VOICE COMMAND RECOGNITION

```
import numpy as np import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB from
sklearn.neural_network import MLPClassifier from
sklearn.metrics import accuracy_score, f1_score

np.random.seed(42)
X = np.random.rand(500, 13)  # 500 samples, 13 features (like MFCCs from speech)
y = np.random.choice(['yes', 'no', 'up', 'down', 'left', 'right', 'on', 'off', 'stop', 'go'], 500)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

models = {
   'KNN': KNeighborsClassifier(n_neighbors=5),
   'Weighted KNN': KNeighborsClassifier(n_neighbors=5, weights='distance'),
   'Naive Bayes': GaussianNB(),
   'ANN': MLPClassifier(hidden_layer_sizes=(50,), max_iter=200)
}
accuracy_results = {} f1_results
= {}

print("Training and evaluating models\n") for
model_name, model in models.items():
   model.fit(X_train, y_train)
y_pred = model.predict(X_test)

   acc = accuracy_score(y_test, y_pred)     f1 =
f1_score(y_test, y_pred, average='weighted')
```

```
    accuracy_results[model_name] = acc
    f1_results[model_name] = f1

    print(f"{model_name} -> Accuracy: {acc:.4f}, F1-Score: {f1:.4f}")

labels = list(accuracy_results.keys()) accuracy_values
= list(accuracy_results.values()) f1_values =
list(f1_results.values())

plt.figure(figsize=(10,6)) plt.plot(labels, accuracy_values, marker='o', linestyle='-',
color='blue', label='Accuracy') plt.plot(labels, f1_values, marker='s', linestyle='--',
color='green', label='F1-Score') plt.title('Model Performance on Voice Command
Recognition (Line Chart)') plt.xlabel('Models') plt.ylabel('Scores') plt.ylim(0, 1)
plt.legend() plt.grid(True) plt.show()

plt.figure(figsize=(8,8))
plt.pie(accuracy_values, labels=labels, autopct='%1.1f%%', startangle=140, colors=['skyblue',
'lightgreen', 'orange', 'pink']) plt.title('Model
Accuracy Distribution (Pie Chart)') plt.axis('equal')
plt.show()

x = np.arange(len(labels)) width = 0.35 fig, ax = plt.subplots(figsize=(10,6)) rects1 =
ax.bar(x - width/2, accuracy_values, width, label='Accuracy', color='orchid') rects2 =
ax.bar(x + width/2, f1_values, width, label='F1-Score', color='lightseagreen')
ax.set_ylabel('Scores') ax.set_title('Model Comparison on Voice Commands (Bar
Chart)') ax.set_xticks(x) ax.set_xticklabels(labels) ax.set_ylim(0, 1) ax.legend()

for rect in rects1 + rects2:     height = rect.get_height()
ax.annotate(f'{height:.2f}',                xy=(rect.get_x() +
rect.get_width()/2, height),                xytext=(0,3),
```

textcoords="offset points", ha='center',

va='bottom')

fig.tight_layout() plt.show()