Only for Reference and basic understanding of flow

PROGRAM ON BREAST CANCER DATASET FROM SKLEARN

Consider load_breast_cancer.csv dataset from sklearn. Assign data as input features and target as target feature. Split test data set 30% of complete dataset. Build a model of support vector classifier in python using sklearn library, use linear kernel with C parameter. Predict test labels and print test accuracy.

```
from sklearn.svm import SVC

from sklearn.datasets import load_breast_cancer

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import StandardScaler

from sklearn.metrics import classification_report, accuracy_score


# Load dataset

data = load_breast_cancer()

X = data.data

y = data.target


# Split dataset into training and testing sets

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


# Scale the features

scaler = StandardScaler()

X_train_scaled = scaler.fit_transform(X_train)

X_test_scaled = scaler.transform(X_test)


# Create and train an SVM classifier

svc_model = SVC(kernel='linear', C=1.0)

svc_model.fit(X_train_scaled, y_train)


# Make predictions
```

```
y_pred = svc_model.predict(X_test_scaled)

# Evaluate the model
print("Accuracy Score:", accuracy_score(y_test, y_pred))
print("Classification Report:")
print(classification_report(y_test, y_pred))
```

Consider Social_Netwok_Ads.csv dataset - (UserID, Gender, Age, EstimatedSalary, Purchased). Use Age and EstimatedSalary as input features and Purchased as target feature. Split test data set 30% of complete dataset. Build two models of support vector classifier in python using sklearn library, one for linear and another for RBF kernel with C and gamma parameters set. Predict test labels and print test accuracy.

```
# Importing necessary libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score

# Load the dataset
df = pd.read_csv("Social_Network_Ads.csv")

# Selecting features and target
X = df[['Age', 'EstimatedSalary']]
y = df['Purchased']

# Splitting the dataset into training and testing sets (70% train, 30% test)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Feature scaling
```

```
scaler = StandardScaler()

X_train = scaler.fit_transform(X_train)

X_test = scaler.transform(X_test)


# Model 1: SVC with linear kernel

linear_svc = SVC(kernel='linear', C=1.0)

linear_svc.fit(X_train, y_train)

y_pred_linear = linear_svc.predict(X_test)

accuracy_linear = accuracy_score(y_test, y_pred_linear)

print(accuracy_linear)


# Model 2: SVC with RBF kernel

rbf_svc = SVC(kernel='rbf', C=1.0, gamma='scale')

rbf_svc.fit(X_train, y_train)

y_pred_rbf = rbf_svc.predict(X_test)

accuracy_rbf = accuracy_score(y_test, y_pred_rbf)

print(accuracy_rbf)
```

Implement an SVC model to classify iris flowers into three species (Setosa, Versicolor, and Virginica) based on their sepal and petal dimensions. The dataset contains 150 samples with four features: sepal length, sepal width, petal length, and petal width. Load the Dataset from sklearn.Split the dataset into training and testing sets.Train the SVC Model using the training data. Visualize the Results

```
# Importing necessary libraries

import numpy as np

import matplotlib.pyplot as plt

from sklearn import datasets

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import StandardScaler

from sklearn.svm import SVC

from sklearn.decomposition import PCA
from sklearn.metrics import accuracy_score, classification_report

# Load the Iris dataset
iris = datasets.load_iris()
X = iris.data  # Features: sepal length, sepal width, petal length, petal width
y = iris.target  # Target: species (Setosa, Versicolor, Virginica)

# Splitting the dataset into training and testing sets (80% train, 20% test)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Feature scaling
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Train the SVC Model with an RBF kernel
svc_model = SVC(kernel='rbf', C=1.0, gamma='scale')
svc_model.fit(X_train, y_train)

# Predict on test set and evaluate
y_pred = svc_model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(accuracy)
print(classification_report(y_test, y_pred, target_names=iris.target_names))

# Reduce data to 2D using PCA for visualization
pca = PCA(n_components=2)
X_reduced = pca.fit_transform(X)

# Train SVC model on 2D reduced data for visualization purposes
svc_model_reduced = SVC(kernel='rbf', C=1.0, gamma='scale')
```

```
svc_model_reduced.fit(pca.transform(X_train), y_train)


# Plotting decision regions
h = 0.02  # Step size in the mesh
x_min, x_max = X_reduced[:, 0].min() - 1, X_reduced[:, 0].max() + 1
y_min, y_max = X_reduced[:, 1].min() - 1, X_reduced[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))
Z = svc_model_reduced.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape)


# Plot the decision boundary
plt.figure(figsize=(10, 6))
plt.contourf(xx, yy, Z, cmap=plt.cm.coolwarm, alpha=0.3)
scatter = plt.scatter(X_reduced[:, 0], X_reduced[:, 1], c=y, cmap=plt.cm.coolwarm, edgecolors='k')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.title("SVC Decision Boundaries on PCA-Reduced Iris Dataset")
plt.legend(handles=scatter.legend_elements()[0], labels=iris.target_names)
plt.show()
```

Develop a Support Vector Classifier to predict whether a tumor is malignant or benign based on 30 features computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. The dataset contains 569 samples with binary labels indicating tumor type. 23 2 Load the Dataset from sklearn.Split the dataset into training and testing sets.Train the SVC Model using the training data. Visualize the Results Explain the role of the 'random_state' parameter in train_test_split

```
# Import necessary libraries
import numpy as np
import matplotlib.pyplot as plt
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
from sklearn.decomposition import PCA
from sklearn.metrics import accuracy_score, classification_report

# Load the Breast Cancer dataset
cancer = datasets.load_breast_cancer()
X = cancer.data  # 30 features
y = cancer.target  # Target: 1 for malignant, 0 for benign

# Split the dataset into training (80%) and testing (20%) sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Feature scaling
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Train the SVC Model with an RBF kernel
svc_model = SVC(kernel='rbf', C=1.0, gamma='scale')
svc_model.fit(X_train, y_train)

# Predict on test set and evaluate
y_pred = svc_model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(accuracy)
print(classification_report(y_test, y_pred, target_names=['Benign', 'Malignant']))

# Reduce data to 2D using PCA for visualization
pca = PCA(n_components=2)
X_reduced = pca.fit_transform(X)
```

```
# Train SVC model on 2D reduced data for visualization purposes
svc_model_reduced = SVC(kernel='rbf', C=1.0, gamma='scale')
svc_model_reduced.fit(pca.transform(X_train), y_train)

# Plotting decision regions
h = 0.02  # Step size in the mesh
x_min, x_max = X_reduced[:, 0].min() - 1, X_reduced[:, 0].max() + 1
y_min, y_max = X_reduced[:, 1].min() - 1, X_reduced[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))
Z = svc_model_reduced.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape)

# Plot the decision boundary
plt.figure(figsize=(10, 6))
plt.contourf(xx, yy, Z, cmap=plt.cm.coolwarm, alpha=0.3)
scatter = plt.scatter(X_reduced[:, 0], X_reduced[:, 1], c=y, cmap=plt.cm.coolwarm, edgecolors='k')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.title("SVC Decision Boundaries on PCA-Reduced Breast Cancer Dataset")
plt.legend(handles=scatter.legend_elements()[0], labels=cancer.target_names)
plt.show()
```

Identify and correct errors in the following code snippet (provide modified version with annotations): python

```
# Incorrect code
svc_model = SVC()
svc_model.fit(X_test, y_test)
predictions = svc_model.predict(X_train)
```

Write a Python program to implement SVM class

```
from sklearn.svm import SVC


# Corrected code
svc_model = SVC()  # Initialize the SVC model
svc_model.fit(X_train, y_train)  # Train the model using training data (X_train, y_train)


# Make predictions on the test set to evaluate the model
predictions = svc_model.predict(X_test)  # Predict on test data (X_test)
```