Lab 2

211112262

VAIBHAV PATEL

Support Vector Machine

In [ ]:
```
import pandas as pd
from sklearn.svm import SVR
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_s
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import GridSearchCV
import glob

# Function to perform SVM Regression with RBF kernel (Radial Basis Functi
def SVRRBFKernelMetrics(x_train, y_train, x_test, y_test, C, gamma):
    scaler = StandardScaler()
    x_train_scaled = scaler.fit_transform(x_train)
    x_test_scaled = scaler.transform(x_test)

    svr = SVR(kernel='rbf', C=C, gamma=gamma)
    svr.fit(x_train_scaled, y_train)

    y_pred = svr.predict(x_test_scaled)

    rmse = mean_squared_error(y_test, y_pred, squared=False)
    mae = mean_absolute_error(y_test, y_pred)
    r2 = r2_score(y_test, y_pred)
    return rmse, mae, r2

folders = ["diabetes-5-fold", "machineCPU-5-fold", "mortgage-5-fold"]

for folder in folders:
    print(folder)

    # Define the file pattern
    file_pattern = "*tra.dat"
    training_files = glob.glob("./" + folder + "/" + file_pattern)
    file_pattern = "*tst.dat"
    testing_files = glob.glob("./" + folder + "/" + file_pattern)

    alpha_values = [2 ** i for i in range(0, 2, 2)]
    degree = [2, 3]

    for d in degree:
        svm_map = {}

        for C in alpha_values:
            for gamma in alpha_values:
```

```
                trmse = 0
                tmae = 0
                tr2 = 0

                for train_file, test_file in zip(training_files, testing_
                    df = pd.read_csv(train_file, delimiter=',', header=No
                    df_test = pd.read_csv(test_file, delimiter=',', heade

                    x_train = df.iloc[:, :-1]
                    y_train = df.iloc[:, -1]
                    x_test = df_test.iloc[:, :-1]
                    y_test = df_test.iloc[:, -1]

                    # SVM applied with RBF Kernel
                    rmse, mae, r2 = SVRRBFKernelMetrics(x_train, y_train,

                    trmse += rmse
                    tmae += mae
                    tr2 += r2

                trmse /= 5
                tmae /= 5
                tr2 /= 5

                svm_map[(folder, d, C, gamma)] = (trmse, tmae, tr2)

#         Print or store the results as needed
        for key, values in svm_map.items():
            print(f"{key}: RMSE={values[0]}, MAE={values[1]}, R2={values[
```

```
diabetes-5-fold
('diabetes-5-fold', 2, 1, 1): RMSE=0.48671909101067873, MAE=0.385495558747
7355, R2=0.34341223350917116
('diabetes-5-fold', 3, 1, 1): RMSE=0.48671909101067873, MAE=0.385495558747
7355, R2=0.34341223350917116
machineCPU-5-fold
('machineCPU-5-fold', 2, 1, 1): RMSE=161.8421465679037, MAE=72.26640426703
452, R2=-0.06745001015434093
('machineCPU-5-fold', 3, 1, 1): RMSE=161.8421465679037, MAE=72.26640426703
452, R2=-0.06745001015434093
mortgage-5-fold
('mortgage-5-fold', 2, 1, 1): RMSE=0.3450650640849083, MAE=0.1493821354448
0547, R2=0.9872723388870728
('mortgage-5-fold', 3, 1, 1): RMSE=0.3450650640849083, MAE=0.1493821354448
0547, R2=0.9872723388870728
```