Lab Practice III

MACHINE LEARNING MINI PROJECT

"Classification Using SVM"

BACHELOR OF ENGINEERING
Computer Engineering

SUBMITTED BY

Shreya Dharmadhikari	41016
Vaishnavi Jadhav	41026
Chaitanya Kulkarni	41043
Manas Sonawane	41072

Progressive Education Society's
Modern College of Engineering
Department of
Computer Engineering Shivajinagar,
Pune- 411005

2021-2022

CERTIFICATE

This is to certify that students from Fourth Year Computer Engineering have successfully completed their mini project work of **Lab Practice III** at P.E.S. Modern College of Engineering in the partial fulfillment of the Bachelor of Engineering Degree in Computer under Savitribai Phule Pune University.

Group Members:

Shreya Dharmadhikari	41016
Vaishnavi Jadhav	41026
Chaitanya Kulkarni	41043
Manas Sonawane	41072

(Prof. Dr. Mrs. S. A. Itkar)
Head Department of Computer Engineering

(Prof. Deipali V. Gore)
Internal Guide

External Examiner

Date:
Place: Pune

Title:

Classification Using SVM

Problem Statement:

Apply the Support vector machine for classification on a dataset obtained from UCI ML repository.

Outcome:

Classification on dataset (Breast Cancer)

Software Requirement:

Operating System	Windows 10
Execution Environment	Google Colab
Programming Language	Python
Version	3.9.5
Dataset	Breast Cancer

Hardware requirements:

Processor	I3
Speed	2.1GHz
RAM	4 GB
Hard Disk	200GB

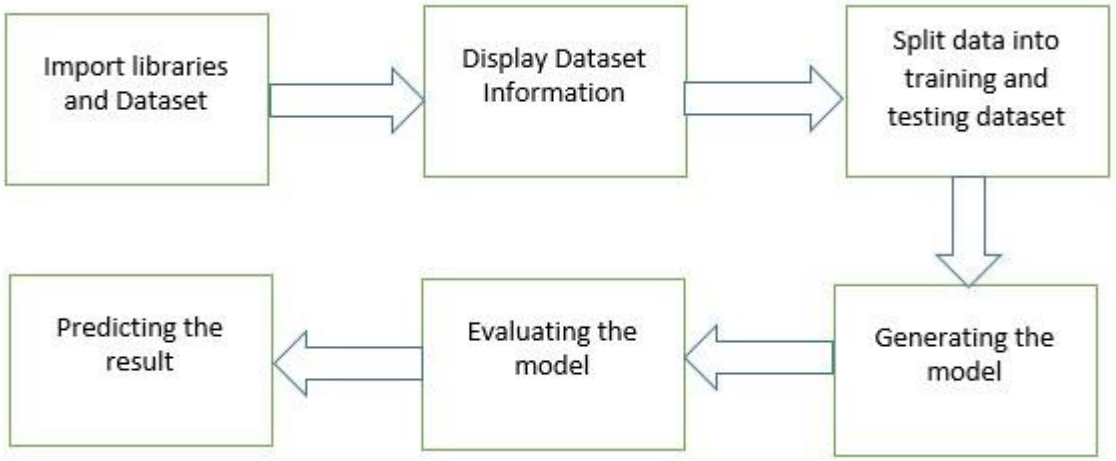
Data:

Scikit-Learn provides seven datasets, which they call toy datasets. Don't be fooled by the word "toy". These datasets are powerful and serve as a strong starting point for learning ML.

Breast Cancer Wisconsin (diagnostic) dataset — use ML to diagnose cancer scans as benign (does not spread to the rest of the body) or malignant (spreads to rest of the body)

Here, we'll be working with the "Breast Cancer Wisconsin" dataset. We will import the data and understand how to read it. As a bonus, we'll build a simple ML model that is able to classify cancer scans either as malignant or benign.

Flow Diagram:



Description:

SVM:

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.

The goal of the SVM algorithm is to create the best line or decision boundary that can segregate ndimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane:

Advantages of support vector machine:

Support vector machine works comparably well when there is an understandable margin of dissociation between classes.

It is more productive in high dimensional spaces.

It is effective in instances where the number of dimensions is larger than the number of specimens.

Support vector machine is comparably memory systematic.

Disadvantages of support vector machine:

Support vector machine algorithm is not acceptable for large data sets.

It does not execute very well when the data set has more sound i.e. target classes are overlapping.

In cases where the number of properties for each data point outstrips the number of training data specimens, the support vector machine will underperform.

As the support vector classifier works by placing data points, above and below the classifying hyperplane there is no probabilistic clarification for the classification.

Applications of support vector machine:

Face observation –

It is used for detecting the face according to the classifier and model. Text

and hypertext arrangement –

In this, the categorization technique is used to find important information or you can say required information for arranging text.

Grouping of portrayals –

It is also used in the Grouping of portrayals for grouping or you can say by comparing the piece of information and take an action accordingly.

Bioinformatics –

In is also used for medical science as well like in laboratory, DNA, research, etc.

Handwriting remembrance – In this, it is used for handwriting recognition.

Algorithm

Step 1: Start

Step 2: Load Dataset

Step 3: Displaying Features.

Step 4: Displaying Size of Dataset.

Step 5: Checking Top 5 Records.

Step 6: Checking Target Set.

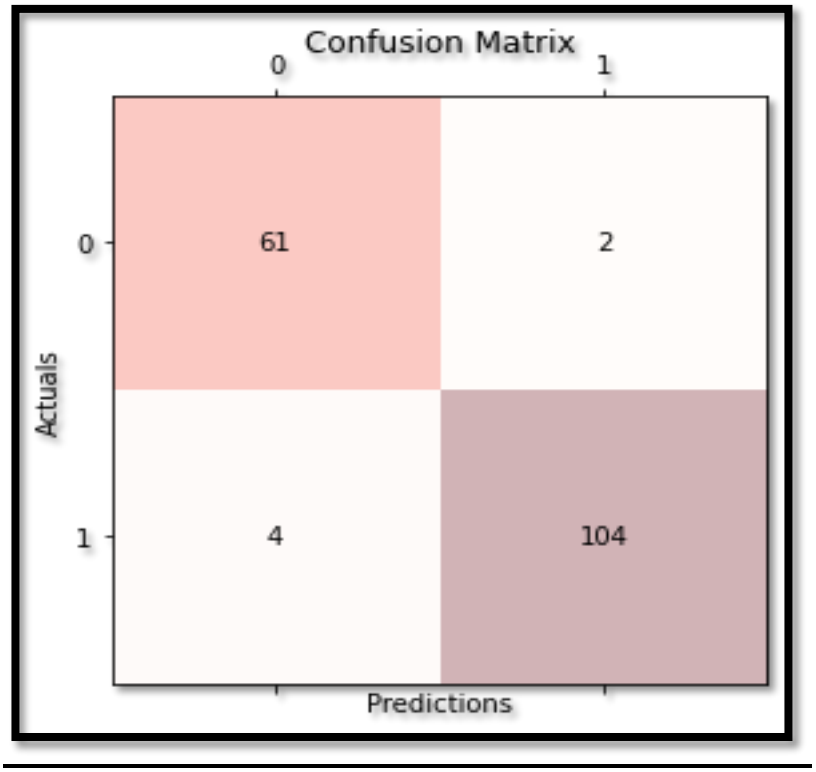
Step 7: Splitting Of Data.

Step 8: Generating Model [Using SVM].

Step 9: Evaluation of Model.

Step 10: STOP

Result:



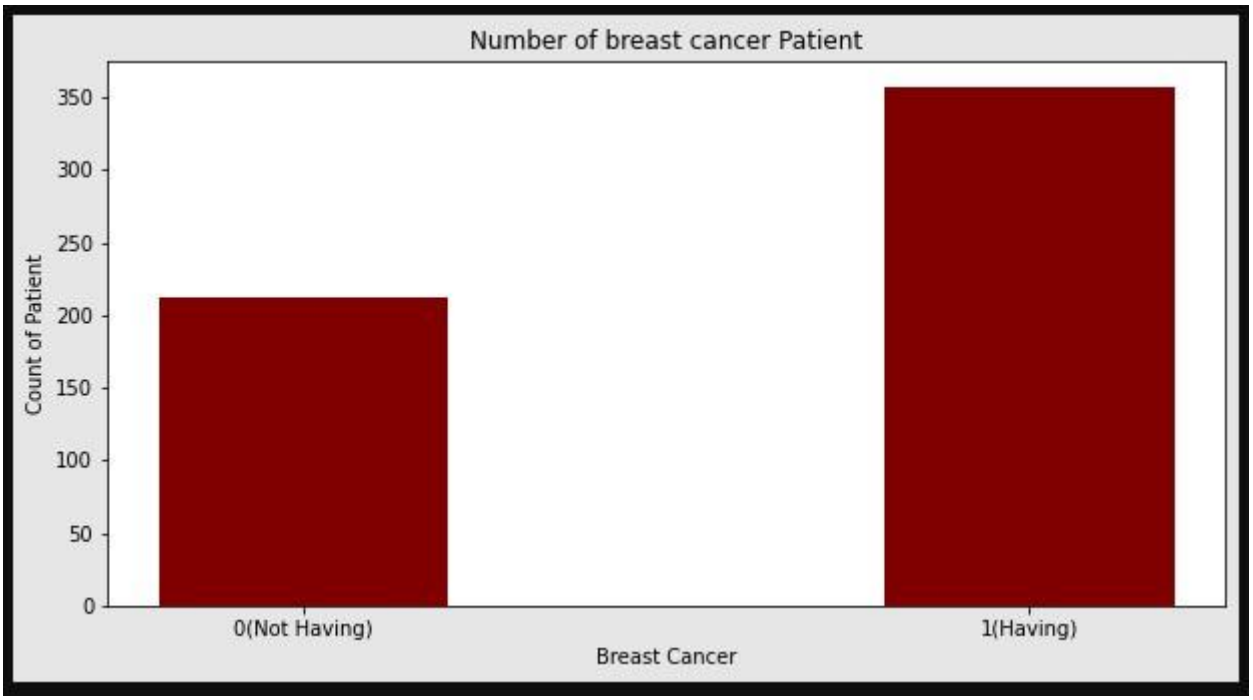

```
Accuracy: 0.9649122807017544
Precision: 0.9811320754716981
Recall: 0.9629629629629629
```

Conclusion:

In this way, we have Implemented Classification on Breast Cancer Dataset Using SVM.

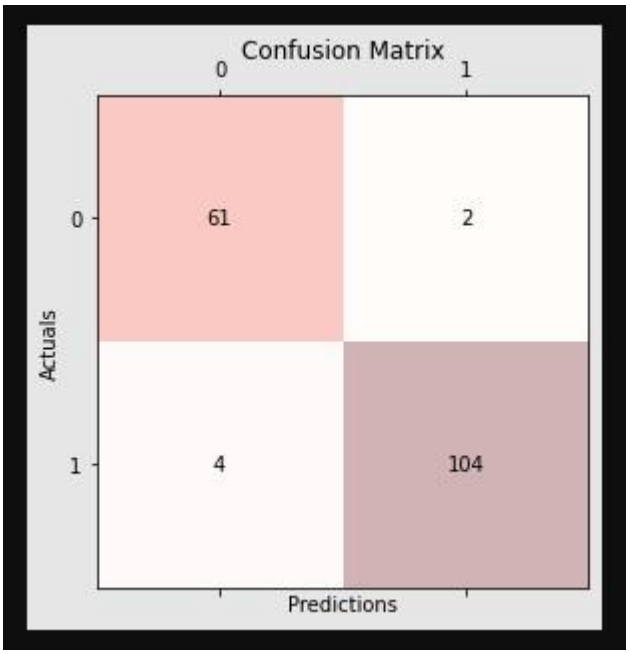
Source Code:

```
#Import scikit-learn dataset library from
sklearn import datasets



#Load dataset
cancer = datasets.load_breast_cancer() #
print the names of the features
print("Features: ", cancer.feature_names)

# print the label type of cancer('malignant' 'benign') print("Labels:
", cancer.target_names)


# print data(feature)shape cancer.data.shape


# print the cancer data features (top 5 records)
print(cancer.data[0:5])


# print the cancer labels (0:malignant, 1:benign) print(cancer.target)


import numpy as np import matplotlib.pyplot as plt
label=['0(Not Having)','1(Having)'] count=[(cancer.target ==
0).sum(),(cancer.target == 1).sum()] fig = plt.figure(figsize
= (10, 5))

# creating the bar plot
plt.bar(label, count, color ='maroon',
width = 0.4)

plt.xlabel("Breast Cancer") plt.ylabel("Count
of Patient") plt.title("Number of breast
cancer Patient") plt.show()
```




```
##Splitting Data

# Import train_test_split function
from sklearn.model_selection import train_test_split

# Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(cancer.data, cancer
.target, test_size=0.3,random_state=109) # 70% training and 30% test


##Generating Model

#Import svm model from
sklearn import svm

#Create a svm Classifier
clf = svm.SVC(kernel='linear') # Linear Kernel

#Train the model using the training sets clf.fit(X_train,
y_train)

#Predict the response for test dataset y_pred
= clf.predict(X_test) ##Evaluating Model

import numpy as np import matplotlib.pyplot
as plt from sklearn.metrics import
confusion_matrix
## plot confusion matrix on test
classes = ['0','1'] tick_marks =
np.arange(len(classes))
```

```
conf_matrix = confusion_matrix(y_test, y_pred)

fig, ax = plt.subplots(figsize=(4, 4))
ax.matshow(conf_matrix, cmap=plt.cm.Reds, alpha=0.3)
for i in range(conf_matrix.shape[0]):     for j in
range(conf_matrix.shape[1]):
        ax.text(x=j, y=i,s=conf_matrix[i, j], va='center', ha='center')

plt.tight_layout() plt.xticks(tick_marks ,
classes, rotation=0) plt.yticks(tick_marks ,
classes) plt.xlabel('Predictions')
plt.ylabel('Actuals') plt.title('Confusion
Matrix', fontsize=12) plt.show()
```




```
#Import scikit-learn metrics module for accuracy calculation from
sklearn import metrics

# Model Accuracy: how often is the classifier correct?/'
print("Accuracy:",metrics.accuracy_score(y_test, y_pred)) # Model
Precision: what percentage of positive tuples are labeled as su ch?
print("Precision:",metrics.precision_score(y_test, y_pred))

# Model Recall: what percentage of positive tuples are labelled as such
? print("Recall:",metrics.recall_score(y_test,
y_pred))
```