Deep Learning

Practical No. 02 Implementing XOR in Deep learning using python

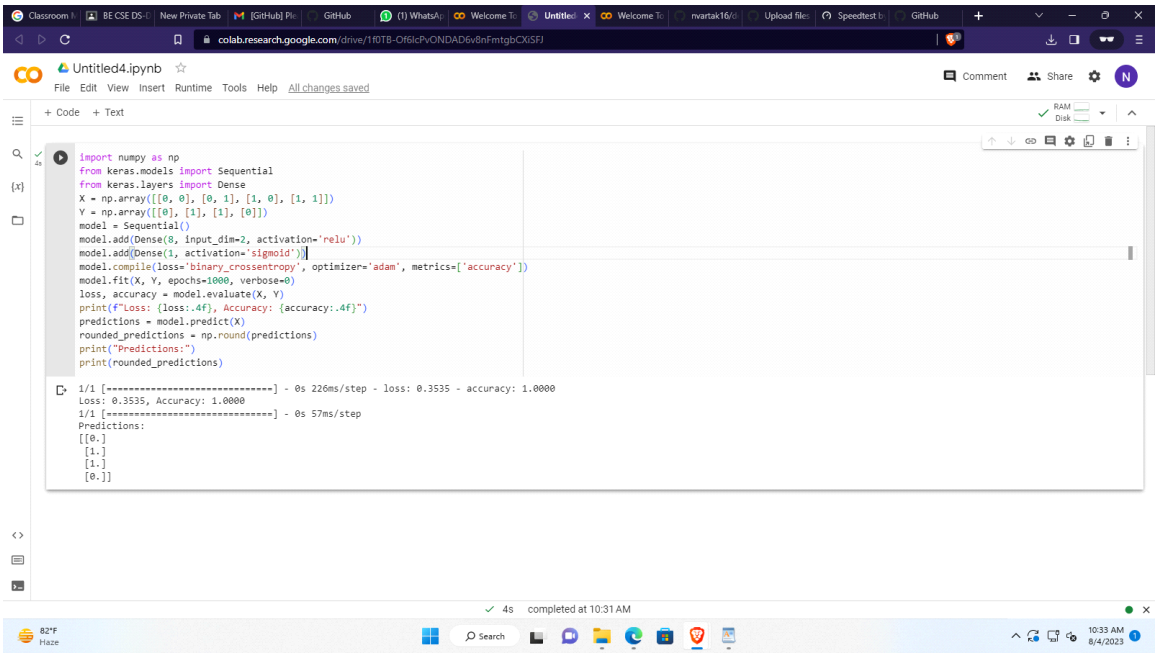
Name: Paritosh

Roll no: 72

Code:

```
import numpy as np
from keras.models import Sequential
from keras.layers import Dense
X = np.array([[0, 0], [0, 1], [1, 0], [1, 1]])
Y = np.array([[0], [1], [1], [0]])
model = Sequential()
model.add(Dense(8, input_dim=2, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
model.fit(X, Y, epochs=1000, verbose=0)
loss, accuracy = model.evaluate(X, Y)
print(f"Loss: {loss:.4f}, Accuracy: {accuracy:.4f}")
predictions = model.predict(X)
rounded_predictions = np.round(predictions)
print("Predictions:")
print(rounded_predictions)
```

Output


Untitled4.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
import numpy as np
from keras.models import Sequential
from keras.layers import Dense
X = np.array([[0, 0], [0, 1], [1, 0], [1, 1]])
Y = np.array([[0], [1], [1], [0]])
model = Sequential()
model.add(Dense(8, input_dim=2, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
model.fit(X, Y, epochs=1000, verbose=0)
loss, accuracy = model.evaluate(X, Y)
print(f"Loss: {loss:.4f}, Accuracy: {accuracy:.4f}")
predictions = model.predict(X)
rounded_predictions = np.round(predictions)
print("Predictions:")
print(rounded_predictions)
1/1 [==============================] - 0s 226ms/step - loss: 0.3535 - accuracy: 1.0000
Loss: 0.3535, Accuracy: 1.0000
1/1 [==============================] - 0s 57ms/step
Predictions:
[[0.]
[1.]
[1.]
[0.]]
4s completed at 10:31 AM