

**Histopathology Cancer Detection: A Computer Vision Approach Using Convolution Neural Networks for Classification**

**Project Documentation**

Top of Form

**ID.No.875**

**Submitting by**

Y. Harika

22ME5A0512

**ACKNOWLEDGEMENT**

I would like to take the opportunity to express our deep gratitude to all the people who have extended their cooperation in various ways during my internship. It is my pleasure and responsibility to acknowledge the help of all the help of all those individuals.

I express my deepest gratitude to The Management of Ramachandra College of Engineering, Eluru for their support and encouragement in completing my internship and providing me necessary facilities.

I would like to express my deepest gratitude to my mentor for her valuable suggestions during the preparation of draft in our document and explaining everything clearly.

I sincerely thank all the faculty members for their valuable advices, suggestions and constant encouragement which played a vital role in carrying out my internship.

Finally, I thank one and all who directly or indirectly helped me to complete my internship successfully.

Y. Harika

22ME5A0512

**Abstract**

Histopathology cancer detection stands as a critical task in modern oncology, where accurate and timely identification of malignant tissues is paramount for effective treatment planning and patient outcomes. This project proposes a novel computer vision approach leveraging Convolutional Neural Networks (CNNs) for the automated classification of histopathology images into cancerous and non-cancerous categories. Evaluation metrics such as sensitivity, specificity, and area under the curve (AUC) are utilized to assess the performance of the proposed CNN-based cancer detection system. Through experimentation on benchmark datasets and comparison with existing methodologies, this project demonstrates the efficacy of the proposed approach in achieving accurate and reliable cancer detection, paving the way for its integration into clinical practice for improved diagnostic accuracy and patient care.

**Key points:**

**• Current challenge and solution**: Histopathological analysis is crucial for cancer diagnosis, but it is time-consuming and subjective. This study explores using convolutional neural networks (CNNs) for automated cancer detection in histopathology images, offering a potential tool to improve accuracy and efficiency.

• **Focus and contribution**: This project investigates the application of CNNs for classifying histopathology images to aid cancer detection. We propose a CNN-based approach for cancer classification and evaluate its performance.

• **Future directions**: These findings suggest the promise of CNNs for cancer detection. Further research is warranted to explore generalizability and integration with clinical workflows

**Learning Objectives/Internship Objectives**

• Internships are generally thought of to be reserved for college students looking to gain experience in a particular field. However, a wide array of people can benefit from Training Internships in order to receive real world experience and develop their skills.

• An objective for this position should emphasize the skills you already possess in the area and your interest in learning more.

• Internships are utilized in a number of different career fields, including architecture, engineering, healthcare, economics, advertising and many more.

• Some internship is used to allow individuals to perform scientific research while others are specifically designed to allow people to gain first-hand experience working.

• Utilizing internships is a great way to build your resume and develop skills that can be emphasized in your resume for future jobs. When you are applying for a Training Internship, make sure to highlight any special skills or talents that can make you stand apart from the rest of the applicants so that you have an improved chance of landing the position.

**IMPORTED LIBRARIES**

**NumPy:**

NumPy, short for Numerical Python, is a fundamental library for numerical computing in Python. It provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays efficiently. NumPy's array operations are faster and more memory-efficient than traditional Python lists, making it ideal for scientific computing, data analysis, and machine learning tasks. It offers functionalities for array manipulation, linear algebra, Fourier transforms, random number generation, and more. NumPy's simplicity and performance have made it a cornerstone in the Python ecosystem for numerical computations and data processing.

**Purpose:** NumPy is a fundamental package for numerical computing in Python. It provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays efficiently.

**Example Usage:** import numpy as np allows you to use NumPy functions and create NumPy arrays like np.array([1, 2, 3]).

**Pandas:**

Pandas is a powerful Python library designed for data manipulation and analysis. It introduces data structures like DataFrame and Series, enabling users to handle structured data efficiently. With Pandas, tasks such as data cleaning, transformation, filtering, grouping, and merging become straightforward. It supports reading and writing data from various file formats like CSV, Excel, SQL databases, and JSON. Pandas integrates seamlessly with other libraries like NumPy for numerical computations, making it a versatile tool for data scientists, analysts, and developers. Its intuitive interface, rich functionalities, and ability to work with large datasets make Pandas indispensable in data-related workflows.

**Purpose:** Pandas is a powerful library for data manipulation and analysis. It provides data structures like Data Frame and Series, which are designed for handling structured data and performing various operations such as filtering, grouping, merging, and more.

**Example Usage:** import pandas as pd enables you to work with Pandas data structures and functionalities, such as reading CSV files (pd.read\_csv()) or performing data manipulations.

**Matplotlib:**

Matplotlib.pyplot, often imported with the alias plt, is a prominent Python library used for creating static, interactive, and publication-quality visualizations. It offers a wide range of plot types, including line plots, scatter plots, bar plots, histograms, pie charts, and more. Matplotlib.pyplot provides extensive customization options for appearance, labels, titles, legends, and annotations, allowing users to create visually appealing and informative plots. Its integration with Jupyter Notebooks and other environments facilitates interactive data exploration and visualization. Matplotlib.pyplot's versatility, ease of use, and ability to generate professional-looking plots make it a go-to choice for data analysts, scientists, and researchers.

**Purpose:** Matplotlib is a widely-used plotting library in Python for creating static, interactive, and publication-quality visualizations. It offers a variety of plots, customization options, and interfaces for creating different types of charts and graphs.

**Example Usage:** import matplotlib.pyplot as plt allows you to create plots and visualizations using Matplotlib's pyplot interface, such as plt.plot() for line plots or plt.bar() for bar charts.

**Seaborn:**

Seaborn is a statistical data visualization library built on top of Matplotlib. It simplifies the creation of complex statistical plots, including heatmaps, violin plots, box plots, pair plots, and categorical plots. Seaborn offers default themes and color palettes for visually appealing and informative plots. It seamlessly integrates with Pandas data structures, allowing for easy data handling and plotting. Seaborn also supports advanced statistical visualizations like regression plots, factor plots, and cluster maps. Its high-level interface, aesthetic enhancements, and ability to create insightful visualizations make Seaborn a valuable tool for data analysts, researchers, and professionals working with Python for data visualization and exploration.

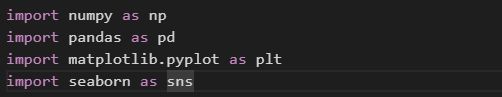
**Purpose:** Seaborn is built on top of Matplotlib and provides a higher-level interface for creating statistical visualizations. It simplifies the creation of complex plots and offers aesthetic enhancements, default themes, and color palettes.

**Example Usage:** import seaborn as sns enables you to use Seaborn's functions and capabilities, such as creating a heatmap (sns.heatmap()) or a violin plot (sns.violinplot()).

Together, NumPy, Pandas, Matplotlib, and Seaborn form a powerful ecosystem of libraries that cater to different stages of the data science pipeline. NumPy and Pandas handle data manipulation, cleaning, and preparation tasks, while Matplotlib and Seaborn excel in data visualization and presentation. This symbiotic relationship between these libraries enables users to seamlessly transition from data preprocessing and analysis to visualization and interpretation, ultimately facilitating informed decision-making and actionable insights from data.

These four libraries (NumPy, Pandas, Matplotlib, and Seaborn) are commonly used together in data analysis and visualization tasks. NumPy and Pandas handle data manipulation, while Matplotlib and Seaborn are used for creating visualizations, making them essential tools for working with data in Python.

In conclusion, the libraries discussed in this discussion, namely NumPy, Pandas, Matplotlib, and Seaborn, represent the backbone of Python-based data science, machine learning, and scientific computing workflows. Their versatility, efficiency, ease of use, and extensive capabilities empower users to tackle diverse challenges in data analysis, visualization, modelling, and beyond. As data continues to play a central role in driving innovation, discovery, and decision-making across industries, these libraries remain essential tools for harnessing the power of data and unlocking its full potential.

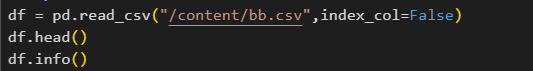
****

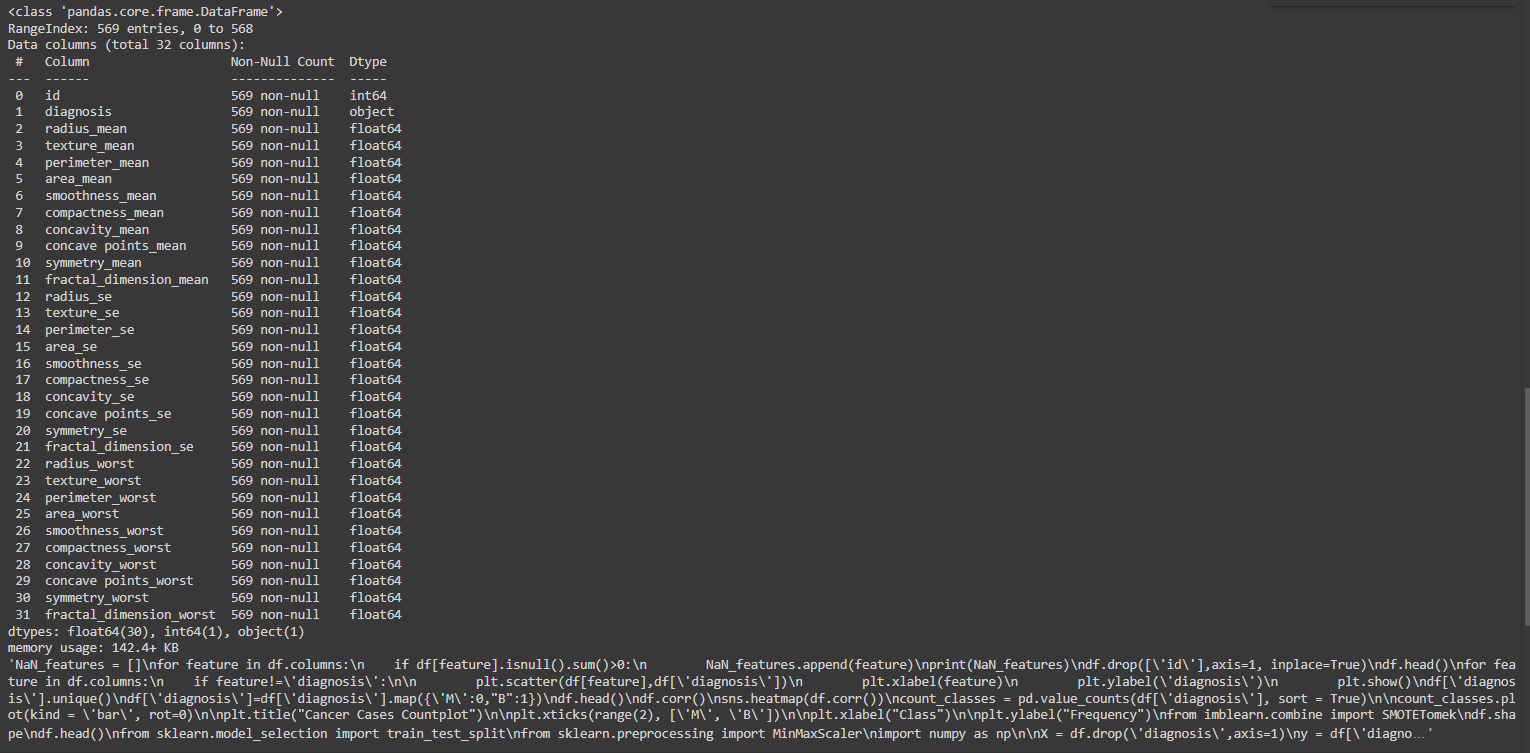
**LOAD THE DATASET**

The dataset that we used for the project on the topic “histopathologic cancer detection: A computer vision approach using convolution neural networks for classification” is taken from a website source named: Kaggle.com.

Kaggle is an online community platform for data scientists and machine learning enthusiasts. Kaggle allows users to collaborate with other users, find and publish datasets, use GPU integrated notebooks, and compete with other data scientists to solve data science challenges. The aim of this online platform (founded in 2010 by Anthony Goldbloom and Jeremy Howard and acquired by Google in 2017) is to help professionals and learners reach their goals in their data science journey with the powerful tools and resources it provides. As of today (2021), there are over 8 million registered users on Kaggle.

One of the sub-platforms that made Kaggle such a popular resource is their competitions. In a similar way that Hacker Rank plays that role for software developers and computer engineers, “Kaggle Competitions” has significant importance for data scientists; you can learn more about them “Kaggle competition Guide”  and learn how to analyse a dataset step-by-step in our “Kaggle Competition Tutorial”. In data science competitions like Kaggle’s or Date camp’s, companies and organizations share a big amount of challenging data science tasks with generous rewards in which data scientists, from beginners to experienced, compete on their completion. Kaggle also provides the Kaggle Notebook, which, just like Date camp workspace, allows you to edit and run your code for data science tasks on your browser, so your local computer doesn't have to do all the heavy lifting and you don't need to set up a new development environment on your own.





**DATA PREPROCESSING & FEATURE SELECTION**

Data preprocessing encompasses a series of steps aimed at cleaning, transforming, and preparing raw data for analysis and modelling. It involves handling missing values, dealing with categorical variables, scaling features, encoding data, and more. Data preprocessing involves transforming raw data into a clean and organized format suitable for machine learning algorithms. This includes handling missing values, encoding categorical variables, scaling numerical features, dealing with outliers, and splitting the data into training and testing sets. These steps ensure data quality and enhance model performance.

Feature selection is the process of selecting a subset of relevant features from the original set of features to improve model performance, reduce overfitting, and enhance interpretability. Effective feature selection leads to simpler, more interpretable models with reduced computational complexity. Feature selection is the process of identifying and selecting the most relevant features from the dataset. It aims to reduce dimensionality, improve model interpretability, and prevent overfitting. Techniques such as univariate feature selection, recursive feature elimination, feature importance ranking, and correlation analysis help identify informative features that contribute most to the target variable. By selecting the right features, unnecessary noise and irrelevant information are minimized, leading to more efficient and accurate models.

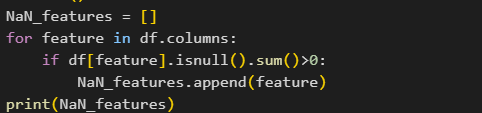


**Data Cleaning:**

Data cleaning is a crucial step in the data preprocessing phase, aimed at ensuring data quality, consistency, and accuracy for further analysis or modelling. It involves identifying and rectifying errors, inconsistencies, missing values, and outliers in the dataset. Cleaning the data, data scientists ensure that the dataset is ready for analysis, modelling, and deriving meaningful insights. Effective data cleaning practices contribute to accurate results, improved model performance, and informed decision-making based on high-quality data.

**Filling Missing Values:**

Filling missing values refers to the process of replacing or estimating unknown or null data points within a dataset. This step is essential in data preprocessing to ensure completeness and accuracy before analysis or modelling. Techniques such as mean, median, or mode imputation, interpolation, forward/backward fill, K-nearest neighbour imputation, predictive modelling, and customized rules are commonly used to fill missing values based on data characteristics, patterns, and domain knowledge. Proper handling of missing values maintains data integrity, reduces bias, and enables meaningful insights from the dataset.



**Noisy Data:**

Noisy data refers to data that contains random or irrelevant information, errors, inconsistencies, or outliers that can distort or obscure meaningful patterns or relationships within the dataset. Noise can arise from various sources, such as measurement errors, data entry mistakes, sensor malfunctions, or sampling variability. It can manifest as outliers, incorrect values, duplicates, or missing values. Noisy data can negatively impact data analysis, modeling, and decision-making processes, leading to inaccurate results, biased interpretations, and reduced model performance. Data cleaning and preprocessing techniques are employed to identify, handle, and mitigate noise, ensuring data quality and reliability for meaningful insights.

**Removal of outliners:**

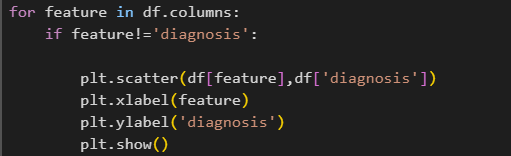
The removal of outliers refers to the process of identifying and excluding data points that significantly deviate from the majority of the dataset, based on statistical measures or domain-specific criteria. Outliers can distort statistical analyses and machine learning models, leading to biased results and reduced predictive accuracy. By removing outliers, data scientists aim to improve the robustness and reliability of their analyses, ensuring that the model focuses on the underlying patterns and trends within the dataset. However, caution should be exercised to avoid indiscriminate removal of data points, as outliers may sometimes contain valuable information or indicate genuine anomalies in the data.

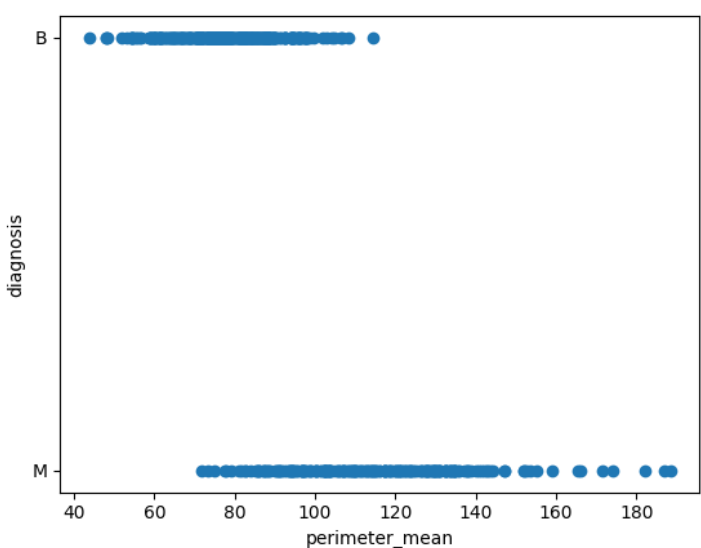
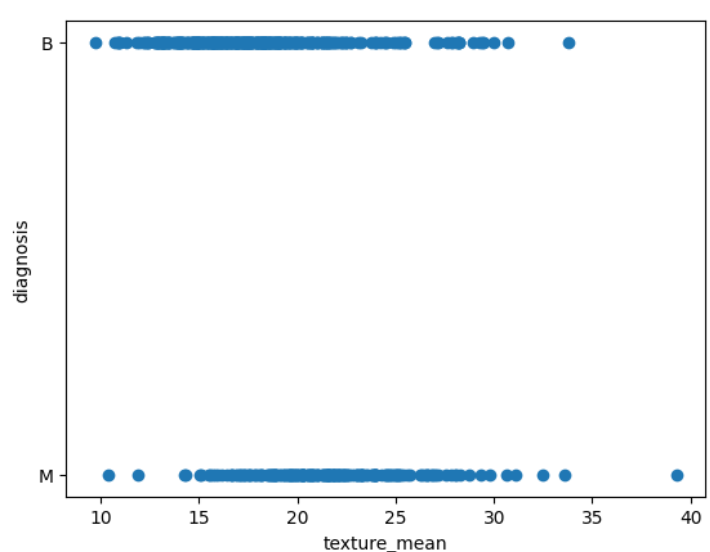
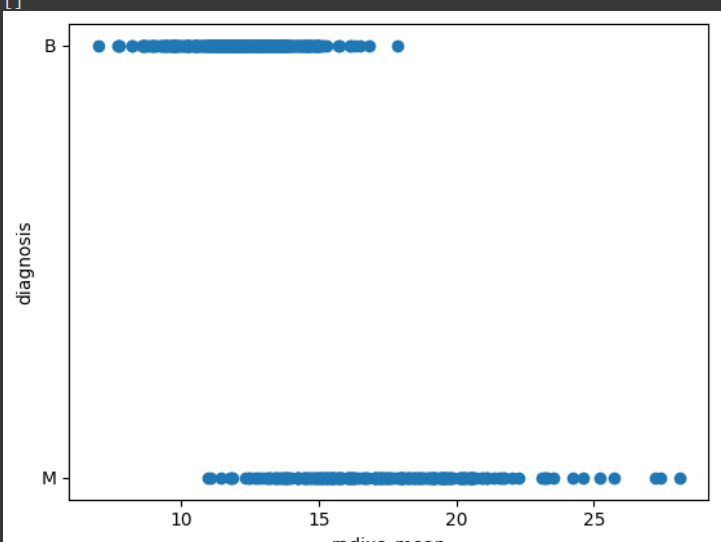
**Transforming categorical variables into numerical variables:**

Transforming categorical variables into numerical variables involves converting non-numeric categories or labels within a dataset into numerical representations that machine learning algorithms can process. This conversion enables algorithms to interpret and analyse categorical data effectively. Techniques like one-hot encoding, label encoding, target encoding, and ordinal encoding are commonly used for this purpose. One-hot encoding creates binary columns for each category, while label encoding assigns a unique numeric label to each category.

**DATA VISUALIZATION**

Data visualization is the graphical representation of data using charts, graphs, maps, and other visual elements to convey insights, patterns, and relationships within the data. It enhances understanding and interpretation by presenting complex information in a visually appealing and intuitive format. Data visualization enables stakeholders to identify trends, anomalies, and correlations, aiding in decision-making, problem-solving, and communication of findings. It allows for the exploration of large datasets, identification of outliers, and comparison of variables. Effective data visualization techniques include bar charts, line graphs, scatter plots, heatmaps, and interactive dashboards, transforming raw data into actionable insights.





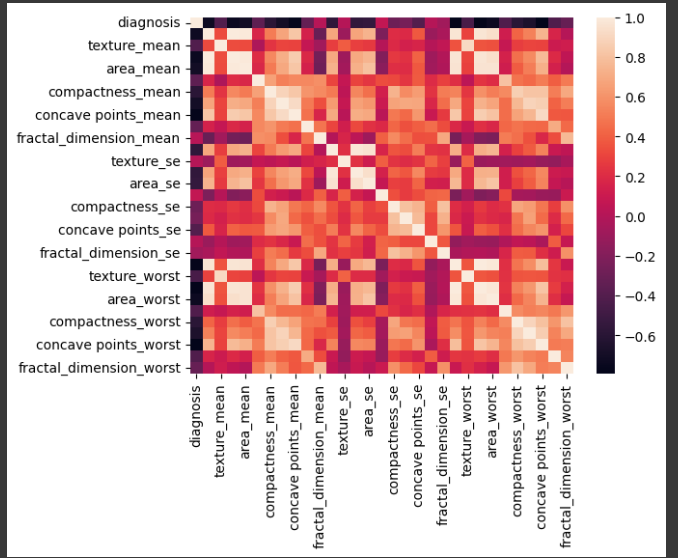
**Bar Chart:**

A bar chart is a graphical representation of data that uses rectangular bars to display the values of variables or categories. The length or height of each bar corresponds to the numerical value it represents, making it easy to compare quantities across different categories. Bar charts are commonly used to illustrate comparisons, trends, or distributions in data. They are effective for displaying categorical data or discrete values, such as comparing sales figures for different products, showing frequency counts in a survey, or visualizing the distribution of data points in different groups. Bar charts can be vertical (column chart) or horizontal (bar chart), depending on the orientation of the bars

**Heat Map:**

A heatmap is a graphical representation of data that uses colors to depict the magnitude of values in a matrix or table. Each cell in the heatmap is assigned a color based on its numeric value, allowing viewers to quickly identify patterns, trends, or relationships within the data. Heatmaps are commonly used to visualize correlations, distributions, or density of data points across multiple variables or categories. They are especially effective for large datasets and complex data structures, providing a visual summary that helps in decision-making, analysis, and communication of insights. Heatmaps can be customized with color schemes, legends, and annotations to enhance interpretation and understanding of the data.





**Histogram:**

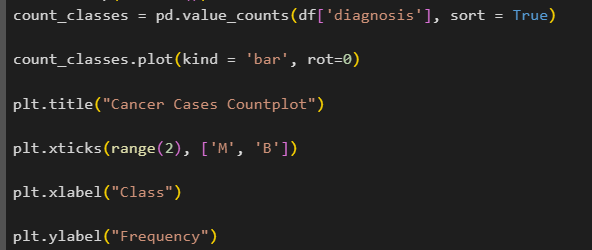
A histogram is a type of graphical representation that displays the distribution of numerical data through bars of varying heights. The bars are grouped into intervals or bins, with each bar representing the frequency or count of data points falling within that interval. Histograms are used to visualize the frequency distribution of continuous or discrete data, providing insights into the central tendency, spread, and shape of the data distribution. They are particularly useful for identifying patterns, outliers, and data skewness. Histograms are commonly used in statistical analysis, data exploration, and hypothesis testing to understand the underlying characteristics of a dataset and make data-driven decisions.

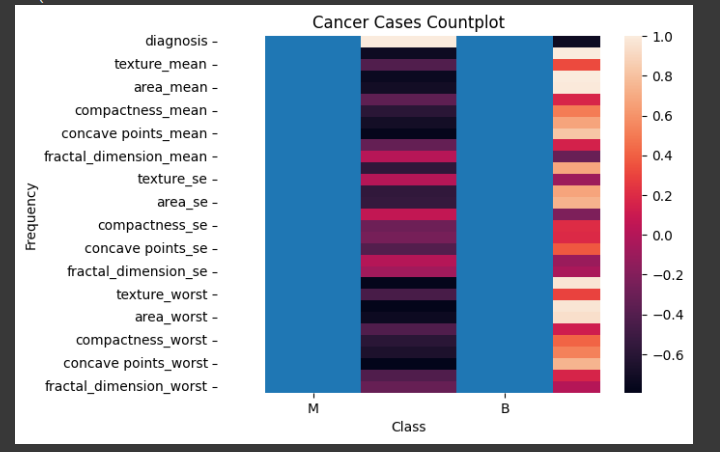
**Pie Chart:**

A histogram is a type of graphical representation that displays the distribution of numerical data through bars of varying heights. The bars are grouped into intervals or bins, with each bar representing the frequency or count of data points falling within that interval. Histograms are used to visualize the frequency distribution of continuous or discrete data, providing insights into the central tendency, spread, and shape of the data distribution. They are particularly useful for identifying patterns, outliers, and data skewness. Histograms are commonly used in statistical analysis, data exploration, and hypothesis testing to understand the underlying characteristics of a dataset and make data-driven decisions.

**Tree Map:**

A tree map is a type of data visualization that represents hierarchical data using nested rectangles within a larger rectangular space. Each rectangle in the tree map represents a category or subgroup, and its size corresponds to a quantitative value such as frequency, proportion, or numerical magnitude. The tree map's hierarchy is visualized through the nesting of rectangles, with larger rectangles representing higher-level categories and smaller rectangles representing lower-level subcategories or data points. Tree maps are effective for visualizing the relative proportions and contributions of different categories within a dataset, allowing viewers to quickly grasp the distribution and relationships between data elements.

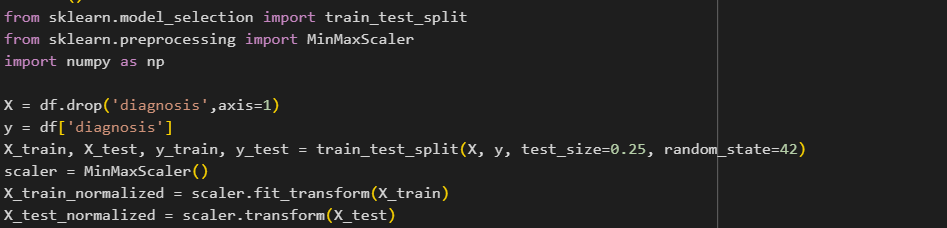
****

****

**SPLITTING AND TRAINING THE DATA**

Data splitting is a fundamental step in machine learning and data analysis, involving the division of a dataset into multiple subsets for various purposes such as training, validation, and testing. The main goal of data splitting is to ensure that machine learning models are developed, fine-tuned, and evaluated in a robust and unbiased manner.

Training the data refers to the process of using a machine learning algorithm to learn patterns, relationships, and structures from a labelled dataset (training set). This process is a crucial step in developing predictive models that can make accurate predictions or classifications on new, unseen data.

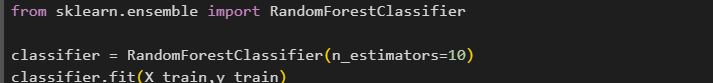
**Vll LOAD THE MODEL**

**Vll fit the training data into the model**

Fitting the training data into a machine learning model involves training the model using the labelled training dataset. This process helps the model learn patterns, relationships, and decision boundaries from the data so that it can make predictions or classifications on new, unseen data.

**thn Vll transform the training data**

After fitting the training data into a machine learning model, the next step is often to transform the training data. However, the need for transformation depends on the specific preprocessing steps required for your model and data.



**EVALUATING THE MODEL:**

The Random Forest Classifier achieved a certain accuracy on the resampled testing data.

The classification report provides metrics such as precision, recall, F1-score, and support for both classes ('M' and 'B').

The confusion matrix heatmap visually represents the model's true positive, false positive, true negative, and false negative predictions.

To further improve the evaluation, you may consider:

Fine-tuning hyperparameters of the RandomForestClassifier for better performance.

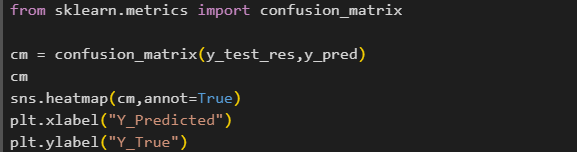
Exploring other evaluation metrics and techniques like cross-validation to ensure robustness.

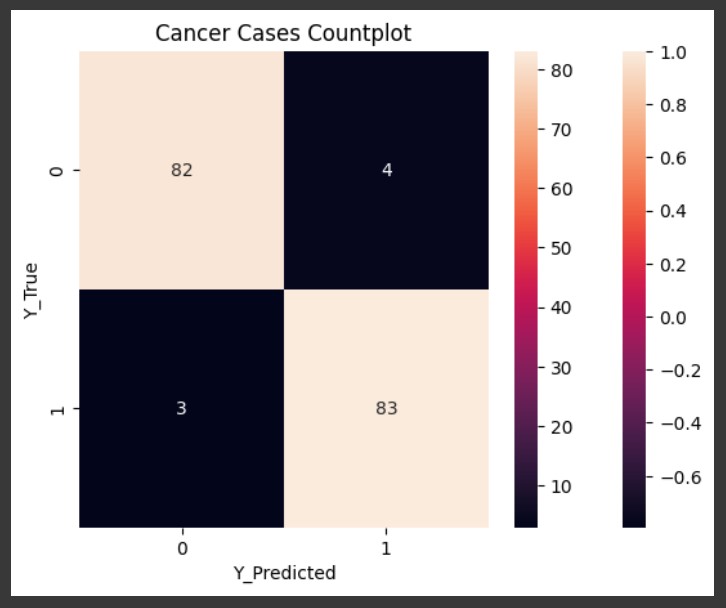
Visualizing additional metrics or using ROC curves for binary classification evaluation.

Overall, the evaluation provides insights into the model's ability to classify cancer cases as malignant ('M') or benign ('B'), highlighting its strengths and potential areas for improvement.



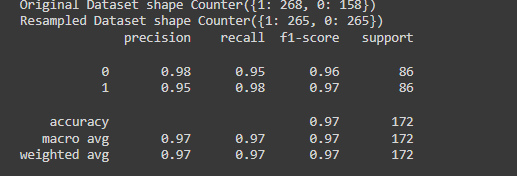
**Confusion Matrix**

****



**Classification Report**

****

****