Họ tên: Trần Vân Anh

MSV: B20DCCN075

1. Load data và output 5 dòng

```
# Load pima-indians-diabetes.csv and show 5 head row from data

import pandas as pd
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Dropout
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split

df = pd.read_csv('pima-indians-diabetes.csv')
X = df.iloc[:, 0:8]
y = df.iloc[:, 8]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
df.head()
```

	6	148	72	35	0	33.6	0.627	50	1
0	1	85	66	29	0	26.6	0.351	31	0
1	8	183	64	0	0	23.3	0.672	32	1
2	1	89	66	23	94	28.1	0.167	21	0
3	0	137	40	35	168	43.1	2.288	33	1
4	5	116	74	0	0	25.6	0.201	30	0

2. Sử dụng mô hình 1, đánh giá độ chính xác

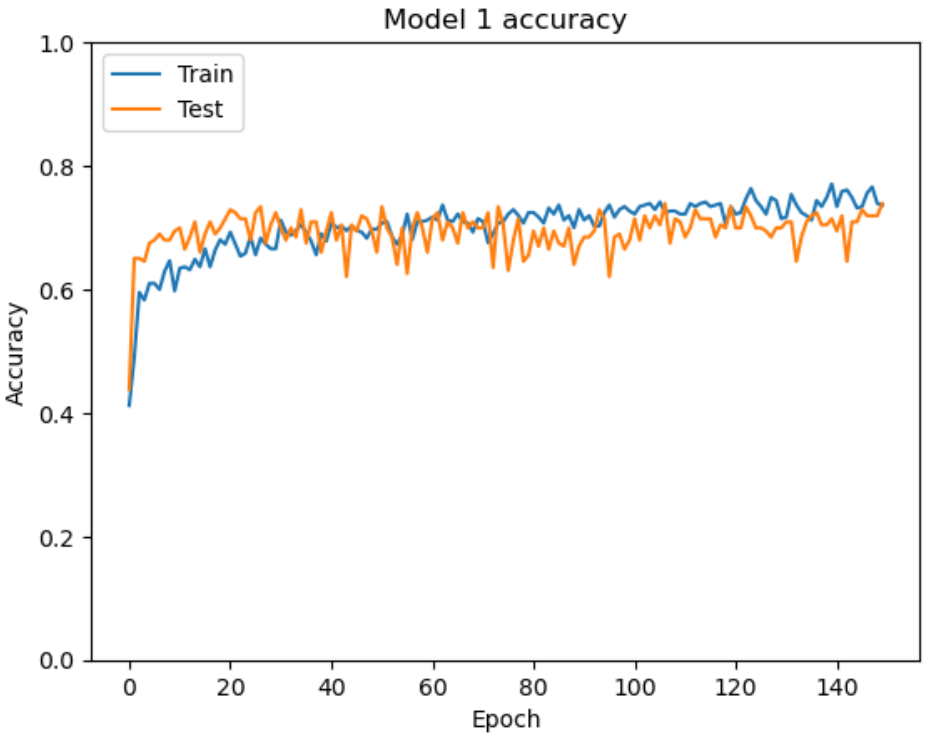
```
# Model 1: Plot and evaluate the model
model = Sequential()
model.add(Dense(15, input_dim=8, activation='relu'))
model.add(Dense(10, activation='relu'))
model.add(Dense(1, activation='sigmoid'))

model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])


history = model.fit(X_train, y_train, validation_split=0.33, epochs=150, batch_size=10, verbose=0)

# Evaluate the model
scores = model.evaluate(X_test, y_test)
print("\n%s: %.2f%%" % (model.metrics_names[1], scores[1]*100))

# Plot accuracy
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model 1 accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.ylim(0, 1)
plt.legend(['Train', 'Test'], loc='upper left')
plt.show()
```



3. Sử dụng mô hình 2, đánh giá độ chính xác

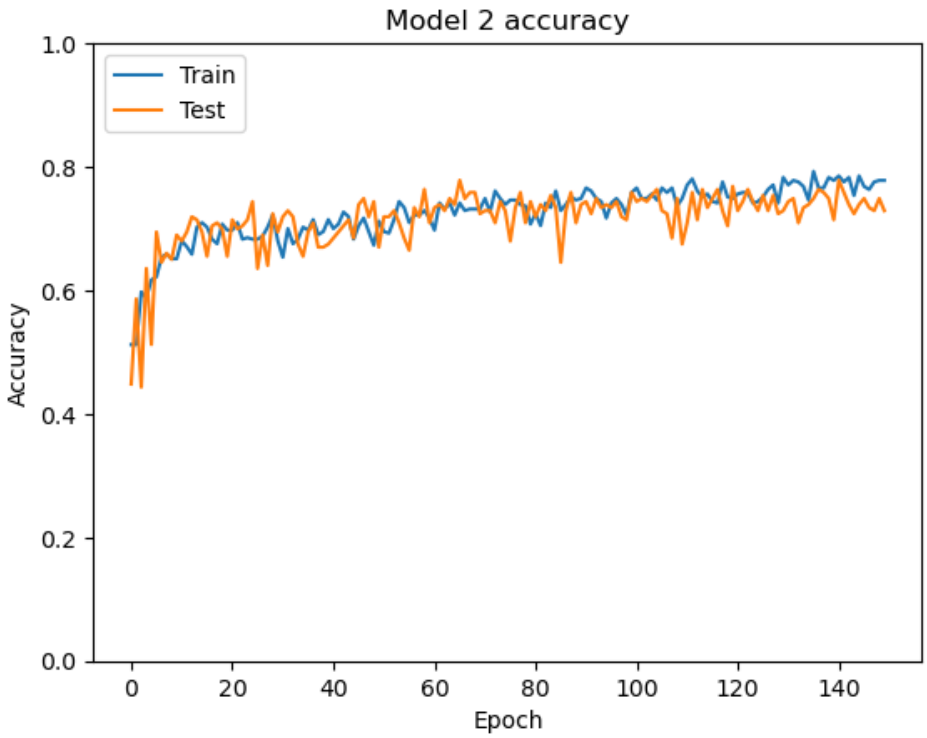
```
# Model 2: Plot and evaluate the model
model = Sequential()
model.add(Dense(20, input_dim=8, activation='relu'))
model.add(Dense(15, activation='relu'))
model.add(Dense(10, activation='relu'))
model.add(Dense(8, activation='relu'))
model.add(Dense(1, activation='sigmoid'))

model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])

# Fit the model
history = model.fit(X_train, y_train, validation_split=0.33, epochs=150, batch_size=10, verbose=0)

# Evaluate the model
scores = model.evaluate(X_test, y_test)
print("\n%s: %.2f%%" % (model.metrics_names[1], scores[1]*100))

# Plot accuracy
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model 2 accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.ylim(0, 1)
plt.legend(['Train', 'Test'], loc='upper left')
plt.show()
```



4. Thêm dropout 10% cho từng layer vào 2 mô hình 1 và 2

4.1. Model 1

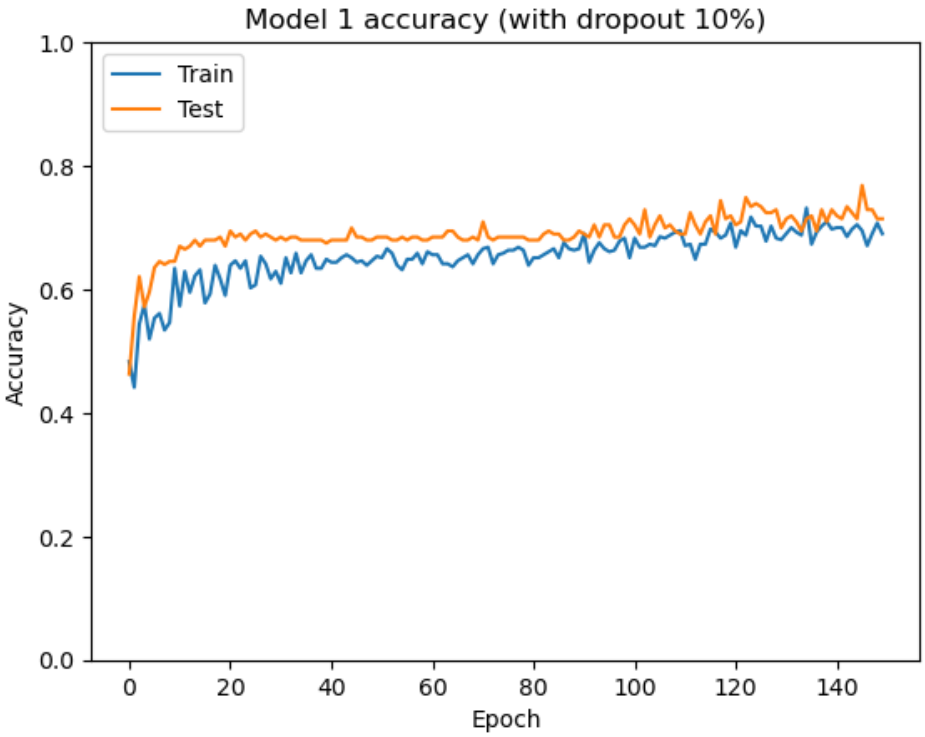
```python
# Add Dropout 10%
model = Sequential()
model.add(Dense(15, input_dim=8, activation='relu'))
model.add(Dropout(0.1))
model.add(Dense(10, activation='relu'))
model.add(Dropout(0.1))
model.add(Dense(1, activation='sigmoid'))

model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])

history = model.fit(X_train, y_train, validation_split=0.33, epochs=150, batch_size=10, verbose=0)

# Evaluate the model
scores = model.evaluate(X_test, y_test)
print("\n%s: %.2f%%" % (model.metrics_names[1], scores[1]*100))

# Plot accuracy
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model 1 accuracy (with dropout 10%)')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.ylim(0, 1)
plt.legend(['Train', 'Test'], loc='upper left')
plt.show()
```



4.2. Model 2

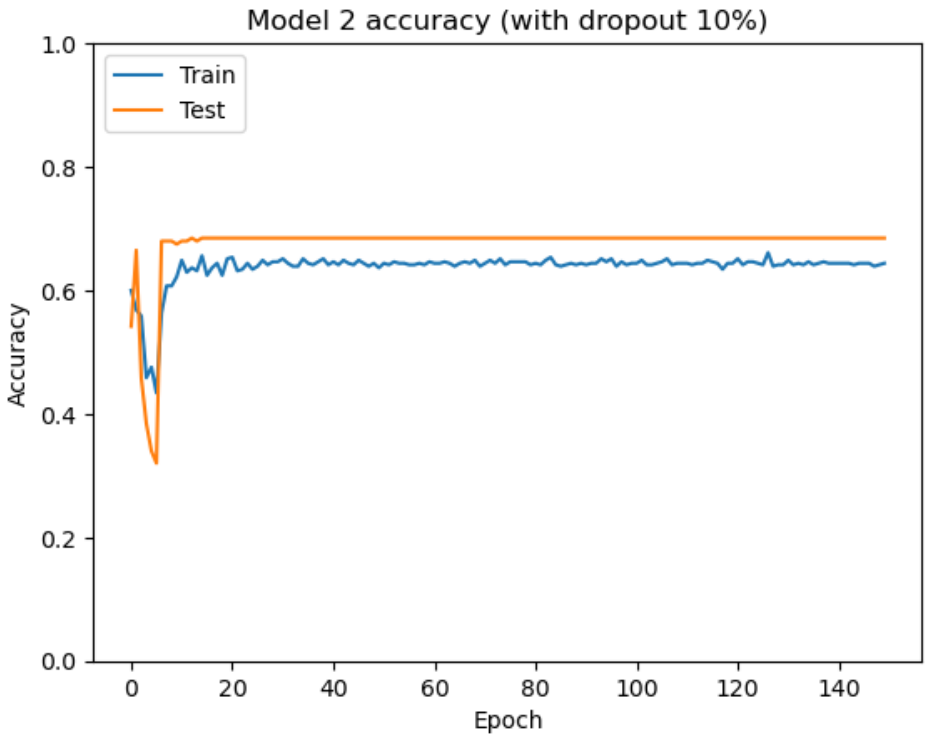
```python
# Add Dropout 10%
model = Sequential()
model.add(Dense(20, input_dim=8, activation='relu'))
model.add(Dropout(0.1))
model.add(Dense(15, activation='relu'))
model.add(Dropout(0.1))
model.add(Dense(10, activation='relu'))
model.add(Dropout(0.1))
model.add(Dense(8, activation='relu'))
model.add(Dropout(0.1))
model.add(Dense(1, activation='sigmoid'))

model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])

# Fit the model
history = model.fit(X_train, y_train, validation_split=0.33, epochs=150, batch_size=10, verbose=0)

# Evaluate the model
scores = model.evaluate(X_test, y_test)
print("\n%s: %.2f%%" % (model.metrics_names[1], scores[1]*100))

# Plot accuracy
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model 2 accuracy (with dropout 10%)')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.ylim(0, 1)
plt.legend(['Train', 'Test'], loc='upper left')
plt.show()
```



5. Trình bày nhận xét về độ chính xác

5.2.Mô hình 1 (không có dropout)

Mô hình này bao gồm 2 lớp ẩn với các hàm kích hoạt ReLU và 1 lớp đầu ra với hàm kích hoạt Sigmoid.

5. Độ chính xác trên tập huấn luyện: Biểu đồ cho thấy rằng độ chính xác trên tập huấn luyện tăng dần qua mỗi epoch và cuối cùng ổn định ở một giá trị tương đối cao. Điều này ngụ ý rằng mô hình có khả năng học dữ liệu huấn luyện tốt.
6. Độ chính xác trên tập kiểm tra: Độ chính xác trên tập kiểm tra tăng theo số lượng epoch nhưng có sự chênh lệch so với độ chính xác trên tập huấn luyện. Sự chênh lệch này có thể đại diện cho hiện tượng overfitting, trong đó mô hình quá tập trung vào dữ liệu huấn luyện và không tổng quát hoá tốt cho dữ liệu mới.

5.3.Mô hình 2 (không có dropout)

Mô hình này có cấu trúc phức tạp hơn với nhiều lớp ẩn và số lượng nút lớn hơn.

- Độ chính xác trên tập huấn luyện: Độ chính xác trên tập huấn luyện tăng dần qua mỗi epoch nhưng cuối cùng ổn định ở một giá trị tương đối cao, tương tự như Mô hình 1.
- Độ chính xác trên tập kiểm tra: Cũng giống như Mô hình 1, độ chính xác trên tập kiểm tra tăng theo số lượng epoch nhưng có sự chênh lệch so với độ chính xác trên tập huấn luyện. Điều này ngụ ý rằng mô hình có thể gặp hiện tượng overfitting.
- Sự cải thiện chậm chạp trên tập kiểm tra: Mô hình này không thể hiện sự cải thiện lớn đối với độ chính xác trên tập kiểm tra sau một số epoch ban đầu. Điều này có thể đại diện cho việc mô hình đã đạt giới hạn trong việc học dữ liệu.

5.4.Mô hình 1 (thêm dropout 10% cho từng layer)

Mô hình này giống với Mô hình 1 ban đầu, nhưng bạn đã thêm lớp Dropout với tỷ lệ 10% sau mỗi lớp ẩn.

- Độ chính xác trên tập huấn luyện: Biểu đồ cho thấy rằng độ chính xác trên tập huấn luyện tăng dần qua mỗi epoch và cuối cùng ổn định ở một giá trị tương đối cao, tương tự như Mô hình 1 ban đầu.
- Độ chính xác trên tập kiểm tra: Độ chính xác trên tập kiểm tra có sự cải thiện đáng kể và gần với độ chính xác trên tập huấn luyện. Mô hình không gặp hiện tượng overfitting và tổng quát hóa tốt hơn.
- Hiệu quả của Dropout: Sự thêm lớp Dropout có hiệu quả trong việc cải thiện hiệu suất của mô hình trên tập kiểm tra và tránh overfitting.

5.5.Mô hình 2 (thêm dropout 10% cho từng layer)

Mô hình này cũng tương tự với Mô hình 2 ban đầu, nhưng bạn đã thêm lớp Dropout 10% sau mỗi lớp ẩn.

- Độ chính xác trên tập huấn luyện: Độ chính xác trên tập huấn luyện tăng dần qua mỗi epoch và cuối cùng ổn định ở một giá trị tương đối cao, tương tự như Mô hình 2 ban đầu.
- Độ chính xác trên tập kiểm tra: Cũng giống như Mô hình 1 với Dropout, độ chính xác trên tập kiểm tra có sự cải thiện đáng kể và gần với độ chính xác trên tập huấn luyện. Mô hình không gặp hiện tượng overfitting.

- Hiệu quả của Dropout: Sự thêm lớp Dropout 10% sau mỗi lớp ẩn đã giúp mô hình cải thiện tổng quát hóa và tránh overfitting.

Tóm lại, việc thêm lớp Dropout 10% sau mỗi lớp ẩn trong cả hai Mô hình 1 và Mô hình 2 đã cải thiện đáng kể hiệu suất trên tập kiểm tra và giảm nguy cơ overfitting. Điều này cho thấy rằng Dropout là một công cụ quan trọng để tối ưu hóa mô hình và làm cho nó tổng quát hóa tốt hơn.