Assignment – 12

# B.Rithwik

# 2303A52330

# Batch – 35

**Read the Data, Find Features and Target Variables and Split into Train and Test Data**

import pandas as pd

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

from sklearn.preprocessing import LabelEncoder, StandardScaler

from sklearn.svm import SVC

from sklearn.neighbors import KNeighborsClassifier

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import accuracy\_score

import matplotlib.pyplot as plt

import seaborn as sns

import numpy as np

from sklearn.utils import resample

# Load your dataset

df = pd.read\_csv('/content/drive/MyDrive/SML Dataset/breast\_cancer\_survival.csv')

# Separate features and target

features = df.drop(columns=['Patient\_Status']) # Replace 'target\_column' with your actual target column name

target = df['Patient\_Status']

# Encode categorical features and target if needed

label\_encoder = LabelEncoder()

target = label\_encoder.fit\_transform(target)

# Convert categorical features to numerical using Label Encoding

for column in features.select\_dtypes(include=['object']).columns:

features[column] = label\_encoder.fit\_transform(features[column])

# Scale features (important for logistic regression to converge faster)

scaler = StandardScaler()

features = scaler.fit\_transform(features)

# Split the data into training and testing sets

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

**SVM ACCURACY**

svm\_model = SVC()

svm\_model.fit(X\_train, y\_train)

svm\_predictions = svm\_model.predict(X\_test)

svm\_accuracy = accuracy\_score(y\_test, svm\_predictions)

print(f"SVM Accuracy: {svm\_accuracy \* 100:.2f}%")

**OUTPUT –**

SVM Accuracy: 77.61%

**KNN ACCURACY**

knn\_model = KNeighborsClassifier()

knn\_model.fit(X\_train, y\_train)

knn\_predictions = knn\_model.predict(X\_test)

knn\_accuracy = accuracy\_score(y\_test, knn\_predictions)

print(f"KNN Accuracy: {knn\_accuracy \* 100:.2f}%")

**OUTPUT –**

KNN Accuracy: 74.63%

**LOGISTIC REGRESSION ACCURACY**

logistic\_model = LogisticRegression(max\_iter=1000)

logistic\_model.fit(X\_train, y\_train)

logistic\_predictions = logistic\_model.predict(X\_test)

logistic\_accuracy = accuracy\_score(y\_test, logistic\_predictions)

print(f"Logistic Regression Accuracy: {logistic\_accuracy \* 100:.2f}%")

**OUTPUT –**

Logistic Regression Accuracy: 77.61%

**Bootstrapping Plots For Each Model**

def bootstrap\_accuracy(model, X\_train, y\_train, X\_test, y\_test, n\_iterations=100):

accuracies = []

for i in range(n\_iterations):

# Sample with replacement

X\_resample, y\_resample = resample(X\_train, y\_train) # resample is now accessible

model.fit(X\_resample, y\_resample)

predictions = model.predict(X\_test)

accuracy = accuracy\_score(y\_test, predictions)

accuracies.append(accuracy)

return accuracies

# Get bootstrapped accuracies for each model

svm\_accuracies = bootstrap\_accuracy(svm\_model, X\_train, y\_train, X\_test, y\_test)

knn\_accuracies = bootstrap\_accuracy(knn\_model, X\_train, y\_train, X\_test, y\_test)

logistic\_accuracies = bootstrap\_accuracy(logistic\_model, X\_train, y\_train, X\_test, y\_test)

# Plot bootstrapped accuracies for each model

plt.figure(figsize=(15, 5))

sns.histplot(svm\_accuracies, kde=True, color='blue', label='SVM', bins=15)

sns.histplot(knn\_accuracies, kde=True, color='green', label='KNN', bins=15)

sns.histplot(logistic\_accuracies, kde=True, color='red', label='Logistic Regression', bins=15)

plt.legend()

plt.title('Bootstrapping Accuracy Distribution for Each Model')

plt.xlabel('Accuracy')

plt.ylabel('Frequency')

plt.show()

**OUTPUT -**

A graph with green and red squares

Description automatically generated with medium confidence