```
AI Externship Assignment 2
Vishwas Saproo
20BBS0217
```

```
import numpy as np
import seaborn as sns
from sklearn.model_selection import train_test_split
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from sklearn.preprocessing import LabelEncoder, StandardScaler
```

```
import pandas as pd
df=pd.read_csv("/content/drug200.csv")
print(df)
```

```
     Age Sex      BP Cholesterol  Na_to_K   Drug
0     23   F    HIGH        HIGH   25.355  DrugY
1     47   M     LOW        HIGH   13.093  drugC
2     47   M     LOW        HIGH   10.114  drugC
3     28   F  NORMAL        HIGH    7.798  drugX
4     61   F     LOW        HIGH   18.043  DrugY
..   ...  ..     ...         ...      ...    ...
195   56   F     LOW        HIGH   11.567  drugC
196   16   M     LOW        HIGH   12.006  drugC
197   52   M  NORMAL        HIGH    9.894  drugX
198   23   M  NORMAL      NORMAL   14.020  drugX
199   40   F     LOW      NORMAL   11.349  drugX

[200 rows x 6 columns]
```

```
df.describe(include='all')
```

	Age	Sex	BP	Cholesterol	Na_to_K	Drug
count	200.000000	200	200	200	200.000000	200
unique	NaN	2	3	2	NaN	5
top	NaN	M	HIGH	HIGH	NaN	DrugY
freq	NaN	104	77	103	NaN	91
mean	44.315000	NaN	NaN	NaN	16.084485	NaN
std	16.544315	NaN	NaN	NaN	7.223956	NaN
min	15.000000	NaN	NaN	NaN	6.269000	NaN
25%	31.000000	NaN	NaN	NaN	10.445500	NaN
50%	45.000000	NaN	NaN	NaN	13.936500	NaN
75%	58.000000	NaN	NaN	NaN	19.380000	NaN
max	74.000000	NaN	NaN	NaN	38.247000	NaN

```
df.isnull().sum()
```

```
Age            0
Sex            0
BP             0
Cholesterol    0
Na_to_K        0
Drug           0
dtype: int64
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 6 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Age          200 non-null    int64
 1   Sex          200 non-null    object
 2   BP           200 non-null    object
 3   Cholesterol  200 non-null    object
 4   Na_to_K      200 non-null    float64
```

```
 5   Drug         200 non-null    object
dtypes: float64(1), int64(1), object(4)
memory usage: 9.5+ KB
```

Data Splitting

```
y = pd.get_dummies(df.iloc[:,5:]).values
x = df.drop('Drug', axis=1)
```

Label Encoding

```
categorical_features = {'Sex', 'BP', "Cholesterol"}
label_encoders={}
for feature in categorical_features:
  label_encoders[feature] = LabelEncoder()
  x[feature] = label_encoders[feature].fit_transform(x[feature])
```

Split data into test and train

```
xtrain,xtest,ytrain,ytest=train_test_split(x,y,test_size=0.2,random_state=21)
```

Task - 2 Build the ANN model with (input layer, min 3 hidden layers & output layer)

Creating ANN model

```
model = Sequential()
model.add(Dense(5,activation='relu'))
model.add(Dense(32,activation='relu'))
model.add(Dense(26,activation='relu'))
model.add(Dense(18,activation='relu'))
model.add(Dense(12,activation='relu'))
model.add(Dense(5,activation='relu'))
```

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

```
model.fit(xtrain,ytrain, batch_size=10, epochs=10, validation_data=(xtest,ytest))
```

```
Epoch 1/10
16/16 [==============================] - 3s 24ms/step - loss: 3.0404 - accuracy: 0.2812 - val_loss: 3.1187 - val_accuracy: 0.3000
Epoch 2/10
16/16 [==============================] - 0s 4ms/step - loss: 2.9617 - accuracy: 0.4313 - val_loss: 2.9362 - val_accuracy: 0.5250
Epoch 3/10
16/16 [==============================] - 0s 4ms/step - loss: 2.8926 - accuracy: 0.4375 - val_loss: 3.2140 - val_accuracy: 0.5250
Epoch 4/10
16/16 [==============================] - 0s 4ms/step - loss: 3.0338 - accuracy: 0.4375 - val_loss: 2.8949 - val_accuracy: 0.5250
Epoch 5/10
16/16 [==============================] - 0s 4ms/step - loss: 2.9868 - accuracy: 0.4375 - val_loss: 3.2112 - val_accuracy: 0.5250
Epoch 6/10
16/16 [==============================] - 0s 4ms/step - loss: 2.8310 - accuracy: 0.5000 - val_loss: 2.9126 - val_accuracy: 0.4500
Epoch 7/10
16/16 [==============================] - 0s 4ms/step - loss: 2.8172 - accuracy: 0.5562 - val_loss: 3.2625 - val_accuracy: 0.4500
Epoch 8/10
16/16 [==============================] - 0s 4ms/step - loss: 2.8125 - accuracy: 0.5312 - val_loss: 2.9585 - val_accuracy: 0.5500
Epoch 9/10
16/16 [==============================] - 0s 4ms/step - loss: 2.8153 - accuracy: 0.5375 - val_loss: 2.9582 - val_accuracy: 0.5750
Epoch 10/10
16/16 [==============================] - 0s 4ms/step - loss: 2.8828 - accuracy: 0.4375 - val_loss: 3.2298 - val_accuracy: 0.6000
<keras.callbacks.History at 0x7f52428434c0>
```

Task - 3 Test the model with random data.

Testing with random values

```
model.predict([[12,1,0,5.146,44]])
```

```
1/1 [==============================] - 0s 456ms/step
array([[7.849716, 0.      , 0.      , 0.      , 0.      ]], dtype=float32)
```

```
model.predict([[76,2,5.876,3,7]])
```

```
1/1 [==============================] - 0s 52ms/step
array([[4.1602273, 0.       , 1.9347914, 2.3882403, 4.1126733]],
      dtype=float32)
```