

## VIVEKANANDA INSTITUTE OF PROFESSIONAL STUDIES - TECHNICAL CAMPUS

### Grade A++ Accredited Institution by NAAC

NBA Accredited for MCA Programme; Recognized under Section 2(f) by UGC; Affiliated to GGSIP University, Delhi; Recognized by Bar Council of India and AICTE An ISO 9001:2015 Certified Institution

**SCHOOL OF ENGINEERING & TECHNOLOGY**

# B.Tech Programme: CSE

Course Title: Supervised Deep Learning

Course Code: ML 463P

# Submitted To: Submitted By:

**Ms. Priyanka**

**Name: Ishaann Jain**

**Enrollment No: 06117702722**

**Branch & Section: CSE B**

**Assistant Professor**



## VISION OF INSTITUTE

To be an educational institute that empowers the field of engineering to build a sustainable future by providing quality education with innovative practices that supports people, planet and profit.

## MISSION OF INSTITUTE

To groom the future engineers by providing value-based education and awakening students' curiosity, nurturing creativity and building

capabilities to enable them to make significant contributions to the world.

**EXPERIMENT 1**

**AIM:** Linear regression: Implement linear regression on a dataset and evaluate the model's performance.

**THEORY:**

Linear Regression is a fundamental statistical and machine learning algorithm used to model the relationship between a dependent variable (Y) and one or more independent variables (X). When there is only one independent variable, it is called Simple Linear Regression. The goal of Simple Linear Regression is to find the best-fitting straight line, also known as the regression line, that minimizes the difference between the actual and predicted values.

The equation of a simple linear regression model is:

𝑌 = w𝑋+ b

Where:

Y = Dependent variable (target/output)

X = Independent variable (input)

w = weight

b = bias (constant term)

The values of w and b are determined using the Least Squares Method, which minimizes the sum of squared differences between actual and predicted values.

**Algorithm:**

The working of Simple Linear Regression involves the following steps:

1. **Data Collection –** Gather historical data containing input-output pairs.
2. **Data Preprocessing –** Handle missing values, remove outliers, and normalize data if necessary.
3. **Splitting Data –** Divide the dataset into training and testing sets to evaluate model performance.
4. **Model Training –** Fit the Simple Linear Regression model to the training data using the Least Squares Method.
5. **Prediction –** Use the trained model to predict new values.
6. **Performance Evaluation –** Measure model accuracy using metrics like Mean Squared Error (MSE) and R-squared (R2) score.

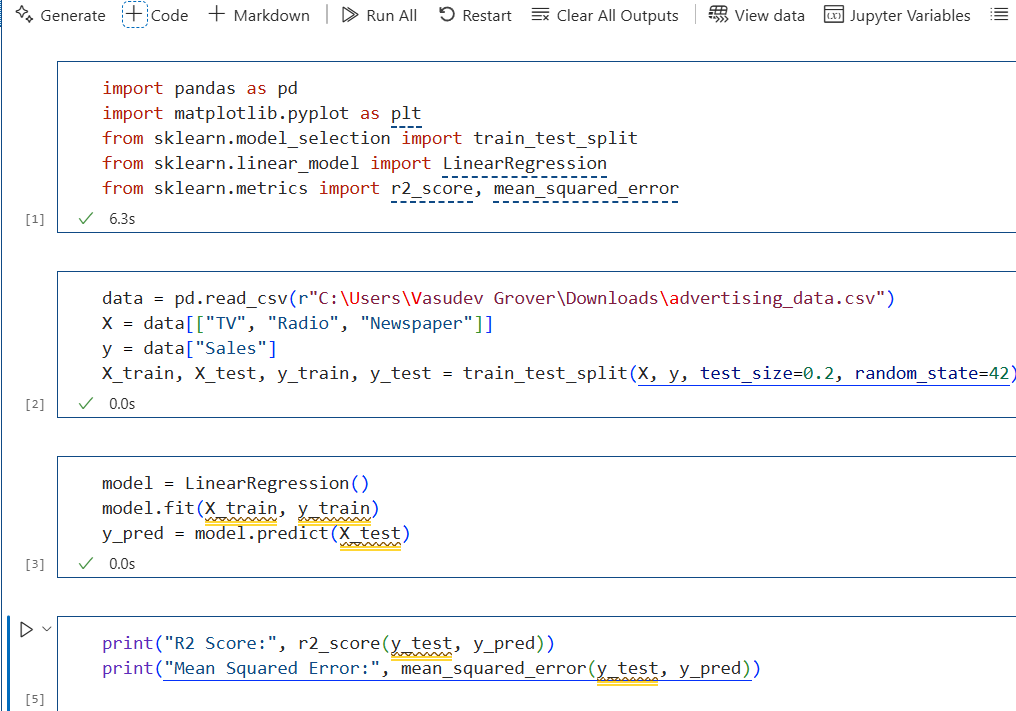
**ABOUT THE DATASET:**

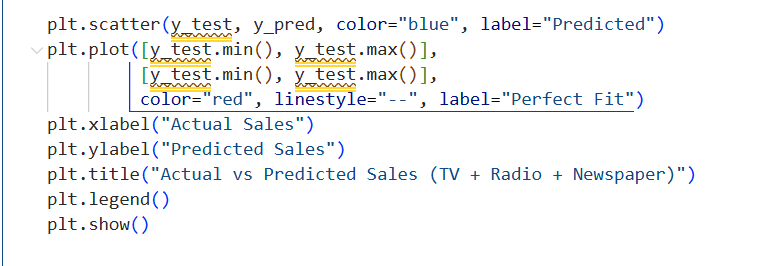
The dataset used in this experiment is the **Advertising Dataset**, which contains information about advertising expenditures across different media and the corresponding sales. It consists of **50 rows** (observations) and **4 columns** (features).

**Attributes**

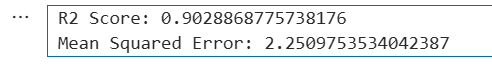
1. **TV** – Advertising budget spent on TV (in thousands of dollars).
2. **Radio** – Advertising budget spent on Radio (in thousands of dollars).
3. **Newspaper** – Advertising budget spent on Newspaper (in thousands of dollars).
4. **Sales** – The dependent variable, representing the sales generated (in thousands of units).

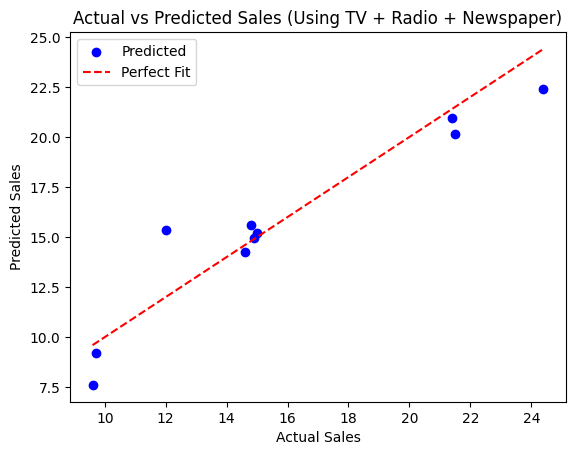
**SOURCE CODE:**





**OUTPUT:**

****

****

**LEARNING OUTCOME:**

**EXPERIMENT 2**

**AIM:** Logistic Regression: Implement logistic regression on a binary classification dataset and evaluate the model’s performance.

**THEORY:**

Logistic Regression is a classification algorithm used in Machine Learning to predict categorical outcomes. It is mainly used for **binary classification**, where the target variable has two possible values (e.g., Yes/No, 0/1, True/False). Instead of fitting a straight line as in **Linear Regression**, it applies the **sigmoid function** to map predicted values between **0 and 1**.

**Mathematical Formulation:**

The logistic function (sigmoid function) is given by:

where z is the linear combination of input features:

z = w0 + w1x1 + w2x2 + ... + wnxn

The output represents the probability of belonging to a particular class. A threshold (usually 0.5) determines the final classification.

**Algorithm:**

* **Data Collection** – Load and analyze the dataset.
* **Preprocessing** – Handle missing values, normalize features, and split the dataset.
* **Model Training** – Apply logistic regression using Gradient Descent to optimize weights.
* **Prediction** – Classify new observations based on computed probabilities.
* **Evaluation** – Assess model performance using metrics like Accuracy, Precision, Recall, and F1-score.

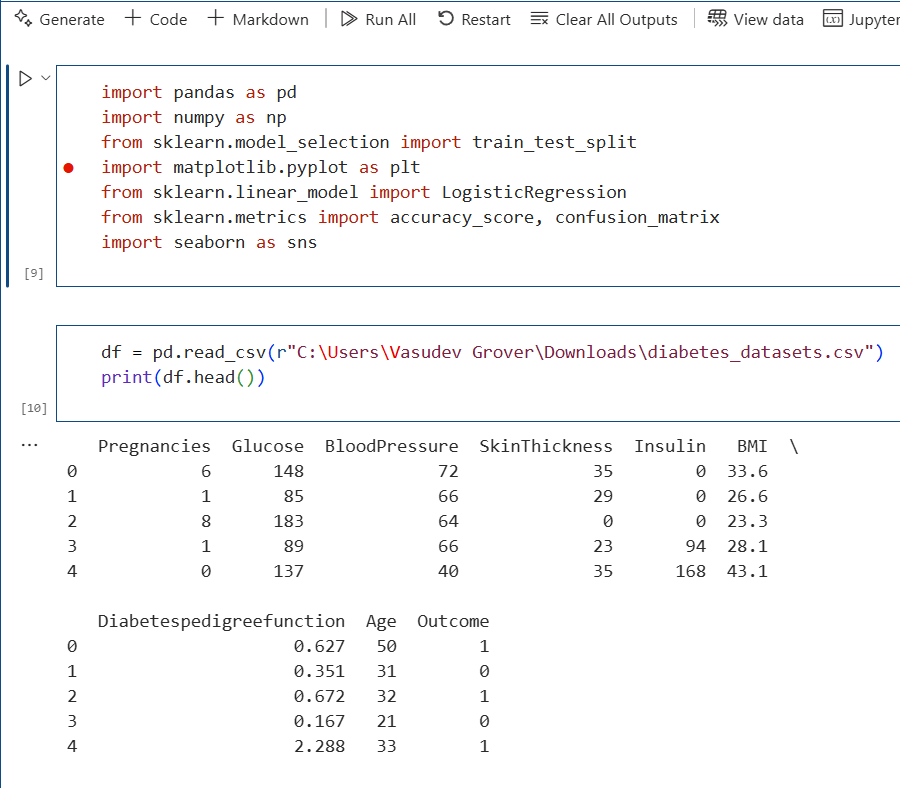
**ABOUT THE DATASET:**

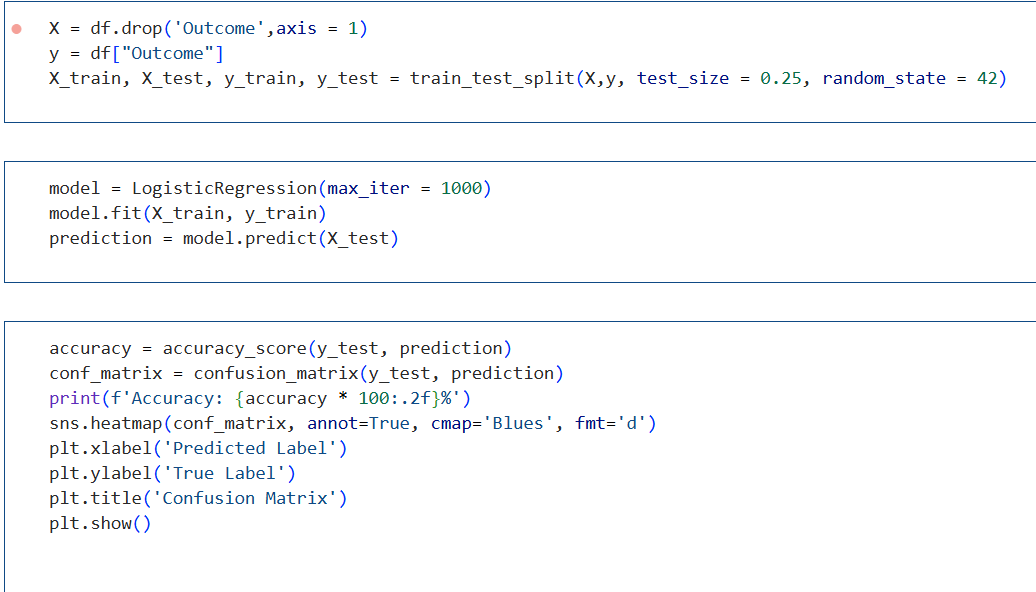
The dataset used in this experiment is the **PIMA Indians Diabetes Dataset**, a well-known medical dataset for diabetes prediction. It contains **768 rows** (patient records) and **9 columns** (attributes). The dataset is often used for binary classification, where the goal is to predict whether a patient has diabetes (**Outcome = 1**) or not (**Outcome = 0**).

**Attributes:**

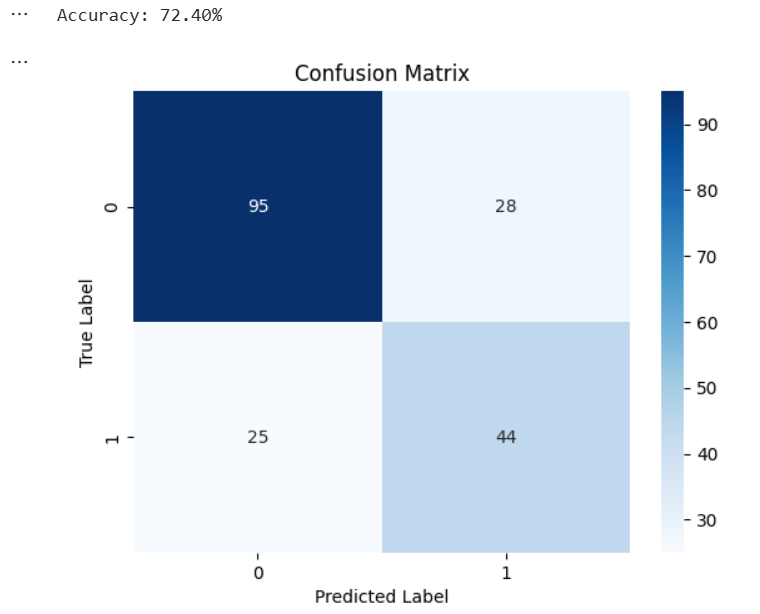
1. **Pregnancies** – Number of times the patient has been pregnant.
2. **Glucose** – Plasma glucose concentration (2 hours after an oral glucose tolerance test).
3. **BloodPressure** – Diastolic blood pressure (mm Hg).
4. **SkinThickness** – Triceps skinfold thickness (mm).
5. **Insulin** – 2-hour serum insulin level (mu U/ml).
6. **BMI** – Body Mass Index, calculated as weight (kg) / height (m²).
7. **DiabetesPedigreeFunction** – A score representing the genetic relationship of diabetes in the family.
8. **Age** – Age of the patient (in years).
9. **Outcome** – Target variable:
   * **0** → Patient does not have diabetes
   * **1** → Patient has diabetes

**SOURCE CODE:**

****

****

**OUTPUT:**

****

**LEARNING OUTCOME:**

**EXPERIMENT 3**

**AIM:** k-Nearest Neighbors (k-NN): Implement k-NN algorithm on a dataset and evaluate the model’s performance.

**THEORY:**

The k-Nearest Neighbours (k-NN) algorithm is a supervised learning technique used for both classification and regression problems. It is one of the simplest machine learning algorithms, based on the principle of similarity. The fundamental idea behind k-NN is that data points with similar features tend to belong to the same category.  
k-NN is a lazy learning algorithm, meaning it does not create an explicit model during the training phase. Instead, it stores the training data and classifies new instances by finding the k closest training examples.

**Key Characteristics of k-NN***:*

* **Instance-based Learning:** The algorithm memorizes training data rather than learning a general model.
* **Non-parametric:**No assumption is made about the underlying data distribution.
* **Versatile:** Can be used for both classification and regression tasks.
* **Computational Complexity:**Increases with dataset size, as it requires searching for nearest neighbors.

**Algorithm:**

1. Choose the value of k (the number of nearest neighbours to consider).  
2. Calculate the distance between the test sample and all training samples using a distance metric such as:

* Euclidean Distance (most common):
* Manhattan Distance
* Minkowski Distance

3. Find the k nearest neighbours (smallest distances).  
4. Assign the class label based on majority voting from the k-nearest neighbours.  
5. Return the predicted class.

**ABOUT THE DATASET:**

The Iris dataset is a well-known dataset used for classification tasks. It contains 150 samples of iris flowers from three species:

• Setosa

• Versicolor

• Virginica

Each sample has four features:

1. Sepal Length (cm)

2. Sepal Width (cm)

3. Petal Length (cm)

4. Petal Width (cm)

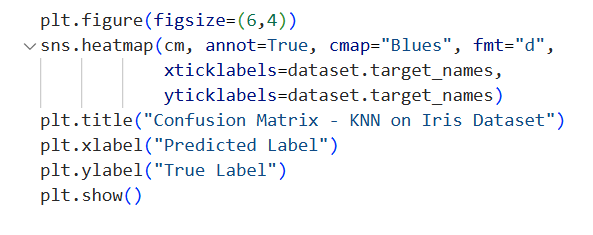
**Target Variable:**

• Species (Categorical: Setosa, Versicolor, Virginica)

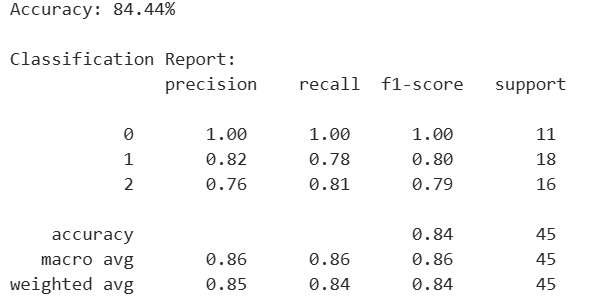
**Dataset Source:**  
The dataset is originally from Fisher’s Iris dataset and is widely used for pattern recognition in Machine Learning.

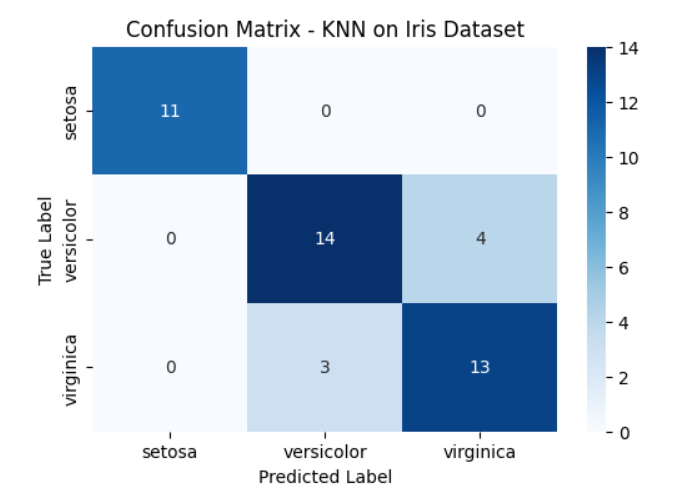
**SOURCE CODE:**

****

****

**OUTPUT:**

****

****

**LEARNING OUTCOME:**

**EXPERIMENT 4**

**AIM:** Decision Trees: Implement decision trees on a dataset and evaluate the model’s performance.

**THEORY:**

A Decision Tree is a supervised machine learning algorithm used for classification and regression tasks. It is a tree-like structure where:

* Each internal node represents a decision based on an attribute.
* Each branch represents the outcome of the decision.
* Each leaf node represents a class label (in classification problems).

The Decision Tree is constructed using recursive partitioning, where the dataset is split into subsets based on the best attribute at each level.

**Advantages of Decision Trees:**

1. Easy to interpret and visualize.
2. Works well with both numerical and categorical data.
3. Requires minimal data preprocessing (no need for feature scaling).
4. Can handle multi-class classification problems effectively.

**Disadvantages of Decision Trees:**

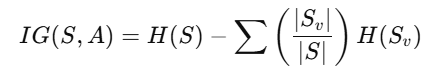
1. Prone to overfitting, especially with deep trees.
2. Small changes in data can lead to different tree structures.
3. Greedy approach may not always lead to the optimal tree.

**Algorithm:**

1. **Compute Entropy**: The entropy of the dataset is calculated to measure

uncertainty.

**2) Calculate Information Gain:**  
Information Gain is used to decide the best attribute for splitting.



**3) Split on the Best Attribute:**  
The attribute with the highest Information Gain is chosen for the next split.

**4) Repeat Until Stopping Condition:**

* If all samples belong to the same class → Stop
* If no more attributes left → Stop

If dataset is empty → Stop

**ABOUT THE DATASET:**

The Iris dataset is a well-known dataset used for classification tasks. It contains 150 samples of iris flowers from three species:

• Setosa

• Versicolor

• Virginica

Each sample has four features:

1. Sepal Length (cm)

2. Sepal Width (cm)

3. Petal Length (cm)

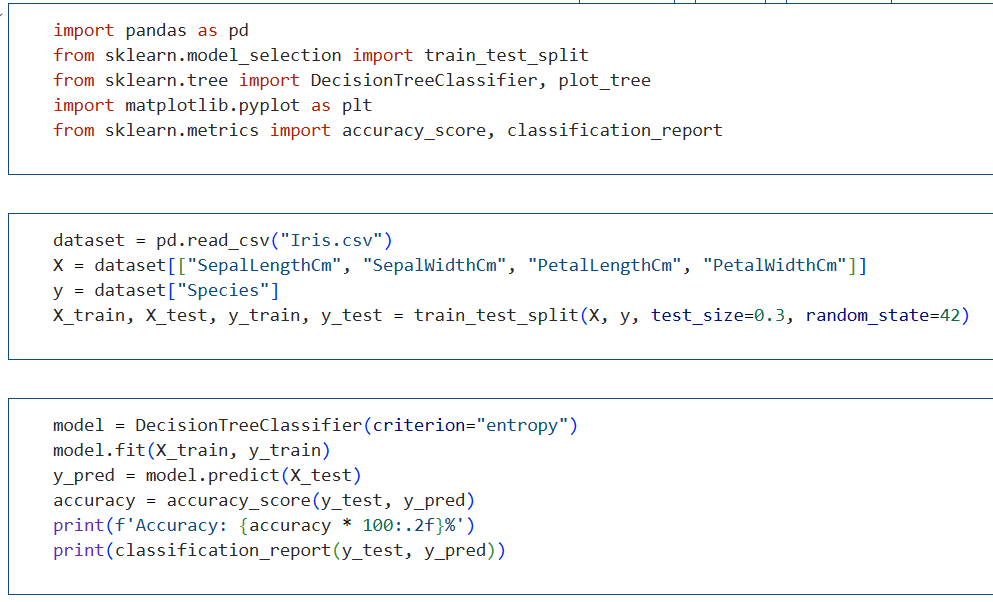
4. Petal Width (cm)

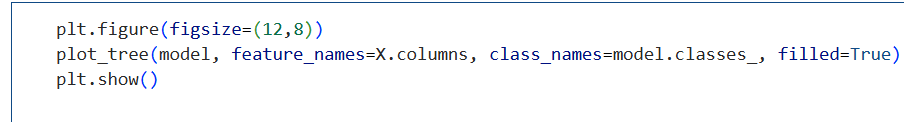
**Target Variable:**

• Species (Categorical: Setosa, Versicolor, Virginica)

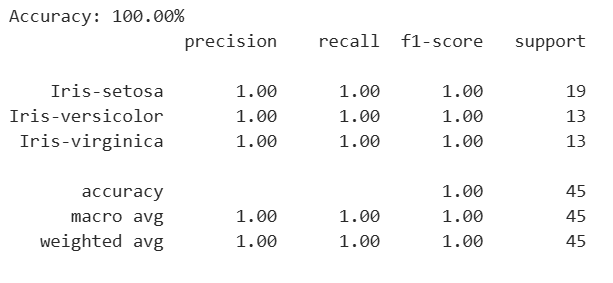
**Dataset Source:**  
The dataset is originally from Fisher’s Iris dataset and is widely used for pattern recognition in Machine Learning.

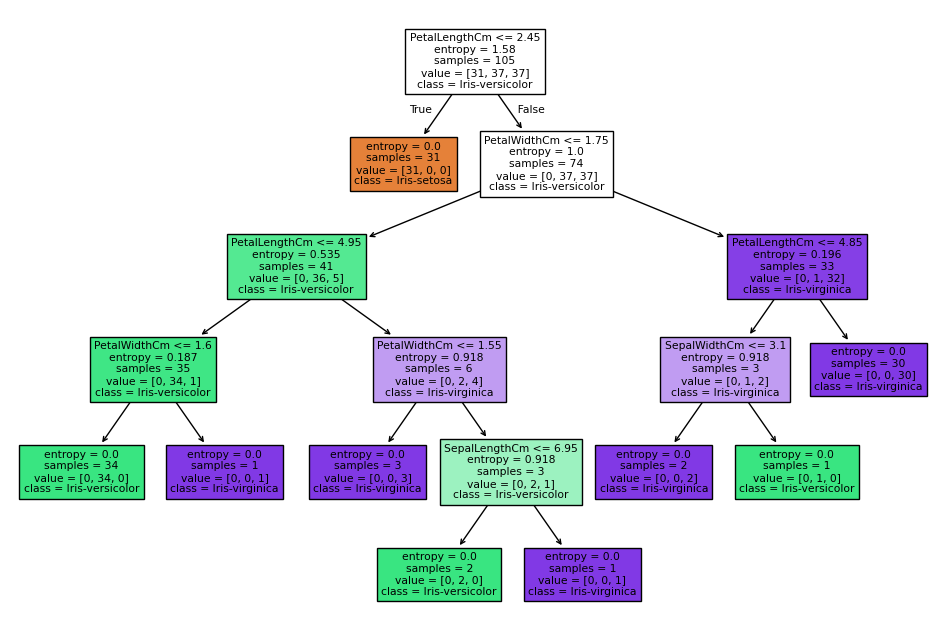
**SOURCE CODE:**

****

****

**OUTPUT:**

****

****

**LEARNING OUTCOME:**

**EXPERIMENT 5 (a)**

**AIM:** Random Forest: Implement random forest algorithm on a dataset and evaluate the model's performance.

**THEORY:**

A Random Forest is a supervised machine learning algorithm used for classification and regression tasks. It is an ensemble learning method that constructs multiple decision trees and combines their outputs to improve predictive performance and reduce overfitting.

**Advantages of Random Forest:**

* High accuracy due to ensemble of multiple trees.
* Reduces overfitting compared to a single decision tree.
* Can handle large datasets with higher dimensionality.
* Provides feature importance for better interpretability.

**Disadvantages of Random Forest:**

* Computationally intensive for large datasets.
* Less interpretable than a single decision tree.
* Training time increases with the number of trees.

**Algorithm:**

1. Create Bootstrap Samples: Random subsets of the dataset are created by sampling with replacement. Each subset is used to train a separate decision tree.
2. Select Random Features at Each Split: At each node of the decision tree, a random subset of features is considered for splitting. This ensures diversity among the trees.
3. Build Decision Trees: Each decision tree is grown fully without pruning using the selected features from step 2.
4. Aggregate Predictions: For classification, the final output is determined by majority voting across all trees. For regression, the final output is the average prediction of all trees.
5. Repeat Until Stopping Condition:

* If the maximum number of trees is reached → Stop.
* If no further improvement in performance → Stop.

**ABOUT THE DATASET:**

The Iris dataset is a well-known dataset used for classification tasks. It contains 150 samples of iris flowers from three species:

• Setosa

• Versicolor

• Virginica

Each sample has four features:

1. Sepal Length (cm)

2. Sepal Width (cm)

3. Petal Length (cm)

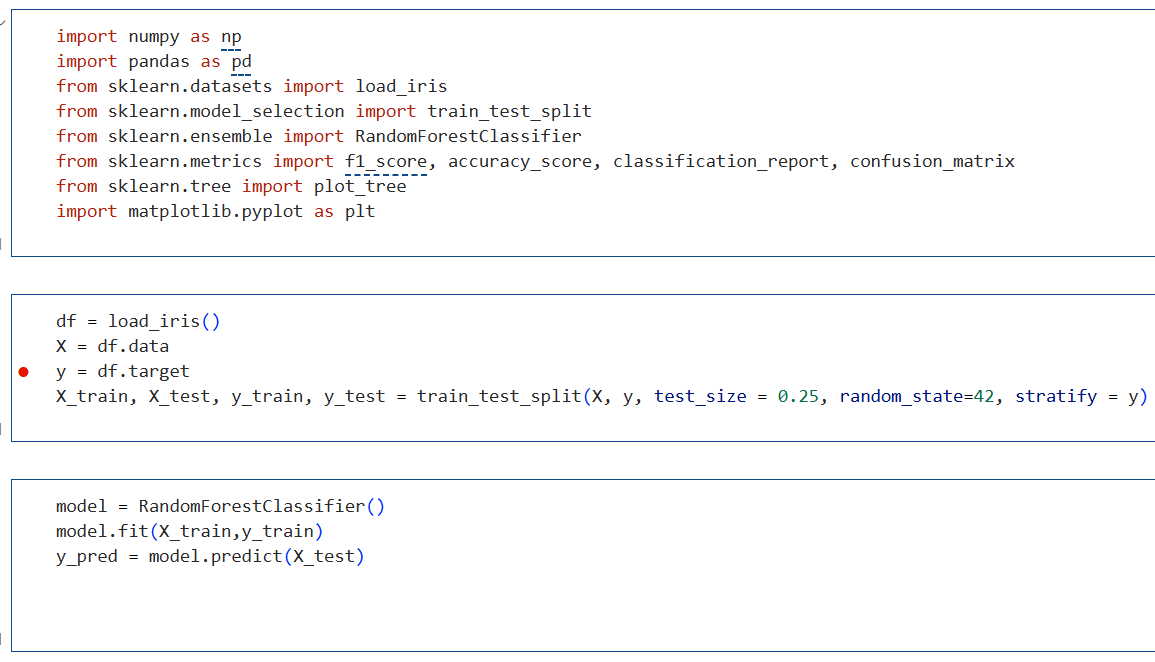
4. Petal Width (cm)

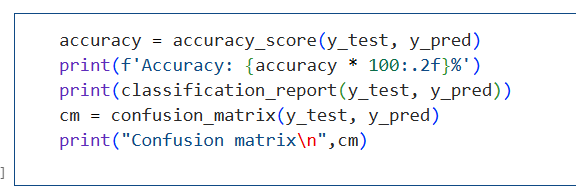
**Target Variable:**

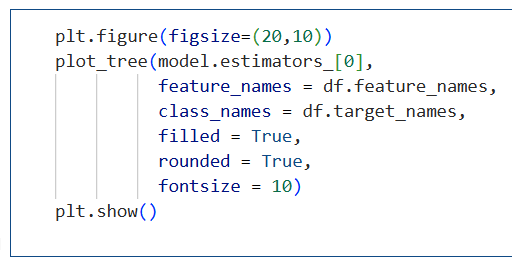
• Species (Categorical: Setosa, Versicolor, Virginica)

**Dataset Source:**  
The dataset is originally from Fisher’s Iris dataset and is widely used for pattern recognition in Machine Learning.

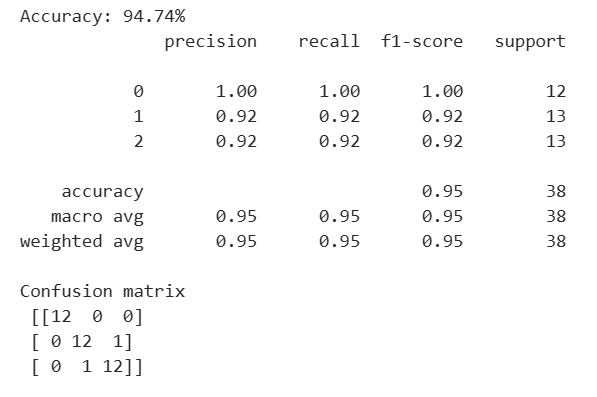
**SOURCE CODE:**

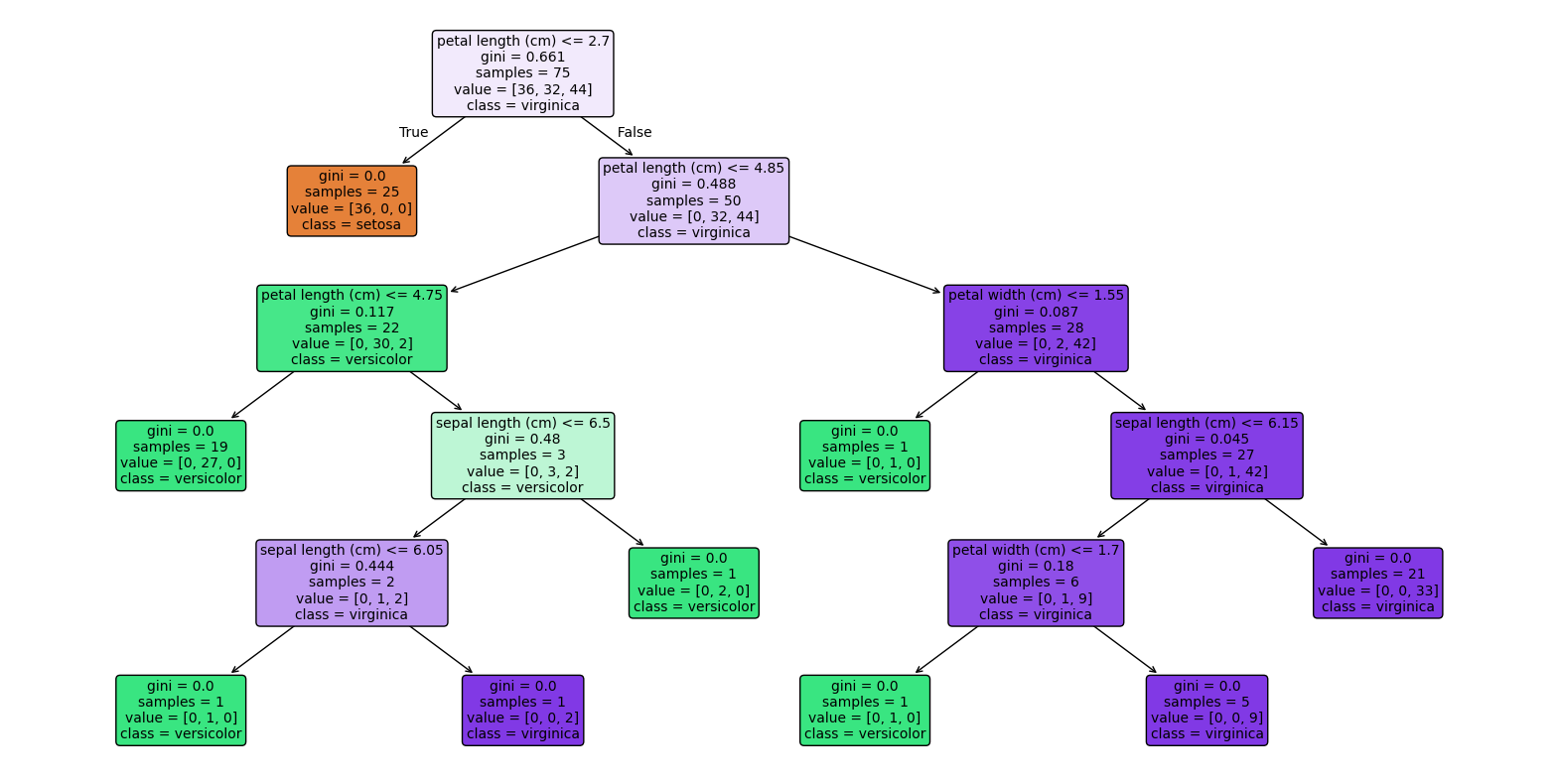
****

****

****

**OUTPUT:**

****

****

**LEARNING OUTCOME:**

**EXPERIMENT 5 (b)**

**AIM:** Random Forest: Implement random forest algorithm on a dataset and evaluate the model's performance.

**THEORY:**

A Random Forest is a supervised machine learning algorithm used for classification and regression tasks. It is an ensemble learning method that constructs multiple decision trees and combines their outputs to improve predictive performance and reduce overfitting.

**Advantages of Random Forest:**

* High accuracy due to ensemble of multiple trees.
* Reduces overfitting compared to a single decision tree.
* Can handle large datasets with higher dimensionality.
* Provides feature importance for better interpretability.

**Disadvantages of Random Forest:**

* Computationally intensive for large datasets.
* Less interpretable than a single decision tree.
* Training time increases with the number of trees.

**Algorithm:**

1. Create Bootstrap Samples: Random subsets of the dataset are created by sampling with replacement. Each subset is used to train a separate decision tree.
2. Select Random Features at Each Split: At each node of the decision tree, a random subset of features is considered for splitting. This ensures diversity among the trees.
3. Build Decision Trees: Each decision tree is grown fully without pruning using the selected features from step 2.
4. Aggregate Predictions: For classification, the final output is determined by majority voting across all trees. For regression, the final output is the average prediction of all trees.
5. Repeat Until Stopping Condition:

* If the maximum number of trees is reached → Stop.
* If no further improvement in performance → Stop.

**ABOUT THE DATASET:**

**Source:** sklearn.datasets.fetch\_california\_housing (based on 1990 California census)

**Purpose:** Predict median house value in California districts using demographic and geographical features.

**Number of Samples:** 20,640

**Number of Features:** 8 (all numerical)

**Features:**

* MedInc – Median income (in $10k)
* HouseAge – Median house age (years)
* AveRooms – Average rooms per household
* AveBedrms – Average bedrooms per household
* Population – Population of the block
* AveOccup – Average household size
* Latitude – Latitude of the block
* Longitude – Longitude of the block

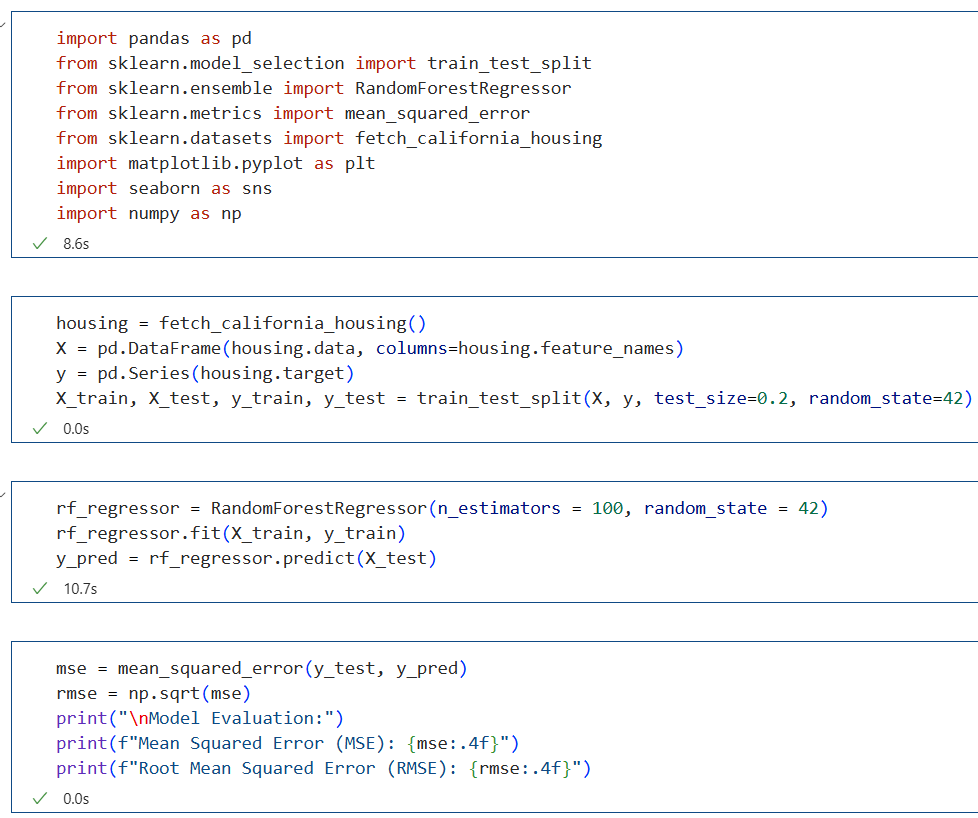
**Target:**

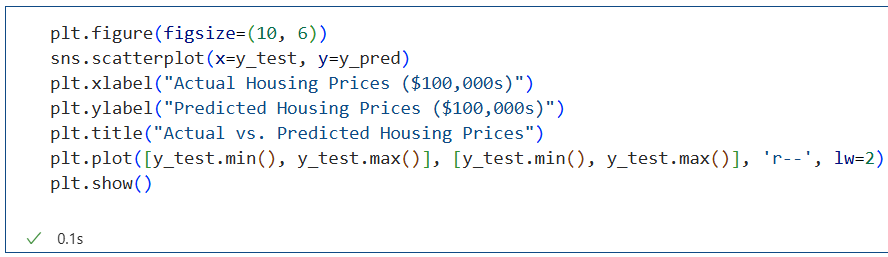
* MedHouseVal – Median house value (in $100k)

**Type:** Regression dataset, no missing values, all numerical.

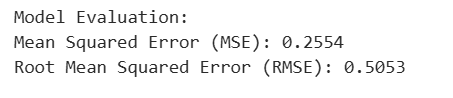
**Use:** Evaluate regression models like Linear Regression, Decision Trees, and Random Forest.

**SOURCE CODE:**

****

****

**OUTPUT:**

****

****

**LEARNING OUTCOME:**

**EXPERIMENT 6(A)**

**AIM:** Support Vector Machines (SVM): Implement SVM on a dataset and evaluate the model’s performance.

**THEORY:**

* SVM is a supervised machine learning algorithm used for both classification and regression.
* It tries to find an optimal hyperplane that separates data points of different classes with the maximum margin.
* For non-linearly separable data, SVM uses the kernel trick to map the data into higher dimensions where a separating hyperplane exists**.**

Key equations:

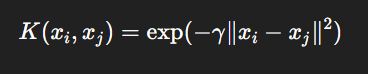
* 1. Decision Function:

F(x) = w.x + b

Where w is the weight vector and b is the bias.

* 1. Optimization Objective:

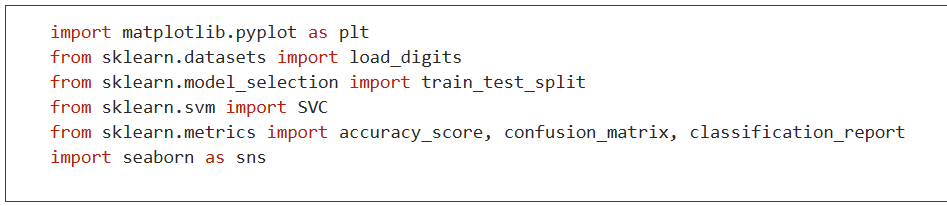
1. Kernel Function:

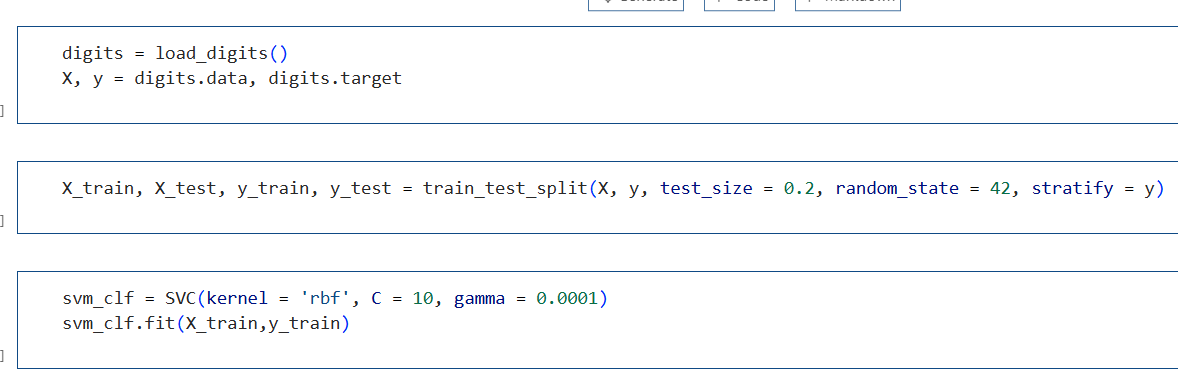


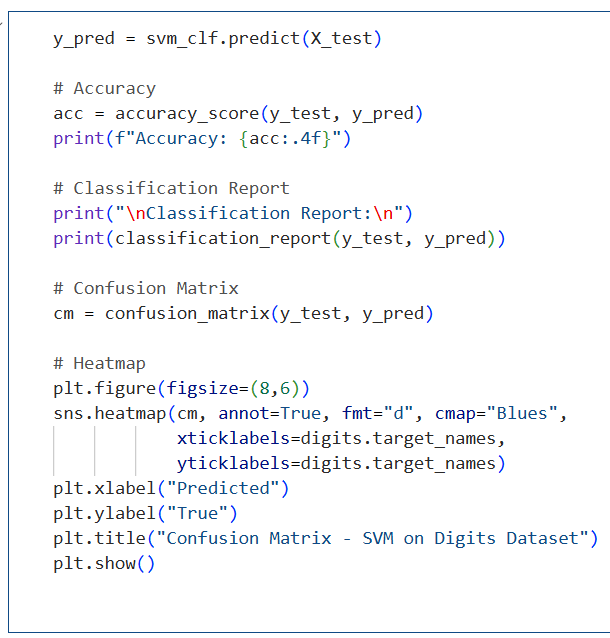
**ABOUT THE DATASET:**

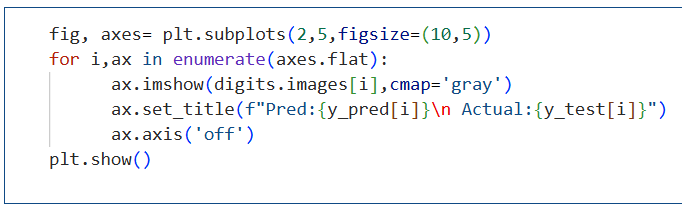
* A small image dataset available in sklearn.datasets.
* Consists of 1,797 grayscale images of handwritten digits (0–9).
* Each image is 8×8 pixels (64 features after flattening).
* Target labels correspond to the digit value.

**SOURCE CODE:**

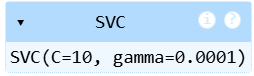


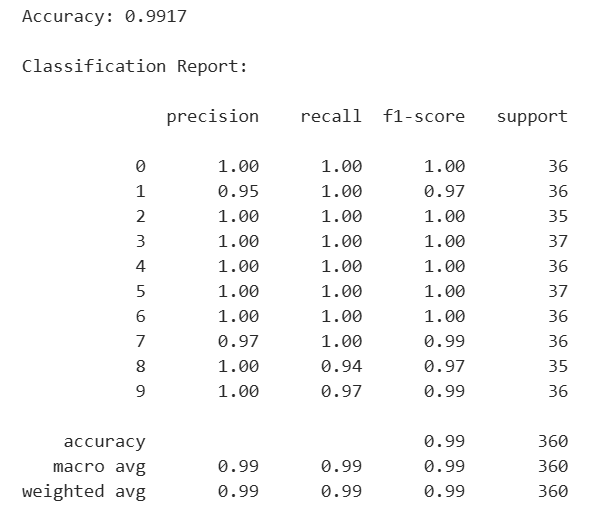


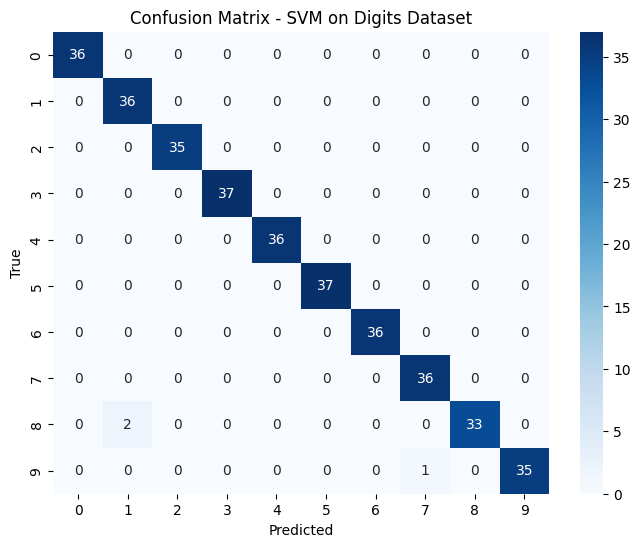


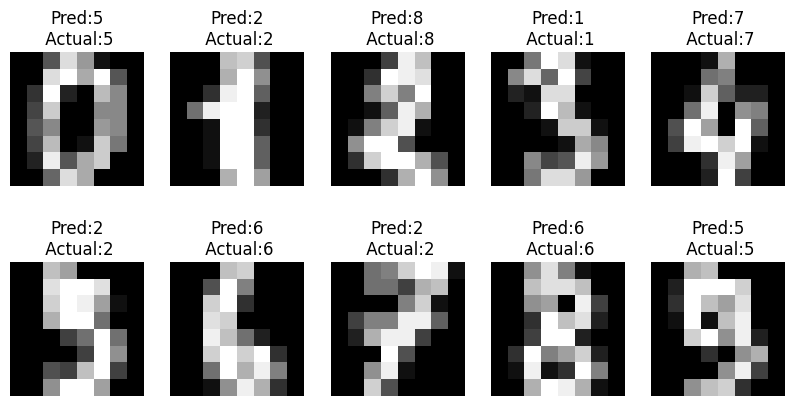


**OUTPUT:**

****

****

****

****

**LEARNING OUTCOMES:**

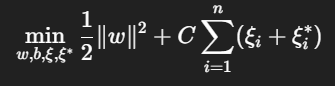
**EXPERIMENT 6(B)**

**AIM:** Support Vector Machines (SVM): Implement SVM on a dataset and evaluate the model’s performance.

**THEORY:**

* SVR is the regression counterpart of Support Vector Machine (SVM).
* Instead of predicting discrete labels, SVR predicts a continuous target variable.
* It tries to fit a function f(x) such that:
  + The deviations from the actual targets are at most ε (epsilon).
  + Complexity of the model is minimized.

Mathematical formulation:



Where:

C: Regularization parameter (controls trade-off between flatness and tolerance to deviations).

ϵ: Tube size for acceptable error.

γ: Parameter of RBF kernel controlling the influence of each support vector.

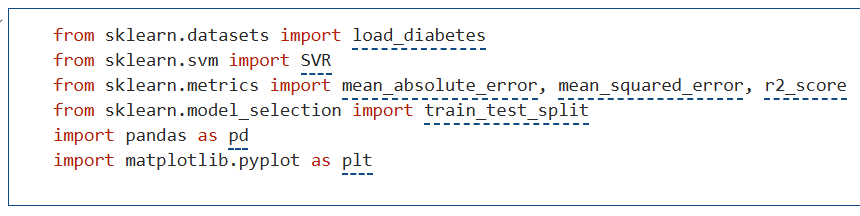
RBF Kernel:

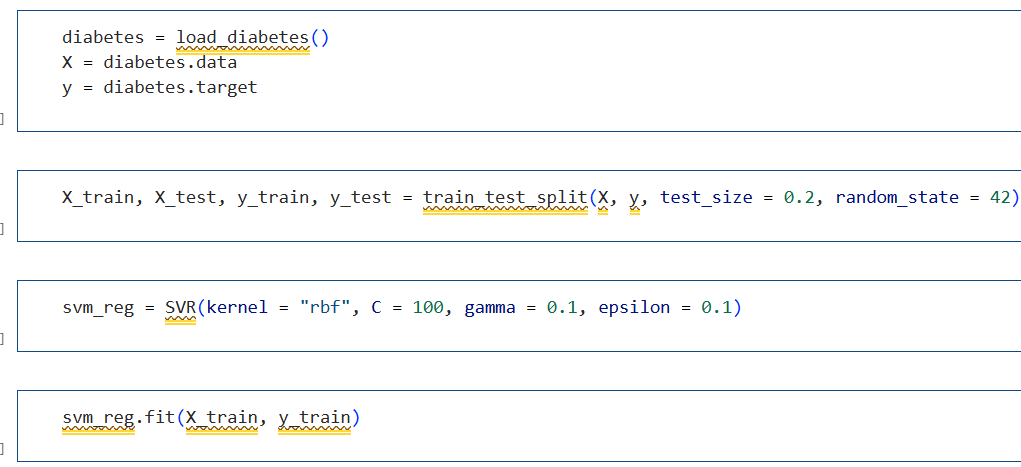


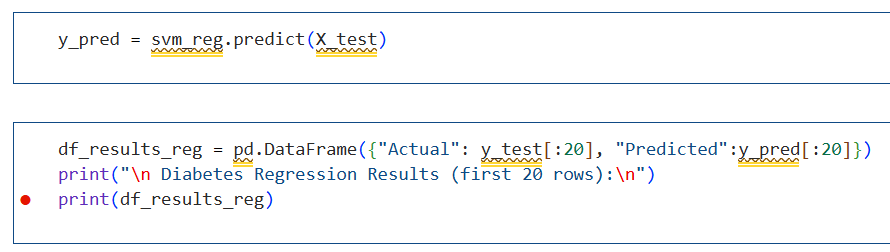
**ABOUT THE DATASET:**

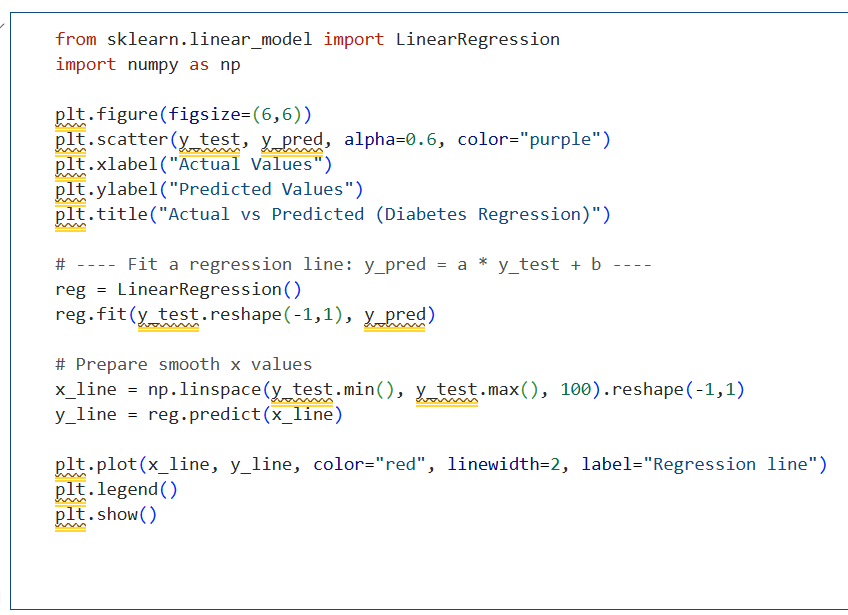
* Available in sklearn.datasets.
* Contains 442 samples and 10 baseline features (e.g., age, BMI, blood pressure, serum measurements).
* Target is a continuous measure of disease progression after one year.

**SOURCE CODE:**

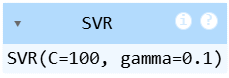


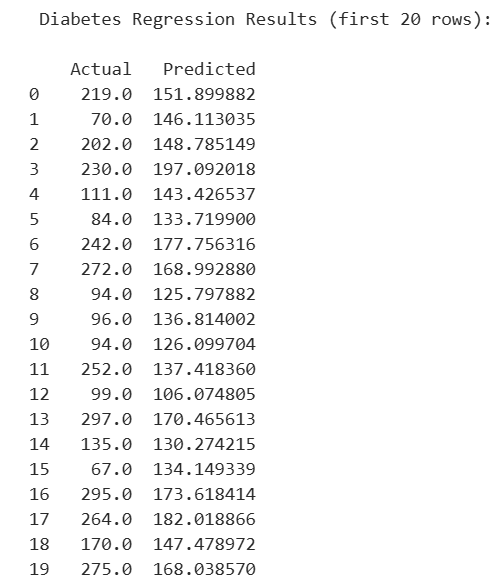


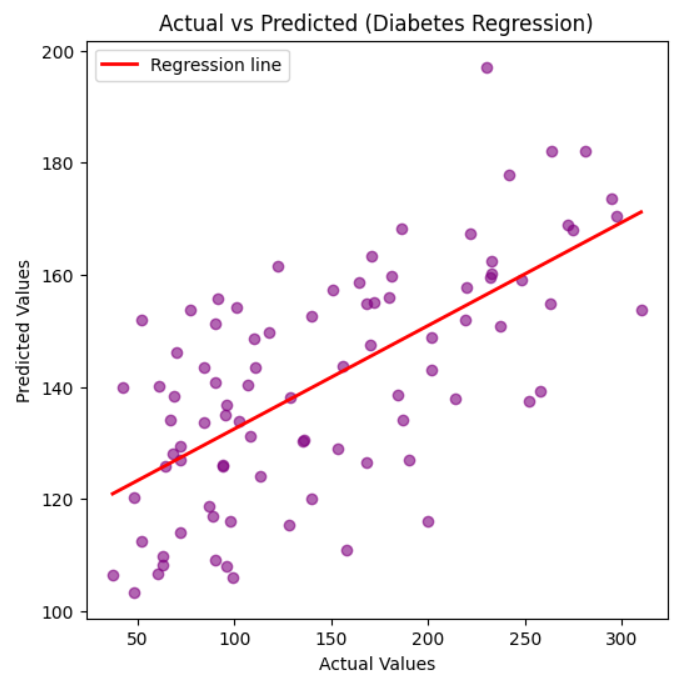




**OUTPUT:**

****

****

****

**LEARNING OUTCOMES:**

**EXPERIMENT 6(C)**

**AIM:** Support Vector Machines (SVM): Implement SVM on a dataset and evaluate the model’s performance.

**THEORY:**

SVM is a supervised machine learning algorithm used for both classification and regression.

It tries to find an optimal hyperplane that separates data points of different classes with the maximum margin.

For non-linearly separable data, SVM uses the kernel trick to map the data into higher dimensions where a separating hyperplane exists.

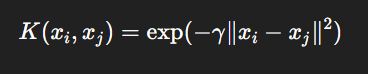
Key equations:

* 1. Decision Function:

F(x) = w.x + b

Where w is the weight vector and b is the bias.

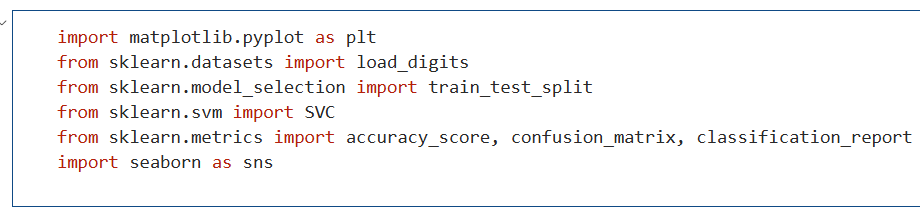
* 1. Optimization Objective:
  2. Kernel Function:

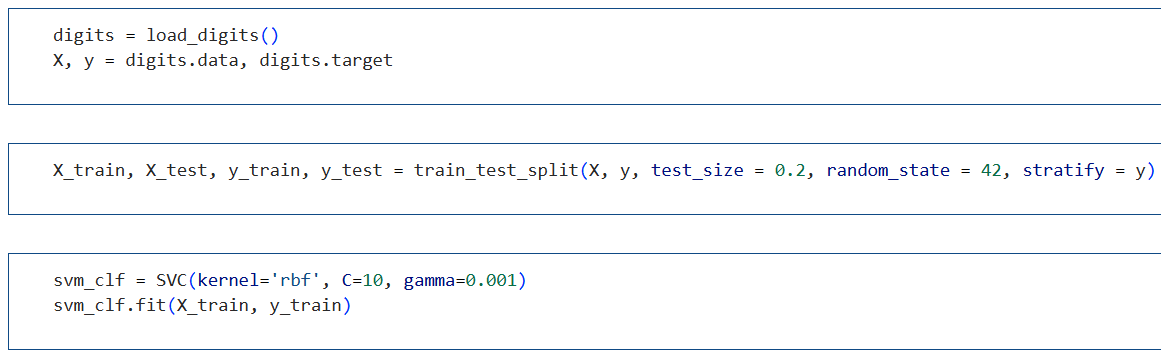


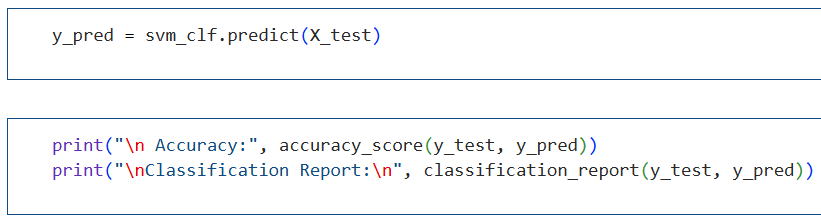
**ABOUT THE DATASET:**

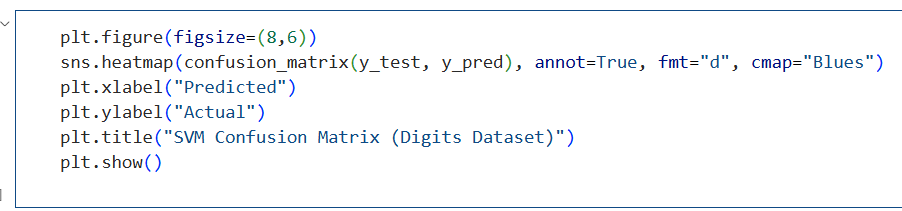
* 1,797 grayscale images of size 8×8 pixels.
* Each image represents a handwritten digit (0–9).
* Features: 64 pixel intensity values per sample.
* Target: digit label (0–9).

**SOURCE CODE:**

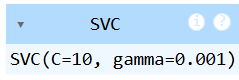


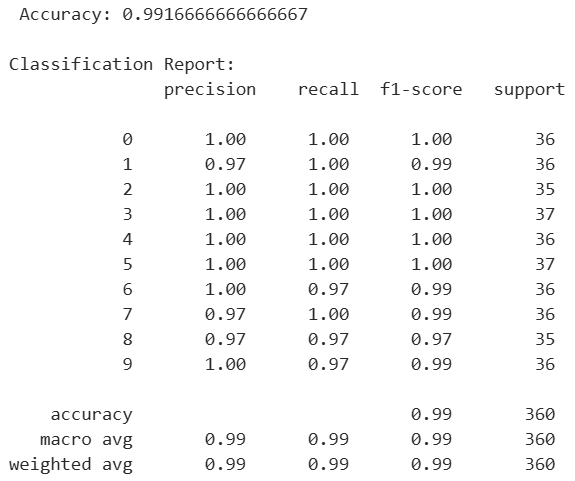


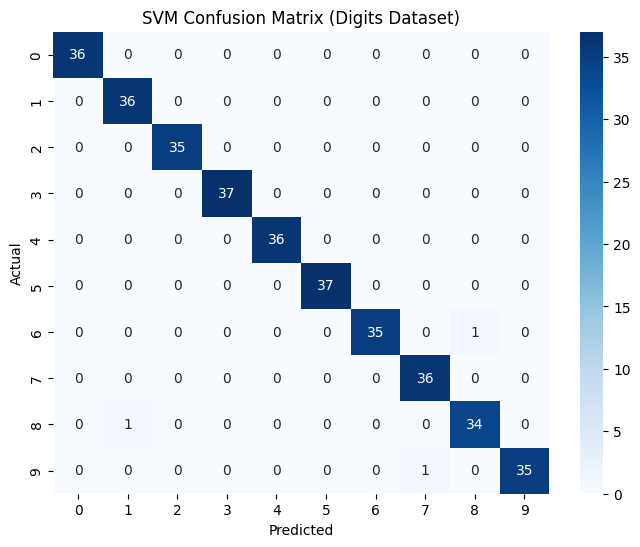




**OUTPUT:**

****

****

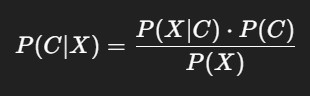
****

**LEARNING OUTCOMES:**

**EXPERIMENT 7**

**AIM: Naïve Bayes:** Implement Nfaïve Bayes on a dataset and evaluate the model’s performance.

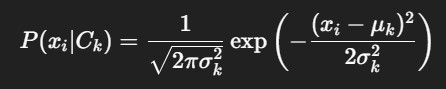
**THEORY:**

* Naïve Bayes is a probabilistic supervised learning algorithm based on Bayes’ theorem.
* It assumes that all features are conditionally independent given the class label (the “naïve” assumption).
* Bayes’ theorem:

Where:

* P(CX): Posterior probability of class CCC given data XXX.
* P(XC): Likelihood of data given class.
* P(C): Prior probability of class.
* P(X): Probability of data.

Gaussian Naïve Bayes assumes that the likelihood P(XC)P(X|C)P(XC) follows a normal distribution:

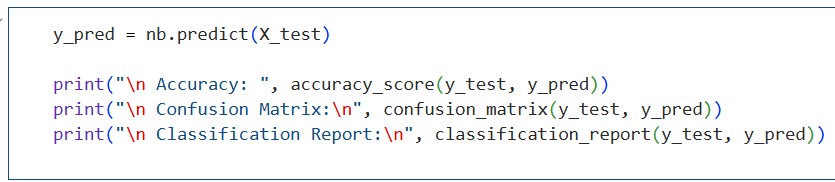
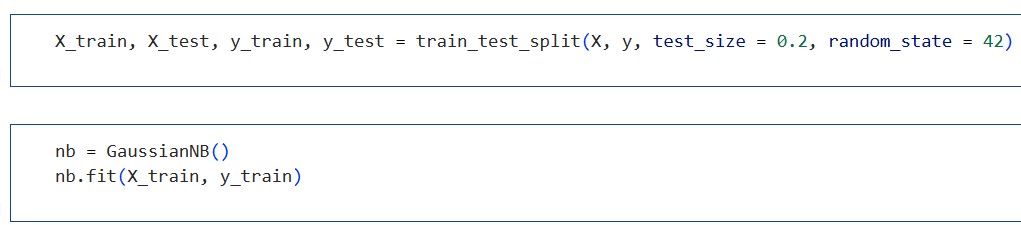
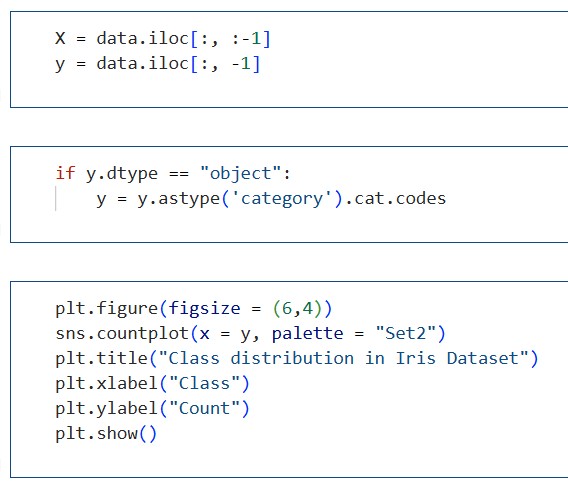
****

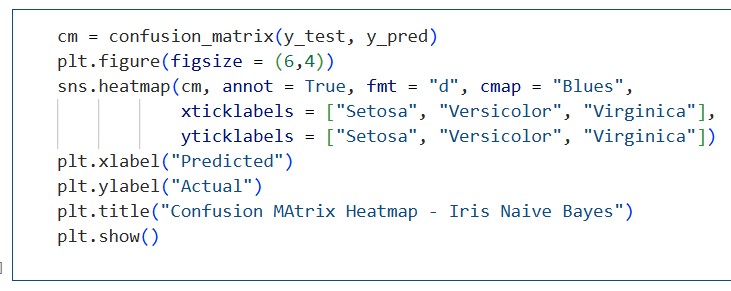
**ABOUT THE DATASET:**

The Iris dataset is a classical dataset in machine learning with:

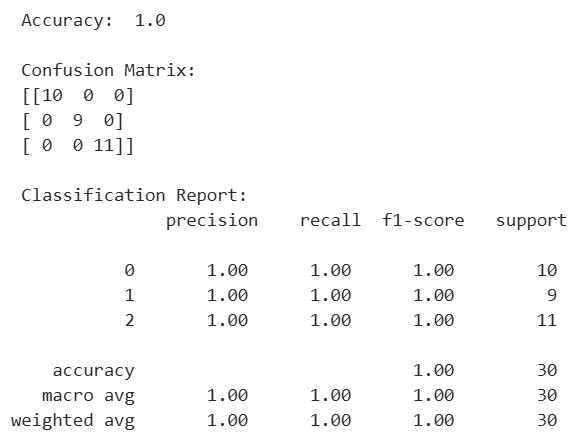
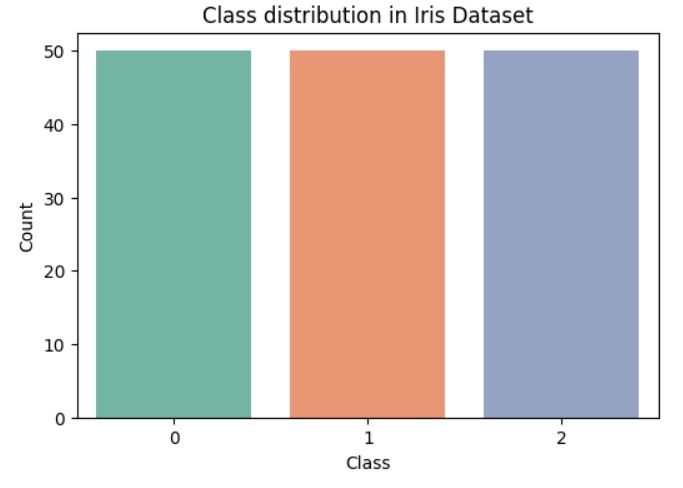
* 150 samples of iris flowers.
* 3 species: Setosa, Versicolor, Virginica.
* 4 features: Sepal Length, Sepal Width, Petal Length, Petal Width. Goal: Classify flowers into their species based on features.

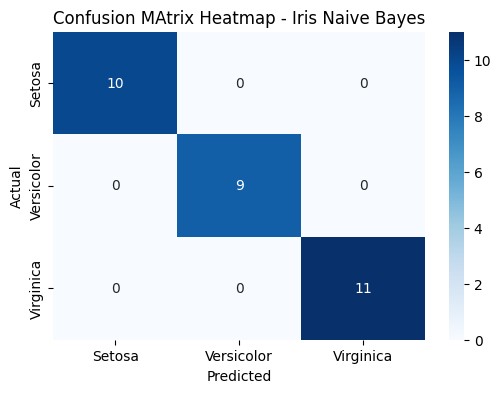
**SOURCE CODE:**



****

**OUTPUT:**



****

**LEARNING OUTCOMES:**