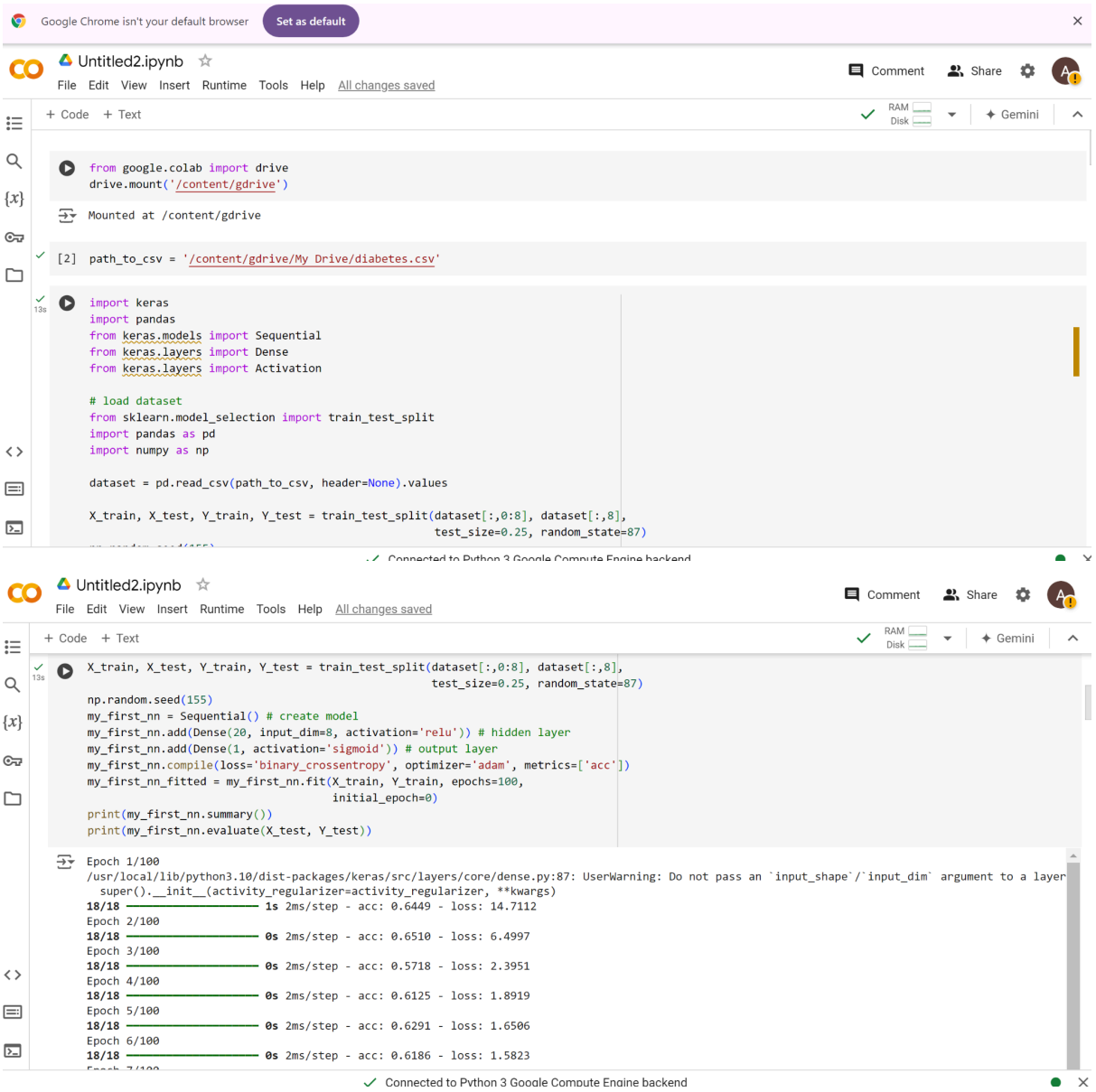
ICP5


Google Chrome isn't your default browser
Set as default
Untitled2.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Gemini
from google.colab import drive
drive.mount('/content/gdrive')
Mounted at /content/gdrive
[2] path_to_csv = '/content/gdrive/My Drive/diabetes.csv'
import keras
import pandas
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Activation
# load dataset
from sklearn.model_selection import train_test_split
import pandas as pd
import numpy as np
dataset = pd.read_csv(path_to_csv, header=None).values
X_train, X_test, Y_train, Y_test = train_test_split(dataset[:,0:8], dataset[:,8],
test_size=0.25, random_state=87)
Connected to Python 3 Google Compute Engine backend

Untitled2.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Gemini
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() # create model
my_first_nn.add(Dense(20, input_dim=8, activation='relu')) # hidden layer
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)
print(my_first_nn.summary())
print(my_first_nn.evaluate(X_test, Y_test))
Epoch 1/100
/usr/local/lib/python3.10/dist-packages/keras/src/layers/core/dense.py:87: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a layer
super().__init__(activity_regularizer=activity_regularizer, **kwargs)
18/18 1s 2ms/step - acc: 0.6449 - loss: 14.7112
Epoch 2/100
18/18 0s 2ms/step - acc: 0.6510 - loss: 6.4997
Epoch 3/100
18/18 0s 2ms/step - acc: 0.5718 - loss: 2.3951
Epoch 4/100
18/18 0s 2ms/step - acc: 0.6125 - loss: 1.8919
Epoch 5/100
18/18 0s 2ms/step - acc: 0.6291 - loss: 1.6506
Epoch 6/100
18/18 0s 2ms/step - acc: 0.6186 - loss: 1.5823
Connected to Python 3 Google Compute Engine backend

Untitled2.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Gemini

```
Epoch 95/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - acc: 0.7470 - loss: 0.5341
Epoch 96/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - acc: 0.7230 - loss: 0.5476
Epoch 97/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - acc: 0.7505 - loss: 0.5300
Epoch 98/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - acc: 0.7425 - loss: 0.5277
Epoch 99/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - acc: 0.7433 - loss: 0.5261
Epoch 100/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - acc: 0.7241 - loss: 0.5686
Model: "sequential"
```

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 20)	180
dense_1 (Dense)	(None, 1)	21

```
Total params: 605 (2.37 KB)
Trainable params: 201 (804.00 B)
Non-trainable params: 0 (0.00 B)
Optimizer params: 404 (1.58 KB)
None
6/6 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - acc: 0.6565 - loss: 0.6996
[0.6950445175170898, 0.6458333134651184]
```

Connected to Python 3 Google Compute Engine backend

Untitled2.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Gemini

```python
import keras
import pandas as pd
import numpy as np
from keras.models import Sequential
from keras.layers import Dense
from sklearn.model_selection import train_test_split

# Load dataset
dataset = pd.read_csv(path_to_csv, header=None).values

# Split the dataset into training and testing sets
X_train, X_test, Y_train, Y_test = train_test_split(dataset[:, 0:8], dataset[:, 8], test_size=0.25, random_state=87)

# Set random seed for reproducibility
np.random.seed(155)

# Create a Sequential model
model = Sequential()

# Add Dense layers with 'relu' activation for hidden layers
model.add(Dense(20, input_dim=8, activation='relu'))  # First hidden layer
model.add(Dense(15, activation='relu'))  # Second hidden layer
model.add(Dense(10, activation='relu'))  # Third hidden layer

# Add output layer with 'sigmoid' activation
```

Connected to Python 3 Google Compute Engine backend

Untitled2.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Gemini

```python
model.add(Dense(10, activation='relu'))  # Third hidden layer

# Add output layer with 'sigmoid' activation
model.add(Dense(1, activation='sigmoid'))

# Compile the model using binary crossentropy and adam optimizer
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['acc'])

# Train the model
model_fitted = model.fit(X_train, Y_train, epochs=100, initial_epoch=0)

# Print model summary and evaluate accuracy on the test set
print(model.summary())
print(model.evaluate(X_test, Y_test))
```

```
Epoch 1/100
/usr/local/lib/python3.10/dist-packages/keras/src/layers/core/dense.py:87: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a layer
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
18/18 ━━━━━━━━━━━━━━━━━━━━ 4s 6ms/step - acc: 0.3444 - loss: 11.4539
Epoch 2/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 6ms/step - acc: 0.3785 - loss: 3.4344
Epoch 3/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - acc: 0.5851 - loss: 1.3650
Epoch 4/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - acc: 0.5424 - loss: 0.8206
```

Connected to Python 3 Google Compute Engine backend

Untitled2.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Gemini

```
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step - acc: 0.7610 - loss: 0.4954
Epoch 97/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - acc: 0.7848 - loss: 0.5198
Epoch 98/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step - acc: 0.7336 - loss: 0.5063
Epoch 99/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - acc: 0.7672 - loss: 0.5110
Epoch 100/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step - acc: 0.7709 - loss: 0.5078
Model: "sequential_2"
```

Layer (type)	Output Shape	Param #
dense_6 (Dense)	(None, 20)	180
dense_7 (Dense)	(None, 15)	315
dense_8 (Dense)	(None, 10)	160
dense_9 (Dense)	(None, 1)	11

```
 Total params: 2,000 (7.82 KB)
 Trainable params: 666 (2.60 KB)
 Non-trainable params: 0 (0.00 B)
 Optimizer params: 1,334 (5.21 KB)
None
6/6 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step - acc: 0.7059 - loss: 0.5874
[0.5887110829353333, 0.7135416865348816]
```

Untitled2.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Gemini

```python
import keras
import pandas as pd
import numpy as np
from keras.models import Sequential
from keras.layers import Dense
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

# Load dataset
dataset = pd.read_csv(path_to_csv, header=None).values

# Split the dataset into features (X) and target (Y)
X = dataset[:, 0:8]
Y = dataset[:, 8]

# Normalize the feature data
sc = StandardScaler()
X = sc.fit_transform(X)

# Split the dataset into training and testing sets
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.25, random_state=87)

# Set random seed for reproducibility
np.random.seed(155)
```

Untitled2.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Gemini

```python
# Set random seed for reproducibility
np.random.seed(155)

# Create a Sequential model
model = Sequential()

# Add Dense layers with 'relu' activation for hidden layers
model.add(Dense(20, input_dim=8, activation='relu'))  # First hidden layer
model.add(Dense(15, activation='relu'))  # Second hidden layer
model.add(Dense(10, activation='relu'))  # Third hidden layer

# Add output layer with 'sigmoid' activation
model.add(Dense(1, activation='sigmoid'))

# Compile the model using binary crossentropy and adam optimizer
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['acc'])

# Train the model
model_fitted = model.fit(X_train, Y_train, epochs=100, initial_epoch=0)

# Print model summary and evaluate accuracy on the test set
print(model.summary())
print(model.evaluate(X_test, Y_test))
```

```
Epoch 1/100
```

Untitled2.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Gemini

```
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - acc: 0.8623 - loss: 0.3159
Epoch 97/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - acc: 0.8393 - loss: 0.3483
Epoch 98/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - acc: 0.8620 - loss: 0.3282
Epoch 99/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - acc: 0.8444 - loss: 0.3321
Epoch 100/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - acc: 0.8594 - loss: 0.3135
Model: "sequential_3"
```

Layer (type)	Output Shape	Param #
dense_10 (Dense)	(None, 20)	180
dense_11 (Dense)	(None, 15)	315
dense_12 (Dense)	(None, 10)	160
dense_13 (Dense)	(None, 1)	11

```
 Total params: 2,000 (7.82 KB)
 Trainable params: 666 (2.60 KB)
 Non-trainable params: 0 (0.00 B)
 Optimizer params: 1,334 (5.21 KB)
None
6/6 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - acc: 0.7210 - loss: 0.5747
[0.5511998534202576, 0.734375]
```

Connected to Python 3 Google Compute Engine backend

ICP5

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Gemini

```python
import keras
import pandas
from keras.models import Sequential
from keras.layers import Dense, Activation

# load dataset
from sklearn.model_selection import train_test_split
#from sklearn.preprocessing import StandardScaler
import pandas as pd
import numpy as np

dataset = pd.read_csv(path_to_csv1, header=None).values

X = dataset[1:, 2:-1]  # Features
Y = dataset[1:, -1]   # Labels (M or B)

# Convert labels to binary format
Y = np.where(Y == 'M', 1, 0)  # M -> 1, B -> 0

#Convert to numeric
X = X.astype(np.float64) # Convert X to numeric

X_train, X_test, Y_train, Y_test = train_test_split(X, Y,
                                                    test_size=0.25, random_state=87)
```

Connected to Python 3 Google Compute Engine backend

ICP5

File np.random.seed <built-in function seed> ols Help All changes saved

Comment Share

+ Code

RAM Disk Gemini

```python
np.random.seed(155)
my_first_nn = Sequential() # create model
my_first_nn.add(Dense(20, input_dim=30, activation='relu')) # hidden layer
my_first_nn.add(Dense(30, activation='relu')) # hidden layer
my_first_nn.add(Dense(40, activation='relu')) # hidden layer
my_first_nn.add(Dense(50, activation='relu')) # hidden layer

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)
print(my_first_nn.summary())
print(my_first_nn.evaluate(X_test, Y_test))
```

```
Epoch 1/100
/usr/local/lib/python3.10/dist-packages/keras/src/layers/core/dense.py:87: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a layer
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
14/14 ━━━━━━━━━━━━━━━━━━━━ 2s 3ms/step - acc: 0.7849 - loss: 0.8910
Epoch 2/100
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - acc: 1.0000 - loss: 4.7385e-10
Epoch 3/100
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - acc: 1.0000 - loss: 1.4251e-10
Epoch 4/100
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - acc: 1.0000 - loss: 5.9871e-11
Epoch 5/100
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - acc: 1.0000 - loss: 1.8965e-10
```

ICP5

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Gemini

```
Epoch 98/100
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - acc: 1.0000 - loss: 8.0804e-11
Epoch 99/100
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - acc: 1.0000 - loss: 2.3542e-11
Epoch 100/100
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - acc: 1.0000 - loss: 6.5664e-11
Model: "sequential_10"
```

Layer (type)	Output Shape	Param #
dense_35 (Dense)	(None, 20)	620
dense_36 (Dense)	(None, 30)	630
dense_37 (Dense)	(None, 40)	1,240
dense_38 (Dense)	(None, 50)	2,050
dense_39 (Dense)	(None, 1)	51

```
Total params: 13,775 (53.81 KB)
Trainable params: 4,591 (17.93 KB)
Non-trainable params: 0 (0.00 B)
Optimizer params: 9,184 (35.88 KB)
None
5/5 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - acc: 1.0000 - loss: 2.0380e-09
[3.926279301680324e-09, 1.0]
```

ICP5

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Gemini

```python
import keras
import pandas as pd
import numpy as np
from keras.models import Sequential
from keras.layers import Dense
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

# Load dataset
dataset = pd.read_csv(path_to_csv1, header=None).values

X = dataset[1:, 2:-1]  # Features
Y = dataset[1:, -1]   # Labels (M or B)

# Convert labels to binary format
Y = np.where(Y == 'M', 1, 0)  # M -> 1, B -> 0

# Convert to numeric
X = X.astype(np.float64)  # Convert X to numeric

# Split data into training and testing sets
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.25, random_state=87)

# Normalize the data
sc = StandardScaler()
```

ICP5

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Gemini

```python
# Normalize the data
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

# Set random seed for reproducibility
np.random.seed(155)

# Create the model
my_first_nn = Sequential()  # create model
my_first_nn.add(Dense(20, input_dim=30, activation='relu'))  # hidden layer
my_first_nn.add(Dense(30, activation='relu'))  # hidden layer
my_first_nn.add(Dense(40, activation='relu'))  # hidden layer
my_first_nn.add(Dense(50, activation='relu'))  # hidden layer
my_first_nn.add(Dense(1, activation='sigmoid'))  # output layer

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

# Train the model
my_first_nn_fitted = my_first_nn.fit(X_train, Y_train, epochs=100, initial_epoch=0)

# Print the model summary and evaluate accuracy on the test set
print(my_first_nn.summary())
print(my_first_nn.evaluate(X_test, Y_test))
```

ICP5

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Gemini

```
/usr/local/lib/python3.10/dist-packages/keras/src/layers/core/dense.py:87: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a layer
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
Epoch 1/100
14/14 ━━━━━━━━━━━━━━━━━━━━ 4s 11ms/step - acc: 0.9556 - loss: 0.5620
Epoch 2/100
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - acc: 1.0000 - loss: 0.2461
Epoch 3/100
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - acc: 1.0000 - loss: 0.0600
Epoch 4/100
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - acc: 1.0000 - loss: 0.0130
Epoch 5/100
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step - acc: 1.0000 - loss: 0.0049
Epoch 6/100
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 6ms/step - acc: 1.0000 - loss: 0.0033
Epoch 7/100
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 6ms/step - acc: 1.0000 - loss: 0.0016
Epoch 8/100
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - acc: 1.0000 - loss: 0.0011
Epoch 9/100
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 13ms/step - acc: 1.0000 - loss: 8.6115e-04
Epoch 10/100
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - acc: 1.0000 - loss: 9.1204e-04
Epoch 11/100
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - acc: 1.0000 - loss: 7.6458e-04
Epoch 12/100
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - acc: 1.0000 - loss: 5.1669e-04
Epoch 13/100
```

Connected to Python 3 Google Compute Engine backend

ICP5

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Gemini

```
Epoch 98/100
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - acc: 1.0000 - loss: 5.6507e-06
Epoch 99/100
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - acc: 1.0000 - loss: 4.5875e-06
Epoch 100/100
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - acc: 1.0000 - loss: 5.1632e-06
Model: "sequential_9"
```

Layer (type)	Output Shape	Param #
dense_30 (Dense)	(None, 20)	620
dense_31 (Dense)	(None, 30)	630
dense_32 (Dense)	(None, 40)	1,240
dense_33 (Dense)	(None, 50)	2,050
dense_34 (Dense)	(None, 1)	51

```
Total params: 13,775 (53.81 KB)
Trainable params: 4,591 (17.93 KB)
Non-trainable params: 0 (0.00 B)
Optimizer params: 9,184 (35.88 KB)
None
5/5 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - acc: 1.0000 - loss: 1.1684e-05
[1.3084764759696554e-05, 1.0]
```

Connected to Python 3 Google Compute Engine backend

ICP5

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Gemini

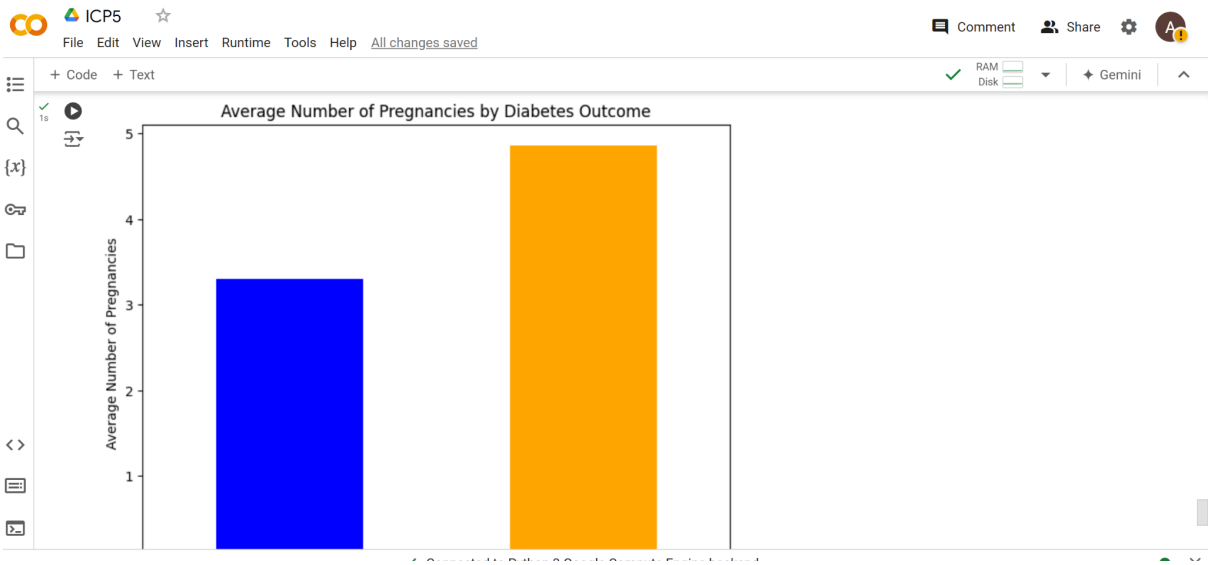
```
import pandas as pd
import matplotlib.pyplot as plt

# Load the dataset
diabetes_data = pd.read_csv('/content/gdrive/My Drive/diabetes.csv')

# Add column names
diabetes_data.columns = ['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
                         'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome']

# Calculate the average number of pregnancies for each diabetes outcome
avg_pregnancies = diabetes_data.groupby('Outcome')['Pregnancies'].mean()

# Plot the bar graph
plt.figure(figsize=(8,6))
avg_pregnancies.plot(kind='bar', color=['blue', 'orange'])
plt.title('Average Number of Pregnancies by Diabetes Outcome')
plt.xlabel('Diabetes Outcome (0 = No, 1 = Yes)')
plt.ylabel('Average Number of Pregnancies')
plt.xticks(rotation=0)
plt.show()
```



My GitHub link:https://github.com/w8162583/bda.git

YouTube link:https://youtu.be/phXHa1d6-V0