PCA compatible with Olivetti, California, iris, cancer except Boston

```
import sklearn
from sklearn.datasets import fetch_olivetti_faces
from sklearn.datasets import fetch_california_housing
from sklearn.datasets import load_iris
from sklearn.datasets import load_breast_cancer
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt

# # Load Olivetti Face dataset
# faces = fetch_olivetti_faces()
# X = faces.data  # shape: (400, 4096)
# y = faces.target

# # california
# data = fetch_california_housing()
# X = data.data
# y = data.target

# # iris
# data = load_iris()
# X = data.data
# y = data.target

# #cancer
# data = load_breast_cancer()
# X = data.data
# y = data.target

# Apply PCA to reduce to 2 dimensions
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X)

print("Original shape:", X.shape)
print("Reduced shape:", X_pca.shape)

# Plot the 2D projection
plt.figure(figsize=(8,6))
scatter = plt.scatter(X_pca[:, 0], X_pca[:, 1], c=y, cmap='tab10', alpha=0.7)
plt.colorbar(scatter, label='Person ID')
plt.title("Olivetti Faces projected to 2D using PCA")
plt.xlabel("Principal Component 1")
plt.ylabel("Principal Component 2")
plt.show()
```

Iris, Olivetti ,breast cancer, using Knn

```
from sklearn.datasets import load_iris
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
from sklearn.datasets import fetch_olivetti_faces
from sklearn.datasets import load_breast_cancer

# Load iris dataset,oliver can also be applied, breast cancer
iris = load_iris()
X, y = iris.data, iris.target

# # for olivetti faces
# faces = fetch_olivetti_faces()
# X, y = faces.data, faces.target

# # for breast cancer
# data = load_breast_cancer()
# X, y = data.data, data.target

# Split into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Create and train KNN classifier
knn = KNeighborsClassifier(n_neighbors=3)
knn.fit(X_train, y_train)
y_pred = knn.predict(X_test)

print("Predicted labels:", y_pred, "True labels:", y_test)
# Test accuracy
accuracy = knn.score(X_test, y_test)
print("Test accuracy:", accuracy)
```

Zoo id3

```
import pandas as pd
from sklearn.tree import DecisionTreeClassifier, plot_tree
import matplotlib.pyplot as plt

# Load the zoo dataset
df = pd.read_csv('zoo.csv')
d=df.info()
head=df.head()
head
# Assume the last column is the class label
X = df.iloc[:, :-1]
y = df.iloc[:, -1]

# Convert categorical features to numeric
X_encoded = pd.get_dummies(X)

# Fit ID3 (DecisionTreeClassifier with entropy)
clf = DecisionTreeClassifier(criterion='entropy', random_state=42)
clf.fit(X_encoded, y)

# Predict on the training data (for demonstration)
y_pred = clf.predict(X_encoded)
accuracy = (y_pred == y).mean()
print("Training accuracy:", accuracy)

# Plot the tree
plt.figure(figsize=(16, 8))
plot_tree(clf, feature_names=X_encoded.columns, class_names=[str(c) for c in set(y)],
filled=True)
plt.title("ID3 Decision Tree for zoo.csv")
plt.show()
```

Breast cancer id3

```
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
# Load dataset
data = load_breast_cancer()
X, y = data.data, data.target

# Split into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

#Decision tree
dec= DecisionTreeClassifier()
dec.fit(X_train, y_train)
y_pred = dec.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(accuracy)
```