STEP 1: Import Libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
```

STEP 2: Load Dataset

```
df = pd.read_csv("/content/digits_pca_dataset.csv")

X = df.drop("label", axis=1)
y = df["label"]
```

STEP 3: Feature Scaling

```
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

STEP 4: Apply PCA with Different Components

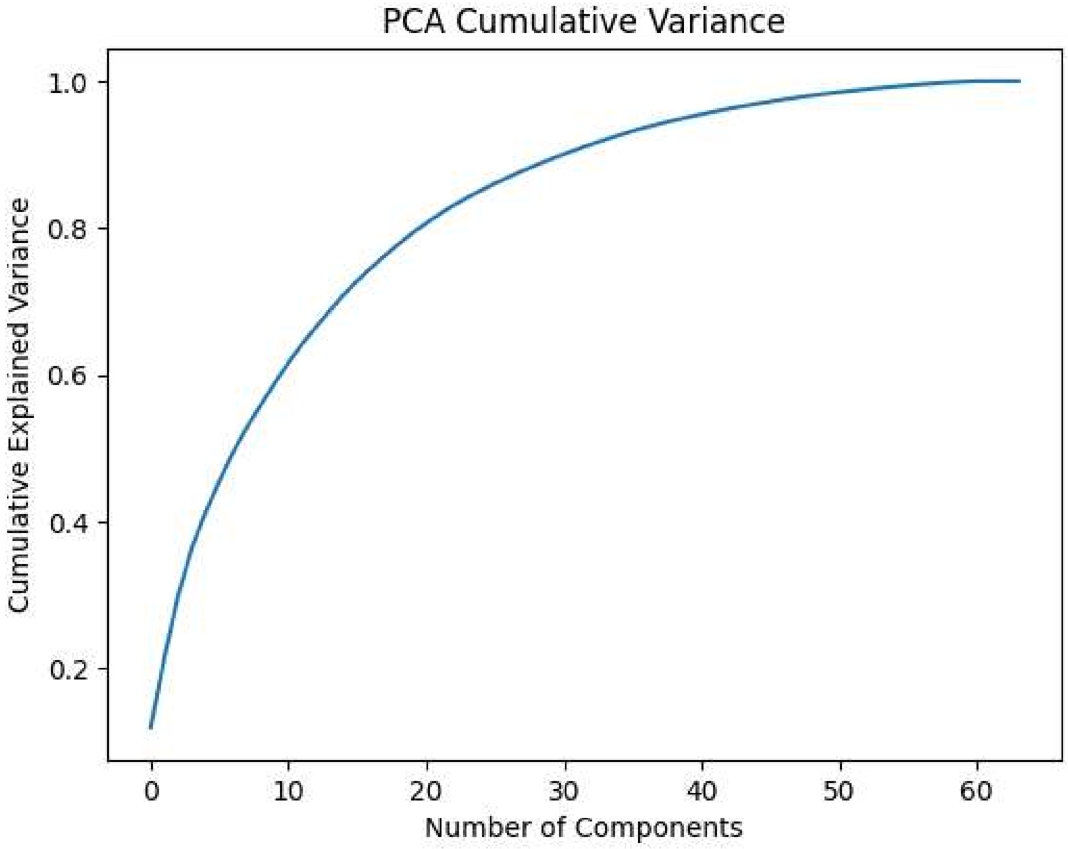
```
components = [2, 10, 30, 50]
explained_variances = []

for n in components:
    pca = PCA(n_components=n)
    pca.fit(X_scaled)
    explained_variances.append(sum(pca.explained_variance_ratio_))
    print(f"Components: {n}, Explained Variance: {sum(pca.explained_variance_ratio_
```

```
Components: 2, Explained Variance: 0.21594970500832805
Components: 10, Explained Variance: 0.5887375533730298
Components: 30, Explained Variance: 0.893208438244957
Components: 50, Explained Variance: 0.982759194602153
```

STEP 5: Plot Cumulative Variance

```
pca_full = PCA()
pca_full.fit(X_scaled)
```

```
cumulative_variance = np.cumsum(pca_full.explained_variance_ratio_)

plt.plot(cumulative_variance)
plt.xlabel("Number of Components")
plt.ylabel("Cumulative Explained Variance")
plt.title("PCA Cumulative Variance")
plt.show()
```



STEP 6: Transform Dataset (Example: 30 Components)

```
pca = PCA(n_components=30)
X_reduced = pca.fit_transform(X_scaled)
```

STEP 7: Train-Test Split

```
X_train, X_test, y_train, y_test = train_test_split(
    X_scaled, y, test_size=0.2, random_state=42, stratify=y
)

X_train_red, X_test_red, _, _ = train_test_split(
    X_reduced, y, test_size=0.2, random_state=42, stratify=y
)
```

STEP 8: Train Logistic Regression (Original Data)

```
model_original = LogisticRegression(max_iter=2000)
model_original.fit(X_train, y_train)

pred_original = model_original.predict(X_test)
print("Original Accuracy:", accuracy_score(y_test, pred_original))
```

```
Original Accuracy: 0.9722222222222222
```

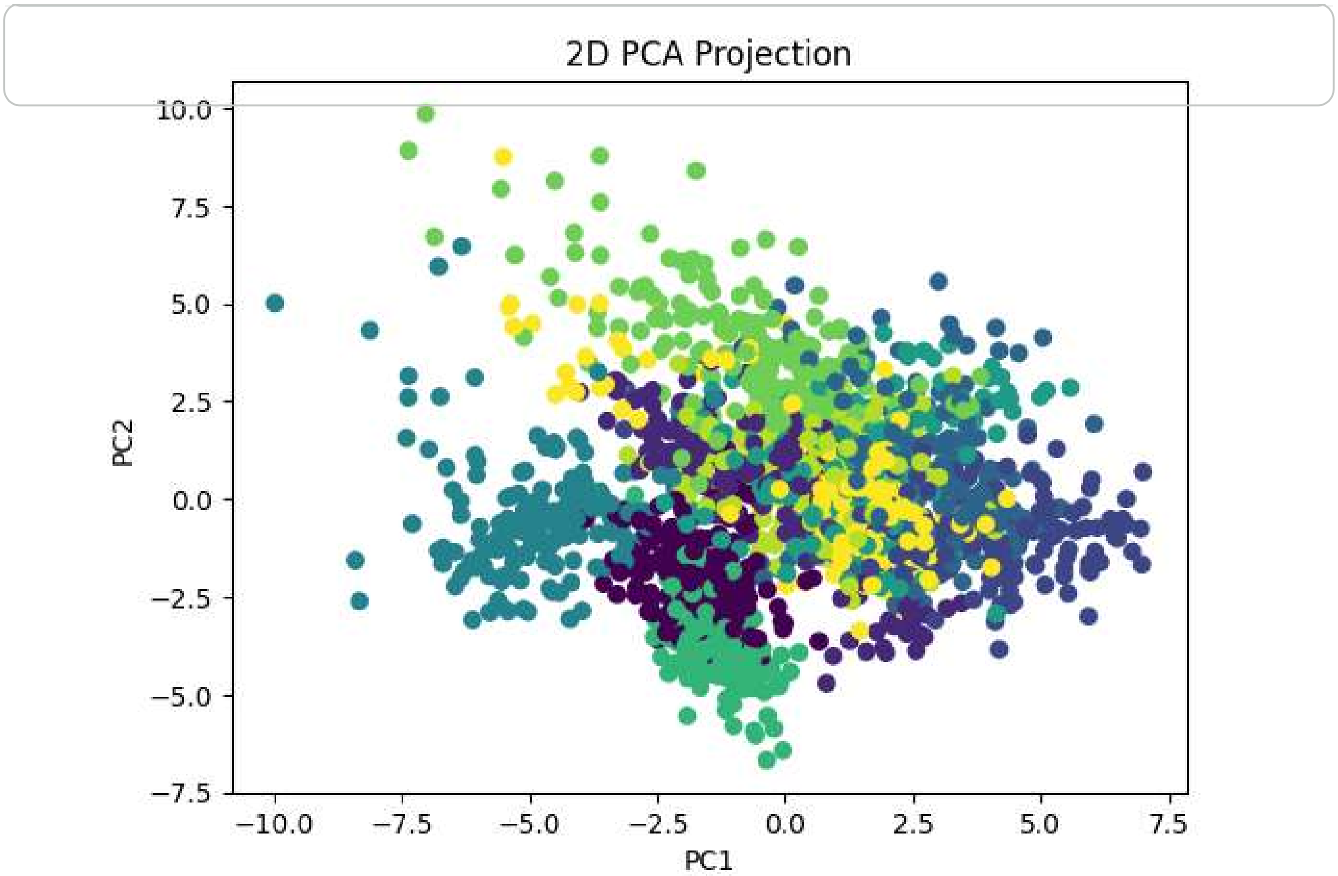
STEP 9: Train Logistic Regression (Reduced Data)

```
model_reduced = LogisticRegression(max_iter=2000)
model_reduced.fit(X_train_red, y_train)

pred_reduced = model_reduced.predict(X_test_red)
print("Reduced Accuracy:", accuracy_score(y_test, pred_reduced))
```

```
Reduced Accuracy: 0.9527777777777777
```

STEP 10: 2D PCA Visualization

```
pca_2d = PCA(n_components=2)
X_2d = pca_2d.fit_transform(X_scaled)

plt.scatter(X_2d[:,0], X_2d[:,1], c=y)
plt.xlabel("PC1")
plt.ylabel("PC2")
plt.title("2D PCA Projection")
plt.show()
```


2D PCA Projection
PC2
PC1
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5