Enhancing Breast Tumor Diagnosis: A Comprehensive Analysis

Data Science II

COSC 4337

Submitted to

Dr. Ricardo Vilalta

Ly Ha 1920058

Erica Hay 2049545

Khuong Ngo 1857210

The diagnosis of tumors has traditionally been performed by a full biopsy, an invasive surgical procedure. Other procedures, such as fine needle aspirations, provide less intrusive methods to examine the tumor tissue. However, the results from these procedures are not always met with success. By examining the characteristics of cells and contextual features, there may be a way to increase the correctness of the diagnosis process by figuring out which features are correlated with malignancy.

Thus, research was conducted at the University of Wisconsin in 1992, jointly in the departments of Computer Science and Surgery to find such a correlation. The primary objective of this study was to diagnose breast tumors using interactive image processing techniques and linear programming-based inductive classification. A total of 569 images were analyzed during the research. Various combinations of features were tested to determine the most effective in distinguishing between benign and malignant tumor samples. The, now named, Breast Cancer Wisconsin data set can be found and is sourced from the[**UC Irvine Machine Learning Repository**](https://archive.ics.uci.edu/dataset/17/breast+cancer+wisconsin+diagnostic).

When taking a closer look at the data set, we have twelve main features. The data set is introduced by two attribute information features: ‘id’ and ‘diagnosis’. Then, ten real-valued features are computed for each cell nucleus. Below is a table that demonstrates each column, their data type, and its description:

|  |  |  |
| --- | --- | --- |
| **Column** | **Data Type** | **Description** |
| ID | int | Unique identification number for each cell nucleus |
| Diagnosis | str | Malignant (M) or Benign (B) tumor diagnosis |
| Radius | float | Mean distance from the center to points on the perimeter |
| Texture | float | Standard deviation of gray-scale values |
| Perimeter | float | Perimeter of the cell nucleus |
| Area | float | Area of the cell nucleus |
| Smoothness | float | Local variation in radius lengths |
| Compactness | float | Perimeter2 area - 1.0 |
| Concavity | float | Severity of concave portions of the contour |
| Symmetry | float | Symmetry of the cell nucleus |
| Fractal Dimension | float | coastline approximation - 1 |

Furthermore, there are 30 columns of the real-valued features due to the addition of ‘mean’, ‘standard error’, and ‘worst’ (mean of the three largest values) variables for each of the features listed in the table above. Meaning, each cell has a column for each of the ten features times the three variables. I.e. ‘radius\_mean’, ‘radius\_se’, and ‘radius\_worst’.

To refine our dataset and gain deeper insights, a pivotal first step involves engaging in both data cleaning and exploratory data analysis. Key steps in data cleaning, such as handling missing values, addressing duplicates and outliers, standardizing data, correcting typos and inconsistencies, dealing with data formats, addressing categorical variables, ensuring data integrity, handling data skewness, and validating data quality, collectively enhance the overall quality, accuracy, and reliability of our data. By conducting exploratory data analysis, we can systematically uncover patterns, relationships, and trends within the refined dataset. This approach contributes to a more comprehensive understanding, fostering informed decision-making and laying a solid foundation for subsequent analyses or modeling methods.

In executing the essential steps of data cleaning, we started by generating a bar chart **(Figure 3)** to visualize the distribution of values across each variable. This graphical representation provided us with an immediate insight into any missing values within the dataset. Upon careful examination of the chart, a notable observation emerged: all variables, with the exception of the one labeled ‘Unnamed: 32’, exhibited a consistent count of 569 values. However, the variable ‘Unnamed: 32’ contained no values, rendering it redundant for our analysis. Consequently, we have opted to eliminate this unnecessary variable from the dataset. To address the issue of duplicates, we employed the built-in “duplicate” function, which revealed the absence of any duplicate entries. Further refinement was accomplished through the utilization of the “head” function **(Figure 4)** to display the initial five observations in our dataset. This enabled us to review the values of each variable for any inconsistencies or typos. During this inspection, we noted the existence of an ‘id’ variable, which we determined unnecessary for further analysis, prompting its removal. Furthermore, our examination through the “head” function revealed that the ‘diagnosis’ variable comprised two distinct categorical values: ‘M’ and ‘B’, as stated earlier on. To ensure consistency in our dataset, we opted to replace the ‘M’ values with 1 and the ‘B’ values with 0, facilitating a standardized approach to our subsequent analyses.

A screenshot of a computer

Description automatically generated

*Figure 3: Bar plot displaying the distribution of values to identify missing values.*

A screenshot of a computer

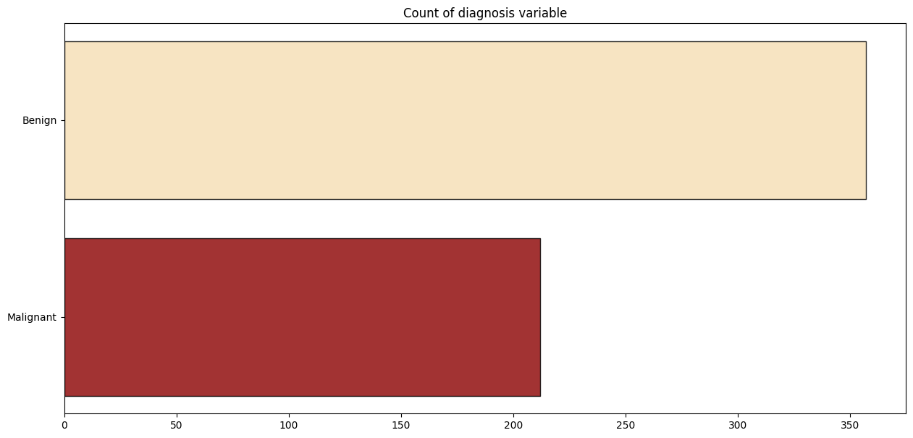
Description automatically generated

A screenshot of a computer

Description automatically generated

*Figure 4: Output from the built-in function data.head().*

Transitioning into exploratory data analysis, it remains paramount to recognize that the insights obtained serve as a guide for subsequent steps in our analysis. These findings assist us in discerning whether feature selection, feature extraction, or classification procedures are warranted. As we unveil patterns, trends, and potential relationships within the data, we are better equipped to tailor our approach, ensuring it aligns with the characteristics of our dataset. We begin by producing a comprehensive chart presenting key statistical measures for each variable, including the count, mean, standard deviation, minimum value, 25th percentile, median (50th percentile), 75th percentile, and maximum value. This representation provides an insightful snapshot of the distribution and central tendencies across our dataset. Building on this newfound insight, a more in-depth exploration of the data is essential for a nuanced understanding. This prompts a shift in attention toward the target variable, ‘diagnosis’. We delve into the distribution of this variable by creating a bar plot **(Figure 5)** illustrating the count (malignant = 212, benign = 357), and a pie chart **(Figure 6)** unveiling the percentage distribution (malignant = 37.3%, benign = 62.7%). Observing this distribution, it becomes apparent that the data exhibits a moderate imbalance, with the benign class prevailing. Recognizing the dataset as a binary classification problem, this insight guides us in selecting an optimal algorithm to navigate and interpret the data effectively. Among the potential candidates are logistic regression, random forest classifier, and ensemble classifier, each tailored to handle the intricacies of our dataset with precision.



A diagram of a number of patients

Description automatically generated

*Figure 5: Bar Plot for Target Count                       Figure 6: Pie Chart for Target Percentage*

Extending our exploration from the distribution of the target variable, our focus now turns to the visualization of predictor variables based on the target variable, employing density plots **(Figure 7)**. These plots serve as a lens through which we examine central tendencies, shape, spread, and overlap. By evaluating the degree of overlap and spread in class distributions concerning predictor variables, we gain valuable insights into feature utility for classification algorithms. Features revealing substantial overlap in distribution are flagged for potential limited discriminatory power. A closer inspection of the plots shines a spotlight on specific attributes – ‘fractal\_dimension\_mean’, ‘texture\_se’, ‘smoothness\_se’, ‘symmetry\_se’, and ‘fractal\_dimension\_se’. These features exhibit pronounced overlap between both classes, signaling a thoughtful consideration for their inclusion in classification algorithms.

A graph of a function

Description automatically generated with medium confidenceA diagram of a normal plot

Description automatically generated

A graph of a function

Description automatically generatedA diagram of a function

Description automatically generated

*Figure 7: Density plots for features’ distribution.*

Alongside assessing the overlap between classes, another critical aspect demanding consideration is multicollinearity. This phenomenon manifests when variables exhibit high, or even perfect, correlation with each other. Such correlation indicates redundancy in the information provided by these variables, leading to potential consequences, including numerical instability and diminished model interpretability. To address this potential issue, we employed a heat map representation **(Figure 8)** of the correlation matrix, providing us with a better understanding of the relationships between variables. Leveraging the accompanying legend, we can readily identify variables that exhibit high positive or negative correlation. In this case, our heat map showcased that many variables had a correlation coefficient close to or even equal to 1.0. This means that these have a high correlation, however, that does not give enough information to show us if these variables should be deemed redundant.

A colorful squares with black and white text

Description automatically generated

*Figure 8: Heat map visualizing the correlation matrix.*

Moving forward, we created a series of scatterplots to show and understand the relationships between different pairs of features. We split these charts **(Figure 9)** into three categories: features with positive correlation, negative correlation, or no correlation. In the positive category, we have ‘perimeter\_mean v. radius\_worst’, ‘area\_mean v. radius\_worst’, ‘texture\_mean v. texture\_worst’, and ‘area\_worst v. radius\_worst’. For the most part, these scatterplots are tightly clustered as they present a perfect correlation coefficient of 1.0. From these plots, we can observe that as both features increase in value, the chance of the cell having a malignant diagnosis also increases. In contrast with the other maps in this category, the ‘texture\_mean v. texture\_worst’ plot carries a slightly loose clustering meaning that there is more variability. However, it still showcases the same relationship as the others. When comparing with the scatter plots illustrating variables with positive correlation, it becomes evident that variables exhibiting the highest negative correlation within the dataset only reach a correlation coefficient of -0.3. Visually, this manifests as a discernibly more scattered arrangement in the scatter plot compared to its positively correlated counterparts. Despite the increased dispersion, the negative correlation scatter plots unmistakably reveal a consistent negative linear relationship between the two variables depicted in their respective plots. As mentioned earlier, the presence of perfect correlation may impose potential issues. Given that the positive correlation plots exhibit such a phenomenon, thoughtful consideration should be given to the possibility of removing one of the variables involved. Doing so will enhance the interpretability and stability of our analytical framework.

A screenshot of a graph

Description automatically generated

A screenshot of a graph

Description automatically generated

*Figure 9: Scatterplots from the Positive and Negative Correlation Categories*

After cleaning the data by removing unnecessary variables and performing exploratory data analysis, we conduct feature engineering on the dataset. Feature engineering is another crucial stage in data preprocessing, as it transforms raw data into informative features that improve the performance of the learning model. In the context of using the Breast Cancer Wisconsin dataset to diagnose breast tumors, feature engineering involves feature selection, transforming existing features, dimensionality reduction, and selecting the top features for predictive modeling.

In the first step of feature engineering, we computed mutual information scores for each feature with respect to the target variable, ‘diagnosis’. Similar to correlation values, mutual information scores help quantify the relationship between the features and the target variable, where higher mutual information scores are more informative and indicate that the feature is significant in predicting the target variable. However, mutual information scores do not have a fixed upper limit and may range from 0 to infinity. This characteristic allows mutual information scores to capture both linear and non-linear relationships between variables. However, it's important to note that low mutual information scores may be misleading without context. Therefore, it's essential to consider the specific characteristics of the dataset when interpreting mutual information scores.

Computing each mutual information score was done using the ‘mutual\_info\_regression’ function, which determines the dependency between two variables. The scores were then plotted in descending order **(Figure 10)**. The barplot of the mutual information scores reveals that ‘perimeter\_worst’, ‘area\_worst’, ‘radius\_worst’, ‘concave points\_mean’, and ‘concave points\_worst’ contain the highest mutual information scores amongst the features. With values all above 0.4, these features indicate a higher potential importance in diagnosing breast tumors. The top ten features shown in **Figure 10** will be heavily considered in feature selection.

A graph of a number of scores

Description automatically generated with medium confidence

*Figure 10: Bar plot displaying the mutual information scores of each feature with respect to ‘diagnosis’.*

Prior to performing principal component analysis, we standardize the dataset using the ‘StandardScaler’ function to ensure that all features have a mean of 0 and a standard deviation of 1. Principal component analysis is then performed on the standardized feature matrix to transform the original features of the dataset into a set of uncorrelated principal components. The best number of components to retain is determined by the ‘plot\_variance’ function, which visualizes the explained variance ratio of each principal component and the cumulative explained variance. **Figure 11** illustrates the stabilization of the explained variance ratio for every additional principal component after two. The cumulative variance graph also shows that two principal components capture a significant amount of variation in the data. Based on these insights, the choice of two principal components was deemed adequate for summarizing the dataset while retaining most of its variability.

**A comparison of a normal and a normal varities

Description automatically generated with medium confidence**

*Figure 11: Explained Variance and Cumulative Variance Ratio Analysis plots.*

The features selected for each principal component were determined by computing the absolute loading values obtained from PCA. The top three features were selected for each principal component, resulting in the following subsets:

* PC1: concave points\_mean, concavity\_mean, concave points\_worst
* PC2: fractal\_dimension\_mean, fractal\_dimension\_se, fractal\_dimension\_worst

The subset of features is standardized and PCA is applied again, and the transformed data is then visualized using a scatter plot **(Figure 12)**, where each point represents a sample in the dataset and its position is determined by principal component values 1 and 2. The visualization of the PCA helps us understand the relationships between the samples based on their feature values. The explained variance ratios of 44% and 19% for PC1 and PC2 show the proportion of variance in the original data that is captured by each principal component. These values indicate that PC1 summarizes a larger portion of the 63% variability present in the dataset compared to PC2.

A screen shot of a graph

Description automatically generated

*Figure 12: Scatter plot showing the visualization of the dataset after performing Principal Component Analysis (PCA) on the top three features for each component.*

The dissimilarity between the top ten features computed using mutual information scores and the features chosen for each principal component capture different aspects of the data. While the top three features for each principal component focus on dimensionality reduction and on features that contribute most to the variability captured, the top ten features computed using mutual information scores prioritize features with the highest predictive power for diagnosing breast cancer, regardless of their contribution to dimensionality reduction. This emphasizes the importance of the multifaceted nature of feature selection in breast cancer diagnosis. Integrating these approaches allows us to take advantage of the strengths of each method while gaining a comprehensive understanding of the dataset.

Overall, the process of data cleaning and exploratory data analysis has improved the quality and understanding of the Breast Cancer Wisconsin data set. Cleaning processes and EDA gave us critical insights that provide a foundation for breast tumor diagnosis analyses. Moreover, feature engineering and the utilization of principal component analysis transformed the raw data into more informative features. Employing mutual information scores prioritized predictive features, while PCA reduced dimensionality. The integration of these approaches emphasizes a multifaceted feature selection strategy, ensuring a comprehensive understanding and facilitating the development of accurate data models in medical diagnostics.