Breast Cancer Diagnosis Classifier

University of Pittsburgh

Final Report for IS 0510 Data Analysis

Ajay Adhithiya Ohm Nathan

Abstract

Can a machine learning model accurately classify breast tumors as malignant or benign based on tumor features? This question is very interesting and raises many possibilities from it. In the real world, early and accurate diagnosis can save lives, and machine learning models can assist in this process. This final report will be useful in aiding medical professionals to make informed decisions. Doctors, researchers, and healthcare institutions would benefit from this model. My report involves data preprocessing, feature selection, model training, and evaluation. The key is to find a database that has hundreds to thousands of patients' information on the different features of the tumor, and its classification as Benign or Malignant.

Introduction

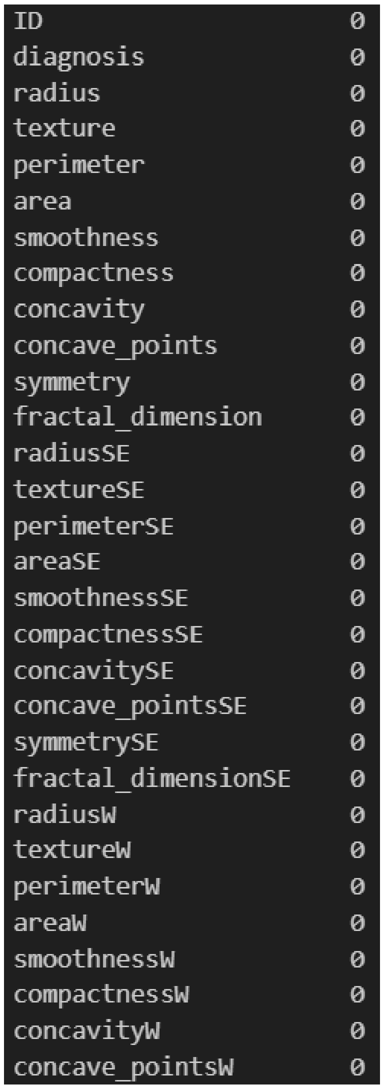
Breast cancer is a type of cancer that starts in the cells of the breast, most commonly affecting women, though men can also develop it. It occurs when abnormal cells in the breast grow uncontrollably, forming a tumor. Some tumors can be felt as lumps, while others are detected through imaging tests such as mammograms. There are various types of breast cancer, with the most common being invasive ductal carcinoma, which begins in the milk ducts and spreads to surrounding tissues. Other types include invasive lobular carcinoma and inflammatory breast cancer, which are often more aggressive forms.

Several factors can increase the risk of developing breast cancer, including gender, age, family history, and certain genetic mutations, such as BRCA1 and BRCA2. Lifestyle factors, such as smoking, excessive alcohol consumption, and obesity, also contribute to the risk. While breast cancer can affect anyone, women are much more likely to develop it, especially as they age. Early diagnosis is crucial, as it greatly improves the chances of successful treatment and survival. When breast cancer is detected at an early stage, it is more localized and has not yet spread, making it easier to treat effectively.

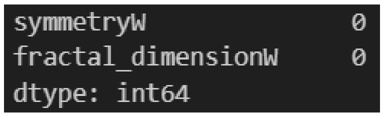
Early detection allows for less aggressive treatments, often resulting in fewer side effects and a better quality of life during and after treatment. It also reduces the risk of cancer spreading to other parts of the body, which can complicate treatment and decrease survival chances. By diagnosing the disease early, patients may be able to undergo less invasive procedures, such as a lumpectomy instead of a mastectomy, and avoid more intense therapies like chemotherapy. For this reason, regular screenings, such as mammograms, and self-exams are essential for identifying potential issues before symptoms appear. Genetic testing may also help individuals with a family history of breast cancer to assess their risk and make informed decisions about their health. With all this being said, can we do even better for early diagnosis of breast cancer? Yes, a machine learning model.

Methodology

In order to achieve an accurate model, we need data. Lots of data. I chose my dataset from the very reliable UC Irvine Machine Learning Repository. It is the Breast Cancer Diagnostic Dataset from Wisconsin from https://archive.ics.uci.edu/dataset/17/breast+cancer+wisconsin+diagnostic.[1] Once we download it as a .csv file, we can use pandas[2] library from Python[3] to load it as a dataframe. Then we check for missing values and print the number of missing values in each of the columns.



```
ID                     0
diagnosis              0
radius                 0
texture                0
perimeter              0
area                   0
smoothness             0
compactness            0
concavity              0
concave_points         0
symmetry               0
fractal_dimension      0
radiusSE               0
textureSE              0
perimeterSE            0
areaSE                 0
smoothnessSE           0
compactnessSE          0
concavitySE            0
concave_pointsSE       0
symmetrySE             0
fractal_dimensionSE    0
radiusW                0
textureW               0
perimeterW             0
areaW                  0
smoothnessW            0
compactnessW           0
concavityW             0
concave_pointsW        0
```

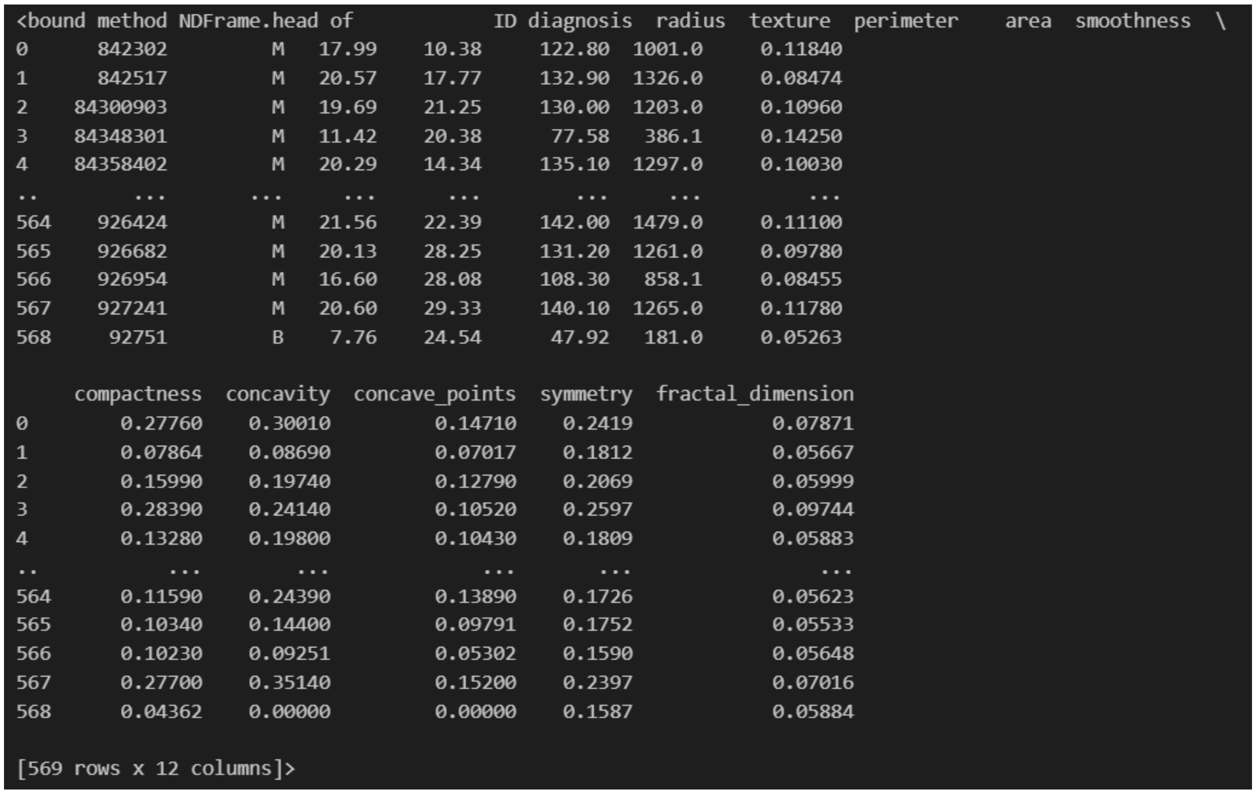


```
symmetryW              0
fractal_dimensionW     0
dtype: int64
```

Table Generated from printing pandas[2] isnull().sum() functionality on dataframe

Considering the table above, it is clear that the data set has no missing values. The columns that are generated here are the several features of a breast. We first have the ID of the patient, then their diagnosis (either M for Malignant or B for Benign). Then we are followed by 30 more columns. There is the radius, texture, perimeter, area, smoothness, compactness, concavity, concave_points, symmetry, and fractal_dimension. These 10 features are repeated twice more, but this time, we are given the standard error and the average of the three worst cases for each feature. Resulting in the last 20 columns.

I dropped the last 20 columns because it is just a repeat of the 10 features, except it ended with W and SE. Ending with W stands for the worst. It just means the largest 3 numbers of that column. And the SE column is just the standard error of each column. I decided to drop these, because for the machine learning model to work properly, we do not want extra numbers regarding the standard error. Also, we do not need the "W" column because that represents the worst cases, which will skew the data greatly. Therefore, we only want to work with raw data with the 10 original features to diagnose breast cancer.



```
<bound method NDFrame.head of           ID diagnosis  radius  texture  perimeter    area  smoothness  \
0      842302         M   17.99    10.38     122.80  1001.0     0.11840
1      842517         M   20.57    17.77     132.90  1326.0     0.08474
2    84300903         M   19.69    21.25     130.00  1203.0     0.10960
3    84348301         M   11.42    20.38      77.58   386.1     0.14250
4    84358402         M   20.29    14.34     135.10  1297.0     0.10030
..        ...       ...     ...      ...        ...     ...         ...
564    926424         M   21.56    22.39     142.00  1479.0     0.11100
565    926682         M   20.13    28.25     131.20  1261.0     0.09780
566    926954         M   16.60    28.08     108.30   858.1     0.08455
567    927241         M   20.60    29.33     140.10  1265.0     0.11780
568     92751         B    7.76    24.54      47.92   181.0     0.05263

     compactness  concavity  concave_points  symmetry  fractal_dimension
0        0.27760    0.30010         0.14710    0.2419            0.07871
1        0.07864    0.08690         0.07017    0.1812            0.05667
2        0.15990    0.19740         0.12790    0.2069            0.05999
3        0.28390    0.24140         0.10520    0.2597            0.09744
4        0.13280    0.19800         0.10430    0.1809            0.05883
..           ...        ...             ...       ...                ...
564      0.11590    0.24390         0.13890    0.1726            0.05623
565      0.10340    0.14400         0.09791    0.1752            0.05533
566      0.10230    0.09251         0.05302    0.1590            0.05648
567      0.27700    0.35140         0.15200    0.2397            0.07016
568      0.04362    0.00000         0.00000    0.1587            0.05884

[569 rows x 12 columns]>
```

Output after printing df.head() after dropping the unnecessary columns

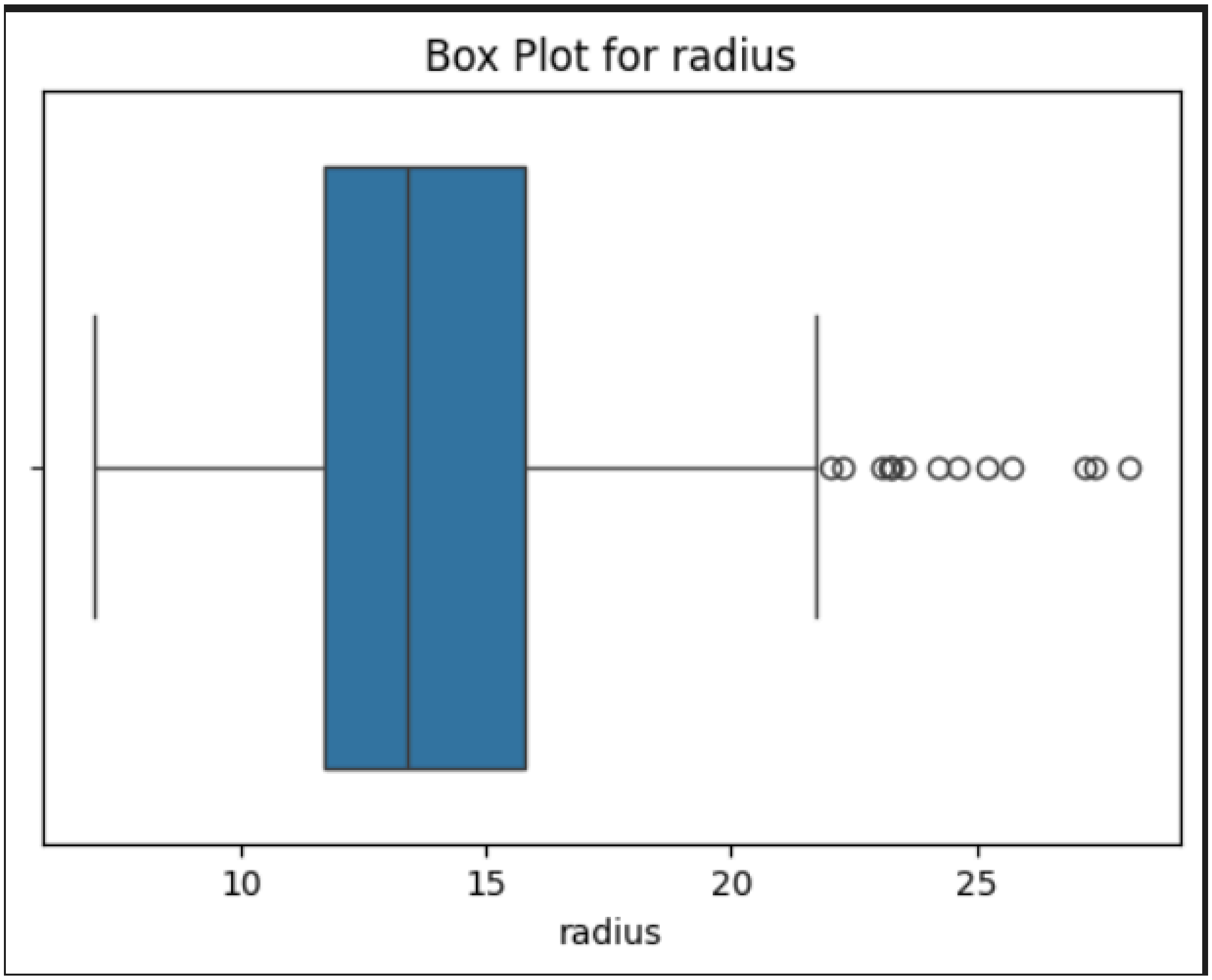
In the context of building a machine learning model to predict whether someone has cancer, it is important to carefully consider the treatment of outliers. While outliers are often removed in some machine learning workflows, there are several compelling reasons why we should keep the outliers in this dataset. Real-World Relevance: Outliers in medical datasets often represent rare but real cases that are highly relevant for prediction. For example, a patient with unusual but possible measurements (e.g., very high tumor area or compactness) could represent a rare form of cancer or an advanced stage of the disease. Removing these outliers could result in the model being less sensitive to these rare, but critical, cases. Cancer Characteristics: In cancer detection, there is significant variability in the size, shape, and texture of tumors across individuals. Tumors can be atypical or aggressive, which might result in extreme measurements. Removing these data points could distort the model's ability to correctly classify all types of cancer, particularly those with more unusual characteristics. Impact on Model Performance: Outliers can sometimes help the model learn better decision boundaries by representing extreme values that the model should recognize and differentiate from other data points. Removing them could limit the model's ability to generalize across the entire range of tumor characteristics, which is crucial for real-world applications where outliers may be encountered frequently. Outliers as Important Indicators: In predictive modeling, particularly in binary classification tasks like cancer prediction, outliers could serve as important indicators of certain conditions. For example, extreme values of certain features (e.g., perimeter or area) could be associated with specific types of cancer. Removing outliers may remove critical information that helps the model distinguish between healthy and cancerous samples.

Alternative Techniques for Handling Outliers: Rather than removing outliers, there are other ways to handle them. Transformation: Applying logarithmic or other transformations can reduce the impact of extreme values without losing information.
Robust Models: Some machine learning algorithms (e.g., Random Forests, XGBoost) are more robust to outliers, meaning they can handle extreme values better without the need to remove

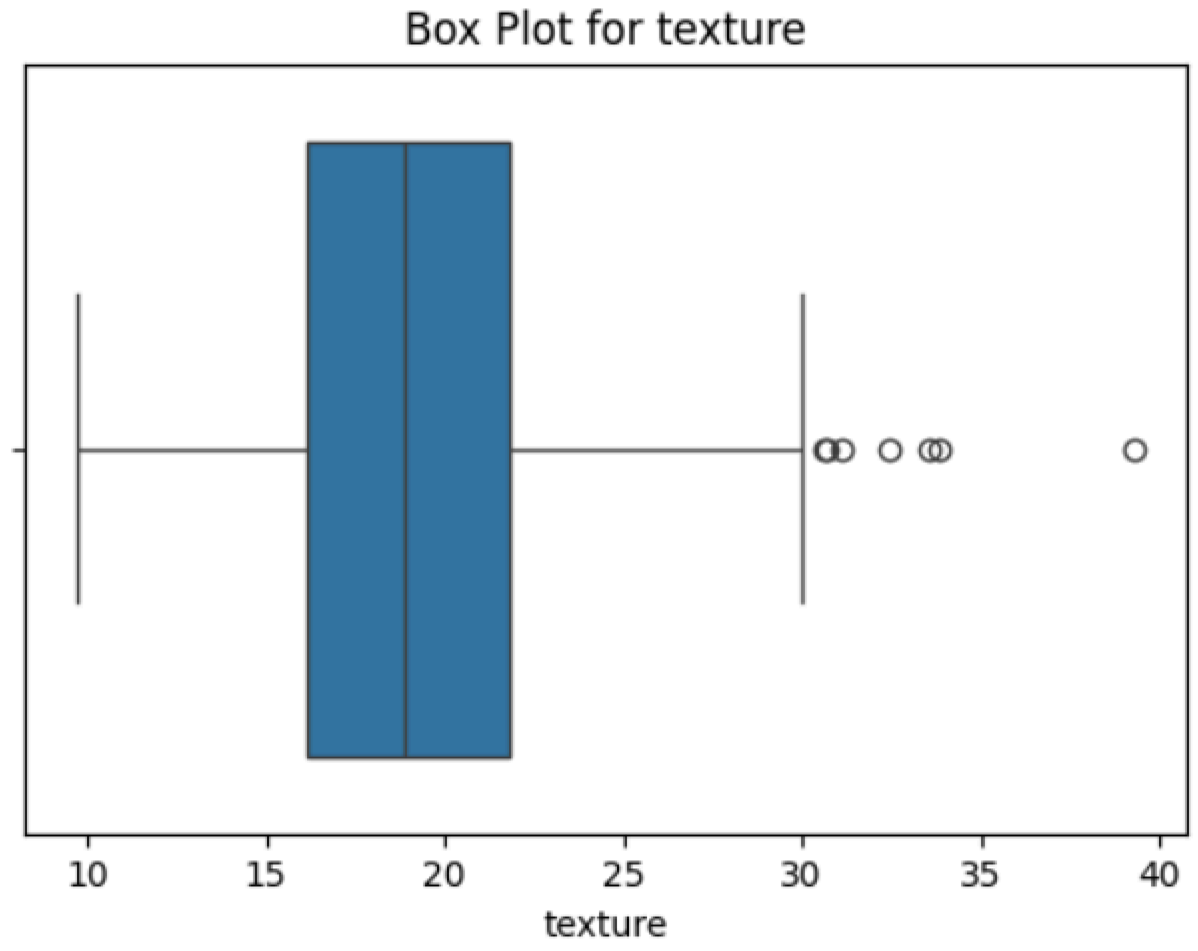
them. Imputation: If outliers are deemed to be errors or inaccuracies, they can be imputed or capped, rather than removed, which preserves the dataset's integrity.

By keeping outliers, the model can be more robust and can perform better in real-world scenarios where the data distribution is unknown or varies. In medical diagnostics, it is important for the model to not only be able to predict common cases but also handle rare and extreme cases without failing. Given the importance of identifying both common and rare cases in cancer prediction, removing outliers from the dataset could lead to loss of valuable information that is necessary for the model to generalize well. Therefore, keeping the outliers can help in building a more accurate, reliable, and robust machine learning model for cancer diagnosis.

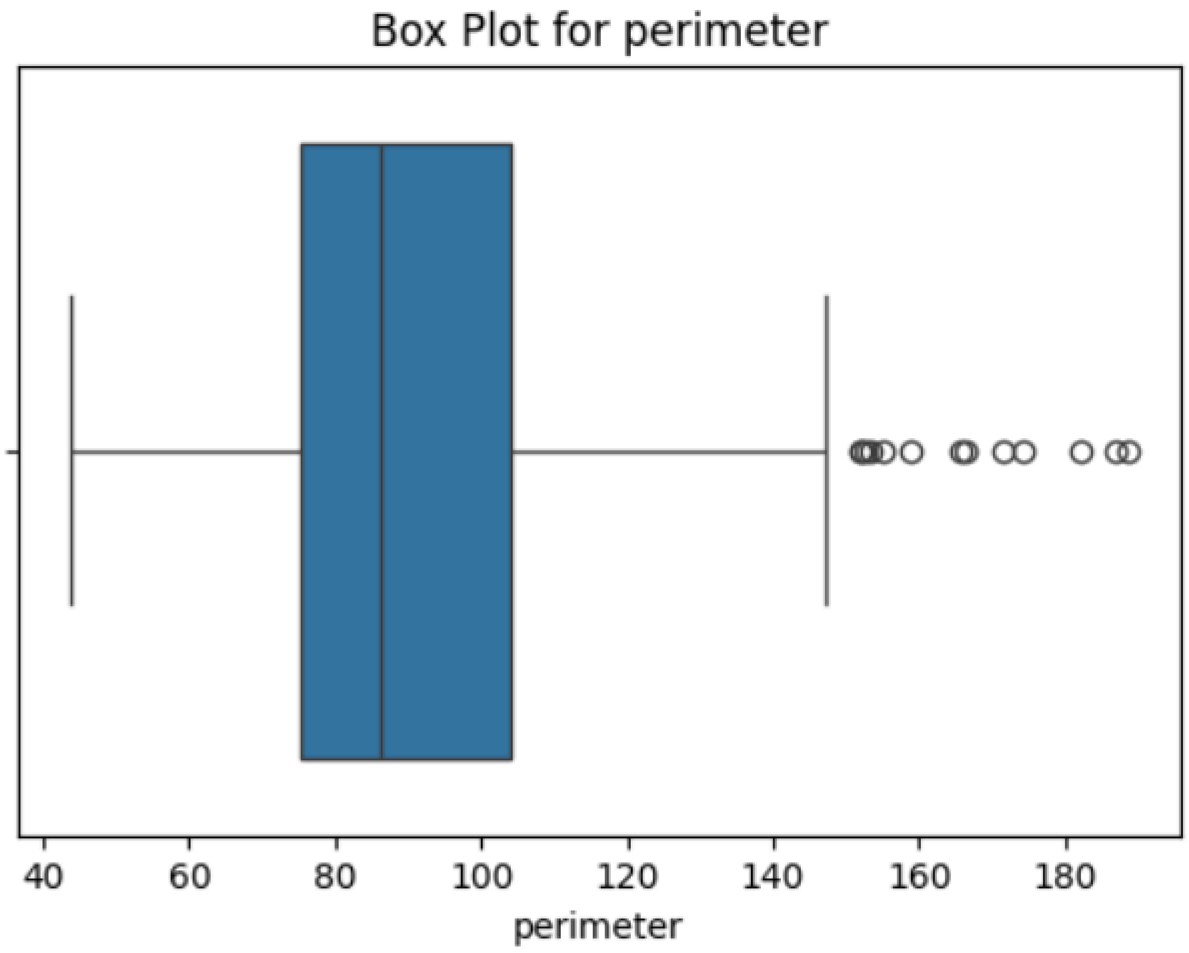
Now that it has been decided to keep the outliers, they must be visualized first. By utilizing Matplotlib[4] and Seaborn[5] we can create a boxplot for each of the 10 features.



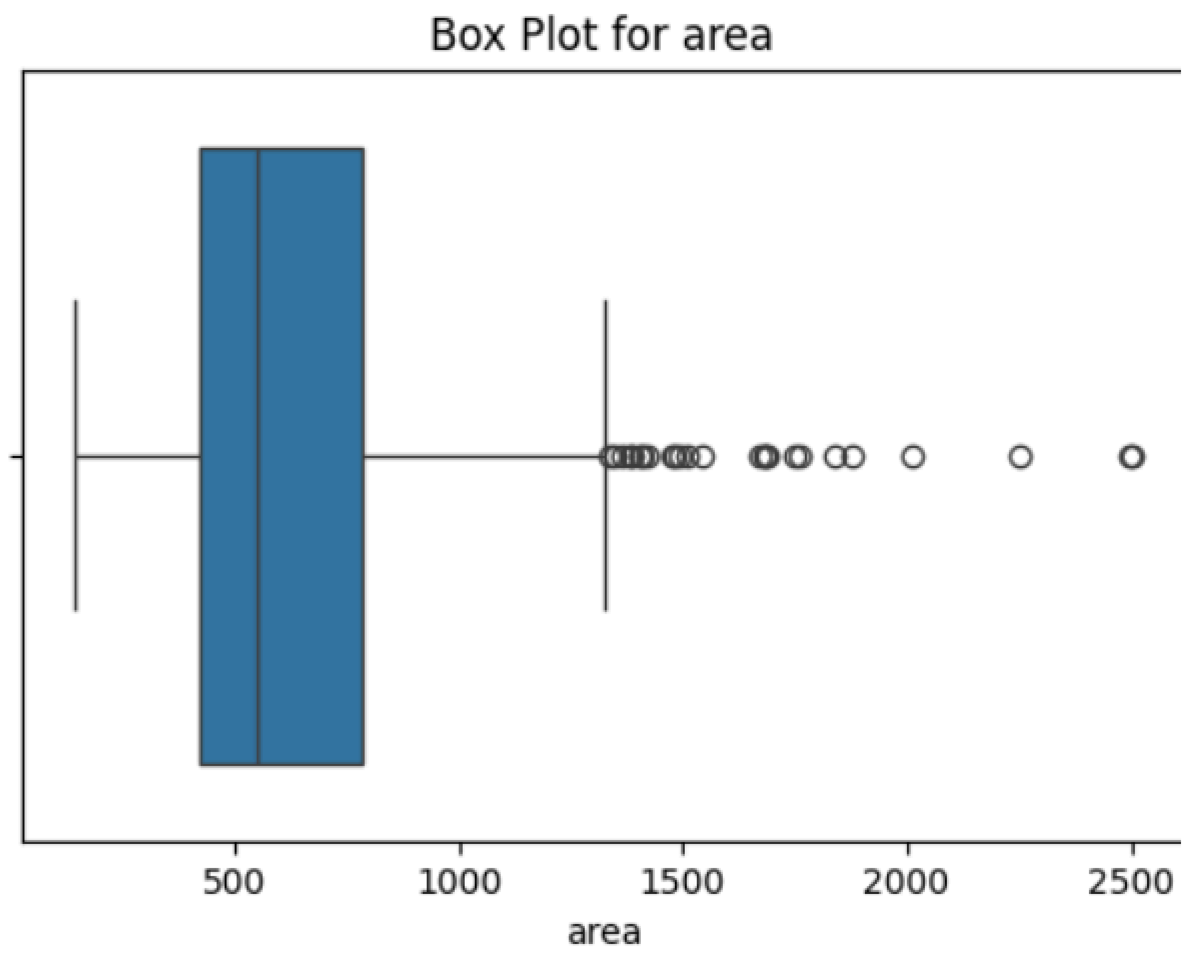
Boxplot for Radius feature



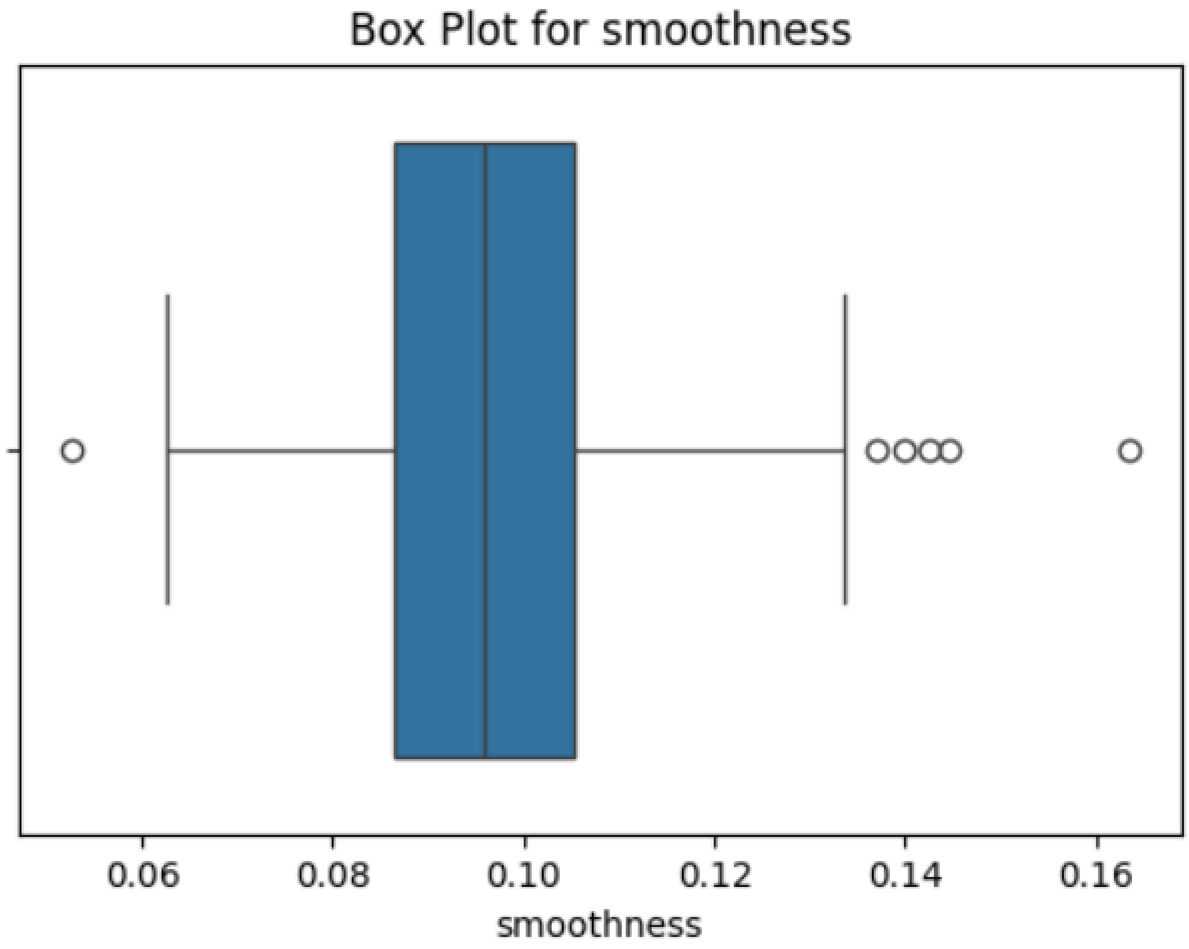
Boxplot for Texture feature



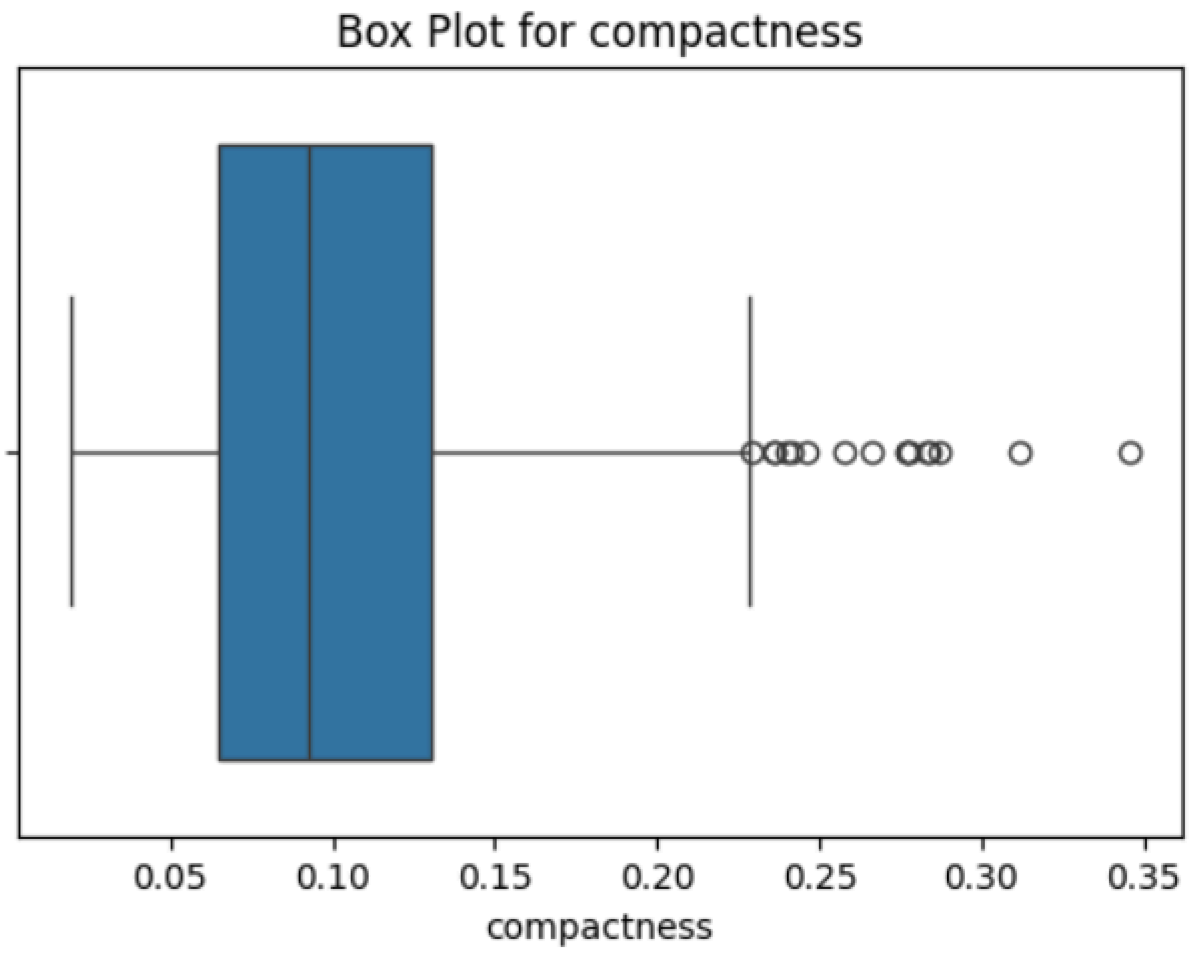
Boxplot for Perimeter feature



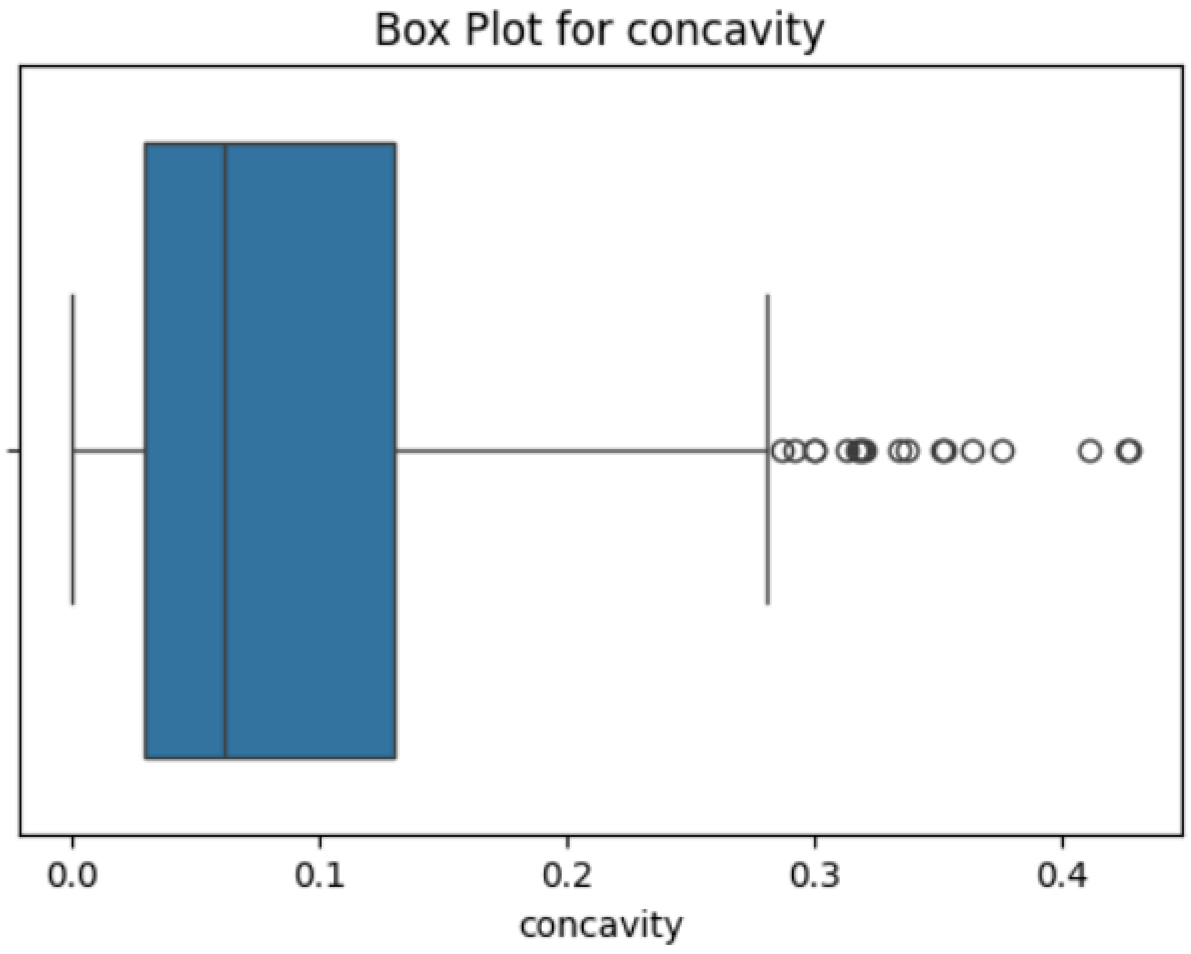
Boxplot for Area feature



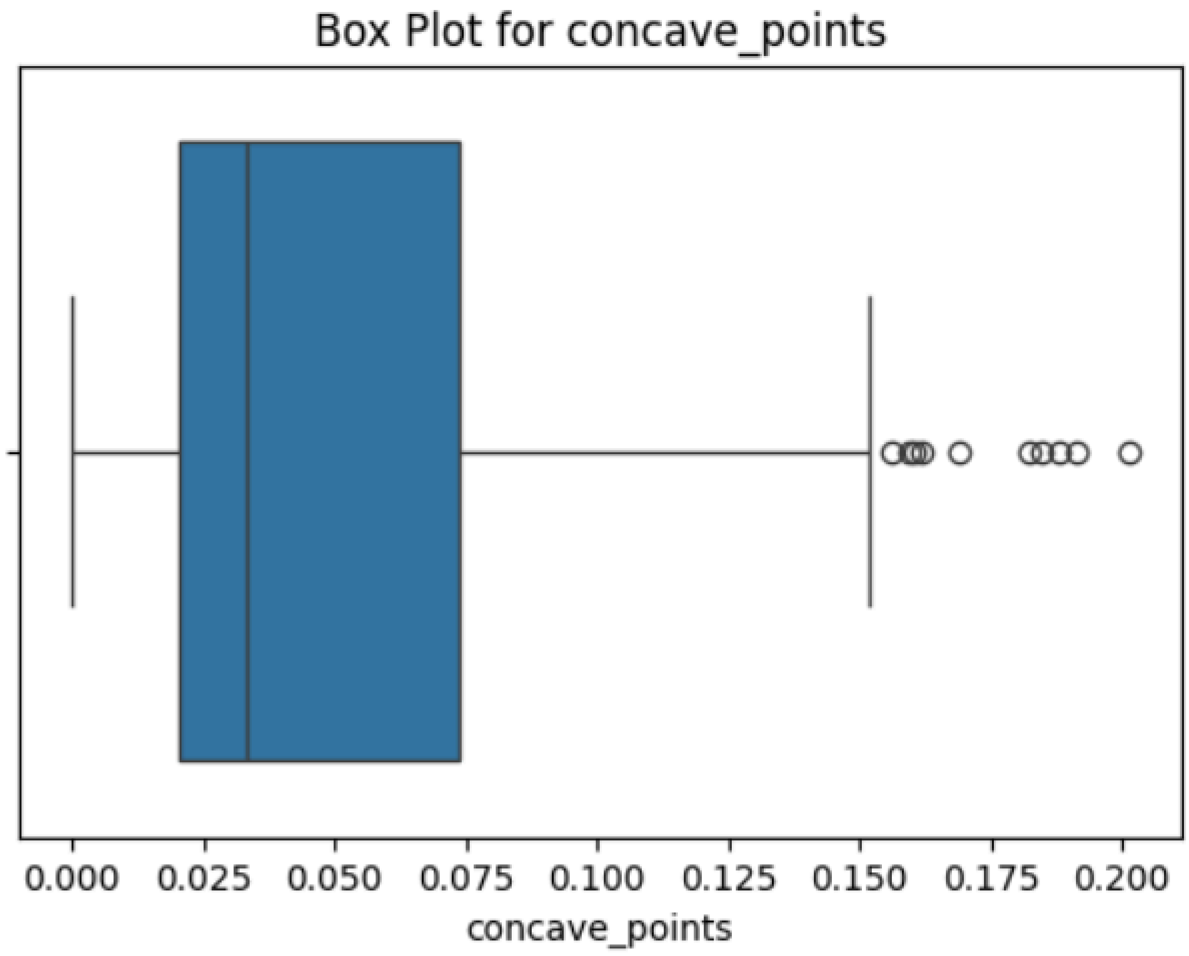
Boxplot for Smoothness texture



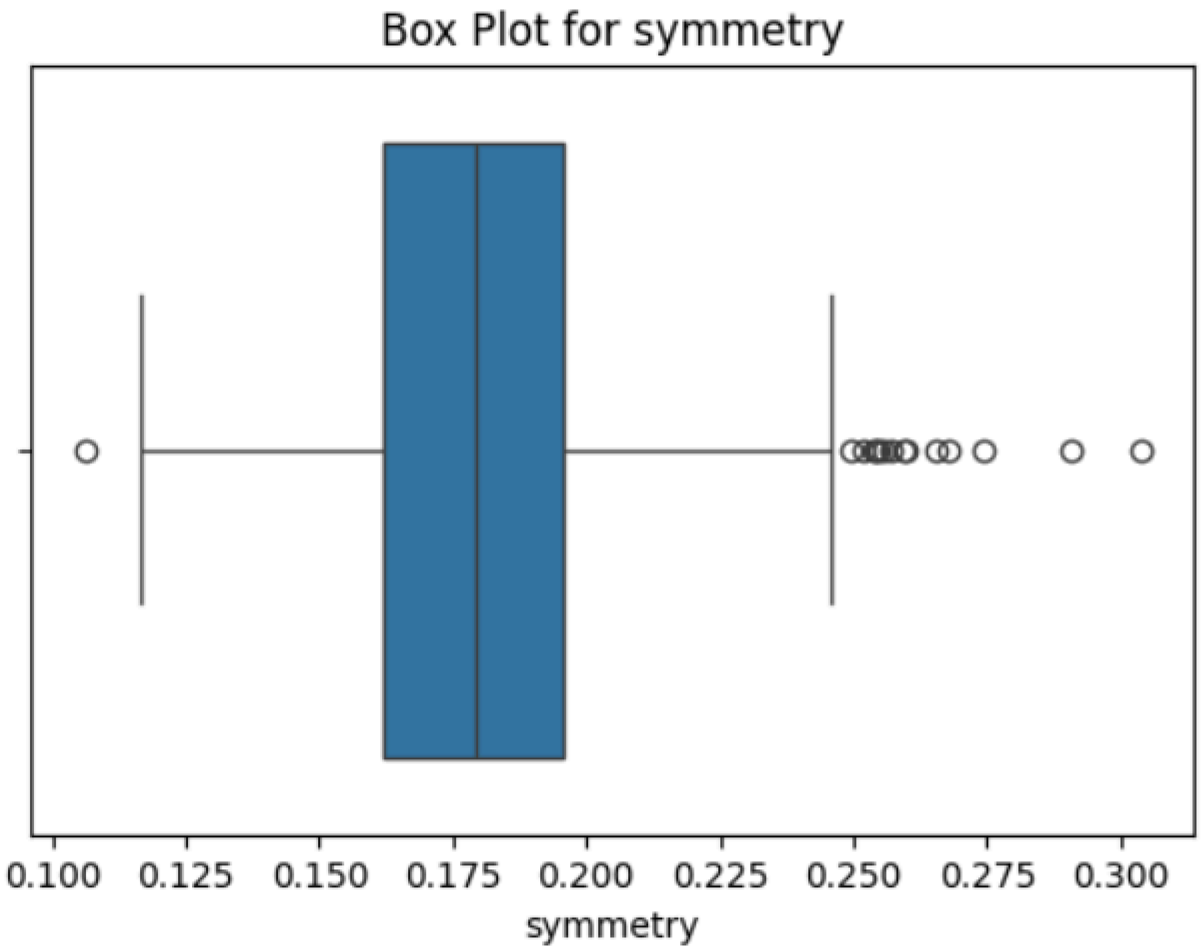
Boxplot for Compactness feature



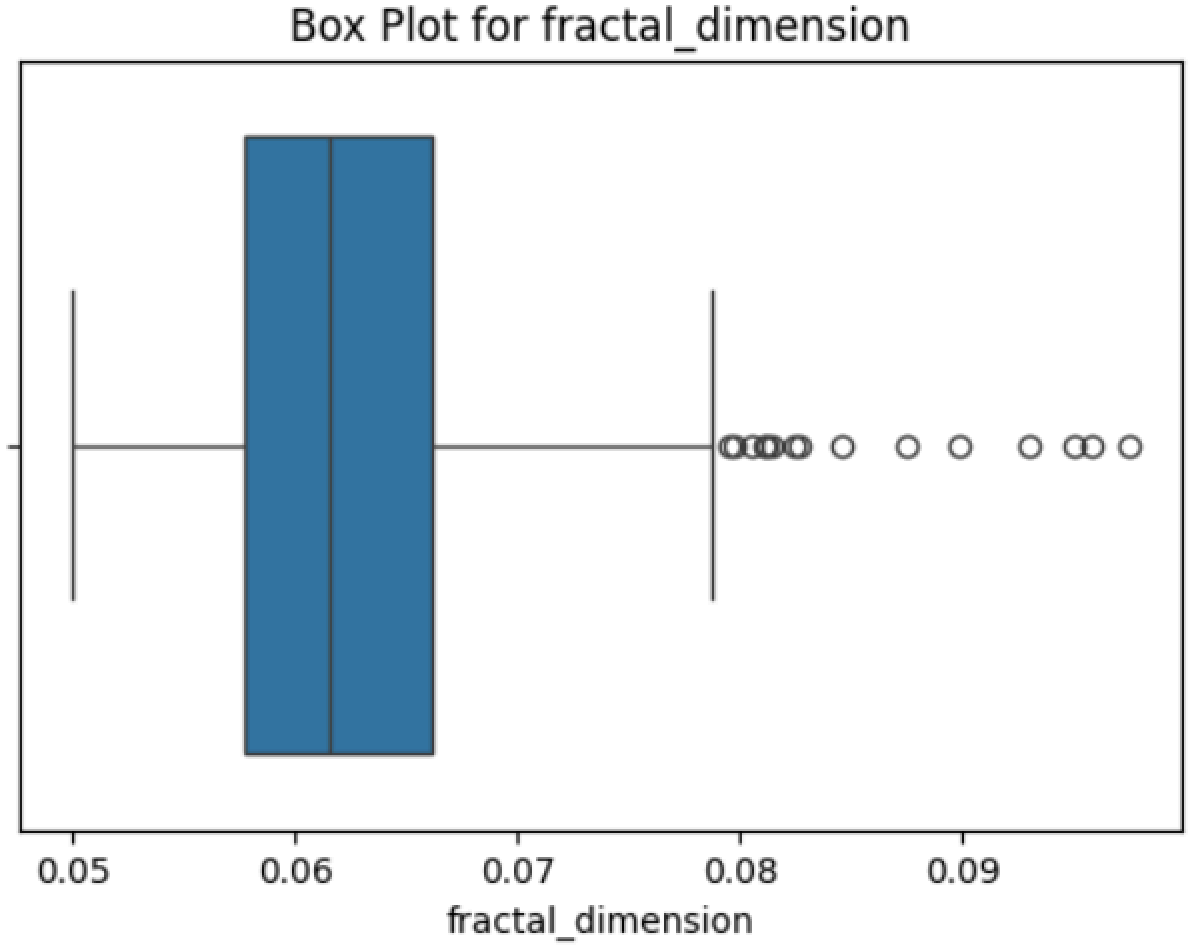
Boxplot for Concavity feature



Boxplot for Concave Points feature

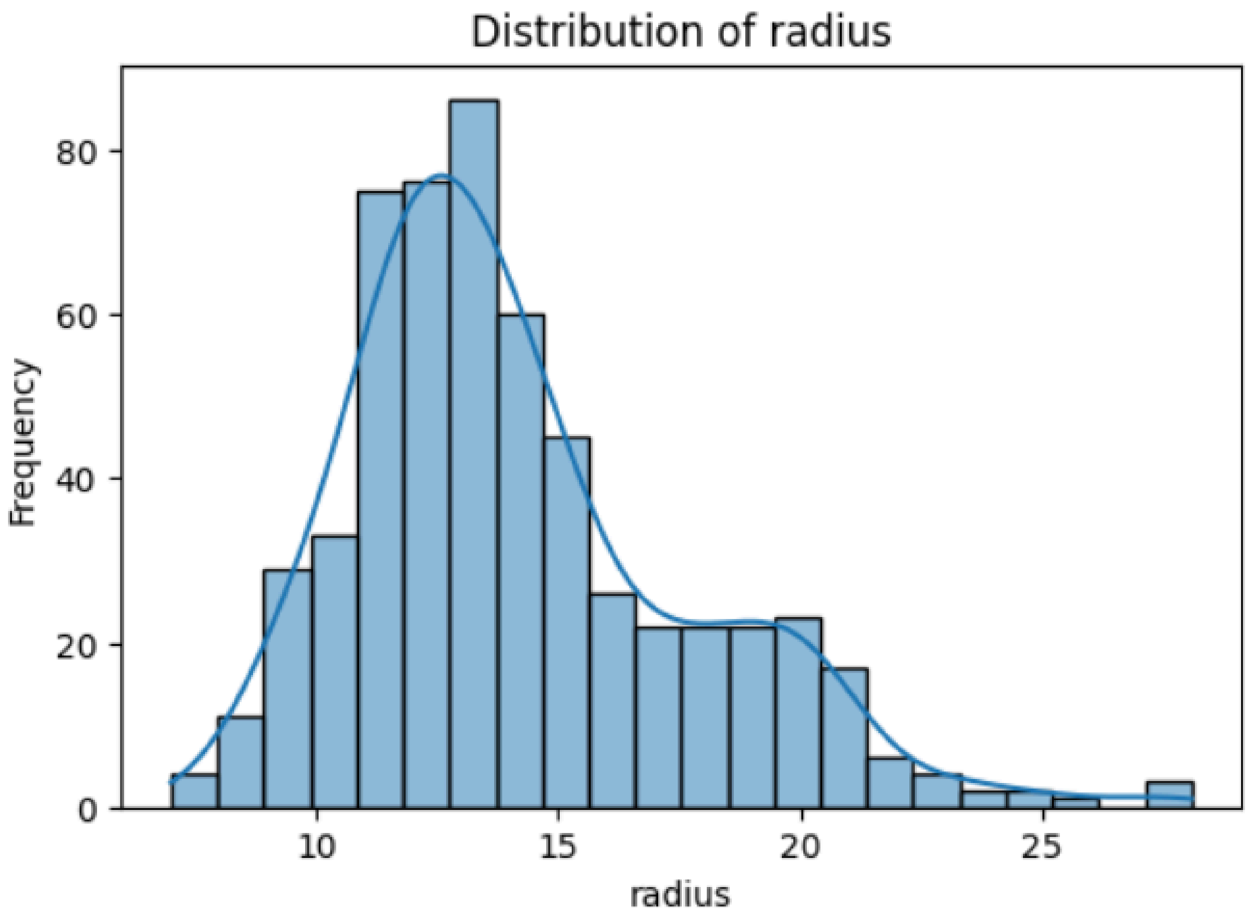


Boxplot for Symmetry feature

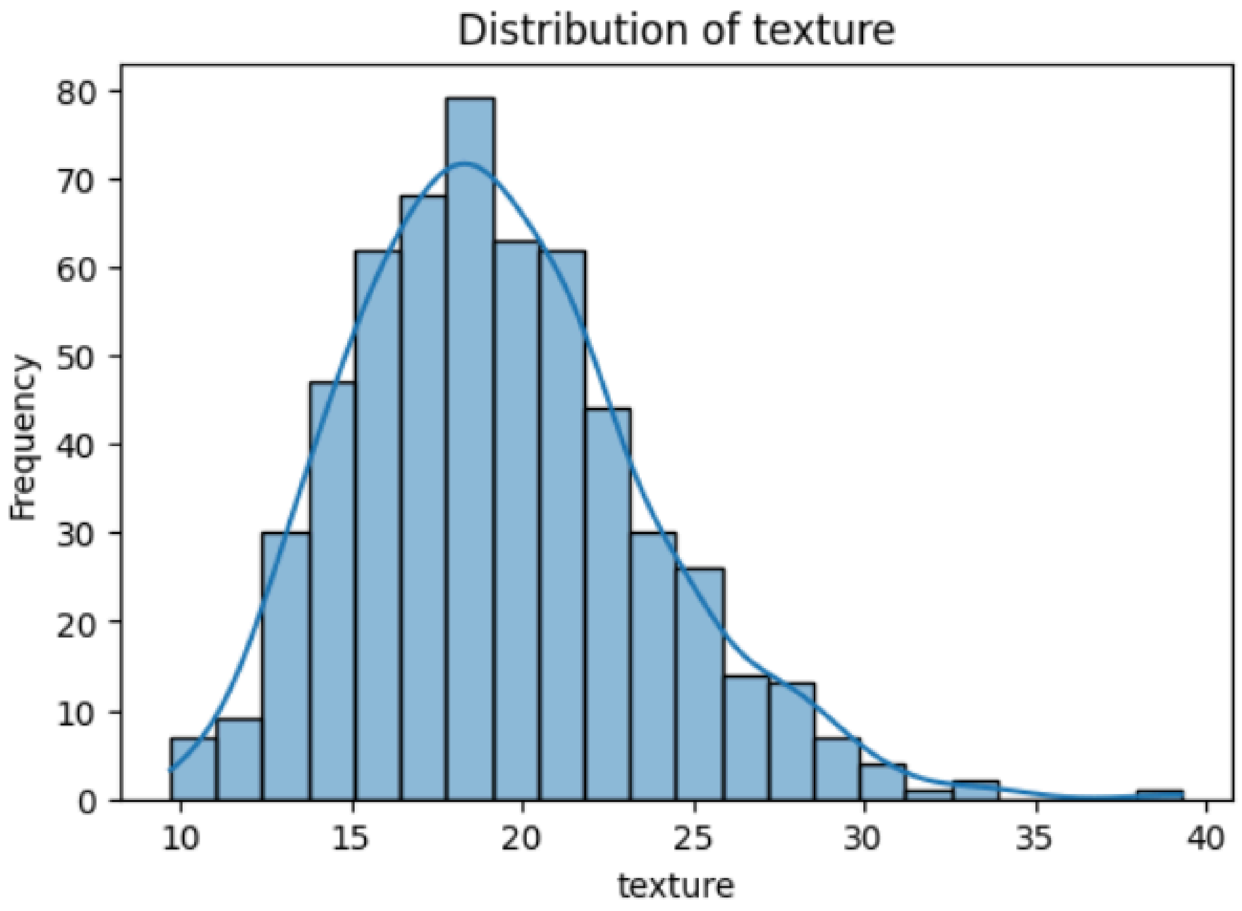


Boxplot for Fractal Dimension feature

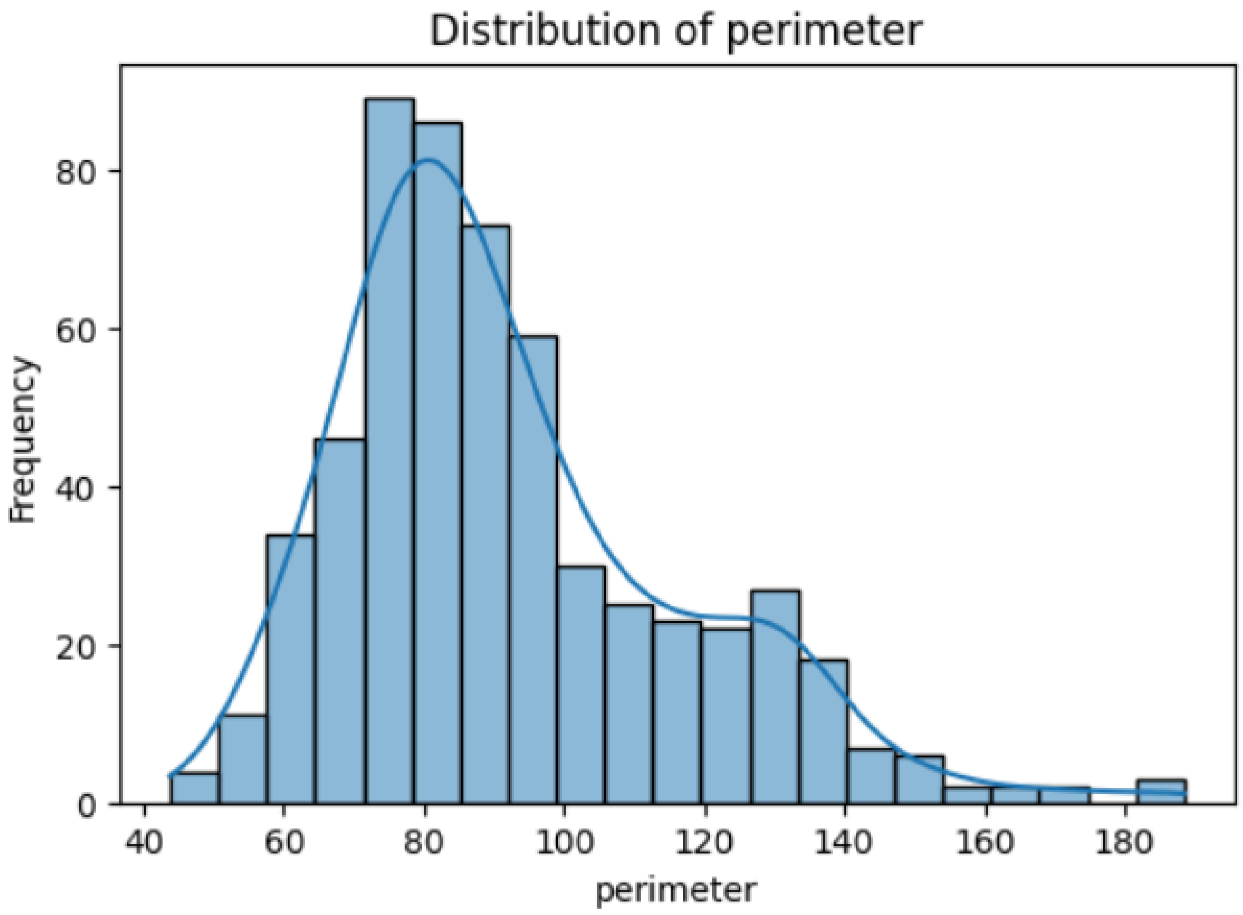
Next, we will use Matplotlib[4] and Seaborn[5] again to generate distribution visualizations for each feature to determine which type of distribution each feature is. In the context of machine learning, it is important that each feature is scaled, centered, and is distributed normally so that the machine learning model can perform optimally. One of the first things we do is to see if the feature is distributed normally amongst the dataset. Therefore, we used and created a distribution plot for each feature, that included a histogram and KDE combined.



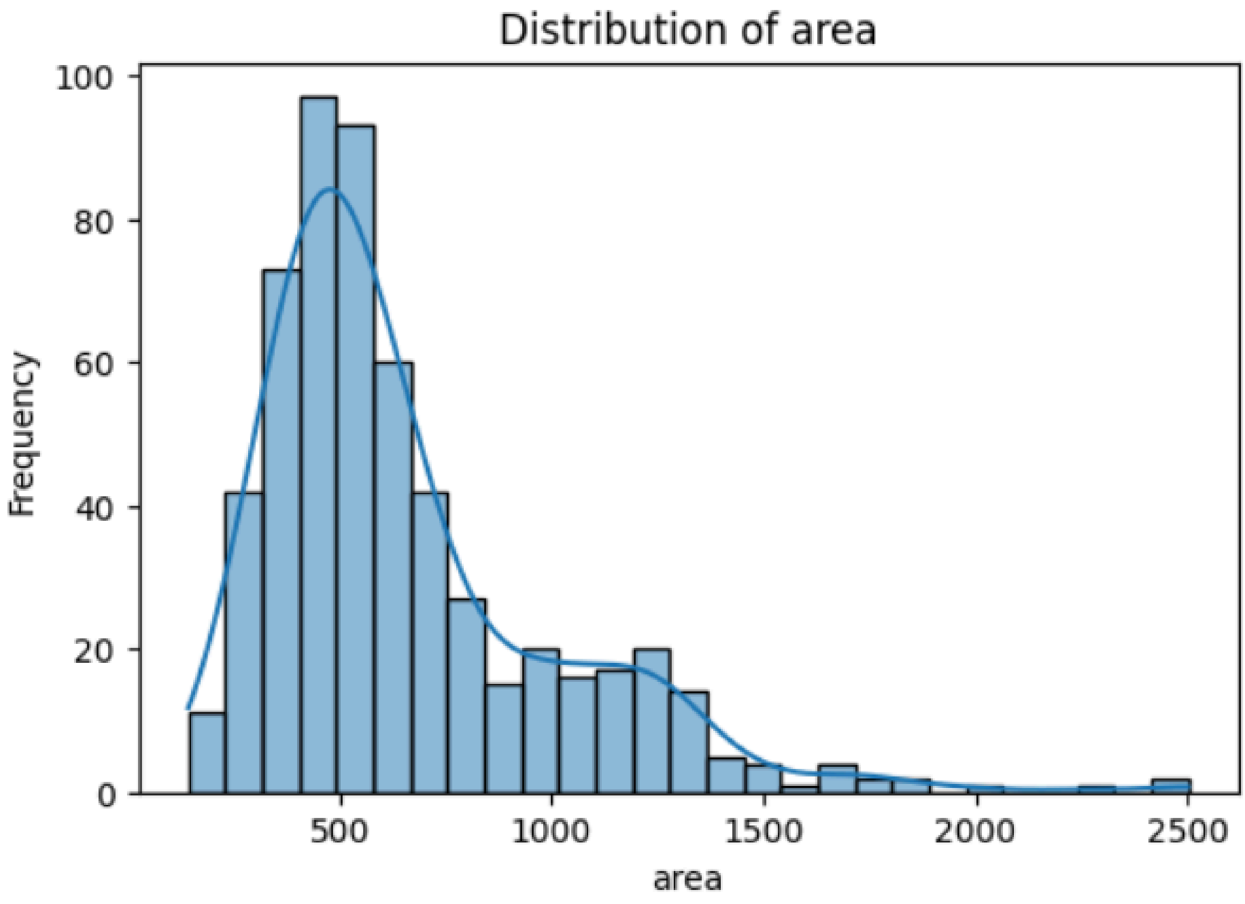
Distribution Plot of Radius feature



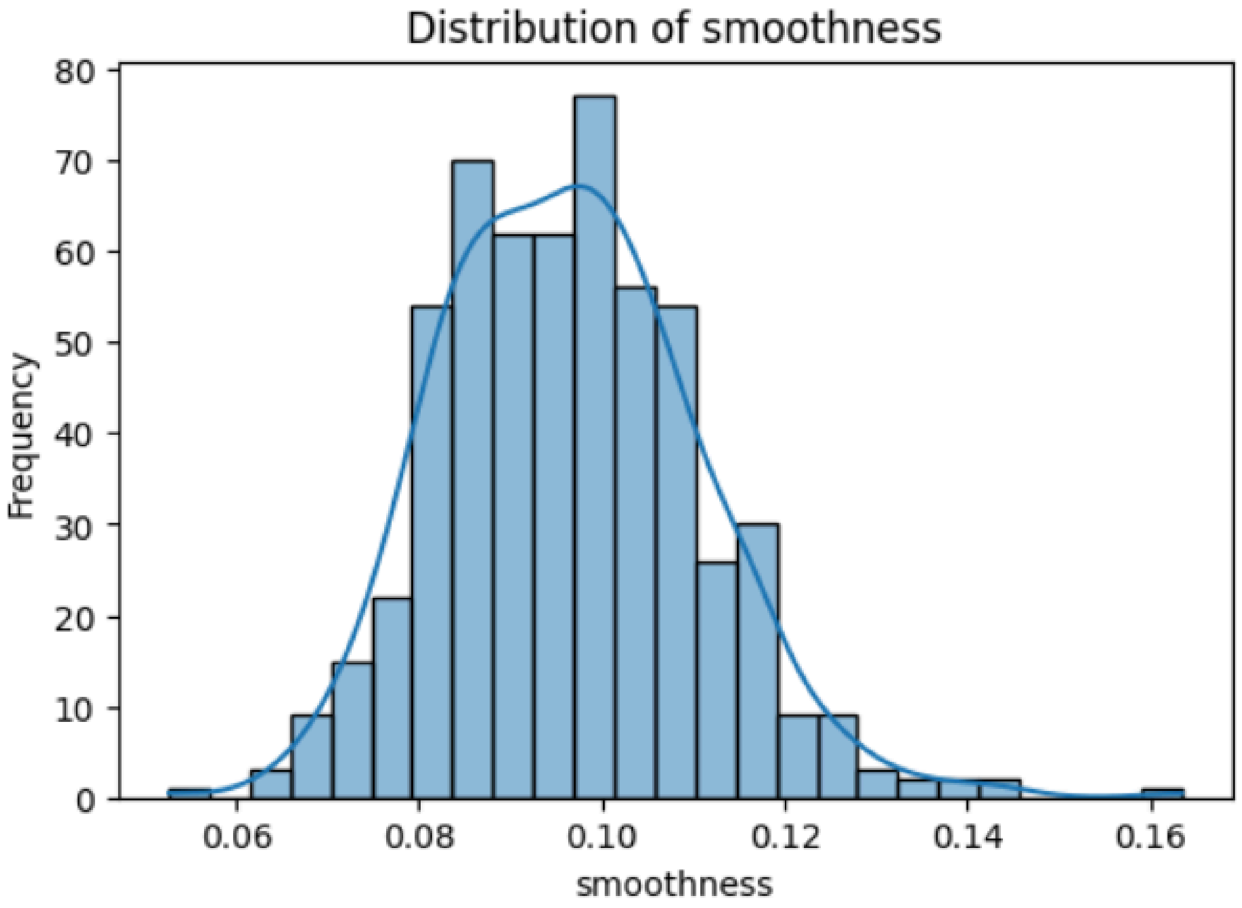
Distribution Plot of Texture feature



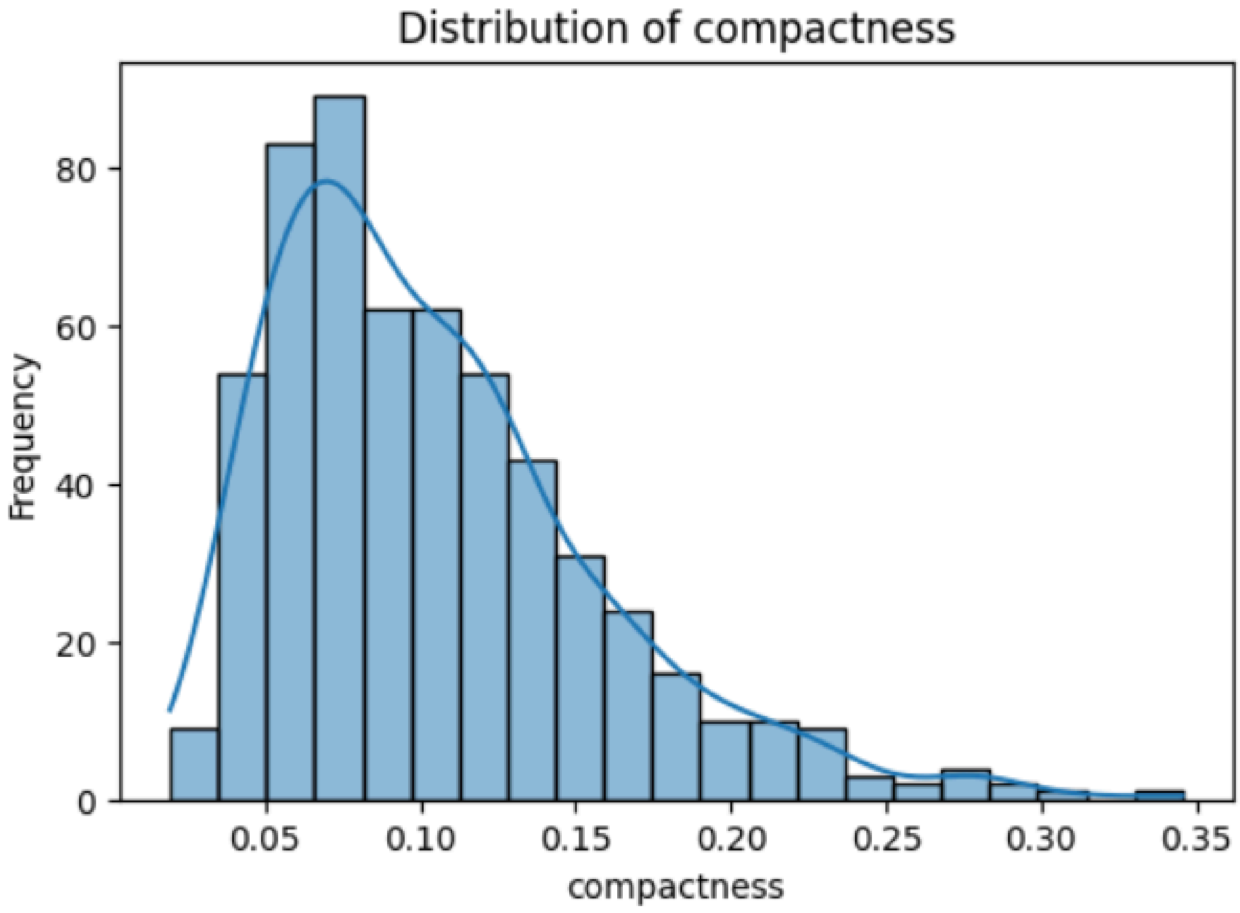
Distribution Plot of Perimeter feature



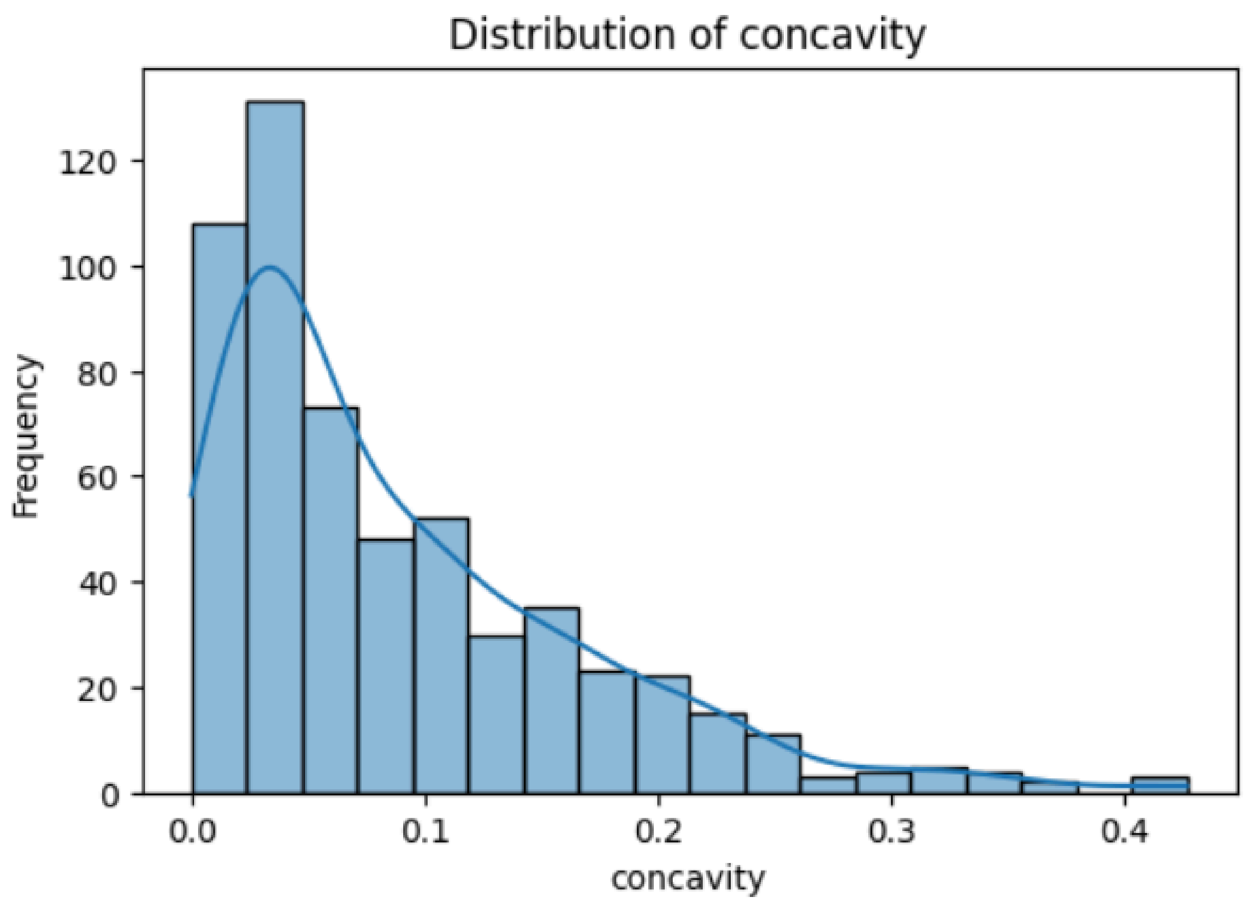
Distribution Plot of Area feature



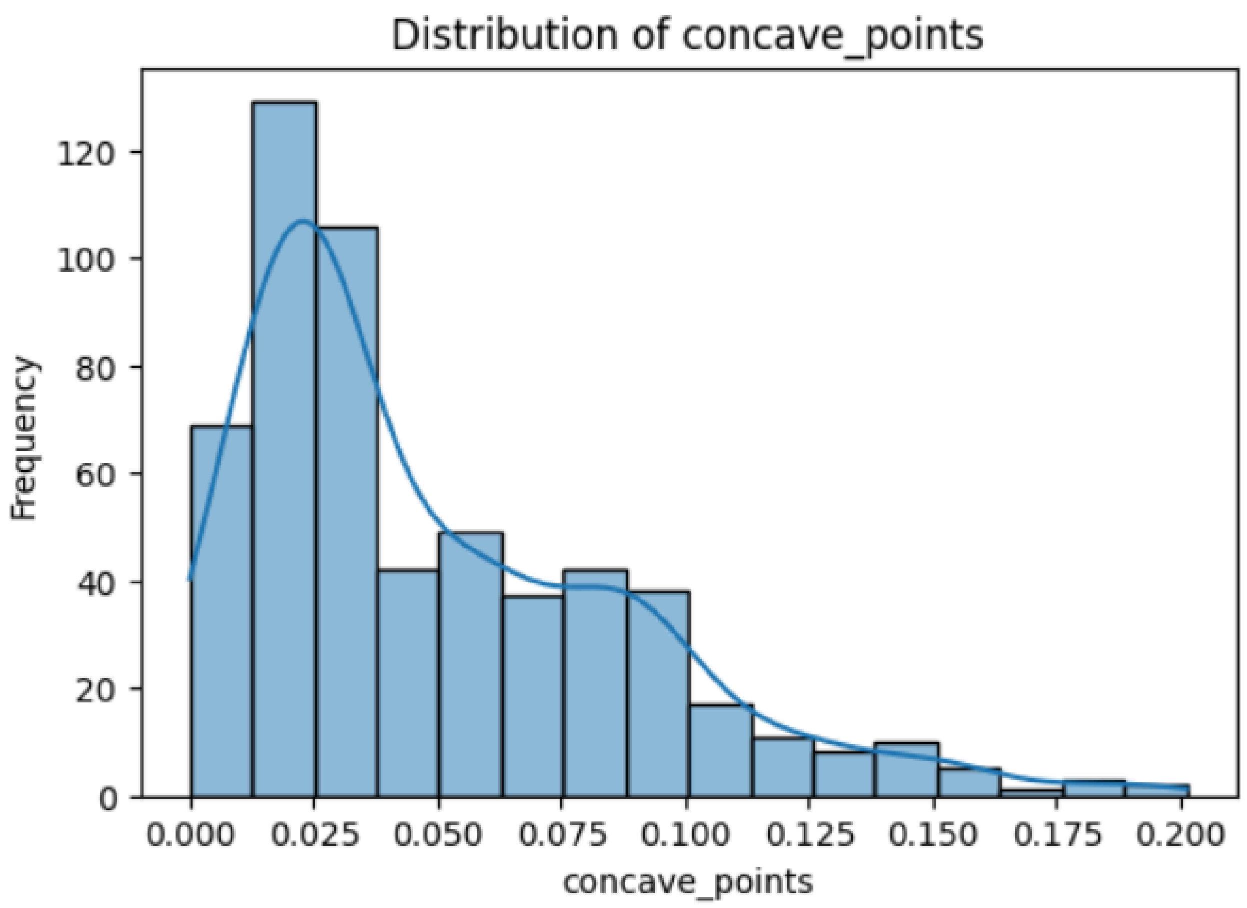
Distribution Plot of Smoothness feature



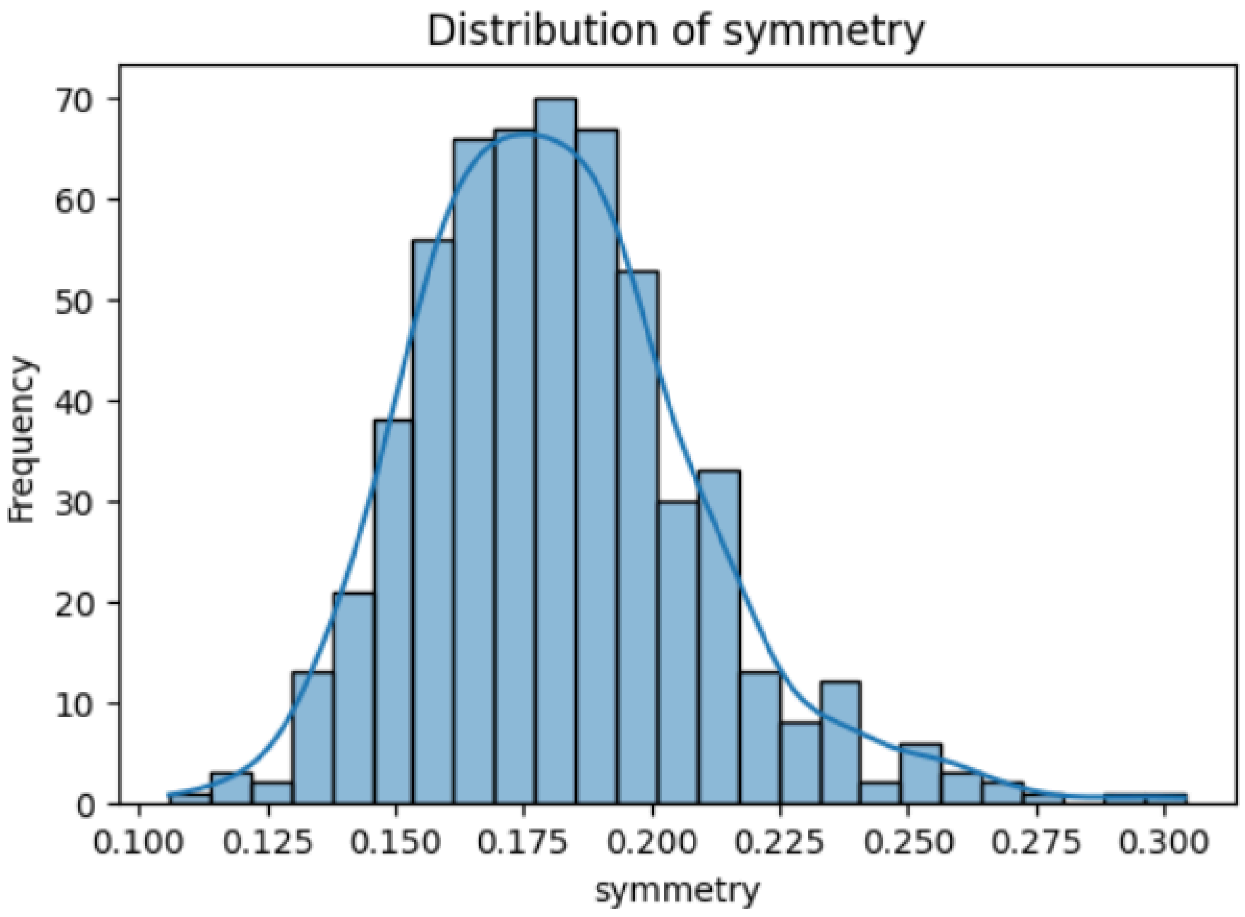
Distribution Plot of Compactness feature



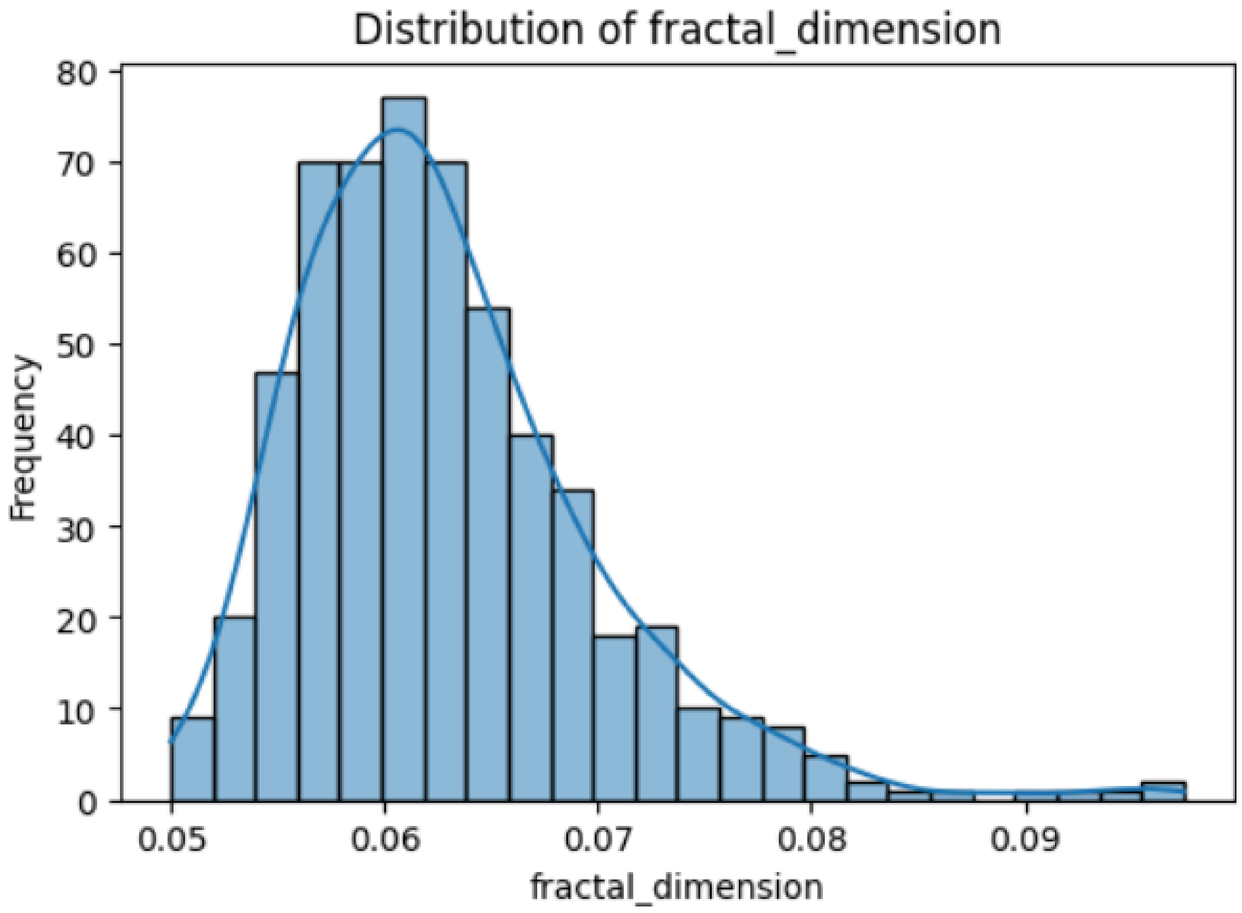
Distribution Plot of Concavity feature



Distribution of Concave Points feature

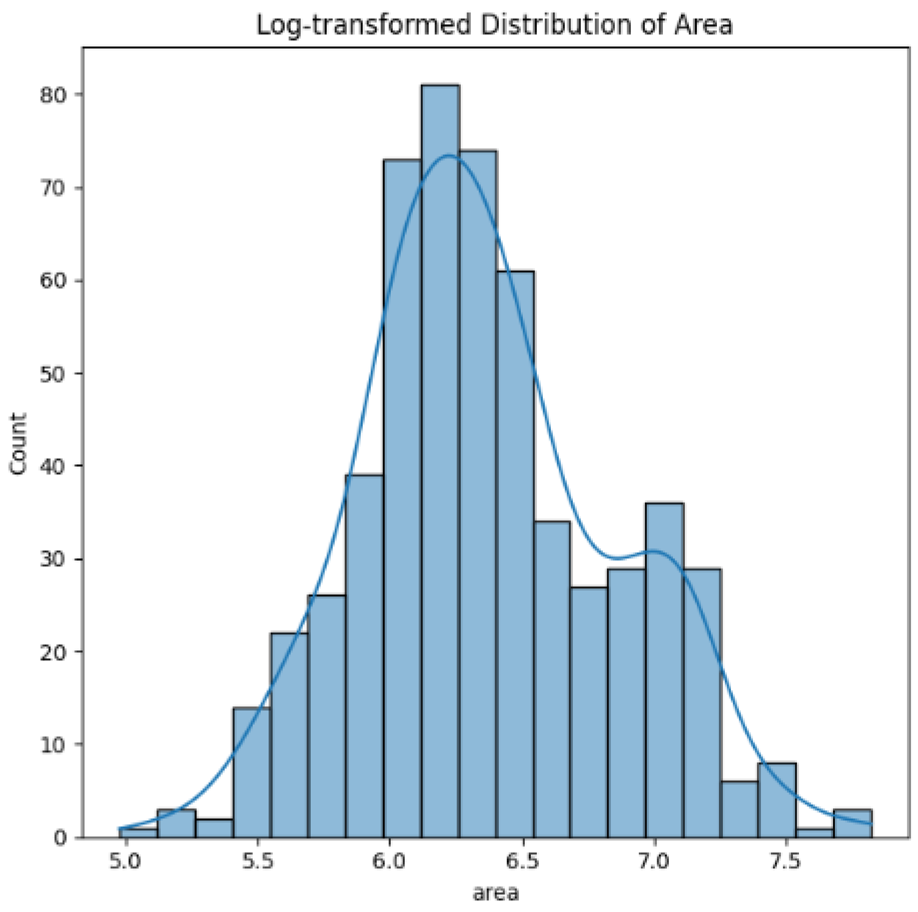


Distribution of Symmetry feature

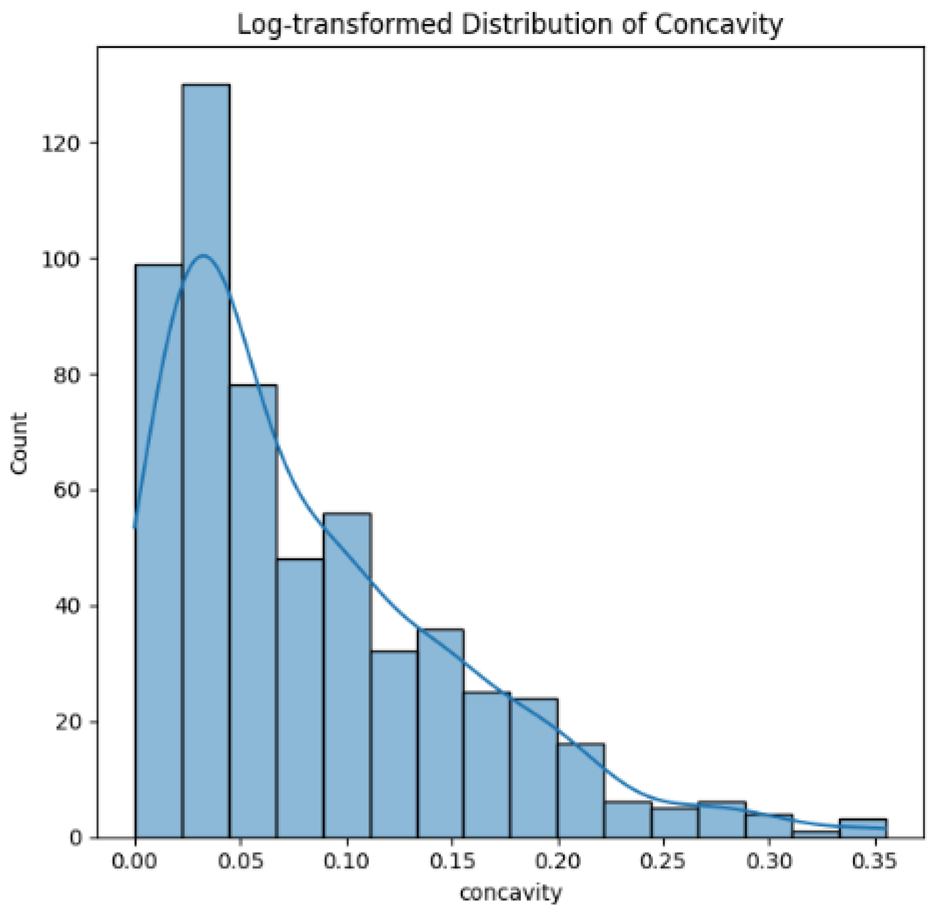


Distribution of Fractal Dimension feature

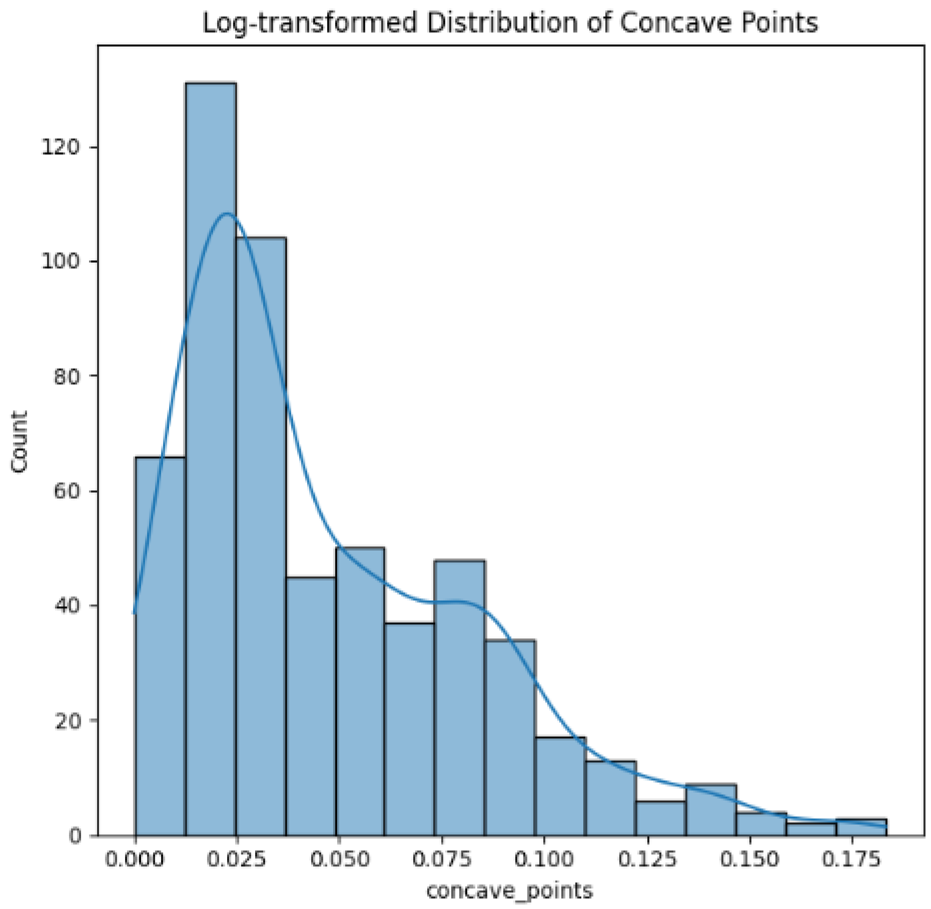
Given the blue line approximation and the histogram bins, we can easily identify if the distributions are normal, or if they need to be transformed. The distribution of these features are fairly normally distributed and do not need any transformation whatsoever: Radius, Texture, Perimeter, Smoothness, Compactness, Symmetry, Fractal Dimension. The three features that do need to be transformed are: Area, Concavity, and Concave Points. As noted in the visualizations, all three of them are skewed right, and so will need logarithmic transformations. Using numpy[6] for mathematical (logarithmic) operations, and Seaborn[5]/Matplotlib[4] for visualization of distribution plots, we can achieve the log transformation of these features seamlessly.



Log-Transformed Distribution Plot of Area feature



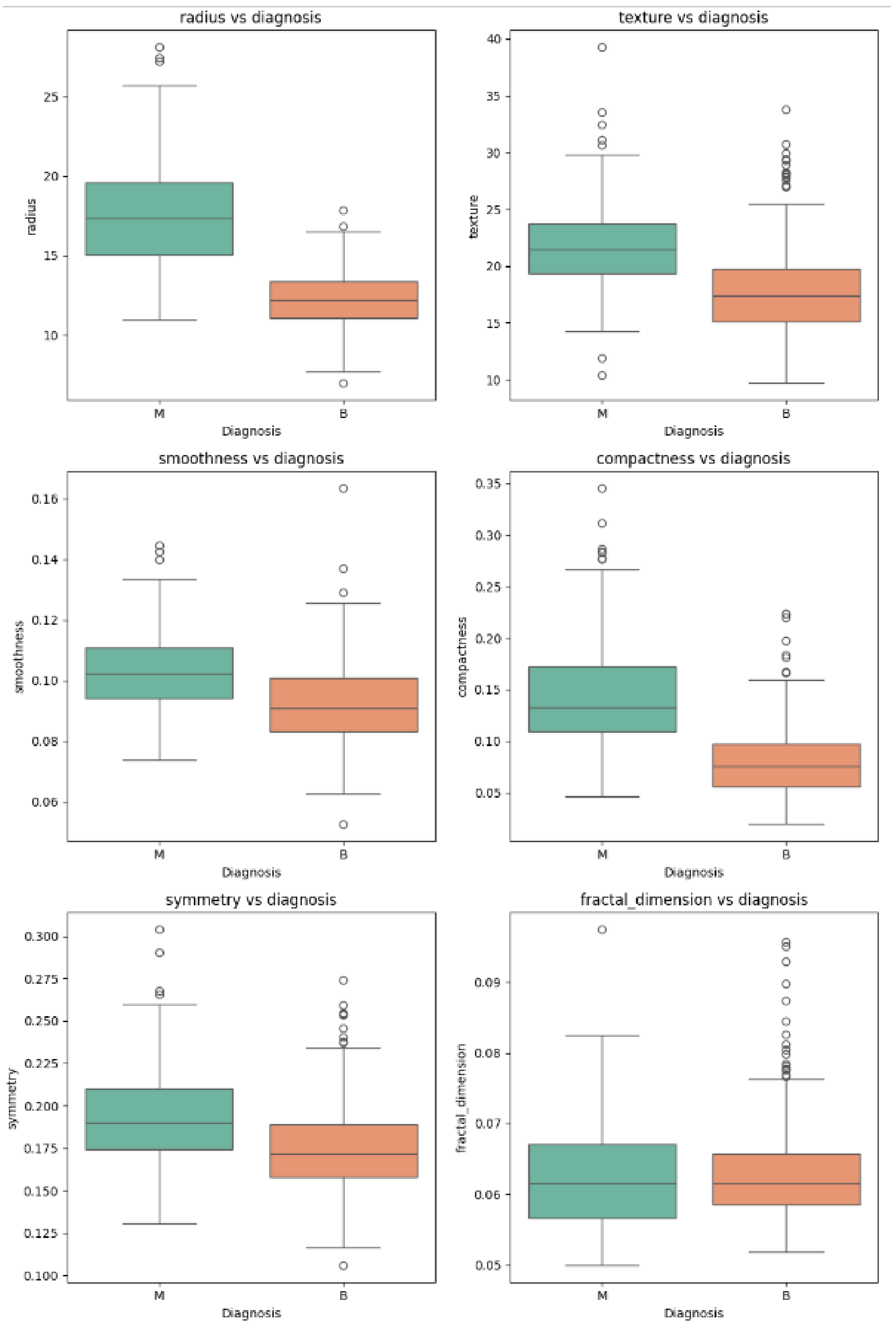
Log-Transformed Distribution Plot of Concavity feature



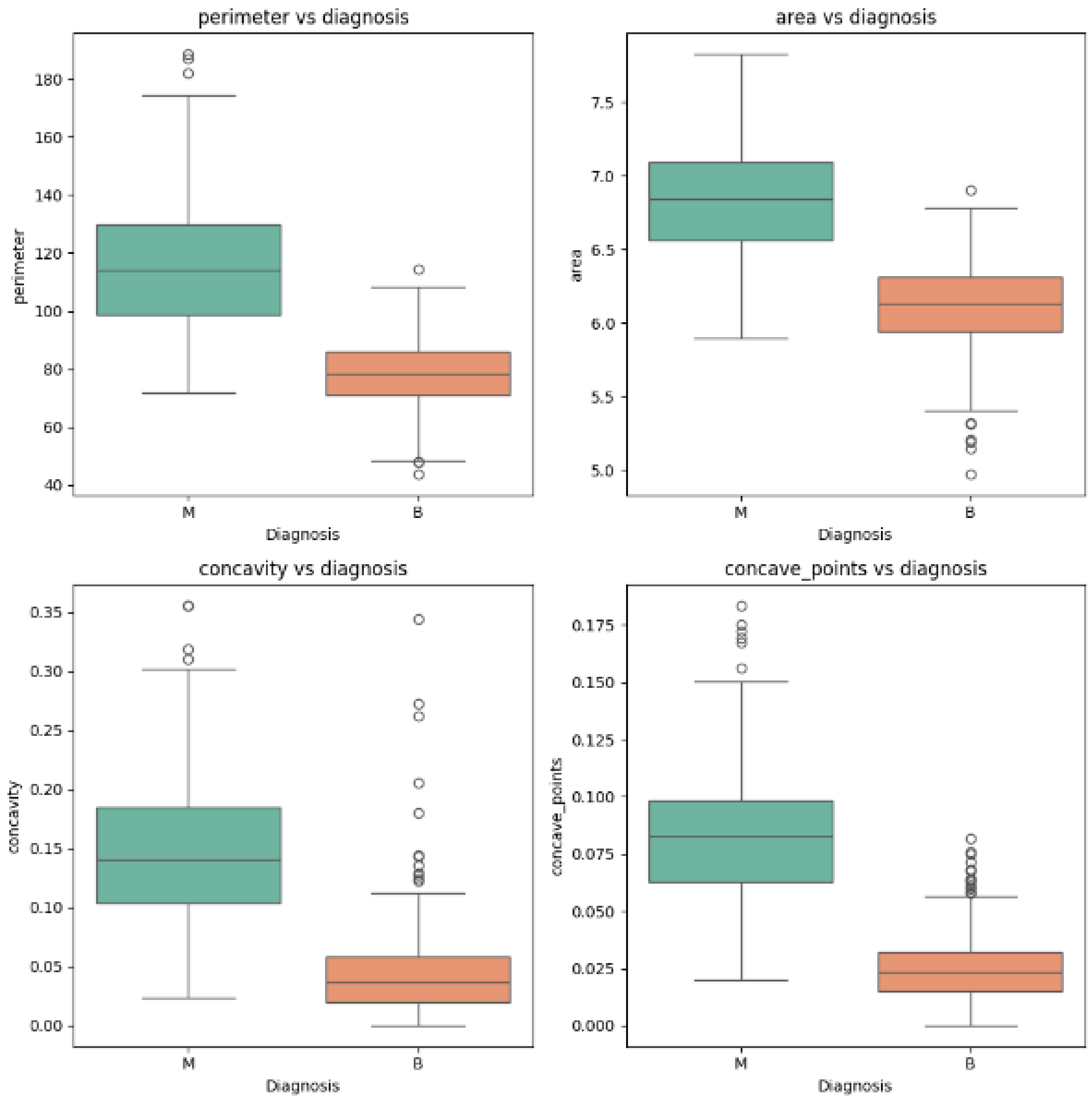
Log-Transformed Distribution Plot of Concave Points feature

After applying log transformations, the distribution of these fields look a lot more normally distributed. Now, these transformed versions are what we will utilize in our final dataframe before training the models. Scaling is another form of changing the features' data before feeding it into and training a machine learning model. However, in this case, scaling may not be necessary for this dataset because the variables represent meaningful physical measurements, such as area, perimeter, and radius. These measurements carry intrinsic significance, and their relative magnitudes might provide important information for distinguishing between malignant and benign diagnoses. Altering their scales could risk losing these natural relationships, which might be vital for understanding the underlying patterns in the data.

Now we need to visualize the data by isolating the relationship between each of the 10 features, and the diagnosis column. The diagnosis column only has 2 possibilities. M (for malignant) or B (for benign). We want to see if there is a relationship between the features, and the diagnosis outcome. The hypothesis here is that the malignant classifications in general have higher average feature values than benign. This makes sense, because malignant tumors are normally bigger and harder.



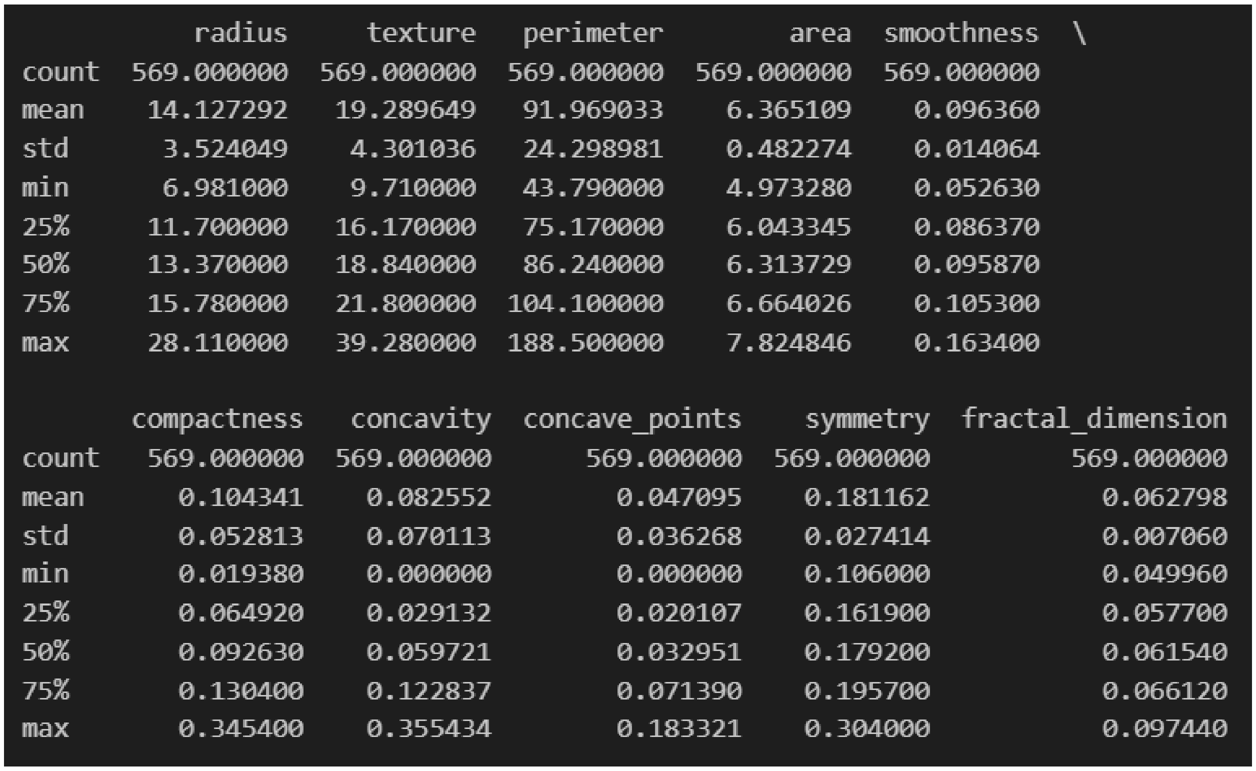
Boxplots of Radius, Texture, Smoothness, Compactness, Symmetry, and Fractal Dimension features in correlation to diagnosis outcome (M or B)



Boxplots of Perimeter, Area, Concavity, and Concave Points features in correlation to diagnosis outcome (M or B)

As obvious as it shows, we can clearly see that when the diagnosis is cancerous (Malignant) the average box plot is considerably higher than when the diagnosis is Benign. Therefore, we already have an idea as to whether these 10 features are going to be perfect for our feature selection because we know all 10 features will positively impact the machine learning model into making more accurate decisions. We know this because we can see a strong correlation between the features' values and the diagnosis outcome.

To summarize, the 10 features I am planning on using for my analysis are: Radius, Texture, Perimeter, Area, Smoothness, Compactness, Concavity, Concave Points, Symmetry, and Fractal Dimension. I am using these particular features, because all these features relate to important information about having (or not) breast cancer. Each of these features plays a very important role in my model to predict if someone will have breast cancer (malignant or benign). I also made sure not to use the last 20 columns of the original data set, because it just includes the worst case, and standard error of each column. Which is irrelevant in this case. To reiterate, our predictors are: Radius, Texture, Perimeter, Area, Smoothness, Compactness, Concavity, Concave Points, Symmetry, and Fractal Dimension. And the response variable is: Diagnosis column (B or M). Almost all of the variables (columns) had outliers, but I decided to still keep them, and their justification was given up above. Some features were skewed right, so I had to apply log transformation, to make them more normally distributed. The exact features, and how I did it is shown above. None of the columns had any missing values. Descriptions with visualizations on how I dealt with the issues are shown above. Before delving into the results of creating, simulating, and determining the accuracy of our three machine learning models, we must display the descriptive statistics for each of the 10 predictors for a better understanding. This will be done using pandas[2]' .describe() method and a screenshot of the stats is provided below.



```
          radius     texture   perimeter        area  smoothness  \
count  569.000000  569.000000  569.000000  569.000000  569.000000
mean    14.127292   19.289649   91.969033    6.365109    0.096360
std      3.524049    4.301036   24.298981    0.482274    0.014064
min      6.981000    9.710000   43.790000    4.973280    0.052630
25%     11.700000   16.170000   75.170000    6.043345    0.086370
50%     13.370000   18.840000   86.240000    6.313729    0.095870
75%     15.780000   21.800000  104.100000    6.664026    0.105300
max     28.110000   39.280000  188.500000    7.824846    0.163400

       compactness   concavity  concave_points   symmetry  fractal_dimension
count   569.000000  569.000000      569.000000  569.000000         569.000000
mean      0.104341    0.082552        0.047095    0.181162           0.062798
std       0.052813    0.070113        0.036268    0.027414           0.007060
min       0.019380    0.000000        0.000000    0.106000           0.049960
25%       0.064920    0.029132        0.020107    0.161900           0.057700
50%       0.092630    0.059721        0.032951    0.179200           0.061540
75%       0.130400    0.122837        0.071390    0.195700           0.066120
max       0.345400    0.355434        0.183321    0.304000           0.097440
```

Descriptive Statistics table for the 10 features which includes: Count, Mean, Standard Deviation, Minimum, 25-50-75th Percentiles, and Maximum – table was obtained using .describe() method from pandas[2]

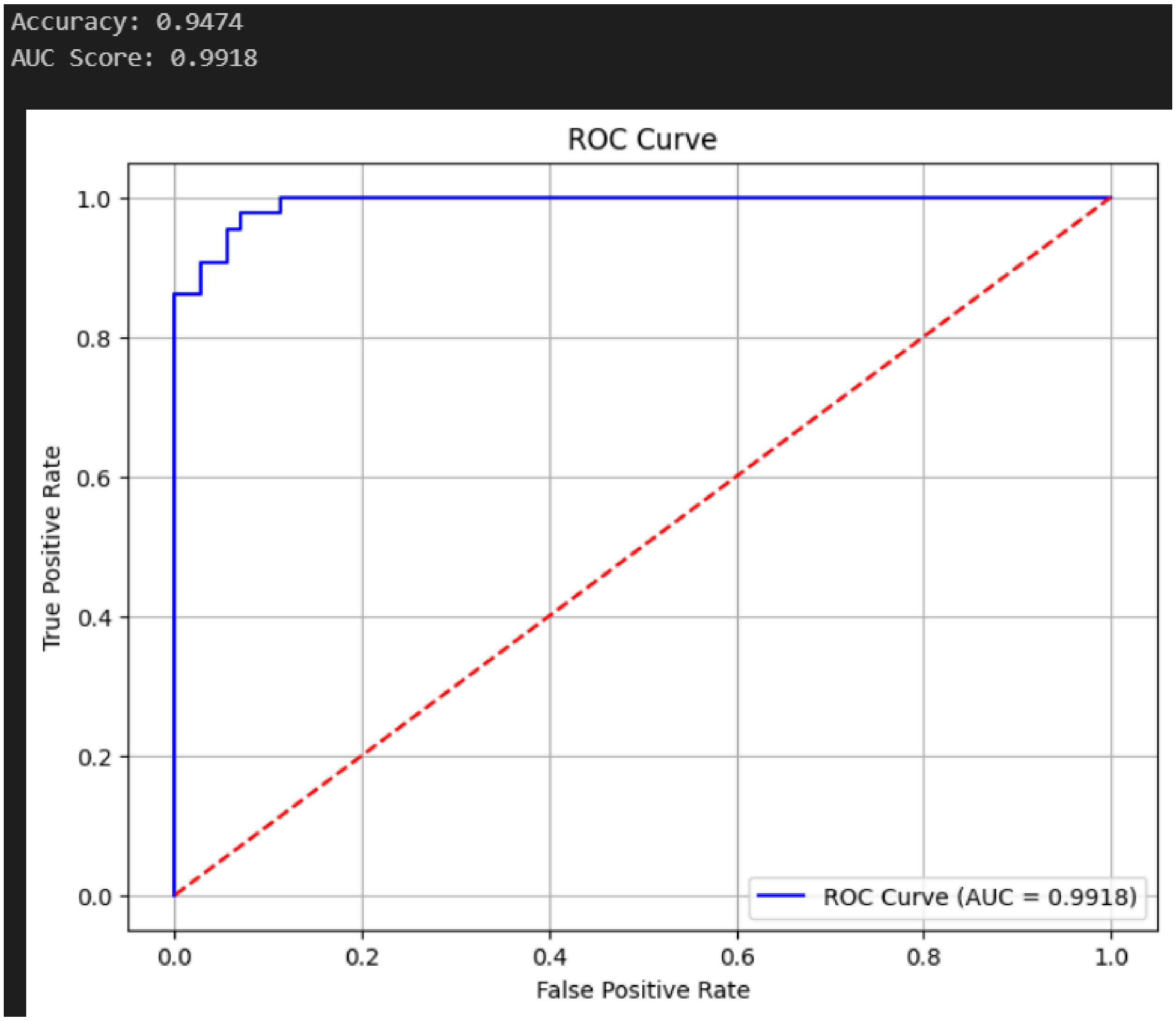
Results

At this point, all of the data has been properly refined to run our machine learning models on this particular dataset. We will be utilizing 3 different machine learning models. Naive Bayes[8], K-Nearest Neighbors (KNN)[9], and Logistic Regression[10]. For all these models, we will be using the Scikit-Learn[7] machine learning library from Python[3]. Our first model is going to be built as a Naive Bayes[8] model from Scikit-Learn[7].

We first split the data into predictors and response variables. Then we split the data again into train and test sets. That way, we can use the same dataset to train the model, and also test its predictive capabilities. Finally, we utilize Matplotlib[4] again to graph the ROC curve to truly visualize the model's accuracy. We will do this exact same process for the other 2 models as well. K-Nearest Neighbors (KNN)[9] and Logistic Regression[10].

Naive Bayes[8] is a simple and computationally efficient machine learning algorithm, making it a suitable choice for tasks like breast cancer diagnosis, especially when starting with a baseline model. Its ability to handle categorical outcomes aligns well with the binary nature of the diagnosis column (malignant or benign), and it provides interpretable outputs in the form of class probabilities, which can help assess prediction certainty. Additionally, Naive Bayes[8] performs well on moderately high-dimensional data and is robust even with relatively small datasets, as long as the data is properly preprocessed. However, the model has notable limitations in this context. A key weakness is its assumption that features are conditionally independent given the class label, which is rarely true in real-world scenarios like breast cancer diagnosis where features such as radius, perimeter, and area are likely correlated. This independence assumption can result in suboptimal performance. Furthermore, while Gaussian Naive Bayes[8] assumes a Gaussian distribution for continuous features, deviations from this assumption may negatively impact the model's effectiveness. Naive Bayes[8] also struggles with capturing complex relationships or interactions between features, which might be critical in accurately diagnosing breast cancer. If the dataset has an imbalanced distribution of malignant and benign cases, the model could be biased toward the majority class unless appropriate preprocessing techniques are applied. Lastly, the probabilities predicted by Naive Bayes[8] can be overconfident due to its simplifying assumptions, which might not accurately reflect the true uncertainty in predictions. Overall, Naive Bayes[8] offers a practical starting point for building a predictive model due to its simplicity and speed. However, given its limitations, it may not be the best choice for capturing the nuanced patterns in the data. Exploring more advanced models, such as Support Vector

Machines, Random Forests, or Gradient Boosting, could provide greater accuracy and reliability, especially in datasets with feature dependencies or complex relationships. Nonetheless, Naive Bayes[8] remains a valuable baseline model for comparative analysis.

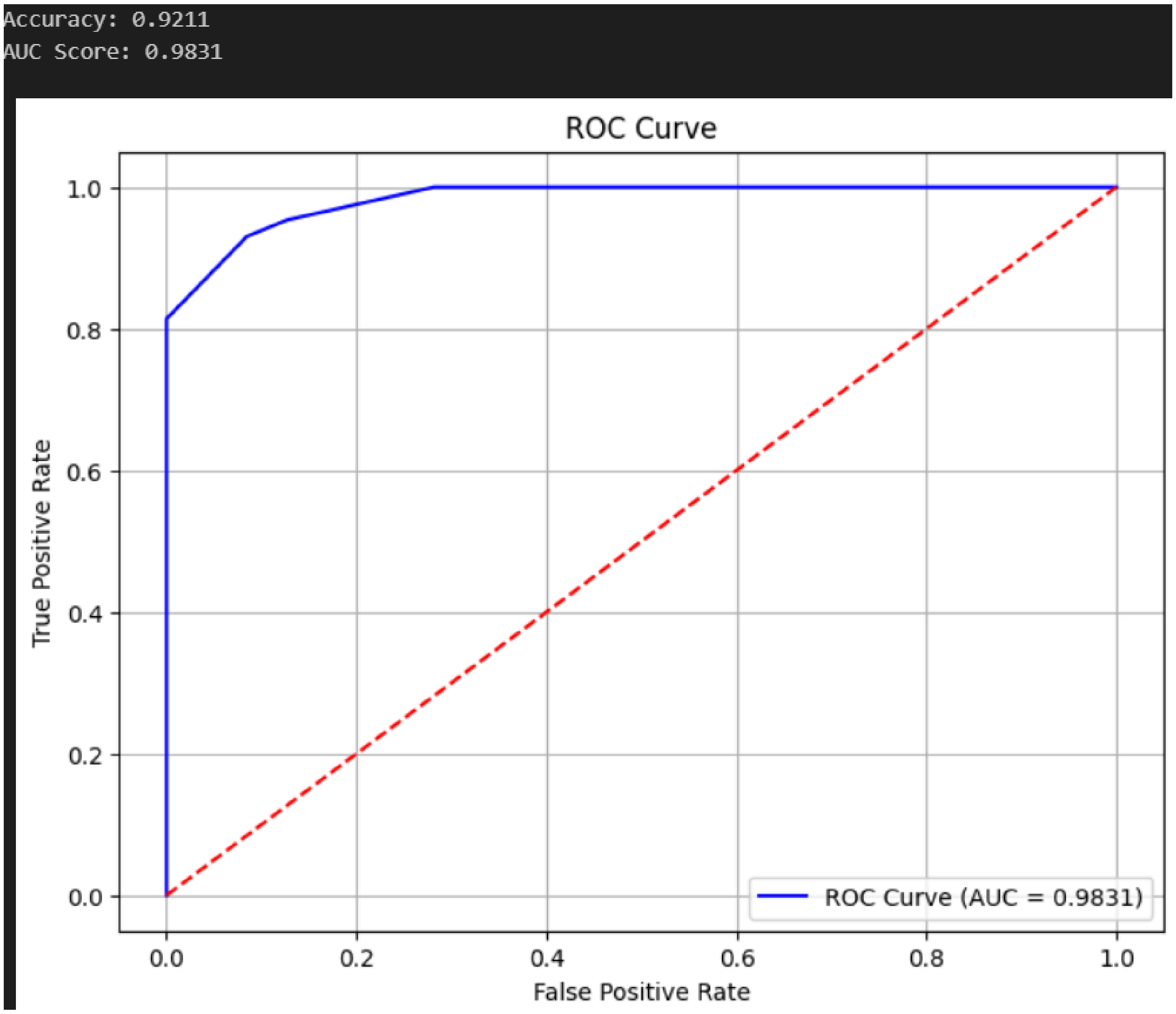


```
Accuracy: 0.9474
AUC Score: 0.9918
```

Accuracy, AUC Score, and ROC Curve visualized after running Scikit-Learn[7]'s Naive Bayes[8] machine learning model on dataset

The performance metrics from the Naive Bayes[8] model—an accuracy of 94.74% and an AUC score of 0.9918—indicate strong predictive performance in diagnosing breast cancer. Accuracy measures the proportion of correctly predicted instances (both malignant and benign) out of the total predictions. An accuracy of 94.74% means that the model correctly classified nearly 95% of the cases in the test set. While this is a high value, accuracy alone can be misleading if the dataset is imbalanced, such as having significantly more benign cases than malignant ones. The AUC score (Area Under the ROC Curve) provides a more robust evaluation of the model's discriminatory ability. It measures how well the model can distinguish between the two classes (malignant and benign) regardless of the chosen decision threshold. An AUC of 0.9918 is extremely high, suggesting that the model is almost perfect at separating malignant tumors from benign ones. For example, if you randomly select a malignant and a benign case, the model would assign a higher probability of being malignant to the malignant case over 99% of the time. These metrics suggest that the Naive Bayes[8] model performs exceptionally well on the given data. The high AUC score particularly indicates that the model is reliable for making predictions, even across varying thresholds. However, it is still essential to consider potential biases, such as whether the data distribution is representative of real-world cases, and validate the model on an independent dataset to ensure generalizability.

K-Nearest Neighbors (KNN)[9] is a simple and intuitive classification algorithm, making it an excellent choice for exploratory analysis in breast cancer diagnosis. Its non-parametric nature means it does not assume any particular distribution of the data, which is advantageous in cases where features do not follow a standard distribution. Additionally, KNN[9] can effectively capture nonlinear decision boundaries, allowing it to identify complex relationships between tumor features and diagnoses. Since the dataset is normalized, KNN[9]'s reliance on distance metrics like Euclidean distance is not hampered by scaling issues, ensuring it can fully utilize the feature space. However, KNN[9] has notable weaknesses in this context. The algorithm is computationally expensive as it must calculate distances between the test point and all training points for each prediction, which can become inefficient as dataset size increases. It is also highly sensitive to noisy data and irrelevant features, as outliers or irrelevant dimensions can significantly skew predictions. The performance of KNN[9] depends heavily on the choice of distance metric; an inappropriate metric may fail to reflect the true relationships in the data. Furthermore, selecting the optimal number of neighbors (k) is crucial, as a small k might lead to overfitting by capturing noise, while a large k could result in underfitting by diluting the influence of relevant neighbors. Like many models, KNN[9] can struggle with imbalanced datasets, as it may bias predictions toward the majority class, particularly with a large k. Additionally, KNN[9] lacks interpretability compared to models like Naive Bayes[8], offering no clear insights into why specific predictions are made. Overall, KNN[9] is a robust choice for capturing complex, non-linear relationships in the dataset and can provide accurate predictions given its simplicity. However, its computational inefficiency, sensitivity to noise, and dependency on careful hyperparameter tuning may limit its practicality as a final model, especially for larger datasets or scenarios requiring fast predictions. While KNN[9] can serve as a useful benchmark model, more advanced techniques, such as Support Vector Machines or ensemble models, may be better suited for real-world applications in breast cancer diagnosis.

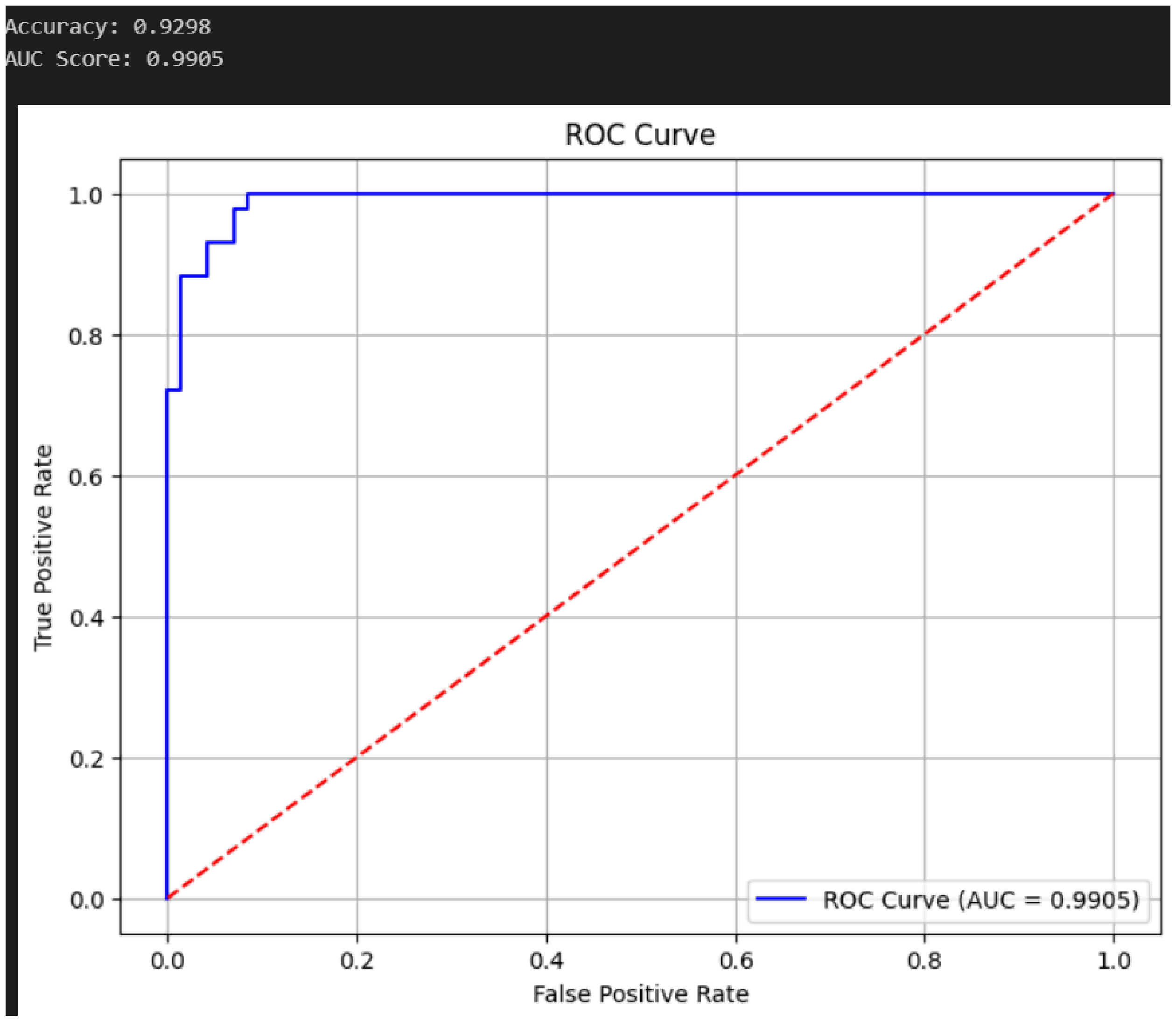


```
Accuracy: 0.9211
AUC Score: 0.9831
```

Accuracy, AUC Score, and ROC Curve visualized after running Scikit-Learn[7]'s K-Nearest Neighbors (KNN)[9] machine learning model on dataset

The performance metrics from the K-Nearest Neighbors (KNN)[9] model—an accuracy of 92.11% and an AUC score of 0.9831—indicate strong, albeit slightly lower, predictive performance compared to the Naive Bayes[8] model. Accuracy, which measures the proportion of correctly predicted instances (both malignant and benign) out of the total predictions, shows that the KNN[9] model correctly classified approximately 92.11% of cases in the test set. While this is a high accuracy, it is marginally lower than the Naive Bayes[8] model's performance. Accuracy, however, should not be interpreted in isolation, especially if the dataset has class imbalance, as it does not account for how well the model distinguishes between classes. The AUC score (Area Under the ROC Curve) provides a more comprehensive measure of the model's ability to separate the two classes (malignant and benign) across all decision thresholds. The AUC score of 0.9831 is exceptionally high, indicating that the model is very effective at distinguishing between malignant and benign tumors. For instance, in randomly selecting one malignant and one benign case, the KNN[9] model would assign a higher probability of malignancy to the malignant case over 98% of the time. While this AUC score is slightly lower than Naive Bayes[8]' AUC of 0.9918, it still reflects strong discriminatory power. In summary, these metrics highlight that KNN[9] performs well in predicting breast cancer diagnoses with high accuracy and excellent AUC. However, its slightly lower performance compared to Naive Bayes[8] may reflect the sensitivity of KNN[9] to the choice of hyperparameters like the number of neighbors (k) or potential noise in the dataset. It underscores the importance of carefully tuning and validating the model to maximize its potential.

Logistic regression[10] is a straightforward and interpretable model that is well-suited for the breast cancer diagnosis task. Its simplicity allows for easy understanding of how changes in tumor features influence the likelihood of malignancy. Logistic regression[10] produces probabilistic outputs, which are valuable for assessing the model's confidence in its predictions and for decision-making based on thresholds. Additionally, its computational efficiency makes it suitable for smaller datasets, like the one in this context, especially when the features are properly normalized and preprocessed. However, logistic regression[10] has some limitations. It assumes that the relationship between the predictors and the log-odds of the outcome is linear. If the relationship is non-linear or complex, the model may struggle to capture it, leading to suboptimal performance. Logistic regression[10] is also sensitive to feature correlations, which can weaken its reliability, particularly if features like radius, perimeter, and area are highly correlated. Its performance can degrade on non-linearly separable data and high-dimensional datasets if there is significant noise or irrelevant features. Moreover, logistic regression[10] can struggle with class imbalance, favoring the majority class unless appropriate adjustments, such as re-sampling or class weighting, are made. Overall, logistic regression[10] is a strong choice for breast cancer diagnosis when the relationships between features and diagnosis are approximately linear. Its simplicity and efficiency make it an excellent baseline model. However, its reliance on linear assumptions and sensitivity to multicollinearity may limit its ability to capture complex relationships fully. Therefore, while logistic regression[10] can perform well in this context, its performance should be compared to more flexible models like Support Vector Machines or ensemble models to ensure optimal diagnostic accuracy.



```
Accuracy: 0.9298
AUC Score: 0.9905
```

Accuracy, AUC Score, and ROC Curve visualized after running Scikit-Learn[7]'s Logistic Regression[10] machine learning model on dataset

The printed values from the logistic regression[10] model—an accuracy of 92.98% and an AUC score of 0.9905—indicate strong performance for predicting breast cancer diagnoses. Accuracy measures the proportion of correct predictions made by the model out of the total predictions. With an accuracy of 92.98%, the logistic regression[10] model correctly classifies approximately 93% of the test cases as either malignant (M) or benign (B). This is a high accuracy, indicating that the model is quite effective at predicting the correct diagnosis for the majority of cases in the test dataset. Compared to K-Nearest Neighbors, this accuracy is slightly higher, suggesting that logistic regression[10] may be better at generalizing from the training data to the test set, possibly due to its simpler and more direct approach. AUC Score (Area Under the ROC Curve) provides a more nuanced measure of the model's discriminative power. An AUC score of 0.9905 is exceptionally high, indicating that the logistic regression[10] model is very capable of distinguishing between malignant and benign tumors across all possible classification thresholds. The AUC measures the model's ability to separate the two classes, with a score close to 1.0 showing excellent performance. This score indicates that the model not only performs well but also does so with high confidence, meaning that its predictions are likely to be correct for the majority of cases. Compared to other models, this AUC score is slightly lower than the Naive Bayes[8] model's 0.9918, but still very strong. In summary, these metrics demonstrate that logistic regression[10] is a robust model for breast cancer diagnosis, offering high accuracy and excellent discriminatory ability as indicated by the AUC score. The high accuracy suggests that the model generalizes well to new, unseen data, while the AUC score reflects the model's ability to distinguish between the two classes, making logistic regression[10] a strong candidate for predictive tasks in medical diagnosis.

Discussion

Given the printed values for accuracy and AUC scores across three different models—Naive Bayes[8], K-Nearest Neighbors (KNN)[9], and Logistic Regression[10]—we can identify the best performing model based on these metrics.

Naive Bayes[8] has the highest accuracy of 94.74% and an AUC score of 0.9918. This indicates that the model is highly effective at predicting the correct diagnosis for breast cancer, with a strong ability to distinguish between malignant and benign tumors. The high AUC score, in particular, suggests excellent discriminatory power, meaning that the model is capable of confidently distinguishing between the two classes across all decision thresholds. Naive Bayes[8]' success can be attributed to its straightforward assumptions of conditional independence among features and its efficient handling of high-dimensional data, which are suitable for the breast cancer dataset.

K-Nearest Neighbors (KNN)[9], while slightly lower in accuracy (92.11%) and AUC (0.9831), is still a strong performer, demonstrating good predictive capability. KNN[9]'s performance is particularly notable due to its ability to capture complex, non-linear relationships between features. However, its computational inefficiency and sensitivity to noise and outliers limit its suitability as the top model for this task. The relatively lower AUC score compared to Naive Bayes[8] indicates that while KNN[9] is capable of good classification, it may struggle slightly in confidently distinguishing between classes in some cases.

Logistic Regression[10] offers an accuracy of 92.98% and an AUC score of 0.9905, making it the second-best performing model. It combines the benefits of interpretability and computational efficiency with strong predictive performance. The simplicity of logistic regression[10], coupled with its calibrated probability outputs, makes it useful for straightforward decision-making. Its performance being slightly less than Naive Bayes[8] in both metrics indicates that while logistic regression[10] is an excellent choice for interpretability and quick, efficient training, it may not fully capture the complexities of the data as well as Naive Bayes[8] does.

Lessons Learned from this analysis show the importance of model selection based on the nature of the dataset. Naive Bayes[8] performed the best, likely due to its effective handling of high-dimensional and complex relationships, and its capability to work well even with imbalanced classes. KNN[9] provided a good baseline for capturing non-linear relationships but was limited by computational cost and sensitivity to data noise. Logistic regression[10] offered a balanced trade-off between simplicity and predictive power, but its linearity assumption prevented it from being the top choice.

Conclusion: The best performing model, Naive Bayes[8], is particularly useful for situations where data is abundant and features are complex but approximately independent. Its simplicity and robust performance across different thresholds make it highly suitable for breast cancer diagnosis. The choice of Naive Bayes[8] reflects the importance of model assumptions aligning well with the underlying data structure, and its higher AUC score signifies superior ability to make accurate predictions consistently.

Limitations of this work include the small size of the dataset, which may not fully challenge the models, and the assumptions made during model training, such as feature independence in Naive Bayes[8]. Additionally, more sophisticated models like Support Vector Machines or ensemble methods were not explored, which could have offered better performance, especially for complex data relationships. Further model validation on larger datasets, and the exploration of hyperparameter tuning, would likely improve the findings and offer deeper insights into model performance.

References

1. "Breast Cancer Wisconsin (Diagnostic)." UCI Machine Learning Repository, archive.ics.uci.edu/dataset/17/breast+cancer+wisconsin+diagnostic.
2. "Pandas." Pandas, pandas.pydata.org/.
3. "Welcome to Python.Org." *Python.Org*, www.python.org/.
4. "Visualization with Python." Matplotlib, matplotlib.org/.
5. "Statistical Data Visualization#." Seaborn, seaborn.pydata.org/.
6. *NumPy*, numpy.org/.
7. "Learn." Scikit, scikit-learn.org/stable/.
8. "Naive Bayes Classifier." Wikipedia, Wikimedia Foundation, 28 Nov. 2024, en.wikipedia.org/wiki/Naive_Bayes_classifier.
9. "K-Nearest Neighbors Algorithm." Wikipedia, Wikimedia Foundation, 23 Nov. 2024, en.wikipedia.org/wiki/K-nearest_neighbors_algorithm.
10. "Logistic Regression." Wikipedia, Wikimedia Foundation, 15 Oct. 2024, en.wikipedia.org/wiki/Logistic_regression.