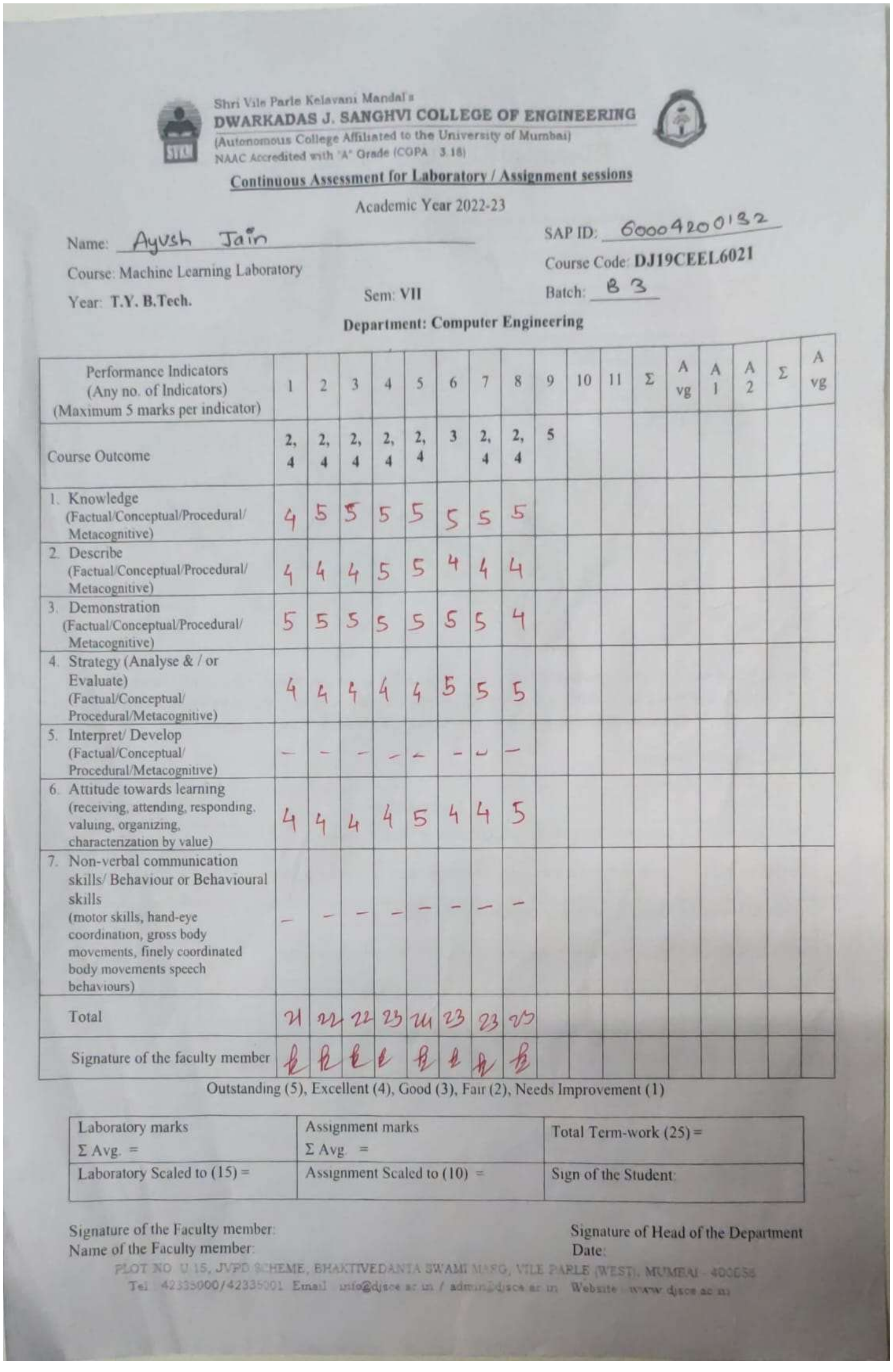


Shri Vile Parle Kelavani Mandal's

DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING

(Autonomous College Affiliated to the University of Mumbai)

NAAC Accredited with "A" Grade (CGPA : 3.18)

Continuous Assessment for Laboratory / Assignment sessions

Academic Year 2022-23

Name: Ayush Jain

SAP ID: 60004200132

Course: Machine Learning Laboratory

Course Code: **DJ19CEEL6021**

Year: **T.Y. B.Tech.**

Sem: **VII**

Batch: B3

Department: Computer Engineering

Performance Indicators (Any no. of Indicators) (Maximum 5 marks per indicator)	1	2	3	4	5	6	7	8	9	10	11	Σ	A vg	A 1	A 2	Σ	A vg
Course Outcome	2, 4	2, 4	2, 4	2, 4	2, 4	3	2, 4	2, 4	5								
1. Knowledge (Factual/Conceptual/Procedural/ Metacognitive)	4	5	5	5	5	5	5	5									
2. Describe (Factual/Conceptual/Procedural/ Metacognitive)	4	4	4	5	5	4	4	4									
3. Demonstration (Factual/Conceptual/Procedural/ Metacognitive)	5	5	5	5	5	5	5	4									
4. Strategy (Analyse & / or Evaluate) (Factual/Conceptual/ Procedural/Metacognitive)	4	4	4	4	4	5	5	5									
5. Interpret/ Develop (Factual/Conceptual/ Procedural/Metacognitive)	–	–	–	–	–	–	–	–									
6. Attitude towards learning (receiving, attending, responding, valuing, organizing, characterization by value)	4	4	4	4	5	4	4	5									
7. Non-verbal communication skills/ Behaviour or Behavioural skills (motor skills, hand-eye coordination, gross body movements, finely coordinated body movements speech behaviours)	–	–	–	–	–	–	–	–									
Total	21	22	22	23	24	23	23	23									
Signature of the faculty member	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]									

Outstanding (5), Excellent (4), Good (3), Fair (2), Needs Improvement (1)

Laboratory marks Σ Avg. =	Assignment marks Σ Avg. =	Total Term-work (25) =
Laboratory Scaled to (15) =	Assignment Scaled to (10) =	Sign of the Student:

Signature of the Faculty member:

Name of the Faculty member:

Signature of Head of the Department

Date:

PLOT NO. U-15, JVPD SCHEME, BHAKTIVEDANTA SWAMI MARG, VILE PARLE (WEST), MUMBAI - 400056

Tel.: 42335000/42335001 Email: info@djsce.ac.in / admin@djsce.ac.in Website: www.djsce.ac.in

Machine Learning

Experiment 7

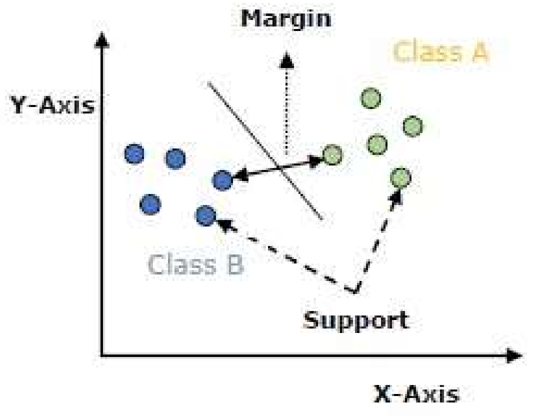
Ayush Jain 60004200132 B3

Aim : Implement SVM

Theory:

Support vector machines (SVMs) are powerful yet flexible supervised machine learning algorithms which are used both for classification and regression. But generally, they are used in classification problems. In 1960s, SVMs were first introduced but later they got refined in 1990. SVMs have their unique way of implementation as compared to other machine learning algorithms. Lately, they are extremely popular because of their ability to handle multiple continuous and categorical variables.

An SVM model is basically a representation of different classes in a hyperplane in multidimensional space. The hyperplane will be generated in an iterative manner by SVM so that the error can be minimized. The goal of SVM is to divide the datasets into classes to find a maximum marginal hyperplane (MMH).



The followings are important concepts in SVM −
Support Vectors − Datapoints that are closest to the hyperplane is called support vectors. Separating line will be defined with the help of these data points.

Hyperplane − As we can see in the above diagram, it is a decision plane or space which is divided between a set of objects having different classes.
Margin − It may be defined as the gap between two lines on the closet data points of different classes. It can be calculated as the perpendicular distance from the line to the support vectors. Large margin is considered as a good margin and small margin is considered as a bad margin.
The main goal of SVM is to divide the datasets into classes to find a maximum marginal hyperplane (MMH) and it can be done in the following two steps −
First, SVM will generate hyperplanes iteratively that segregates the classes in best way.

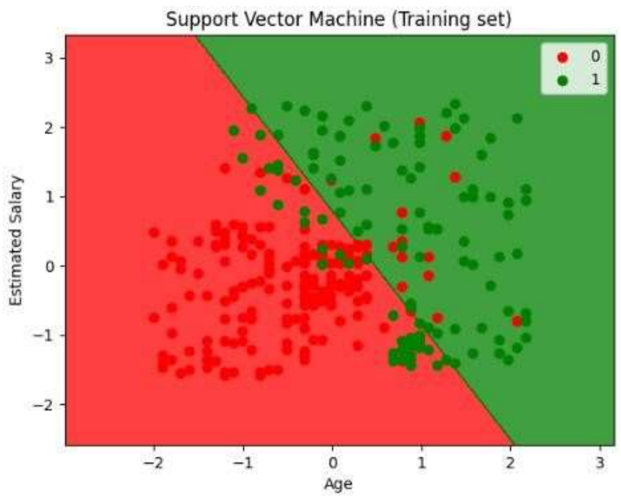
Then, it will choose the hyperplane that separates the classes correctly.

Code:

```
import pandas as pd import numpy as np import
matplotlib.pyplot as plt from sklearn.datasets import
load_digits from sklearn.model_selection import
train_test_split
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix

# Load the dataset digits
= load_digits()

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(digits.data, digits.target,
test_size=0.3, random_state=42)

# Define the list of kernel functions to be used
kernels = ['linear', 'poly', 'rbf', 'sigmoid']
best_accuracy = 0

# Loop through each kernel function and train the SVM for
kernel in kernels:
   # Create an SVM with the chosen kernel function
svm = SVC(kernel=kernel)

   # Fit the SVM to the training data
svm.fit(X_train, y_train)
```

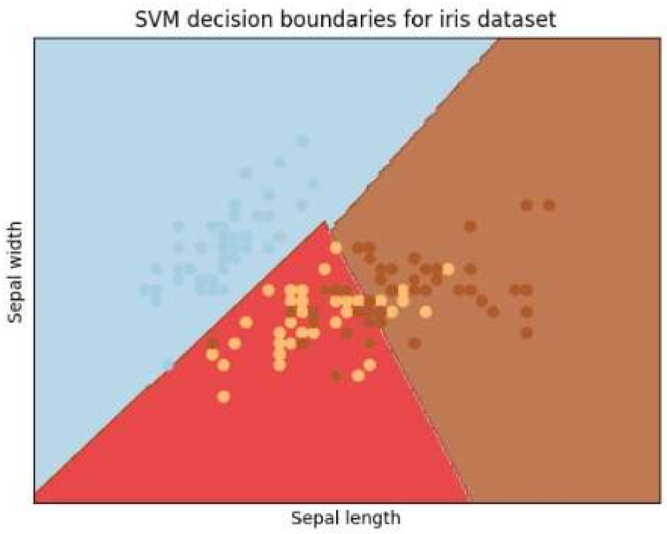
```
    # Evaluate the performance of the SVM on the testing data
y_pred = svm.predict(X_test)     cm =
confusion_matrix(y_test, y_pred)

    # Update the best-performing kernel     if
np.mean(np.diag(cm)) > best_accuracy:
best_accuracy = np.mean(np.diag(cm))
        best_kernel = kernel
        best_svm = svm

# Plot the decision boundary for the best-performing kernel
plt.figure(figsize=(8, 6))
plt.scatter(X_test[:, 0], X_test[:, 1], c=y_test, cmap=plt.cm.get_cmap('jet', 10),
edgecolors='k') plt.colorbar(ticks=range(10))
plt.title('Decision boundary for SVM with {} kernel'.format(best_kernel))
plt.show()
```

Output:



Multi-dimensional:

```
import numpy as np import
matplotlib.pyplot as plt from
sklearn import svm, datasets

data='/content/Test.csv'
X = iris.data[:, :2] y =
iris.target

C = 1.0
clf = svm.SVC(kernel='linear', C=C, decision_function_shape='ovr') clf.fit(X,
y)

x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1 xx,
yy = np.meshgrid(np.arange(x_min, x_max, 0.02),
                    np.arange(y_min, y_max, 0.02))

Z = clf.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape) plt.contourf(xx, yy, Z,
cmap=plt.cm.Paired, alpha=0.8) plt.scatter(X[:, 0], X[:,
1], c=y, cmap=plt.cm.Paired) plt.xlabel('Sepal length')
plt.ylabel('Sepal width') plt.xlim(xx.min(), xx.max())
plt.ylim(yy.min(), yy.max())
plt.xticks(()) plt.yticks(())
plt.title('SVM decision boundaries for iris dataset') plt.show()
```



Conclusion: We successfully implemented SVM.