Objective:

The goal of this project is to develop a predictive model for breast cancer classification using logistic regression. The dataset used in this project contains various features extracted from breast cancer biopsies, and the target variable is the diagnosis (Malignant or Benign).

1. Libraries and Setup:

import warnings

warnings.filterwarnings("ignore")

import numpy as np

import scipy as sp

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import sklearn

%matplotlib inline

1. Data Loading:

df1 = pd.read\_csv(r"C:\Users\salma elbadry\Desktop\Projects\BREAST\_CANSER\_CLASS\_SVM\data.csv")

.3 Data Exploration and Preprocessing:

Check for missing values:

df1.isna().sum()

Explore the dataset:

df1.info()

4. Data Cleaning and Label Encoding:

label\_encoder = LabelEncoder()

df1["diagnosis"] = label\_encoder.fit\_transform(df1["diagnosis"])

5. Data Splitting:

x = df1.drop(columns=["diagnosis", "Unnamed: 32"], axis=1)

y = df1["diagnosis"]

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size=0.33, shuffle=True, random\_state=42)

6. Logistic Regression Model Training:

model = LogisticRegression()

model.fit(x\_train, y\_train)

7. Model Evaluation:

model.score(x\_train, y\_train)

y\_pred = model.predict(x\_test)

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

precision = precision\_score(y\_test, y\_pred)

recall = recall\_score(y\_test, y\_pred)

f1 = f1\_score(y\_test, y\_pred)

conf\_matrix = confusion\_matrix(y\_test, y\_pred)

8. Hyperparameter Tuning with GridSearchCV:

param\_grid = {

'C': [0.001, 0.01, 0.1, 1, 10, 100],

'solver': ['lbfgs', 'liblinear', 'newton-cg', 'sag', 'saga'],

'penalty': ['l1', 'l2']

}

grid\_search = GridSearchCV(LogisticRegression(), param\_grid, cv=5, scoring='accuracy')

grid\_search.fit(x\_train, y\_train)

best\_params = grid\_search.best\_params\_

best\_model = grid\_search.best\_estimator\_

9. Model Evaluation with Tuned Hyperparameters:

y\_pred = best\_model.predict(x\_test)

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

precision = precision\_score(y\_test, y\_pred)

recall = recall\_score(y\_test, y\_pred)

f1 = f1\_score(y\_test, y\_pred)

conf\_matrix = confusion\_matrix(y\_test, y\_pred)

10. Visualizations:

Countplot:

sns.countplot(data=df1, x="diagnosis")

A graph with a blue and orange rectangle

Description automatically generated

Barplot:

sns.barplot(data=df1, x="diagnosis", y="radius\_mean")

A graph of a diagram

Description automatically generated with medium confidence

Confusion Matrix Heatmap before tuning:

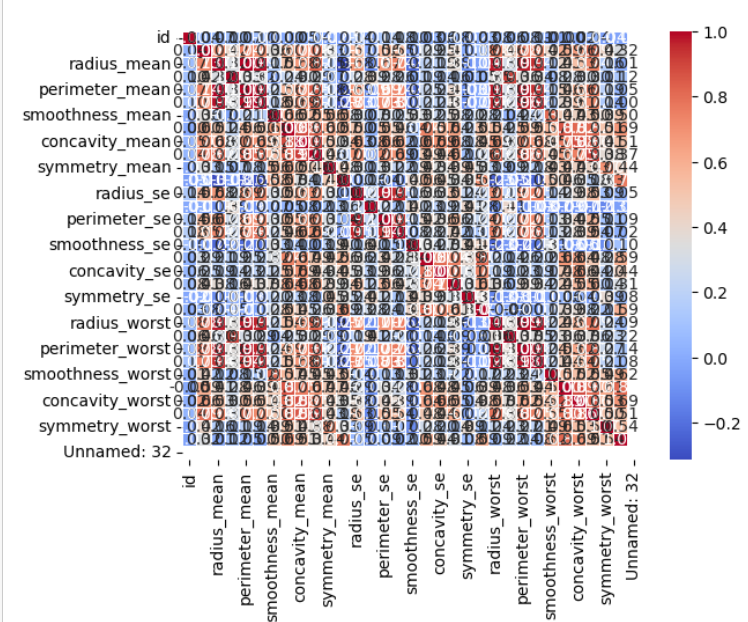
sns.heatmap(conf\_matrix, annot=True, fmt="d", cmap="Blues", cbar=False)

A blue squares with white numbers

Description automatically generated

Correlation Matrix Heatmap before tuning:

sns.heatmap(correlation\_matrix, annot=True, cmap="coolwarm", fmt=".2f", linewidths=0.5)



Confusion Matrix Heatmap after tuning:

A screenshot of a graph

Description automatically generated

This documentation provides an overview of the breast cancer classification project, covering data loading, preprocessing, model training, evaluation, and visualizations. Make sure to customize the documentation based on the specific details and insights derived from your analysis.