## 23MCA0346

Shylesh Kumar. S

## Question 1

Naïve Bayes classification

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.datasets import load_iris
from sklearn.metrics import accuracy_score
```

```
data = load_iris()
data.target[[10, 25, 50]]
list(data.target_names)
```

```
['setosa', 'versicolor', 'virginica']
```

```
X=data.data
y=data.target
```

```
X_train,X_test,y_train,y_test=train_test_split(X, y, test_size=0.3, random_state=42)
```

```
from sklearn.naive_bayes import GaussianNB
model =GaussianNB()
```

```
model.fit(X_train, y_train)
```

```
GaussianNB
GaussianNB()
```

```
y_pred = model.predict(X_test)
```

```
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

```
Accuracy: 0.9777777777777777
```

## Question 2

Neural Network

```
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.datasets import load_iris
from sklearn.preprocessing import StandardScaler
import tensorflow as tf
from tensorflow.keras import layers, models
```

```
scaler = StandardScaler() #standardizing features
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

```
model = models.Sequential([   #architecture
    layers.Dense(64, activation='relu', input_shape=(X_train.shape[1],)),
    layers.Dense(64, activation='relu'),
    layers.Dense(3, activation='softmax')
])


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


history = model.fit(X_train, y_train, epochs=100, validation_split=0.3)
```

```
3/3 [==============================] - 0s 25ms/step - loss: 0.0773 - accuracy: 0.9726 - val_loss: 0.1842 - val_accuracy: 0.90
Epoch 73/100
3/3 [==============================] - 0s 23ms/step - loss: 0.0762 - accuracy: 0.9863 - val_loss: 0.1938 - val_accuracy: 0.90
Epoch 74/100
3/3 [==============================] - 0s 26ms/step - loss: 0.0768 - accuracy: 0.9726 - val_loss: 0.1924 - val_accuracy: 0.90
Epoch 75/100
3/3 [==============================] - 0s 26ms/step - loss: 0.0759 - accuracy: 0.9863 - val_loss: 0.1867 - val_accuracy: 0.90
Epoch 76/100
3/3 [==============================] - 0s 26ms/step - loss: 0.0751 - accuracy: 0.9863 - val_loss: 0.1807 - val_accuracy: 0.90
Epoch 77/100
3/3 [==============================] - 0s 17ms/step - loss: 0.0746 - accuracy: 0.9863 - val_loss: 0.1728 - val_accuracy: 0.93
Epoch 78/100
3/3 [==============================] - 0s 17ms/step - loss: 0.0750 - accuracy: 0.9589 - val_loss: 0.1689 - val_accuracy: 0.93
Epoch 79/100
3/3 [==============================] - 0s 26ms/step - loss: 0.0751 - accuracy: 0.9589 - val_loss: 0.1679 - val_accuracy: 0.93
Epoch 80/100
3/3 [==============================] - 0s 16ms/step - loss: 0.0756 - accuracy: 0.9589 - val_loss: 0.1698 - val_accuracy: 0.93
Epoch 81/100
3/3 [==============================] - 0s 17ms/step - loss: 0.0741 - accuracy: 0.9589 - val_loss: 0.1683 - val_accuracy: 0.93
Epoch 82/100
3/3 [==============================] - 0s 27ms/step - loss: 0.0738 - accuracy: 0.9589 - val_loss: 0.1695 - val_accuracy: 0.93
Epoch 83/100
3/3 [==============================] - 0s 17ms/step - loss: 0.0729 - accuracy: 0.9726 - val_loss: 0.1723 - val_accuracy: 0.93
Epoch 84/100
3/3 [==============================] - 0s 20ms/step - loss: 0.0721 - accuracy: 0.9726 - val_loss: 0.1776 - val_accuracy: 0.90
Epoch 85/100
3/3 [==============================] - 0s 27ms/step - loss: 0.0716 - accuracy: 0.9863 - val_loss: 0.1844 - val_accuracy: 0.90
Epoch 86/100
3/3 [==============================] - 0s 17ms/step - loss: 0.0710 - accuracy: 0.9863 - val_loss: 0.1840 - val_accuracy: 0.90
Epoch 87/100
3/3 [==============================] - 0s 26ms/step - loss: 0.0711 - accuracy: 0.9863 - val_loss: 0.1811 - val_accuracy: 0.90
Epoch 88/100
3/3 [==============================] - 0s 28ms/step - loss: 0.0711 - accuracy: 0.9863 - val_loss: 0.1774 - val_accuracy: 0.90
Epoch 89/100
3/3 [==============================] - 0s 17ms/step - loss: 0.0713 - accuracy: 0.9863 - val_loss: 0.1760 - val_accuracy: 0.93
Epoch 90/100
3/3 [==============================] - 0s 27ms/step - loss: 0.0687 - accuracy: 0.9863 - val_loss: 0.1888 - val_accuracy: 0.90
Epoch 91/100
3/3 [==============================] - 0s 20ms/step - loss: 0.0690 - accuracy: 0.9726 - val_loss: 0.2126 - val_accuracy: 0.90
Epoch 92/100
3/3 [==============================] - 0s 28ms/step - loss: 0.0759 - accuracy: 0.9589 - val_loss: 0.2259 - val_accuracy: 0.90
Epoch 93/100
3/3 [==============================] - 0s 16ms/step - loss: 0.0767 - accuracy: 0.9589 - val_loss: 0.2172 - val_accuracy: 0.90
Epoch 94/100
3/3 [==============================] - 0s 16ms/step - loss: 0.0745 - accuracy: 0.9726 - val_loss: 0.2067 - val_accuracy: 0.90
Epoch 95/100
3/3 [==============================] - 0s 16ms/step - loss: 0.0714 - accuracy: 0.9726 - val_loss: 0.2010 - val_accuracy: 0.90
Epoch 96/100
3/3 [==============================] - 0s 27ms/step - loss: 0.0703 - accuracy: 0.9726 - val_loss: 0.1839 - val_accuracy: 0.90
Epoch 97/100
3/3 [==============================] - 0s 20ms/step - loss: 0.0670 - accuracy: 0.9726 - val_loss: 0.1700 - val_accuracy: 0.90
Epoch 98/100
3/3 [==============================] - 0s 26ms/step - loss: 0.0687 - accuracy: 0.9863 - val_loss: 0.1572 - val_accuracy: 0.90
Epoch 99/100
3/3 [==============================] - 0s 18ms/step - loss: 0.0690 - accuracy: 0.9726 - val_loss: 0.1545 - val_accuracy: 0.90
Epoch 100/100
3/3 [==============================] - 0s 17ms/step - loss: 0.0686 - accuracy: 0.9726 - val_loss: 0.1576 - val_accuracy: 0.90
```

```
test_loss, test_accuracy = model.evaluate(X_test, y_test)
print('Test accuracy:', test_accuracy)
```

```
2/2 [==============================] - 0s 8ms/step - loss: 0.0276 - accuracy: 1.0000
Test accuracy: 1.0
```

```
import matplotlib.pyplot as plt
labels = ['Neural Network', 'Naive']
accuracies = [test_accuracy, accuracy]
plt.bar(labels, accuracies, color=['blue', 'green'])
plt.xlabel('Model')
plt.ylabel('Accuracy')
plt.title('Comparison of Test Accuracies')
plt.show()
```