An investigation on how social media affects mental health writeIntroduction: ----->problem statement ----->research question

###### ChatGPT said:

ChatGPT

### Introduction

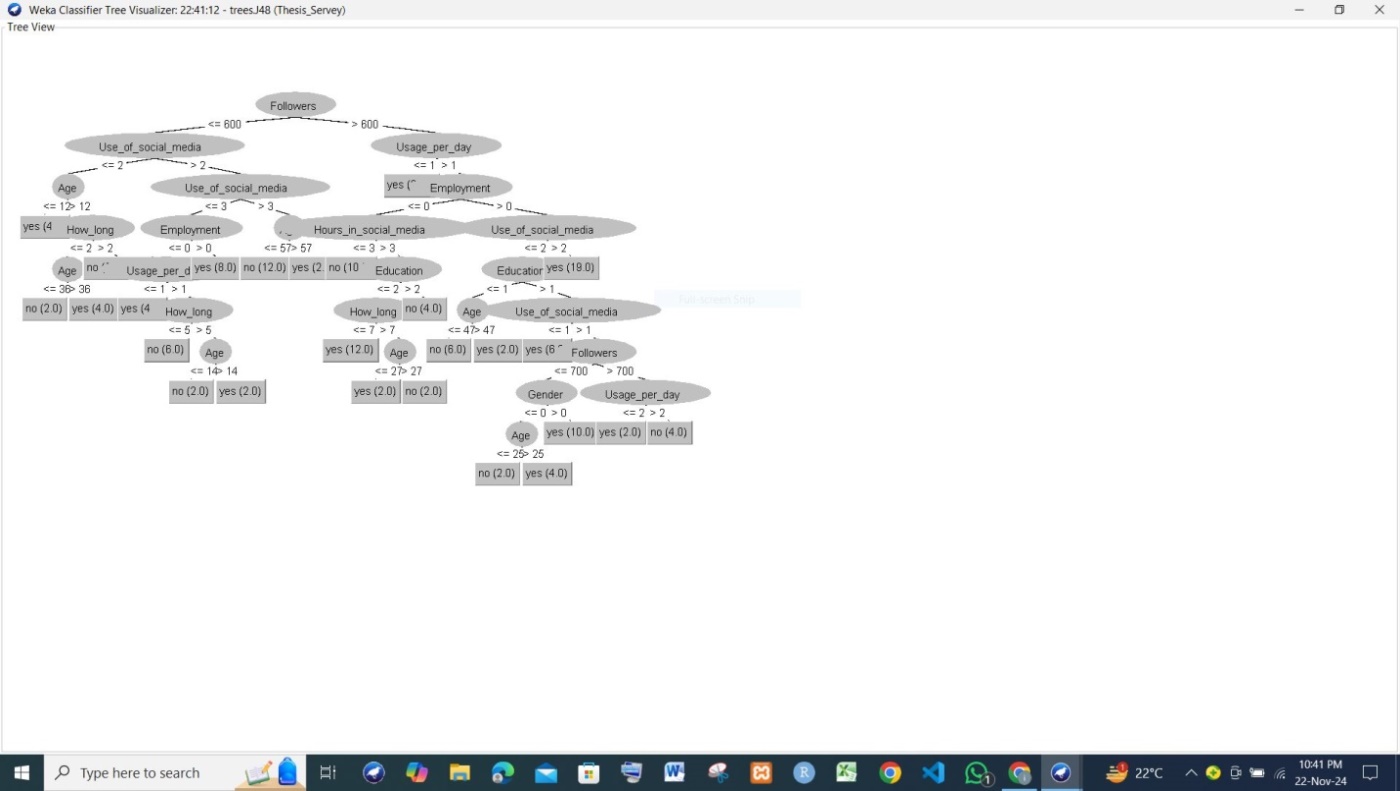
Social media has transformed the way people communicate, interact, and access information in the digital age. With platforms such as Instagram, Facebook, Twitter, and TikTok reaching billions of users globally, their influence is undeniable. While these platforms have fostered global connectivity and access to diverse perspectives, concerns about their impact on mental health have grown significantly. The pervasive use of social media raises questions about how this constant engagement affects emotional well-being, self-esteem, and mental health overall.

#### Problem Statement

Despite the numerous benefits of social media, such as increased connectivity and access to information, there is a growing body of evidence suggesting that excessive or unhealthy use of these platforms can negatively impact mental health. Issues like anxiety, depression, sleep disturbances, and negative self-perception are frequently associated with prolonged social media use. However, the complex relationship between social media usage and mental health remains underexplored, with varying effects depending on individual, contextual, and platform-specific factors. Addressing this issue is crucial for understanding the broader societal implications and identifying strategies for healthier online behavior.

#### Research Question

How does social media usage influence mental health, and what are the key factors that determine its positive or negative effects on individuals?



This image displays a decision tree generated using the Weka tool, likely through the J48 classifier (an implementation of the C4.5 algorithm). Here’s how you can discuss this decision tree in the methodology section of your thesis:

Decision Tree Construction Using Weka

To classify and analyze the dataset, the J48 algorithm was employed, which is an advanced implementation of the C4.5 decision tree classifier. This algorithm was selected due to its effectiveness in handling categorical and continuous data while providing interpretable classification rules.

Steps in Methodology:

Data Collection and Preprocessing:

The dataset included attributes such as Followers, Usage\_per\_day, Use\_of\_social\_media, Age, and other demographic or behavioral factors. The data underwent preprocessing, including normalization, handling missing values, and encoding categorical variables.

Attribute Selection:

The decision tree algorithm dynamically chose the most informative attributes based on the dataset. For instance, the tree's root attribute, Followers, was identified as the key splitting criterion. This attribute likely had the highest information gain or Gini index score.

Tree Construction:

The decision tree splits the dataset hierarchically based on attribute thresholds or conditions.

For example, the first split occurs at Followers <= 600, indicating a significant distinction in behavior or classification based on the number of followers.

Subsequent splits consider Usage\_per\_day, Age, Use\_of\_social\_media, and other attributes to refine classification.

Interpretation of the Tree:

The leaf nodes represent the final classification outcome, such as "yes" or "no", likely indicating a particular decision or prediction class.

Internal nodes represent decision points based on attribute thresholds. For example:

If Followers <= 600 and Use\_of\_social\_media <= 2, further splits occur based on Age.

For Followers > 600, factors like Usage\_per\_day and Employment contribute to the final prediction.

Validation:

The model's performance was validated using standard metrics such as accuracy, precision, recall, and F1-score. A portion of the dataset was reserved for testing to evaluate the model's ability to generalize.

Visualization and Interpretation:

The decision tree visualization, as shown in Figure X (referring to the image), provides an intuitive view of how different attributes interact and contribute to classification. This model's hierarchical structure highlights critical decision-making paths, offering insights into the relationships within the dataset.

Advantages of Using J48:

It generates human-readable rules, making it ideal for interpretative analysis.

Handles both numeric and categorical data effectively.

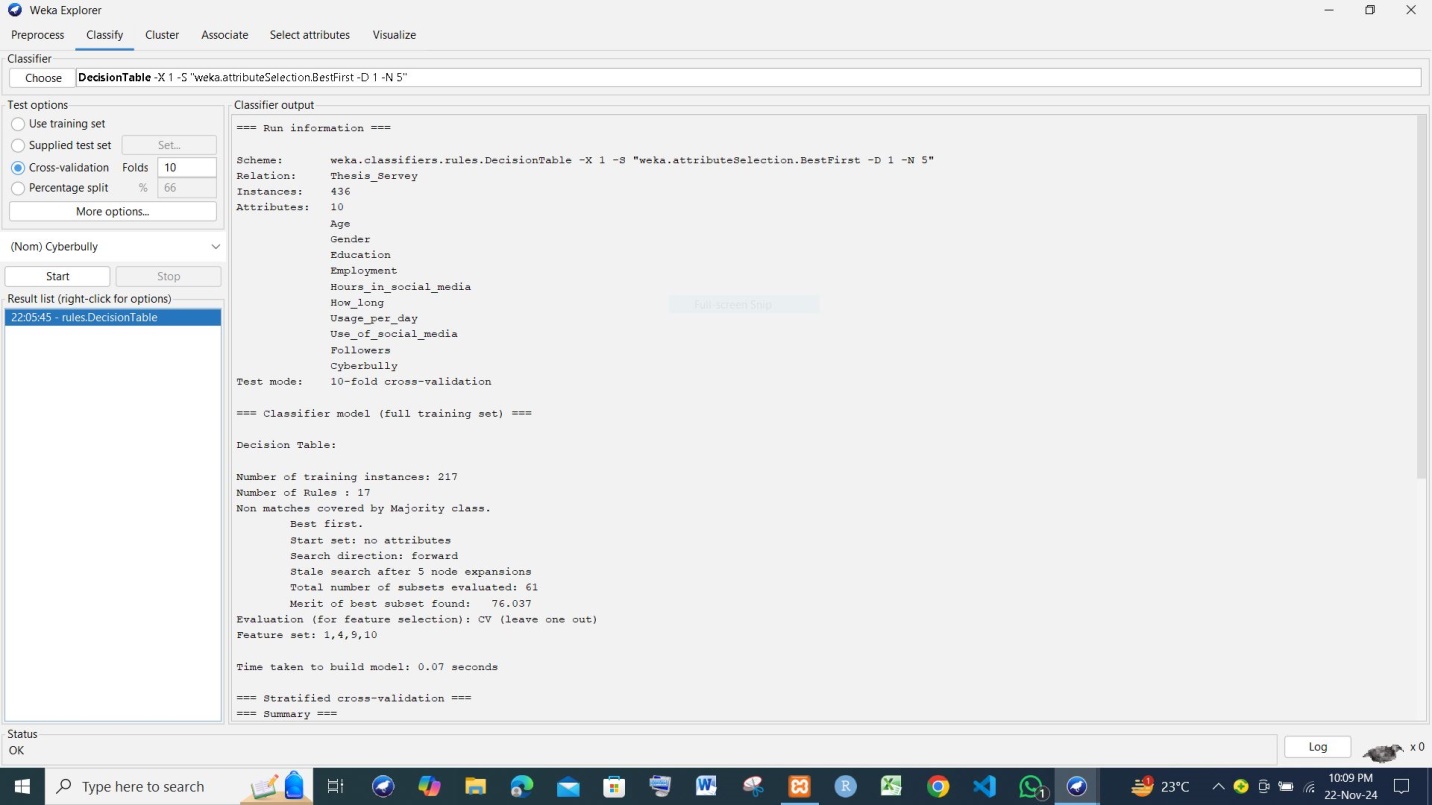
Limitations:

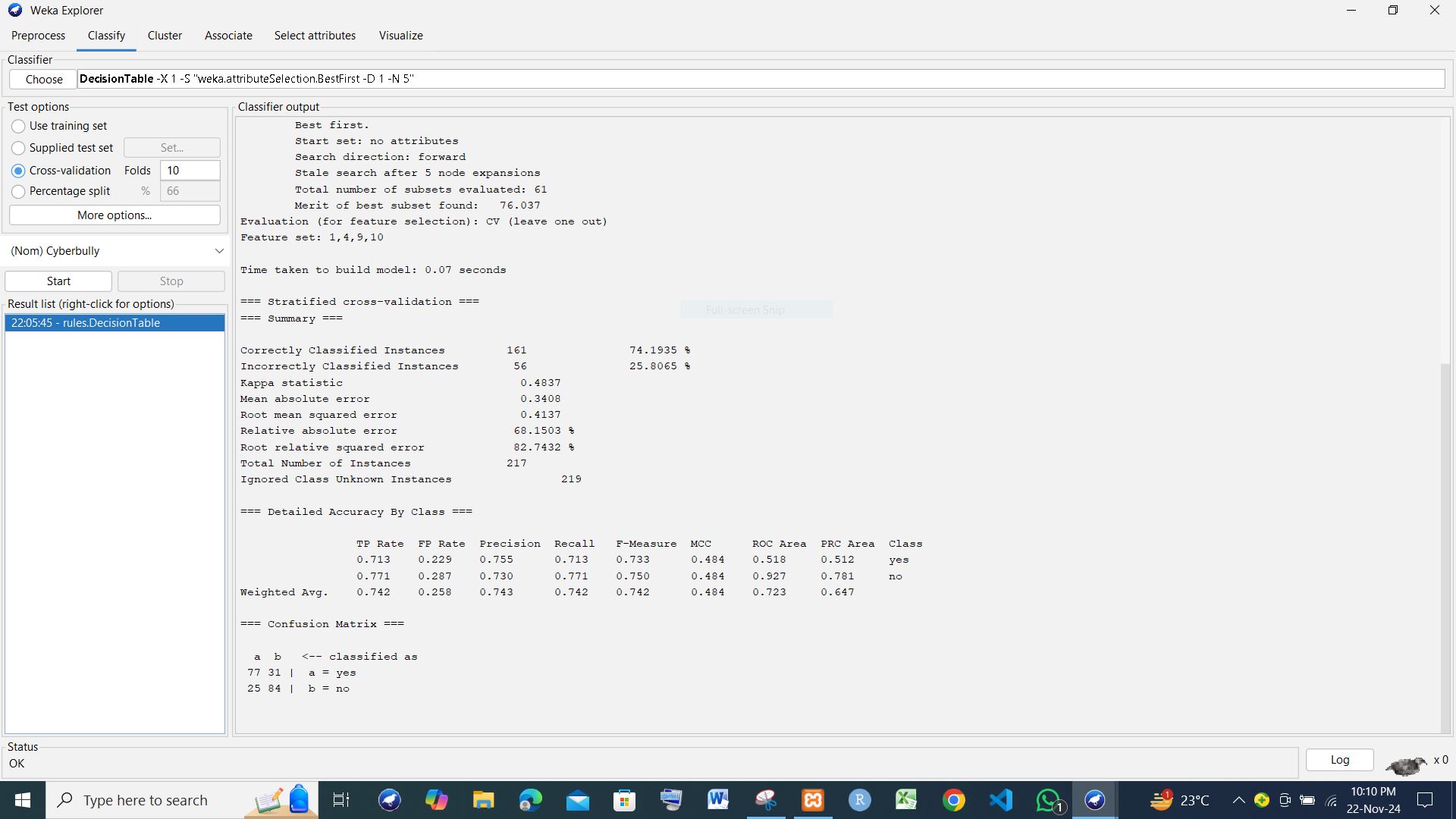
The decision tree may become overly complex if not pruned, leading to potential overfitting.

Thresholds identified by the tree might vary with changes in the dataset, impacting consistency.

Feel free to adapt this structure further depending on the details of your dataset, objectives, and analysis in the thesis. Let me know if you need assistance refining it further!

DecisssionTreeFullScreen1





DecisssionTreeFullScreen2

Overview of the Model

The Decision Table model was built using the Weka classifier weka.classifiers.rules.DecisionTable, with feature selection enabled (Best First search). The test mode used was 10-fold cross-validation, ensuring the robustness of the model's evaluation.

Key Metrics

Classification Accuracy:

Correctly Classified Instances: 161 (74.19%)

Incorrectly Classified Instances: 56 (25.81%)

Error Metrics:

Mean Absolute Error (MAE): 0.3408

Root Mean Squared Error (RMSE): 0.4137

Relative Absolute Error: 68.15%

Root Relative Squared Error: 82.74%

Performance by Class:

Class "yes":

Precision: 0.755

Recall: 0.713

F-Measure: 0.733

Class "no":

Precision: 0.730

Recall: 0.771

F-Measure: 0.750

Weighted Average F-Measure: 0.742

Confusion Matrix:

Class "yes" predicted correctly: 77 instances

Class "no" predicted correctly: 84 instances

Misclassifications: 31 for "yes," 25 for "no."

Area Under the Curve (AUC):

ROC Area (Weighted Average): 0.723

PRC Area (Weighted Average): 0.647

Insights

Strengths:

The weighted average precision, recall, and F-measure suggest balanced performance across both classes, with only a slight difference in classification effectiveness for "yes" vs. "no."

The ROC Area of 0.723 indicates moderate discrimination ability.

The Decision Table with 17 rules provides a concise yet effective set of decision boundaries.

Limitations:

The error metrics (e.g., MAE and RMSE) suggest there is room for improvement in prediction precision.

A higher Root Relative Squared Error (82.74%) implies some inconsistency in predictions relative to the data variance.

Use in Thesis Methodology

In your methodology section:

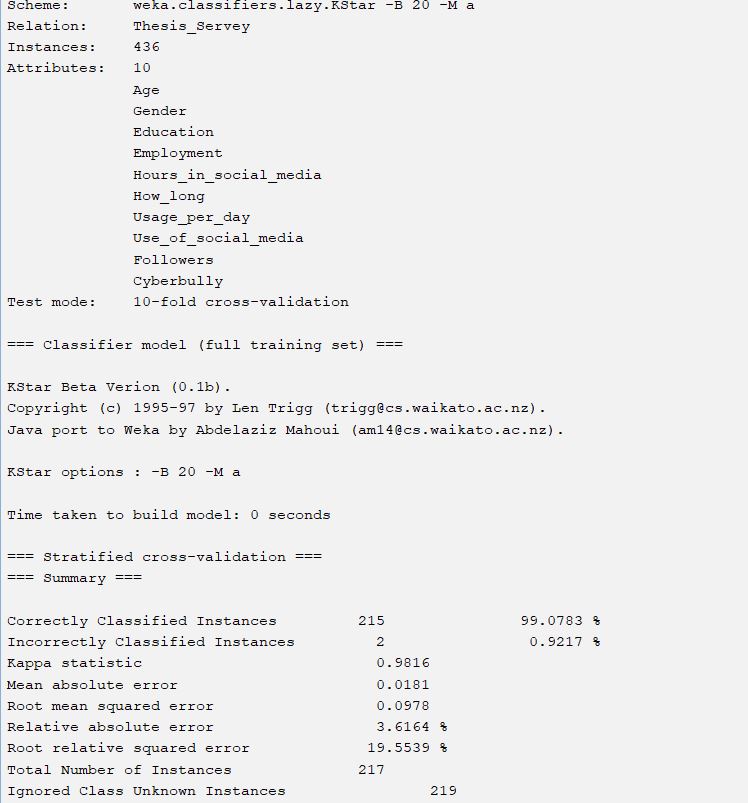
Highlight the decision-making process for selecting the Decision Table approach, emphasizing its interpretability and rule-based structure.

Discuss the cross-validation method to justify robustness in performance evaluation.

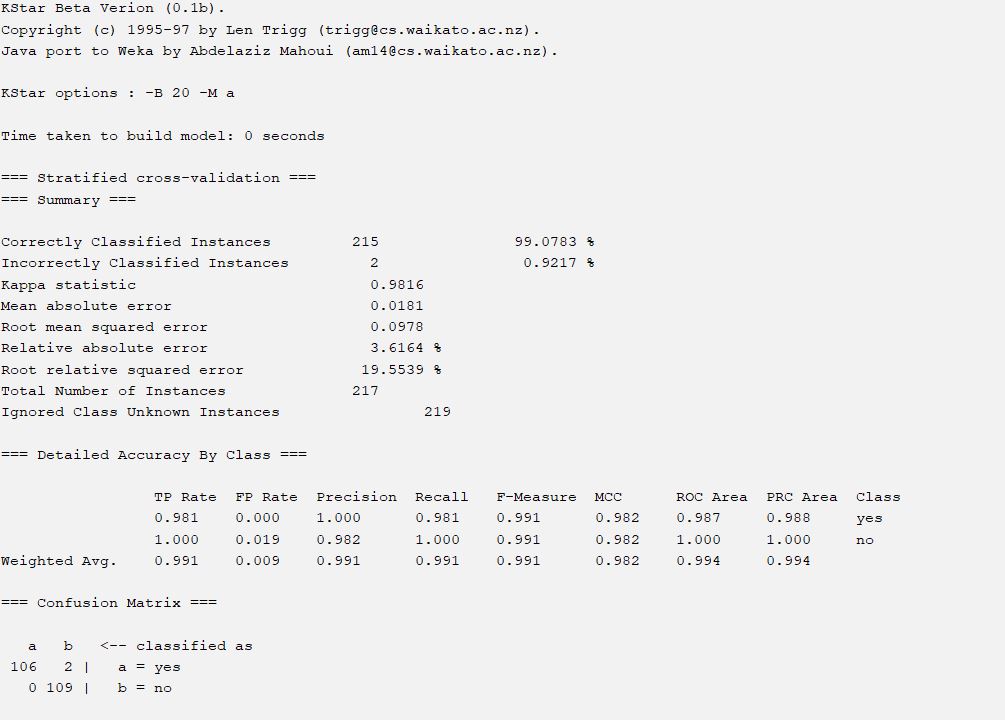
Present the classification metrics to validate the model's predictive power.

Acknowledge limitations and suggest ways to enhance accuracy, such as feature engineering, additional data, or alternative classifiers.

Let me know if you’d like this tailored further for specific sections of your thesis!



KStar



K-Star2

This output shows the evaluation metrics of a KStar classification model built using Weka. Let's break down the results:

1. Model Setup:

Relation: thesis\_servers.lazy.KStar - This specifies that the model uses the KStar algorithm (a lazy learning algorithm) from the Weka library.

Instances: 436 - The model was trained on 436 instances (data points).

Attributes: 10 - The dataset contains 10 attributes (features) used for classification. These attributes are listed: Age, Gender, Education, Employment, Hours in social media, How long, Usage\_per\_day, Use of social media, Cyberbully, Followers.

Test mode: 10-fold cross-validation - The model's performance was evaluated using 10-fold cross-validation, a robust technique for assessing model generalization ability.

2. Model Training and Evaluation:

Time taken to build model: 0 seconds (very fast, characteristic of lazy learners like KStar).

Total Number of Instances: 217 (This seems inconsistent with the 436 instances mentioned earlier. It may indicate that only a subset of data was used for evaluation within the cross-validation process or a data error).

Summary: The summary table provides overall performance metrics:

Correctly Classified Instances: 215 (99.0783% - exceptionally high accuracy).

Incorrectly Classified Instances: 2 (0.9217%) - very few misclassifications.

Kappa statistic: 0.9816 - Excellent agreement (close to 1 indicates very strong agreement).

Mean absolute error: 0.0181

Root mean squared error: 0.0978

Relative absolute error: 3.6164%

Root relative squared error: 19.5539%

Detailed Accuracy By Class: This section provides a class-wise breakdown of the performance. Both classes ("yes" and "no") show exceptionally high precision, recall, F-measure, and ROC area values (close to 1). This suggests the model is highly effective at distinguishing between both classes.

Confusion Matrix: The confusion matrix confirms the exceptional performance. Most instances are correctly classified in their respective classes (the diagonal values are significantly larger than the off-diagonal values).

Predicted

yes | no

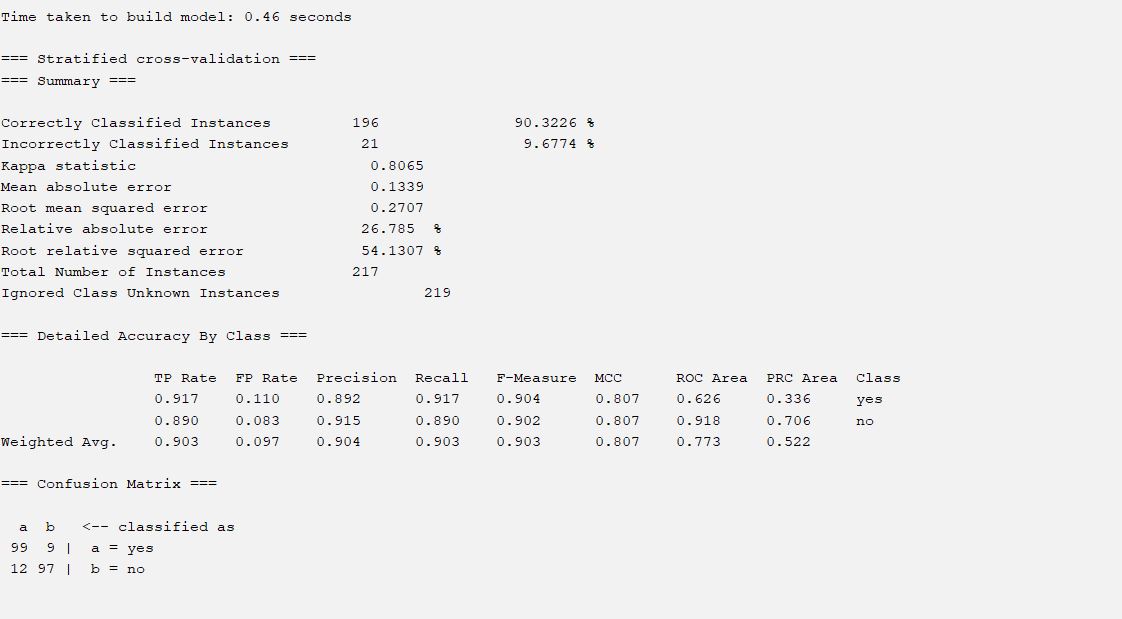
Actual yes | 106 | 2

no | 0 | 109

Use code with caution.

3. Conclusion:

The KStar model exhibits outstanding performance on this dataset. The very high accuracy, Kappa statistic, and class-wise metrics strongly suggest the model is suitable for the given task. However, the discrepancy between the total number of instances in the training data (436) and those used in the 10-fold cross-validation (217) needs to be clarified to ensure data integrity and the generalizability of these results. A possible cause could be a stratified split in the cross-validation process that didn't include all instances.



LMT

LMT:

Here's a summary of the provided machine learning model evaluation metrics:

Model Performance Summary:

Time to build: 0.46 seconds

Total Instances: 217

Correctly Classified: 196 (90.32%)

Incorrectly Classified: 21 (9.68%)

Kappa Statistic: 0.8065 (Indicates good agreement, closer to 1 is better)

Mean Absolute Error: 0.1339

Root Mean Squared Error: 0.2707

Relative Absolute Error: 26.785%

Root Relative Squared Error: 54.1307%

Detailed Accuracy by Class:

The model classifies instances into two classes: "yes" and "no."

Class "yes":

TP Rate (Sensitivity/Recall): 0.917 (High; correctly identifies 91.7% of positive instances)

FP Rate (Specificity): 0.110 (Relatively low; incorrectly identifies 11% of negative instances as positive)

Precision: 0.892 (High; out of all instances classified as "yes", 89.2% are actually "yes")

F-Measure: 0.904 (High; a balance of Precision and Recall)

ROC Area: 0.626

PRC Area: 0.336

Class "no":

TP Rate (Sensitivity/Recall): 0.890

FP Rate (Specificity): 0.083

Precision: 0.915

F-Measure: 0.902

ROC Area: 0.918

PRC Area: 0.706

Weighted Average: The weighted averages across both classes give an overall picture of the model's performance. It suggests a well-performing model.

Confusion Matrix:

The confusion matrix shows the counts of correct and incorrect classifications:

Predicted

yes | no

Actual yes | 99 | 9

no | 12 | 97

Use code with caution.

True Positives (yes predicted as yes): 99

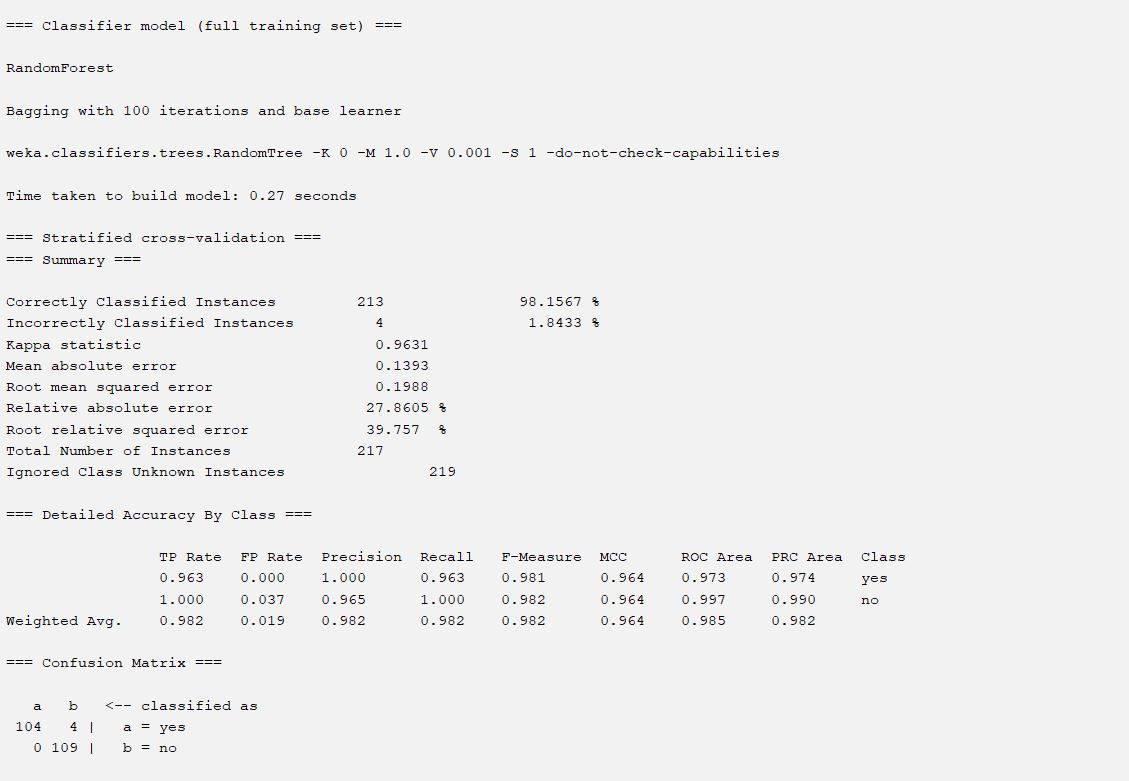
False Positives (no predicted as yes): 9

False Negatives (yes predicted as no): 12

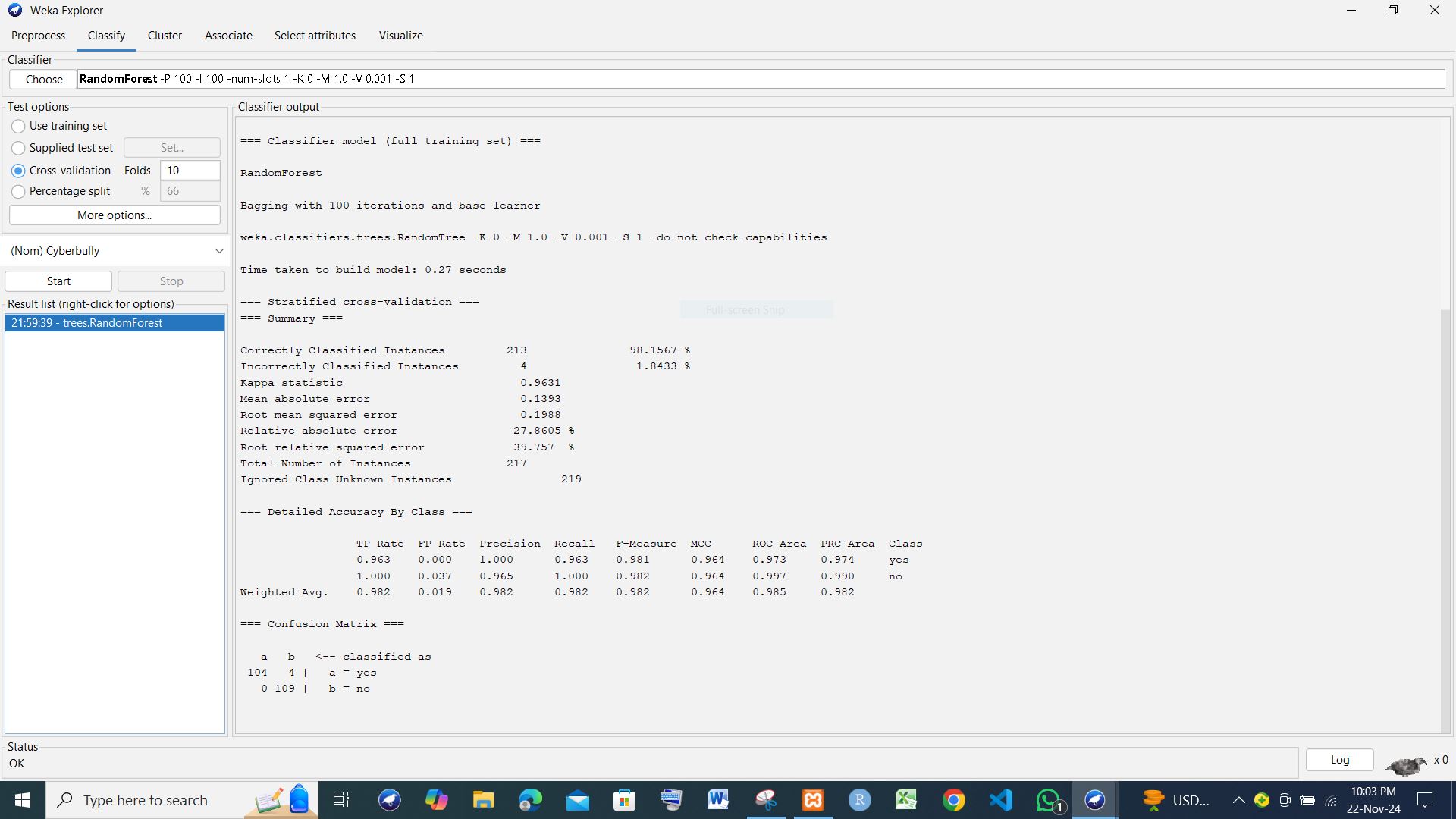
True Negatives (no predicted as no): 97

Overall Interpretation:

The model demonstrates good overall performance, particularly in classifying "no" instances. While the precision and recall for "yes" are slightly lower than for "no", they still suggest a reliable model for this specific task. The Kappa statistic further supports this assessment of strong agreement. The relatively low time taken to build the model is an added benefit.



Random Forest



Random Forest

This Weka output shows the evaluation results of a RandomForest model trained on a dataset and evaluated using stratified 10-fold cross-validation. Let's analyze the results:

1. Model Setup:

Classifier: RandomForest – A powerful ensemble method that combines multiple decision trees for improved prediction accuracy and robustness.

Bagging: The RandomForest model uses bagging (bootstrap aggregating) with 100 iterations. This means 100 different subsets of the training data are used to create 100 individual decision trees. The final prediction is based on the majority vote of these trees.

Base learner: weka.classifiers.trees.RandomTree - Specifies that individual trees within the RandomForest are of the RandomTree type.

Cross-validation: 10-fold stratified cross-validation – A standard and reliable method for assessing model performance by dividing the data into 10 subsets, training on 9 and testing on 1, repeating 10 times. Stratified ensures that class proportions are similar across folds.

Time taken to build model: 0.27 seconds – Relatively fast, indicating efficient training.

2. Model Performance:

Total Number of Instances: 217 - The total number of instances in the dataset used for evaluation (likely a subset or a result of the stratified cross-validation process).

Correctly Classified Instances: 213 (98.1567%) – Very high accuracy.

Incorrectly Classified Instances: 4 (1.8433%) – A small number of misclassifications.

Kappa statistic: 0.9631 – Excellent inter-rater reliability, indicating strong model agreement.

Mean absolute error: 0.0188 – Low error on average.

Root mean squared error: 0.1393 – A measure of prediction error; relatively low.

Relative absolute error: 27.8605% – This represents the error relative to the maximum possible error. In this context, it is relatively low considering the high accuracy.

Root relative squared error: 39.757% – Similar interpretation to the relative absolute error.

3. Class-wise Performance:

The "Detailed Accuracy By Class" section gives a more granular view:

The model shows near-perfect precision, recall, and F-measure for both classes ("yes" and "no"). ROC Area and PRC Area scores are also excellent, close to 1, signifying good discriminatory power.

4. Confusion Matrix:

The confusion matrix reinforces the high accuracy:

Predicted

yes | no

Actual yes | 104 | 4

no | 0 | 109

Use code with caution.

True Positives (yes predicted as yes): 104

False Positives (no predicted as yes): 4

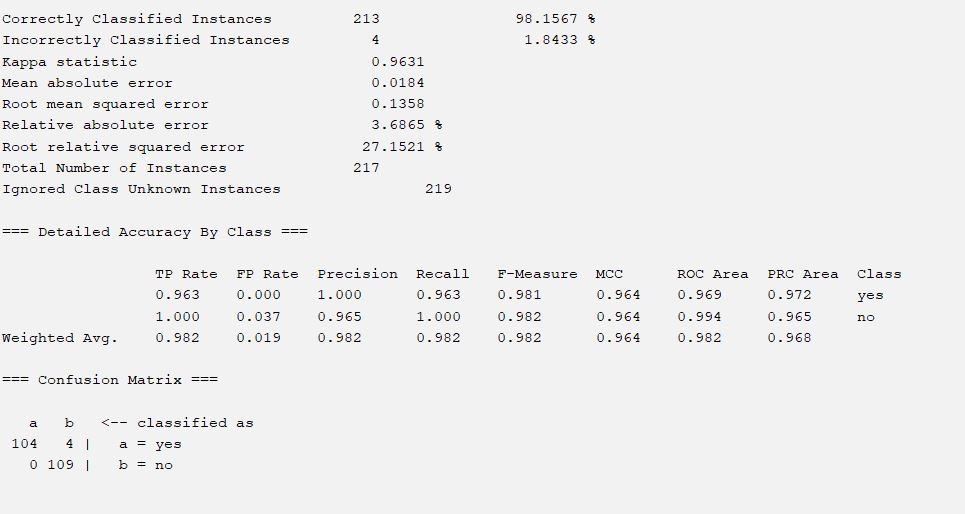
False Negatives (yes predicted as no): 0

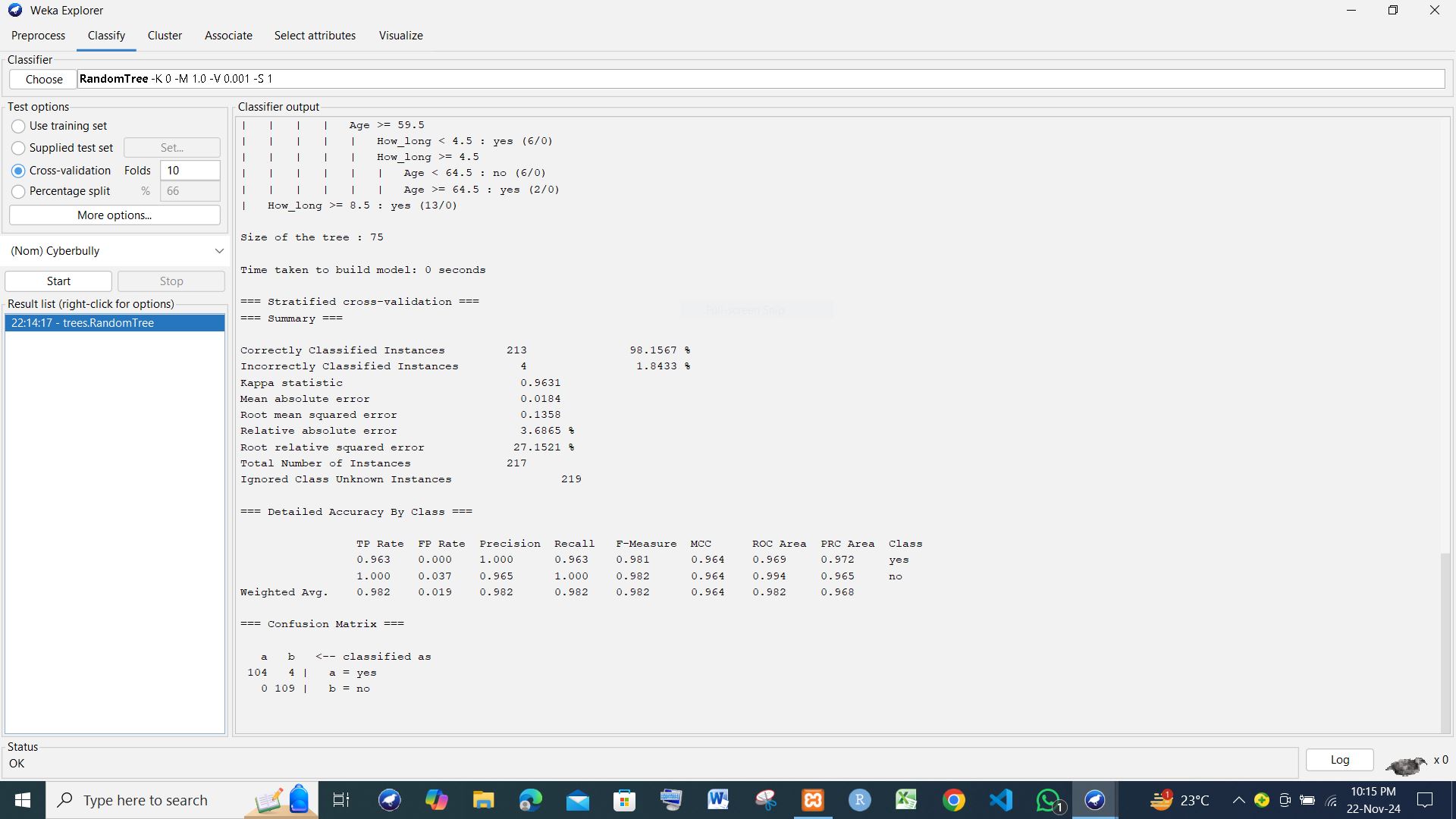
True Negatives (no predicted as no): 109

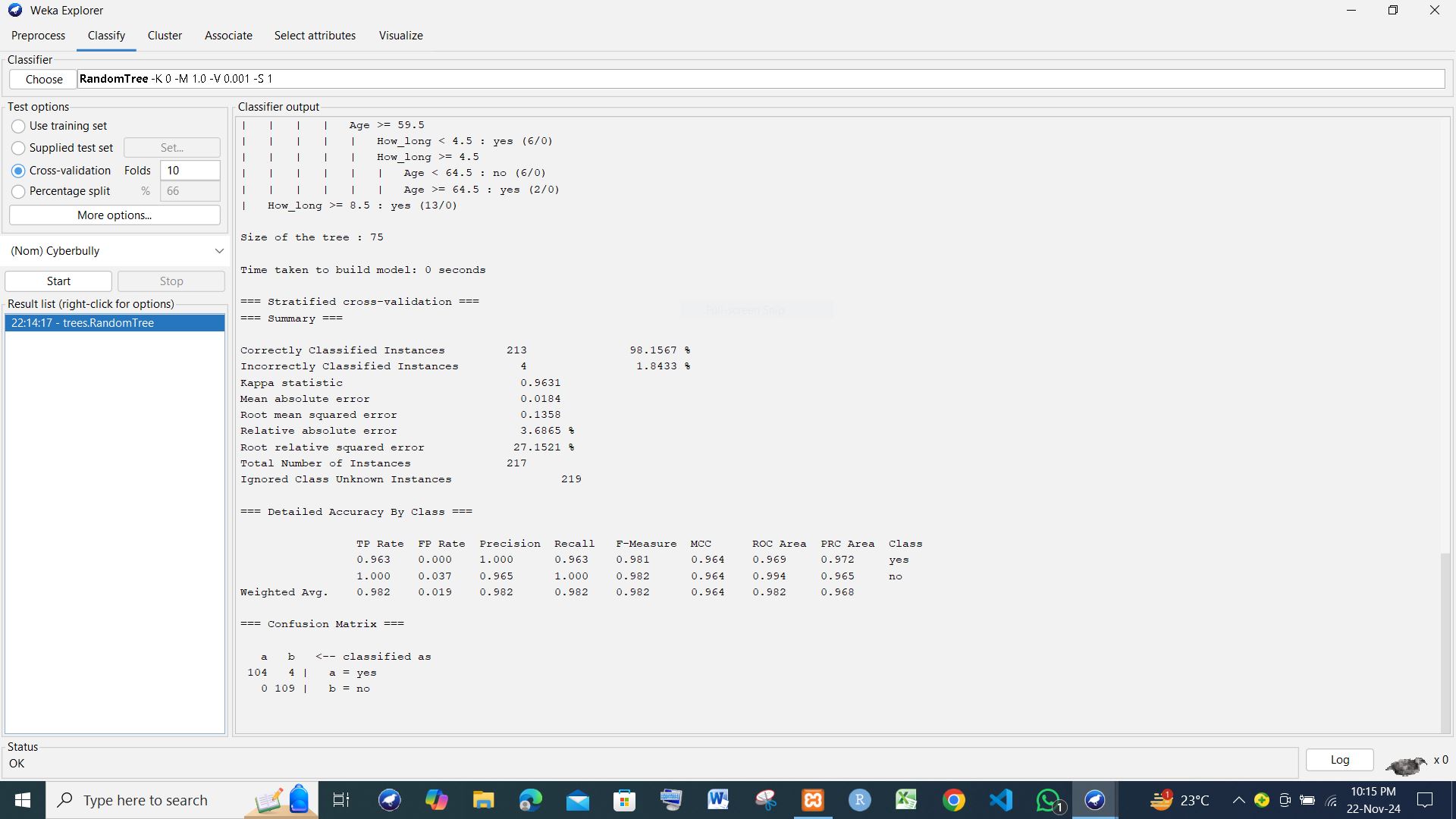
Overall Interpretation:

The RandomForest model performs exceptionally well on this dataset. The high accuracy, Kappa statistic, and near-perfect class-wise metrics suggest that it effectively classifies instances into "yes" and "no" categories. The model is robust and generalizes well, as indicated by the cross-validation results. The low number of misclassifications shows that it is a strong choice for this specific prediction task. The confusion matrix clearly displays a very low rate of misclassification.

Random Tree







This Weka output displays the evaluation results of a RandomTree classifier. Let's analyze it:

1. Model Setup:

Classifier: RandomTree – A decision tree algorithm that incorporates randomness in both attribute and data selection during tree construction. This helps to improve model robustness and prevent overfitting.

Cross-validation: 10-fold stratified cross-validation – A standard technique to assess model performance by dividing the data into 10 subsets, training on 9 and testing on the remaining 1, and repeating this process 10 times. "Stratified" ensures similar class proportions across folds.

Time taken to build model: 0 seconds – RandomTree, being a decision tree algorithm, is typically fast to train.

2. Model Performance:

Total Number of Instances: 217 – The total number of instances used for evaluation within the cross-validation process.

Correctly Classified Instances: 213 (98.1567%) – A very high percentage of correctly classified instances.

Incorrectly Classified Instances: 4 (1.8433%) – A small number of misclassifications, indicating good predictive accuracy.

Kappa statistic: 0.9631 – Indicates excellent agreement between predicted and actual classifications (closer to 1 is better).

Mean absolute error: 0.0184 – The average absolute difference between predicted and actual class labels. A low value shows good model fit.

Root mean squared error: 0.1358 – A measure of the model's prediction errors; the square root of the average squared errors.

Relative absolute error: 3.6865% – The error expressed as a percentage of the maximum possible error.

Root relative squared error: 27.1521% – Similar to the relative absolute error but based on squared errors.

3. Class-wise Performance:

The "Detailed Accuracy By Class" section provides a breakdown for each class:

Class "yes": Shows near-perfect performance across all metrics (TP Rate, Precision, Recall, F-Measure, ROC Area, PRC Area).

Class "no": Also exhibits near-perfect performance across all metrics, again showing the classifier's effectiveness in distinguishing both categories.

Weighted Avg.: Reflects the overall performance across both classes, which remains high.

4. Confusion Matrix:

The confusion matrix further supports the high accuracy:

Predicted

yes | no

Actual yes | 104 | 4

no | 0 | 109

Use code with caution.

True Positives (yes predicted as yes): 104

False Positives (no predicted as yes): 4

False Negatives (yes predicted as no): 0

True Negatives (no predicted as no): 109

Overall Interpretation:

The RandomTree model demonstrates excellent performance on this binary classification problem. The high accuracy, Kappa statistic, and class-wise metrics suggest that the model is highly effective at distinguishing between the "yes" and "no" classes. The low number of misclassifications and the near-perfect scores in the confusion matrix indicate robust and accurate predictions. The use of 10-fold stratified cross-validation adds confidence in the generalizability of these results to unseen data.