ipykernel)

In [26]:

```python
import keras
from keras.datasets import mnist
from keras.models import Sequential
from keras.layers import Dense, Dropout
import matplotlib.pyplot as plt
import numpy as np

# load MNIST dataset
(x_train, y_train), (x_test, y_test) = mnist.load_data()

# normalize pixel values to range [0, 1]
x_train = x_train.astype('float32') / 255
x_test = x_test.astype('float32') / 255

# convert class labels to binary class matrices
num_classes = 10
y_train = keras.utils.to_categorical(y_train, num_classes)
y_test = keras.utils.to_categorical(y_test, num_classes)
```

Content | 15351254 | (5) WhatsApp | Home Page - Select or cr | icp6_700752861 - Jupyte

localhost:8888/notebooks/icp6_700752861.ipynb

Gmail | YouTube | Maps | News | MyFlixer | telegrem

jupyter icp6_700752861 Last Checkpoint: 10 hours ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help | Trusted | Python 3 (ipykernel)

```python
# create a simple neural network model
model = Sequential()
model.add(Dense(512, activation='relu', input_shape=(784,)))
model.add(Dropout(0.2))
model.add(Dense(512, activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(num_classes, activation='softmax'))

model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
model.fit(x_train.reshape(-1, 784), y_train, validation_data=(x_test.reshape(-1, 784), y_test),
          epochs=20, batch_size=128)

# plot one of the images in the test data
plt.imshow(x_test[0], cmap='gray')
plt.show()

# make a prediction on the image using the trained model
prediction = model.predict(x_test[0].reshape(1, -1))
print('Model prediction:', np.argmax(prediction))
```

Content | 15351254 | (5) WhatsApp | Home Page - Select or cr | icp6_700752861 - Jupyte

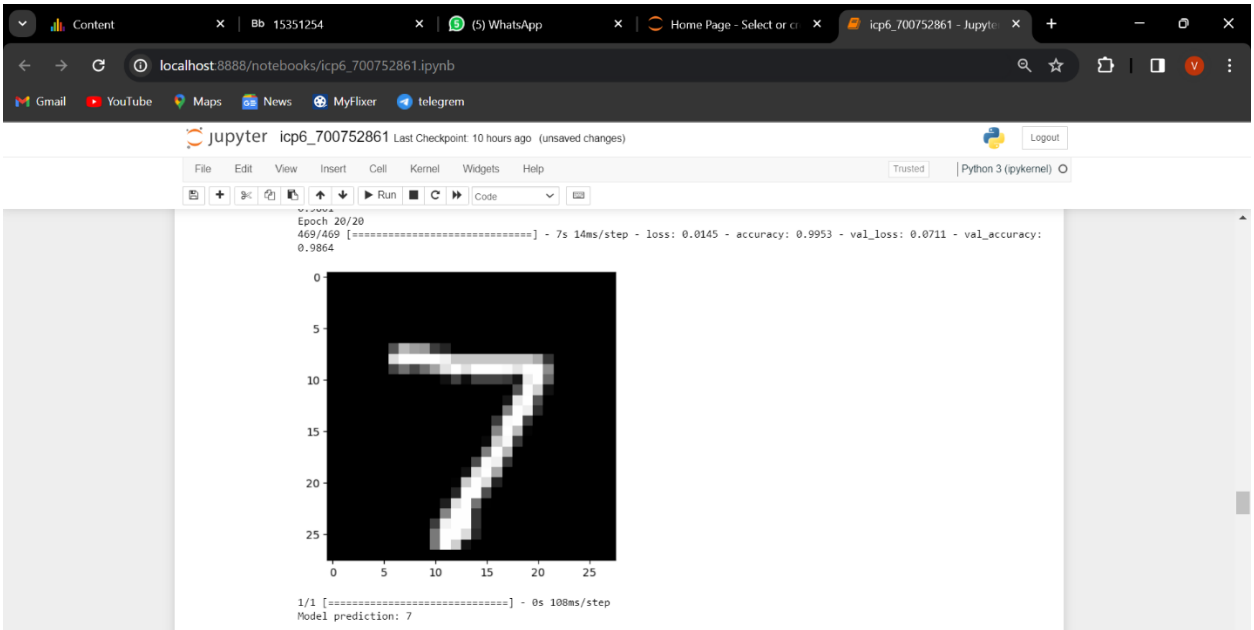
localhost:8888/notebooks/icp6_700752861.ipynb

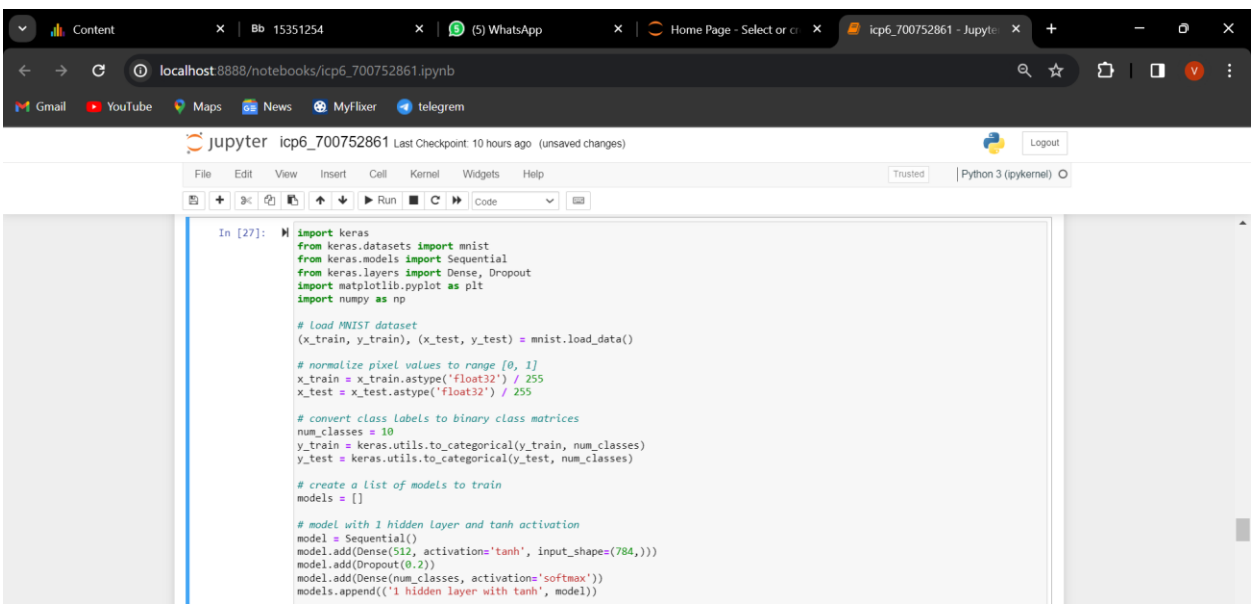
Gmail | YouTube | Maps | News | MyFlixer | telegrem

jupyter icp6_700752861 Last Checkpoint: 10 hours ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help | Trusted | Python 3 (ipykernel)

```
Epoch 1/20
469/469 [==============================] - 8s 15ms/step - loss: 0.2521 - accuracy: 0.9242 - val_loss: 0.1119 - val_accuracy:
0.9671
Epoch 2/20
469/469 [==============================] - 6s 14ms/step - loss: 0.1026 - accuracy: 0.9687 - val_loss: 0.0769 - val_accuracy:
0.9762
Epoch 3/20
469/469 [==============================] - 6s 13ms/step - loss: 0.0726 - accuracy: 0.9776 - val_loss: 0.0685 - val_accuracy:
0.9775
Epoch 4/20
469/469 [==============================] - 6s 13ms/step - loss: 0.0565 - accuracy: 0.9821 - val_loss: 0.0687 - val_accuracy:
0.9790
Epoch 5/20
469/469 [==============================] - 6s 14ms/step - loss: 0.0482 - accuracy: 0.9840 - val_loss: 0.0647 - val_accuracy:
0.9806
Epoch 6/20
469/469 [==============================] - 7s 14ms/step - loss: 0.0388 - accuracy: 0.9879 - val_loss: 0.0626 - val_accuracy:
0.9807
Epoch 7/20
469/469 [==============================] - 6s 13ms/step - loss: 0.0358 - accuracy: 0.9884 - val_loss: 0.0688 - val_accuracy:
0.9802
Epoch 8/20
469/469 [==============================] - 7s 15ms/step - loss: 0.0320 - accuracy: 0.9895 - val_loss: 0.0690 - val_accuracy:
0.9803
Epoch 9/20
469/469 [==============================] - 6s 13ms/step - loss: 0.0246 - accuracy: 0.9916 - val_loss: 0.0675 - val_accuracy:
0.9813
Epoch 10/20
469/469 [==============================] - 6s 13ms/step - loss: 0.0257 - accuracy: 0.9917 - val_loss: 0.0809 - val_accuracy:
0.9804
Epoch 11/20
469/469 [==============================] - 6s 13ms/step - loss: 0.0238 - accuracy: 0.9922 - val_loss: 0.0814 - val_accuracy:
0.9807
Epoch 12/20
```


Content
15351254
(5) WhatsApp
Home Page - Select or cr
icp6_700752861 - Jupyte
localhost:8888/notebooks/icp6_700752861.ipynb
Gmail
YouTube
Maps
News
MyFlixer
telegrem
jupyter icp6_700752861 Last Checkpoint: 10 hours ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code
Epoch 20/20
469/469 [==========] - 7s 14ms/step - loss: 0.0145 - accuracy: 0.9953 - val_loss: 0.0711 - val_accuracy: 0.9864
0
5
10
15
20
25
0 5 10 15 20 25
1/1 [==========] - 0s 108ms/step
Model prediction: 7


Content
15351254
(5) WhatsApp
Home Page - Select or cr
icp6_700752861 - Jupyte
localhost:8888/notebooks/icp6_700752861.ipynb
Gmail
YouTube
Maps
News
MyFlixer
telegrem
jupyter icp6_700752861 Last Checkpoint: 10 hours ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code
In [27]:
import keras
from keras.datasets import mnist
from keras.models import Sequential
from keras.layers import Dense, Dropout
import matplotlib.pyplot as plt
import numpy as np
# load MNIST dataset
(x_train, y_train), (x_test, y_test) = mnist.load_data()
# normalize pixel values to range [0, 1]
x_train = x_train.astype('float32') / 255
x_test = x_test.astype('float32') / 255
# convert class labels to binary class matrices
num_classes = 10
y_train = keras.utils.to_categorical(y_train, num_classes)
y_test = keras.utils.to_categorical(y_test, num_classes)
# create a list of models to train
models = []
# model with 1 hidden layer and tanh activation
model = Sequential()
model.add(Dense(512, activation='tanh', input_shape=(784,)))
model.add(Dropout(0.2))
model.add(Dense(num_classes, activation='softmax'))
models.append(('1 hidden layer with tanh', model))

Content | 15351254 | (5) WhatsApp | Home Page - Select or cr | icp6_700752861 - Jupyte

localhost:8888/notebooks/icp6_700752861.ipynb

Gmail YouTube Maps News MyFlixer telegrem

jupyter icp6_700752861 Last Checkpoint: 10 hours ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

```
# model with 1 hidden layer and sigmoid activation
model = Sequential()
model.add(Dense(512, activation='sigmoid', input_shape=(784,)))
model.add(Dropout(0.2))
model.add(Dense(num_classes, activation='softmax'))
models.append(('1 hidden layer with sigmoid', model))

# model with 2 hidden layers and tanh activation
model = Sequential()
model.add(Dense(512, activation='tanh', input_shape=(784,)))
model.add(Dropout(0.2))
model.add(Dense(512, activation='tanh'))
model.add(Dropout(0.2))
model.add(Dense(num_classes, activation='softmax'))
models.append(('2 hidden layers with tanh', model))

# model with 2 hidden layers and sigmoid activation
model = Sequential()
model.add(Dense(512, activation='sigmoid', input_shape=(784,)))
model.add(Dropout(0.2))
model.add(Dense(512, activation='sigmoid'))
model.add(Dropout(0.2))
model.add(Dense(num_classes, activation='softmax'))
models.append(('2 hidden layers with sigmoid', model))
```

Content | 15351254 | (5) WhatsApp | Home Page - Select or cr | icp6_700752861 - Jupyte

localhost:8888/notebooks/icp6_700752861.ipynb

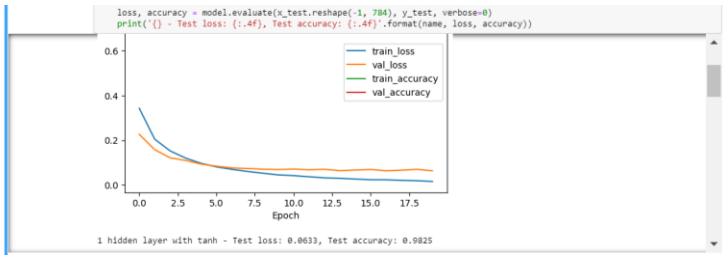
Gmail YouTube Maps News MyFlixer telegrem

jupyter icp6_700752861 Last Checkpoint: 10 hours ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

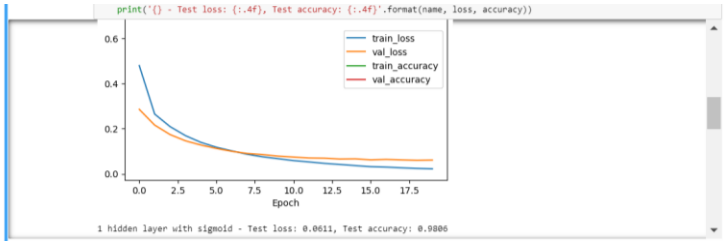
```
# train each model and plot loss and accuracy curves
for name, model in models:
    model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
    history = model.fit(x_train.reshape(-1, 784), y_train, validation_data=(x_test.reshape(-1, 784), y_test),
                        epochs=20, batch_size=128, verbose=0)
    # plot loss and accuracy curves
    plt.plot(history.history['loss'], label='train_loss')
    plt.plot(history.history['val_loss'], label='val_loss')
    plt.plot(history.history['accuracy'], label='train_accuracy')
    plt.plot(history.history['val_accuracy'], label='val_accuracy')
    plt.title(name)
    plt.xlabel('Epoch')
    plt.legend()
    plt.show()

    # evaluate the model on test data
    loss, accuracy = model.evaluate(x_test.reshape(-1, 784), y_test, verbose=0)
    print('{} - Test loss: {:.4f}, Test accuracy: {:.4f}'.format(name, loss, accuracy))
```

```
loss, accuracy = model.evaluate(x_test.reshape(-1, 784), y_test, verbose=0)
print('{} - Test loss: {:.4f}, Test accuracy: {:.4f}'.format(name, loss, accuracy))
```



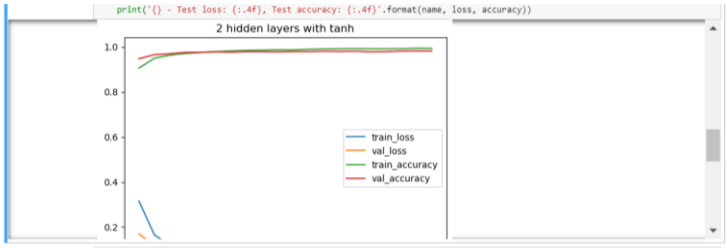

```
1 hidden layer with tanh - Test loss: 0.0633, Test accuracy: 0.9825
```

```
print('{} - Test loss: {:.4f}, Test accuracy: {:.4f}'.format(name, loss, accuracy))
```

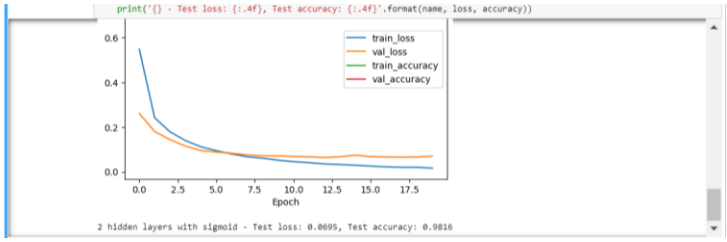



```
1 hidden layer with sigmoid - Test loss: 0.0611, Test accuracy: 0.9806
```

```
print('{} - Test loss: {:.4f}, Test accuracy: {:.4f}'.format(name, loss, accuracy))
```




```
print('{} - Test loss: {:.4f}, Test accuracy: {:.4f}'.format(name, loss, accuracy))
```




```
2 hidden layers with sigmoid - Test loss: 0.0695, Test accuracy: 0.9816
```

Content | 15351254 | (5) WhatsApp | Home Page - Select or cr | icp6_700752861 - Jupyte

localhost:8888/notebooks/icp6_700752861.ipynb

Gmail YouTube Maps News MyFlixer telegrem

jupyter icp6_700752861 Last Checkpoint: 11 hours ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3 (ipykernel)

Run Code

In [28]:

```python
import keras
from keras.datasets import mnist
from keras.models import Sequential
from keras.layers import Dense, Dropout
import matplotlib.pyplot as plt
import numpy as np

# load MNIST dataset
(x_train, y_train), (x_test, y_test) = mnist.load_data()

# convert class labels to binary class matrices
num_classes = 10
y_train = keras.utils.to_categorical(y_train, num_classes)
y_test = keras.utils.to_categorical(y_test, num_classes)

# create a list of models to train
models = []

# model with 1 hidden layer and tanh activation
model = Sequential()
model.add(Dense(512, activation='tanh', input_shape=(784,)))
model.add(Dropout(0.2))
model.add(Dense(num_classes, activation='softmax'))
models.append(('1 hidden layer with tanh', model))

# model with 1 hidden layer and sigmoid activation
model = Sequential()
model.add(Dense(512, activation='sigmoid', input_shape=(784,)))
model.add(Dropout(0.2))
model.add(Dense(num_classes, activation='softmax'))
models.append(('1 hidden layer with sigmoid', model))

# model with 2 hidden layers and tanh activation
model = Sequential()
model.add(Dense(512, activation='tanh', input_shape=(784,)))
model.add(Dropout(0.2))
model.add(Dense(512, activation='tanh'))
model.add(Dropout(0.2))
model.add(Dense(num_classes, activation='softmax'))
models.append(('2 hidden layers with tanh', model))

# model with 2 hidden layers and sigmoid activation
```

Content | 15351254 | (5) WhatsApp | Home Page - Select or cr | icp6_700752861 - Jupyte

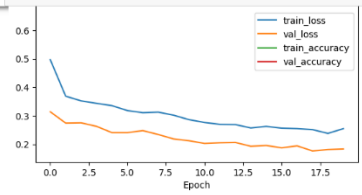
localhost:8888/notebooks/icp6_700752861.ipynb

Gmail | YouTube | Maps | News | MyFlixer | telegrem

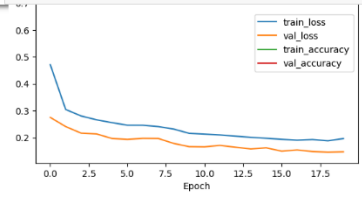
jupyter icp6_700752861 Last Checkpoint: 11 hours ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help

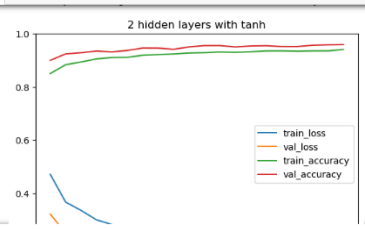
```python
# model with 2 hidden layers and sigmoid activation
model = Sequential()
model.add(Dense(512, activation='sigmoid', input_shape=(784,)))
model.add(Dropout(0.2))
model.add(Dense(512, activation='sigmoid'))
model.add(Dropout(0.2))
model.add(Dense(num_classes, activation='softmax'))
models.append(('2 hidden layers with sigmoid', model))

# train each model and plot loss and accuracy curves
for name, model in models:
    model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
    history = model.fit(x_train.reshape(-1, 784), y_train, validation_data=(x_test.reshape(-1, 784), y_test),
                        epochs=20, batch_size=128, verbose=0)
    # plot loss and accuracy curves
    plt.plot(history.history['loss'], label='train_loss')
    plt.plot(history.history['val_loss'], label='val_loss')
    plt.plot(history.history['accuracy'], label='train_accuracy')
    plt.plot(history.history['val_accuracy'], label='val_accuracy')
    plt.title(name)
    plt.xlabel('Epoch')
    plt.legend()
    plt.show()

    # evaluate the model on test data
    loss, accuracy = model.evaluate(x_test.reshape(-1, 784), y_test, verbose=0)
    print('{} - Test loss: {:.4f}, Test accuracy: {:.4f}'.format(name, loss, accuracy))
```




```
1 hidden layer with tanh - Test loss: 0.1831, Test accuracy: 0.9467
```




```
1 hidden layer with sigmoid - Test loss: 0.1475, Test accuracy: 0.9572
```




```
2 hidden layers with tanh - Test loss: 0.1362, Test accuracy: 0.9589
```