**Project Overview – breast cancer diagnosis**

The key question to answer is: "Can we accurately classify tumors as benign or malignant based on features derived from imaging data?"

We know that breast cancer diagnosis is critical for early detection and treatment. The dataset includes various features (e.g., radius\_mean, texture\_mean, perimeter\_mean) that are related to the physical characteristics of the tumor, which are crucial for determining whether a tumor is benign or malignant.

The desired outcome is a model that can accurately predict the diagnosis (benign or malignant) based on the provided features.

Factors influencing the diagnosis include the size and shape of the tumor (e.g., radius\_mean, area\_mean), and the texture of the tumor (e.g., texture\_mean). These are key features known to be associated with malignancy.

.

**Data Preparation**

The breast cancer dataset consists of 569 observations and 30 continuous predictor variables.

The binary outcome variable denotes the tumor's malignancy status.

We can proceed directly to the analysis phase, as the data is clean and complete. All columns contain continuous numerical values.

**Exploratory data analysis (EDA)**

The EDA focuses on understanding the distribution of various features in the dataset and how they differ between benign and malignant tumors. The analysis uses skewness to identify parametric and non-parametric features, applies statistical tests (T-tests and Wilcoxon rank-sum tests), and visualizes the data distributions.

1. **Parametric Features:**

Features such as smoothness\_mean, texture\_worst, and concave points\_worst show significant differences between benign and malignant tumors. These features are likely to be critical for developing the classification model.

1. **Non-Parametric Features:**

Despite their skewed distributions, non-parametric features like radius\_mean, area\_mean, and perimeter\_mean also exhibit significant differences between the two groups, confirming their importance in diagnosis.

1. **Skewness and Distribution:**

High skewness values indicate that many features are not normally distributed.

Therefore, in the next steps, I need to conduct tests and select models that are robust to nonparametric data.

**Data Cleansing**

**Missing Values**

### The dataset is clean, with no missing data. All variables are continuous numerical and do not necessitate normalization. Outlier detection and handling will be the next step

**Outliers for parametric distribution**

### Identify outliers using the Z-score method For all the columns analyzed the decision was made not to remove the outliers because they do not significantly affect the distribution or the correlation with the target variable.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **feature** | **Percentage of Outliers (%)** | **KS test p-value**  **(P<0.05 distribution has changed)** | **Correlation Change** | **Fill NaN?** |
| smoothness\_mean | 1.58 | 0.999999974 | 0.0001 | FALSE |
| texture\_worst | 1.41 | 0.999999973 | 0.0009 | FALSE |
| smoothness\_worst | 1.23 | 1 | 0.0028 | FALSE |
| concave points\_worst | 0.53 | 1 | 0 | FALSE |

### **Outliers for nonparametric distribution**

### Identifies outliers using the Interquartile Range (IQR) method for non- parametric For all the columns analyzed the decision was made not to remove the outliers because they do not significantly affect the distribution or the correlation with the target variable.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **column** | **Percentage of Outliers (%)** | **KS test p-value**  **(P<0.05 distribution has changed)** | **Correlation Change** | **Fill NaN ?** |
| radius\_mean | 2.46 | 0.9934 | 0.0141 | FALSE |
| texture\_mean | 1.23 | 1 | 0.0028 | FALSE |
| perimeter\_mean | 2.28 | 0.9975 | 0.0121 | FALSE |
| area\_mean | 4.39 | 0.6335 | 0.0265 | FALSE |
| compactness\_mean | 2.81 | 0.973 | 0.0249 | FALSE |
| concavity\_mean | 3.16 | 0.9299 | 0.0007 | FALSE |
| concave points\_mean | 1.76 | 0.9999 | 0.0079 | FALSE |
| symmetry\_mean | 2.64 | 0.9935 | 0.0083 | FALSE |
| fractal\_dimension\_mean | 2.64 | 0.9857 | 0.0027 | FALSE |
| radius\_se | 6.68 | 0.1623 | 0.0586 | FALSE |
| texture\_se | 3.51 | 0.8628 | 0.0113 | FALSE |
| perimeter\_se | 6.68 | 0.1623 | 0.0623 | FALSE |
| area\_se | 11.42 | 0.0017 | 0.094 | FALSE |
| smoothness\_se | 5.27 | 0.4056 | 0.044 | FALSE |
| compactness\_se | 4.92 | 0.4914 | 0.0171 | FALSE |
| concavity\_se | 3.87 | 0.7775 | 0.0275 | FALSE |
| concave points\_se | 3.34 | 0.8991 | 0.0041 | FALSE |
| symmetry\_se | 4.75 | 0.5375 | 0.0525 | FALSE |
| fractal\_dimension\_se | 4.92 | 0.4914 | 0.0247 | FALSE |
| radius\_worst | 2.99 | 0.9546 | 0.0132 | FALSE |
| perimeter\_worst | 2.64 | 0.9857 | 0.011 | FALSE |
| area\_worst | 6.15 | 0.2349 | 0.0305 | FALSE |
| compactness\_worst | 2.81 | 0.973 | 0.0251 | FALSE |
| concavity\_worst | 2.11 | 0.9993 | 0.0016 | FALSE |
| symmetry\_worst | 4.04 | 0.7306 | 0.0528 | FALSE |
| fractal\_dimension\_worst | 4.22 | 0.6823 | 0.0316 | FALSE |

**Conclusion:**  
 Model Robustness: This suggests that your model may be robust to the presence of  
 these outliers, or that the outliers themselves might represent valuable variations in the data that   
 are worth keeping

**Feature engineering**

## Adding new features:

In this dataset, there are various measurements related to breast cancer diagnosis.

For each type of measurement, there are three columns: mean value, worst value, and SE value.

For each such set of columns, we added 4 calculated columns based on the existing three columns.

The following is a detailed explanation of the 4 columns created for each measurement:

**\_mean\_to\_worst\_ratio:**

Calculation: Divides the mean of the feature by the sum of the worst-case value. This ratio indicates how much the mean deviates from the worst-case value

**\_se\_to\_mean\_ratio:**

Calculation: Divides the standard error of the feature by the mean). Interpretation: This ratio can be used to assess relative variability.

**\_worst\_mean\_diff:**

Calculation: Subtracts the mean from the worst-case value. Interpretation: This difference directly shows how much the worst-case value deviates from the average.

**\_z\_score\_worst":**

Calculation: Calculates the z-score for the worst-case value, which measures how many standard deviations away from the mean it is. Interpretation: A higher z-score indicates that the worst-case value is further away from the mean in terms of standard deviations.

[5]:

**Feature selection**

## Multivariable feature selection

At this stage, I want to filter out columns that do not contribute and reach 30 columns out of 70

using multivariable selection with the following models:

Lasso, Ridge, SVM, GradientBoost, RandomForest, XGBoost

### results

The models consistently favored the original features.

More or equal 5: This category contains 11 columns that have a score of 5 or higher, indicating they are considered relatively important.

More or equal 4: This category contains 48 columns with a score of 4 or higher, including those from the previous category.

**The goal of reducing the number of columns to 30 has not been achieved using this method.**

**We will explore a univariate approach where each feature's relationship with the target variable is assessed**

Univariable feature selection

Since this method examines each column individually in relation to the target column, we must first assess the distribution of each column. Based on the distribution, we will employ the appropriate statistical test.

For columns with a **normal distribution, we will use a t-test.**

For columns with a **non-normal distribution, we will use the Wilcoxon Rank-Sum test.**

### results

Engineered features:

Our univariate analysis showed that the engineered features did not improve the model's ability to classify the target variable as effectively as the original features.

# In summary

Removed the 'id' column as it was not relevant for prediction.

Removed calculated columns

**Started with 32 original columns**

**We are left with 31 columns for model building.**

**Model Selection**

## When dealing with non-normally distributed data, it's crucial to choose statistical methods that are robust to such deviations. Machine learning algorithms that are less sensitive to data distribution, such as decision trees, random forests, or support vector machines, are often appropriate. Additionally, ensemble methods like bagging or boosting can improve model performance and robustness.

## Therefore, I've decided to evaluate the performance of the following classification models: SVC, XGBoost, Gradient Boosting, and Random Forest

## For each model, it prints the confusion matrix and the classification report, which provides precision, recall, F1-score, and support for each class.

**Model selection results**

XGBoost has the highest overall accuracy (0.98) and performs consistently well across all metrics (precision, recall, F1-score). It's the most balanced and robust model among the four.

SVM has slightly lower recall for Class 1, which could be a concern if capturing all positives in this class is crucial.

Gradient Boosting and Random Forest are also strong contenders, both achieving high accuracy and balanced precision and recall. However, they fall slightly short of XGBoost in overall performance.

Continue to the next step of hyperparameter tuning with XGBoost and SVC

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Model** | **Precision (0)** | **Recall (0)** | **F1-Score (0)** | **Precision (1)** | **Recall (1)** | **F1-Score (1)** | **Accuracy** |
| **SVM** | 0.91 | 1 | 0.95 | 1 | 0.83 | 0.9 | 0.94 |
| **XGBoost** | 0.98 | 0.99 | 0.99 | 0.98 | 0.97 | 0.98 | 0.98 |
| **Gradient Boosting** | 0.96 | 0.97 | 0.97 | 0.95 | 0.94 | 0.94 | 0.96 |
| **Random Forest** | 0.96 | 0.99 | 0.98 | 0.98 | 0.94 | 0.96 | 0.97 |

### **Hyperparameters tuning**

Defines Hyperparameter Grids: Specifies ranges of hyperparameters for SVC and XGBoost to be tuned.

Uses GridSearchCV: To find the best hyperparameters by evaluating all possible combinations using cross-validation.

Evaluates Models: After tuning, the models are evaluated on the test set, and results are printed, including confusion matrices and classification reports, to understand their performance.

**Tuning result:**

XGBoost (not tuned) shows the best overall performance

High accuracy, precision, recall, and F1-scores for both classes. It appears to generalize well and could be the preferred model if you want high performance across the board.

The tuned XGBoost performs slightly worse than the not-tuned version, which suggests that the tuning might not have been effective

Tuned XGBoost and Tuned SVC both offer good performance but with slightly different strengths and weaknesses.  
Depending on your focus (e.g., precision vs. recall), you might choose between these models.

SVC (not tuned), less reliable due to lower recall.

In a medical context where detecting malignant cases accurately is crucial, XGBoost or Tuned SVC might be preferable due to their higher recall for malignant cases.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **del** | **Precision (0)** | **Recall (0)** | **F1-Score (0)** | **Precision (1)** | **Recall (1)** | **F1-Score (1)** | **Accuracy** |
| **Tuned XGBoost** | 0.97 | 0.96 | 0.97 | 0.94 | 0.95 | 0.94 | 0.96 |
| **XGBoost** | 0.98 | 0.99 | 0.99 | 0.98 | 0.97 | 0.98 | 0.98 |
| **Tuned SVC** | 0.95 | 0.98 | 0.96 | 0.97 | 0.9 | 0.93 | 0.95 |
| **SVC** | 0.91 | 1 | 0.95 | 1 | 0.83 | 0.9 | 0.94 |

**CROSS-VALIDATION**

**Models:**

XGBoost (Tuned and Not Tuned) SVC (Tuned and Not Tuned)

**Conclusion:**

XGBoost:

Both tuned and not-tuned models show excellent performance, with high mean accuracy scores and relatively low variability. Tuning had a minor impact on the performance of XGBoost in this case. Both versions of XGBoost are performing similarly well.

SVC:

Tuning the SVC model significantly improved its performance. The not tuned model had much lower accuracy compared to the tuned version. Performance variability is higher in the tuned SVC model but still maintains a good average accuracy. Recommendations XGBoost: Both versions perform well. If further improvement is needed, consider exploring additional hyperparameter tuning or advanced techniques such as feature engineering or ensemble methods.

SVC: The tuned SVC model is significantly better than the not tuned version. Continue using the tuned model for its improved performance. However, be mindful of the variability in performance and consider adjusting the hyperparameters further if necessary.

|  |  |  |  |
| --- | --- | --- | --- |
| **odel** | **Cross-Validation Scores** | **Mean CV Accuracy** | **Standard Deviation of CV Accuracy** |
| **XGBoost** | [0.9737, 0.9561, 0.9912, 0.9825, 0.9823] | 0.9772 | 0.0119 |
| **XGBoost TUNED** | [0.9649, 0.9649, 0.9912, 0.9737, 0.9735] | 0.9736 | 0.0096 |
| **SVC** | [0.6228, 0.6228, 0.6316, 0.6316, 0.6283] | 0.92968 | 0.01577 |

# **Selected model**

## XGBoost consistently outperformed other models, making it the preferred choice

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **del** | **Precision (0)** | **Recall (0)** | **F1-Score (0)** | **Precision (1)** | **Recall (1)** | **F1-Score (1)** | **Accuracy** |
| **XGBoost** | 0.98 | 0.99 | 0.99 | 0.98 | 0.97 | 0.98 | 0.98 |