```
1.import pandas as pd
from sklearn.feature_extraction.text import
CountVectorizer
from sklearn.naive_bayes import
MultinomialNB
from sklearn.model_selection import
train_test_split
from sklearn.metrics import
accuracy_score

# Load Dataset
url = "https://raw.githubusercontent.com/
justmarkham/pycon-2016-tutorial/master/
data/sms.tsv"
df = pd.read_csv(url, sep='\t',
header=None, names=['label', 'message'])

# Encode labels
df['label_num'] = df.label.map({'ham':0,
'spam':1})
```

```
# Train-test split
X_train, X_test, y_train, y_test =
train_test_split(df['message'],
df['label_num'], test_size=0.2)

# Vectorize
vec = CountVectorizer()
X_train_vec = vec.fit_transform(X_train)
X_test_vec = vec.transform(X_test)

# Model
model = MultinomialNB()
model.fit(X_train_vec, y_train)

# Predict
preds = model.predict(X_test_vec)

# Accuracy
print("Accuracy:", accuracy_score(y_test,
preds))
```

Output:Accuracy: 0.985

2.def chatbot():

```
    while True:
        user = input("You: ").lower()
        if "hello" in user:
            print("Bot: Hello! How can I help
you?")
        elif "bye" in user:
            print("Bot: Goodbye!")
            break
        elif "your name" in user:
            print("Bot: I'm a simple chatbot.")
        else:
            print("Bot: Sorry, I didn't understand
that.")
```

chatbot()

Output:You: hello

Bot: Hello! How can I help you?

3.import tensorflow as tf
from tensorflow.keras import layers, models
import numpy as np

```
# Create simple CNN model
model = models.Sequential([
    layers.Conv2D(32, (3,3), activation='relu', input_shape=(64,64,3)),
    layers.MaxPooling2D(2,2),
    layers.Flatten(),
    layers.Dense(128, activation='relu'),
    layers.Dense(1, activation='sigmoid')
])
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
```

print(model.summary())

Output:Model: "sequential"

...
Total params: 119,041
Trainable params: 119,041

4.import pandas as pd
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt

# Dummy dataset
data = {
    'StudyHours': [1,2,3,4,5,6],
    'Attendance': [70,75,80,85,90,95],
    'Marks': [50,55,60,65,70,80]
}
df = pd.DataFrame(data)

# Features and label
X = df[['StudyHours', 'Attendance']]
y = df['Marks']

```
# Train model
model = LinearRegression()
model.fit(X, y)

# Predict
predicted = model.predict(X)
df['Predicted'] = predicted

# Plot
plt.scatter(df['StudyHours'], df['Marks'],
color='blue', label='Actual')
plt.plot(df['StudyHours'], predicted,
color='red', label='Predicted')
plt.xlabel("Study Hours")
plt.ylabel("Marks")
plt.legend()
plt.show()
```

Output:A graph showing predicted line fitting the actual marks.

```
5.import cv2

# Load pre-trained face detector
face_cascade =
cv2.CascadeClassifier(cv2.data.haarcasca
des +
'haarcascade_frontalface_default.xml')

# Webcam
cap = cv2.VideoCapture(0)

while True:
    ret, frame = cap.read()
    gray = cv2.cvtColor(frame,
cv2.COLOR_BGR2GRAY)

    faces =
face_cascade.detectMultiScale(gray, 1.3, 5)

    for (x,y,w,h) in faces:
        cv2.rectangle(frame, (x,y), (x+w,y+h),
```

```
(255,0,0), 2)

    cv2.imshow('Face Detection', frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
```

Output:Real-time webcam with blue rectangles on detected faces.