#(1)sklearn linear regression example using diabetes with plot

import numpy as np

import matplotlib.pyplot as plt

from sklearn import datasets, linear\_model

from sklearn.metrics import mean\_squared\_error, r2\_score

diabetes = datasets.load\_diabetes()

diabetes\_X = diabetes.data[:, np.newaxis, 2]

diabetes\_X\_train = diabetes\_X[:-20]

diabetes\_X\_test = diabetes\_X[-20:]

diabetes\_y\_train = diabetes.target[:-20]

diabetes\_y\_test = diabetes.target[-20:]

regr = linear\_model.LinearRegression()

regr.fit(diabetes\_X\_train, diabetes\_y\_train)

diabetes\_y\_pred = regr.predict(diabetes\_X\_test)

plt.scatter(diabetes\_X\_test, diabetes\_y\_test,  color='black')

plt.plot(diabetes\_X\_test, diabetes\_y\_pred, color='blue', linewidth=3)

plt.xlabel('Scaled BMI')

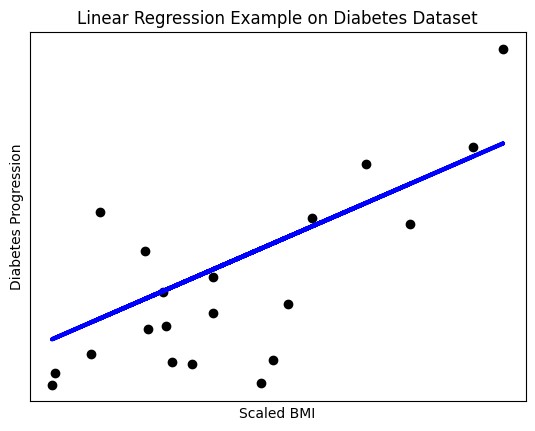
plt.ylabel('Diabetes Progression')

plt.title('Linear Regression Example on Diabetes Dataset')

plt.xticks(())

plt.yticks(())

plt.show()



#(2)sklearn linear regression  using iris dataset

import numpy as np

import matplotlib.pyplot as plt

from sklearn import datasets, linear\_model

iris = datasets.load\_iris()

X = iris.data[:, np.newaxis, 0]

y = iris.target

regr = linear\_model.LinearRegression().fit(X, y)

y\_pred = regr.predict(X)

plt.scatter(X, y, color='black')

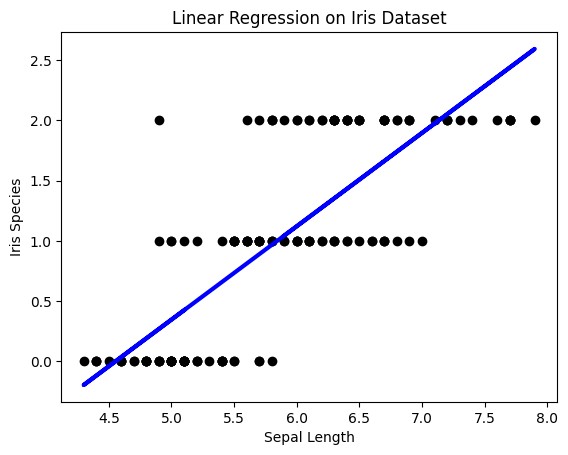
plt.plot(X, y\_pred, color='blue', linewidth=3)

plt.xlabel('Sepal Length')

plt.ylabel('Iris Species')

plt.title('Linear Regression on Iris Dataset')

plt.show()



#(3) Python program using Iris dataset before fitting it to a Linear Regression Model with plot

import numpy as np

import matplotlib.pyplot as plt

from sklearn import datasets

iris = datasets.load\_iris()

X = iris.data[:, np.newaxis, 0]

y = iris.target

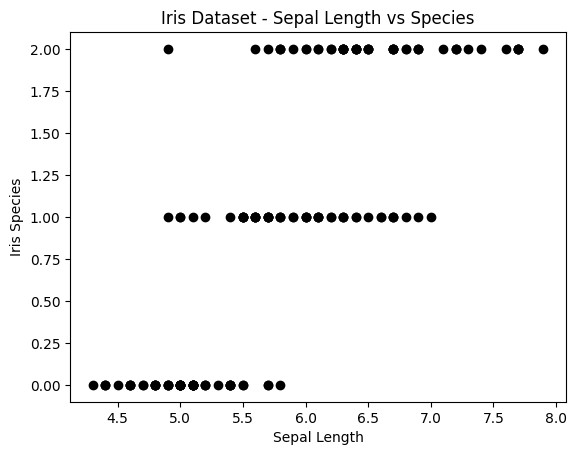
plt.scatter(X, y, color='black')

plt.xlabel('Sepal Length')

plt.ylabel('Iris Species')

plt.title('Iris Dataset - Sepal Length vs Species')

plt.show()



#(6) Python code on sklearn decision tree using breast\_cancer with plot.

import matplotlib.pyplot as plt

from sklearn.datasets import load\_breast\_cancer

from sklearn.tree import DecisionTreeClassifier, plot\_tree

from sklearn.model\_selection import train\_test\_split

breast\_cancer = load\_breast\_cancer()

X = breast\_cancer.data

y = breast\_cancer.target

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

clf = DecisionTreeClassifier(random\_state=42)

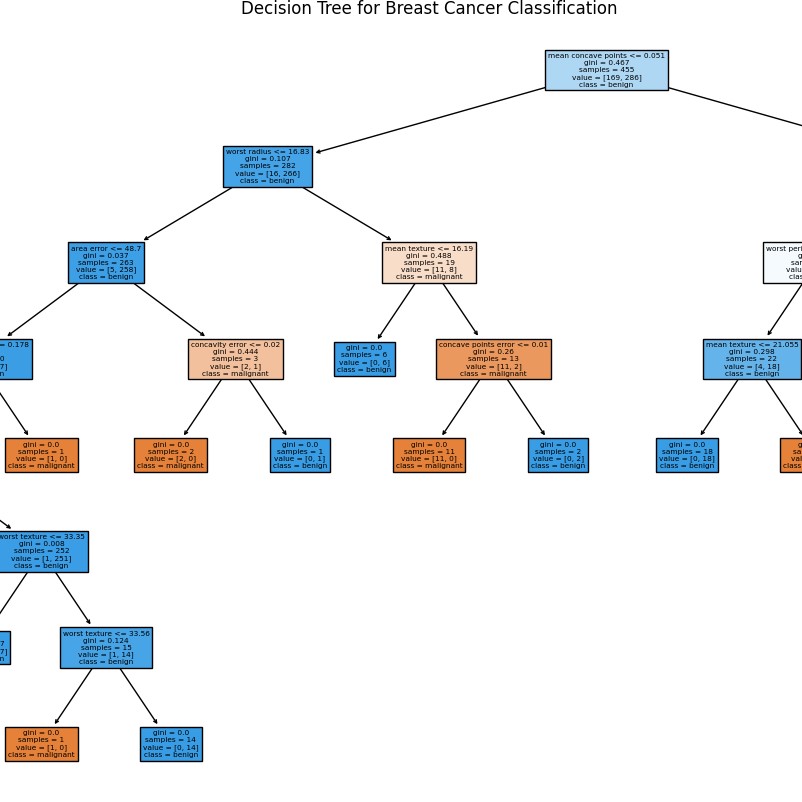
clf.fit(X\_train, y\_train)

plt.figure(figsize=(20, 10))

plot\_tree(clf, filled=True, feature\_names=breast\_cancer.feature\_names, class\_names=breast\_cancer.target\_names)

plt.title('Decision Tree for Breast Cancer Classification')

plt.show()



#(7) Python code on sklearn Kmeans classifier using Iris dataset with plot.

import matplotlib.pyplot as plt

from sklearn.datasets import load\_iris

from sklearn.cluster import KMeans

iris = load\_iris()

X = iris.data[:, :2]

kmeans = KMeans(n\_clusters=3, random\_state=42)

kmeans.fit(X)

plt.figure(figsize=(8, 6))

plt.scatter(X[:, 0], X[:, 1], c=kmeans.labels\_, cmap='viridis', edgecolor='k')

plt.scatter(kmeans.cluster\_centers\_[:, 0], kmeans.cluster\_centers\_[:, 1], marker='x', s=300, c='red', label='Centroids')

plt.title('KMeans Clustering on Iris Dataset')

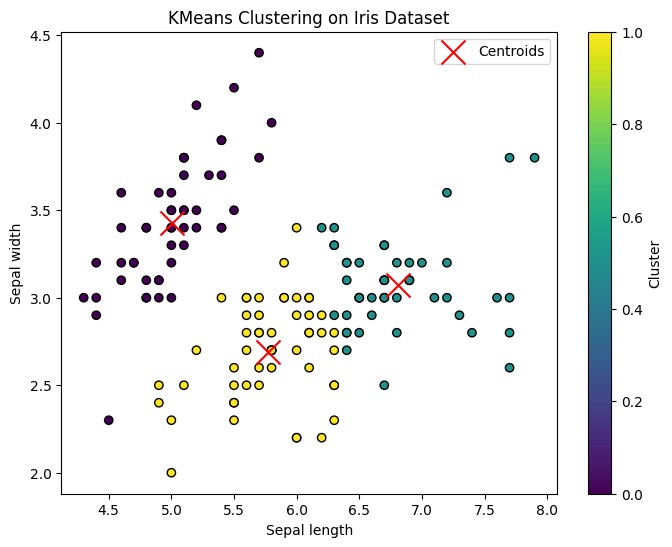
plt.xlabel('Sepal length')

plt.ylabel('Sepal width')

plt.legend()

plt.colorbar(label='Cluster')

plt.show()



#(8) Python code on sklearn Random Forest using breast\_cancer with plot

import numpy as np

import matplotlib.pyplot as plt

from sklearn.datasets import load\_breast\_cancer

from sklearn.ensemble import RandomForestClassifier

breast\_cancer = load\_breast\_cancer()

X = breast\_cancer.data

y = breast\_cancer.target

clf = RandomForestClassifier(n\_estimators=100, random\_state=42)

clf.fit(X, y)

plt.figure(figsize=(10, 6))

plt.barh(range(X.shape[1]), clf.feature\_importances\_, align='center')

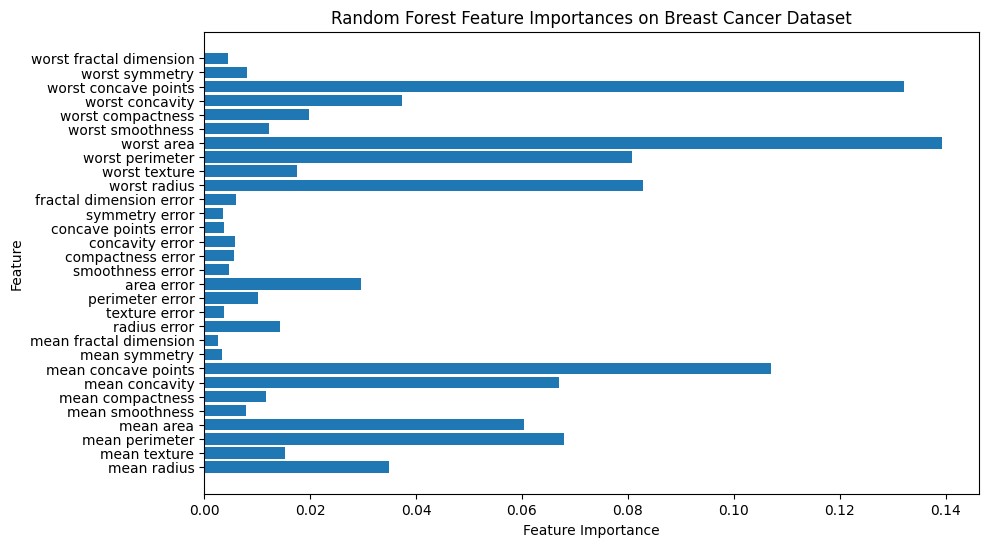
plt.yticks(range(X.shape[1]), breast\_cancer.feature\_names)

plt.xlabel('Feature Importance')

plt.ylabel('Feature')

plt.title('Random Forest Feature Importances on Breast Cancer Dataset')

plt.show()



#(9) Python code on sklearn Kmeans classifier using breast\_cancer dataset with plot.

import matplotlib.pyplot as plt

from sklearn.datasets import load\_breast\_cancer

from sklearn.cluster import KMeans

from sklearn.decomposition import PCA

breast\_cancer = load\_breast\_cancer()

X = breast\_cancer.data

pca = PCA(n\_components=2)

X\_pca = pca.fit\_transform(X)

kmeans = KMeans(n\_clusters=2, random\_state=42)

kmeans.fit(X\_pca)

plt.figure(figsize=(8, 6))

plt.scatter(X\_pca[:, 0], X\_pca[:, 1], c=kmeans.labels\_, cmap='viridis', edgecolor='k')

plt.scatter(kmeans.cluster\_centers\_[:, 0], kmeans.cluster\_centers\_[:, 1], marker='x', s=300, c='red', label='Centroids')

plt.title('KMeans Clustering on Breast Cancer Dataset')

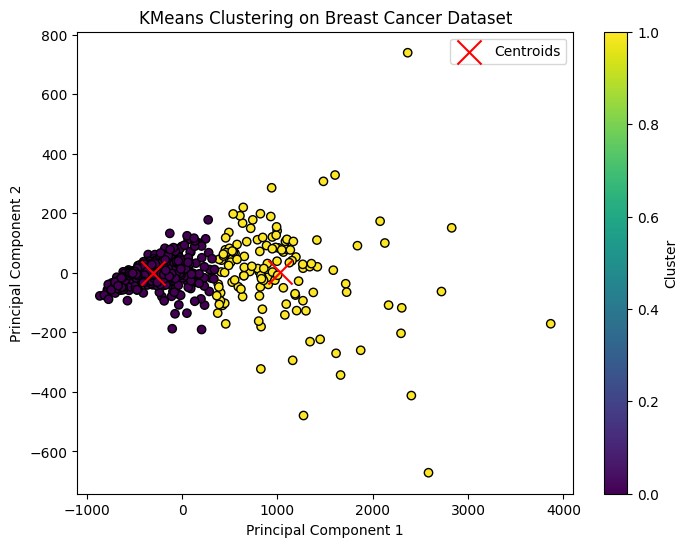
plt.xlabel('Principal Component 1')

plt.ylabel('Principal Component 2')

plt.legend()

plt.colorbar(label='Cluster')

plt.show()



#(10) Python code on sklearn Random Forest using Iris dataset with plot

import numpy as np

import matplotlib.pyplot as plt

from sklearn.datasets import load\_iris

from sklearn.ensemble import RandomForestClassifier

iris = load\_iris()

X = iris.data[:, :2]

y = iris.target

clf = RandomForestClassifier(n\_estimators=100, random\_state=42)

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, 0.1), np.arange(y\_min, y\_max, 0.1))

Z = clf.predict(np.c\_[xx.ravel(), yy.ravel()])

Z = Z.reshape(xx.shape)

plt.contourf(xx, yy, Z, alpha=0.8)

plt.scatter(X[:, 0], X[:, 1], c=y, edgecolors='k', s=20)

plt.xlabel('Sepal length')

plt.ylabel('Sepal width')

plt.title('Random Forest Decision Boundaries on Iris Dataset')

plt.show()

##5. Python code on sklearn logistic regression using breast\_cancer with plot

import numpy as np

import matplotlib.pyplot as plt

from sklearn import datasets

from sklearn.linear\_model import LogisticRegression

breast\_cancer = datasets.load\_breast\_cancer()

X = breast\_cancer.data

y = breast\_cancer.target

logreg = LogisticRegression()

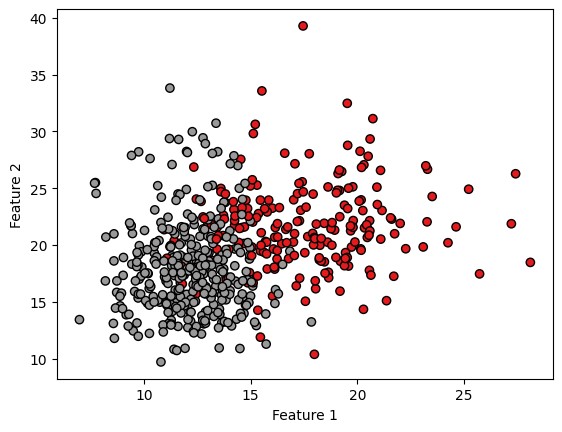
logreg.fit(X, y)

plt.scatter(X[:, 0], X[:, 1], c=y, cmap=plt.cm.Set1, edgecolor='k')

plt.xlabel('Feature 1')

plt.ylabel('Feature 2')

plt.show()



##4. Python code on sklearn logistic regression using diabetes with plot

import numpy as np

import matplotlib.pyplot as plt

from sklearn import datasets

from sklearn.linear\_model import LogisticRegression

diabetes = datasets.load\_diabetes()

X = diabetes.data

y = diabetes.target

logreg = LogisticRegression()

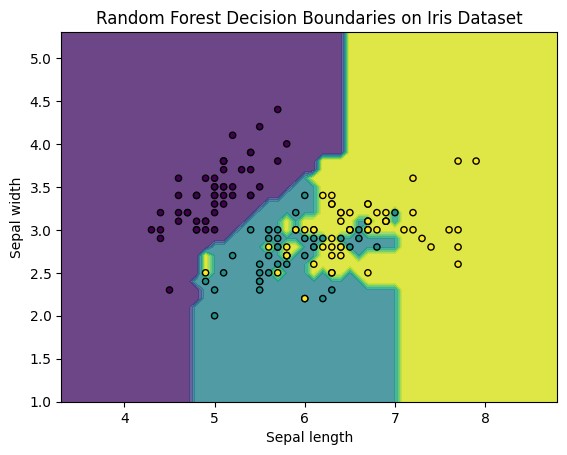
logreg.fit(X, y)

plt.scatter(X[:, 0], X[:, 1], c=y, cmap=plt.cm.Set1, edgecolor='k')

plt.xlabel('Feature 1')

plt.ylabel('Feature 2')

plt.show()



##4. Python code on sklearn logistic regression using diabetes with plot

import numpy as np

import matplotlib.pyplot as plt

from sklearn import datasets

from sklearn.linear\_model import LogisticRegression

diabetes = datasets.load\_diabetes()

X = diabetes.data

y = diabetes.target

logreg = LogisticRegression()

logreg.fit(X, y)

plt.scatter(X[:, 0], X[:, 1], c=y, cmap=plt.cm.Set1, edgecolor='k')

plt.xlabel('Feature 1')

plt.ylabel('Feature 2')

plt.show()

