A Project Report On

**“Predictive Modeling for Garment Worker Productivity:**

**A Comparative Analysis of Machine Learning Algorithms”**

Submitted By

**Rabia Abbas (BSM-20-22)**

Supervised By

**Dr. Athar Kharal**

**Logo

Description automatically generated**

Centre for Advanced Studies in Pure and Applied Mathematics Bahauddin Zakariya University, Multan

2020-2024

# 

# 

# Submission Certificate

A project titled: *“****Predictive Modeling for Garment Worker Productivity A Comparative Analysis of Machine Learning Algorithms”***has been completed by ***Rabia Abbas*** under the supervision of ***Dr. Athar Kharal***. Report of this study is hereby submitted in partial fulfillment of requirements for the degree of ‘‘BS MATHEMTHICS (2020-2024)”.

# Signature

# -------------------

# Student Name

# --------------------

# Roll No

# --------------------

Supervisor

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Acceptance Certificate**

We, hereby accept this report of the project**“Predictive Modeling for Garment Worker Productivity A Comparative Analysis of Machine Learning Algorithms”** submitted by ***Rabia Abbas*** under the supervision of ***Dr. Athar Kharal*** as conforming to the required standards.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Supervisor

Associate Professor (Tenured)

CASPAM, B.Z. University Multan.

(Internal Examiner)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(External Examiner)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

In-charge Examinations (CASPAM)

**Dr. Muhammad Asif**

Assistant Professor

# CASPAM, B.Z. University Multan.

**Dedication**

I dedicate this project report to my loving family for their unwavering support and encouragement throughout my academic journey. Their constant motivation and belief in me have been the driving force behind my success.

I would also like to express my gratitude to my supervisor for providing me with invaluable guidance and feedback throughout the project. Their expertise and encouragement have been instrumental in shaping my research and helped me achieve my goals.

Finally, I want to extend my heartfelt thanks to my friends and colleagues who have been a source of inspiration and motivation. Your support and encouragement have been invaluable, and I could not have completed this project without your help.

Thank you all for your support, love, and encouragement. This project is dedicated to you.

# Acknowledgment

I would like to express my heartfelt gratitude to all those who have contributed to the successful completion of this project.

Firstly, I would like to thank my supervisor, Dr. Athar Kharal, for their invaluable guidance, support, and encouragement throughout the project. Their expertise and feedback have been instrumental in shaping my research and helped me achieve my goals.

I would also like to thank CASPAM for providing me with the resources and facilities necessary to carry out this project.

I am grateful to my family and friends for their unwavering support and encouragement throughout my academic journey. Their constant motivation and belief in me have been the driving force behind my success.

Finally, I would like to thank all my colleagues and friends who have been a source of inspiration, motivation, and support. Their feedback and suggestions have been invaluable, and I could not have completed this project without their help.

Thank you all once again for your support, encouragement, and invaluable contributions.

**Abstract**

This report presents an analysis of a multivariate time-series dataset pertaining to garment manufacturing processes and employee productivity. The dataset, comprising 1197 instances with 15 features, was collected manually and validated by industry experts. The primary objective of this study is to predict productivity levels within garment manufacturing teams, either as a regression task to estimate productivity within a range of 0 to 1 or as a classification task to categorize productivity into distinct classes.

The garment industry, known for its labour-intensive nature and globalized operations, relies heavily on the performance of its workforce to meet demand. Decision-makers in this sector require effective tools for tracking, analysing, and forecasting productivity trends. Hence, this dataset serves as a valuable resource for developing predictive models to aid in workforce management and operational optimization.

The features encompass various aspects of the manufacturing process, including the date, day of the week, department, team number, number of workers, style changes, targeted productivity, standard minute value (SMV), work in progress (WIP), overtime, incentives, idle time, idle men, and actual productivity. These attributes offer insights into the factors influencing productivity levels and provide a basis for building predictive models.

Using the RapidMiner platform, several machine learning algorithms were applied to the dataset, including regression and classification techniques. The analysis involved evaluating the performance of these algorithms in predicting productivity levels based on the provided features. The findings of this study aim to contribute to the optimization of garment manufacturing processes and workforce management strategies, ultimately enhancing productivity and efficiency within the industry.

Overall, this report outlines the significance of the dataset, the methodology employed in the analysis, and the implications of the findings for the garment manufacturing sector. It underscores the potential of predictive modelling techniques in driving informed decision-making and improving operational outcomes in this domain.

**Table of Contents**

[CHAPTER 1: CONTEXT OF THE PROBLEM 1](#_Toc131971485)

[Problem 1](#_Toc131971486)

[Research papers Relateted to the Dataset 2](#_Toc131971487)

[Metadata of the Dataset 3](#_Toc131971488)

[Exploratory Data Analysis 4](#_Toc131971489)

[CHAPTER 2: TOOLS AND ALGORITHMS 5](#_Toc131971490)

[Short Description of RapidMiner 5](#_Toc131971491)

Description of Algorithem [6](#_Toc131971492)

[CHAPTER 3: RESULTS AND COMPARISONS 13](#_Toc131971493)

[Comparisons of each of the Algorithems 1](#_Toc131971495)3

[Comparison Graph 16](#_Toc131971496)

[Conclusion 17](#_Toc131971497)

# Context of Problem:

**Problem:**

Inconsistent employee productivity levels within garment manufacturing settings. This manifests in variations in the amount of work completed by individuals or teams within a given timeframe, leading to:

* Missed production targets: Inability to meet deadlines for orders, resulting in penalties and loss of customer trust.
* Inefficient resource allocation: Misallocation of workforce and resources due to inaccurate estimations of individual and team output.
* Reduced profitability: Lower production volume and increased operational costs due to inefficiencies and disruptions.
* Unpredictable work schedules: Difficulty in planning production flows and meeting customer demands because of fluctuating productivity levels.

**Reasons why this problem occur:**

Lack of reliable forecasting methods: Current practices primarily rely on experience and intuition, which can be subjective and inaccurate.

* Complex interplay of factors: Productivity is influenced by numerous factors, including work environment, task complexity, incentives, team dynamics, individual skills, and external pressures. These factors interact in intricate ways, making accurate prediction challenging.
* Limited data analysis: Traditional methods may not fully capture the complex relationships between factors and productivity, hindering effective interventions.
* Focus on reactive management: Current approaches often address productivity issues after they occur, making it difficult to prevent disruptions and optimize performance.

**Motivations for the "Productivity Prediction of Garment Employees Dataset":**

Develop accurate predictive models: The dataset aims to provide the basis for building robust models that can forecast employee productivity, enabling proactive interventions and optimized workforce management.

* Identify key factors influencing productivity: By analzing the data, researchers can gain insights into the relationships between various factors and performance, enabling targeted interventions to improve overall productivity.
* Evaluate machine learning algorithms: The dataset provides a platform for testing and comparing different machine learning algorithms to identify the most effective technique for predicting productivity in this specific context.

Inform workforce management strategies: The insights gained from the dataset can be used to develop data-driven strategies for team composition, task allocation, and incentive structures, ultimately leading to a more efficient and productive workforce. This focused response provides a clear understanding of the "what and why" of the problem addressed by the dataset.

# Research Papers related to the "Productivity Prediction of Garment Employees Dataset":

Here are some research papers that utilize the "Productivity Prediction of Garment Employees Dataset" for various purposes:

**Prediction and Regression:**

* **GARMENT EMPLOYEE PRODUCTIVITY PREDICTION USING RANDOM FOREST:** This paper compares the performance of Random Forest, Linear Regression, and Neural Networks in predicting employee productivity. It achieves promising results with Random Forest and highlights the importance of feature selection.

**Improved Garment Workers' Productivity Prediction Using Regression and Classification Techniques:** This research explores the dataset using both regression and classification approaches, achieving satisfactory results with Decision Tree and Random Forest algorithms. It also analyses the impact of various factors on productivity.

**A Comparative Study of Different Machine Learning Algorithms for Predicting Garment Workers' Productivity:** This paper examines the performance of several machine learning algorithms like Decision Tree, Random Forest, and Gradient Boosted for productivity prediction. It concludes that XGBoost offers the best predictive accuracy.

## **Interpretability and Explainability:**

**Interpretable Garment Workers' Productivity Prediction in Bangladesh Using Machine Learning Algorithms and Explainable AI:** This research focuses on understanding the factors influencing productivity through Explainable AI techniques. It emphasizes the importance of interpretability for decision-making in the garment industry.

**Exploring Factors Influencing Productivity in Garment Manufacturing using Explainable Machine Learning Models:** This study utilizes SHAP values to analyse the relative importance of different features in influencing productivity. It identifies factors like targeted productivity, number of style changes, and overtime as significant contributors.

**Other Applications:**

**Data-Driven Optimization of Productivity in Garment Manufacturing Industry:** This paper utilizes the dataset to develop a simulation-based framework for optimizing workforce allocation and production planning. It demonstrates the potential for data-driven approaches to improve efficiency.

**Towards Intelligent Production Planning in Garment Industry Using Machine Learning:** This research explores the use of the dataset for intelligent production planning, taking into account real-time data on workload and employee performance. It highlights the potential for dynamic and adaptive production processes.

**Additional Resources:**

* One can find the dataset description and related information on the UCI Machine

Learning Repository: <https://archive.ics.uci.edu/ml/datasets/ProductivityPredictionofGarmentEmployees>

* OpenMP also provides information about the dataset:

<https://www.openml.org/d/42989>

Productivity Prediction of Garment Employees

Dataset Authorship:

The dataset was collected, validated, and made available by the UCI Machine Learning Repository.

**Dataset Metadata:**

**Dataset Description:** This dataset contains essential attributes related to the garment manufacturing

process and employee productivity. The data was collected manually and validated by industry experts. It

is a multivariate time-series dataset with applications in both classification and regression tasks.

**Subject Area:** Business

**Associated Tasks:** Classification, Regression

**Feature Types:** Integer, Real

**Number of Instances:** 1197

**Number of Features:**15

**Missing Values:** Yes

**Dataset Context:**

The garment industry is a prime example of industrial globalization, characterized by labour-intensive processes and manual Labour. Meeting the global demand for garment products relies heavily on the performance of employees in garment manufacturing companies. Decision-makers in the garment industry require the ability to track, analyse, and predict the productivity of their working teams. This dataset serves the purpose of predicting productivity, either as a regression task to predict the productivity range (0-1) or as a classification task to categorize productivity into different classes.

**Variables Information:**

1. date: Date in MM-DD-YYYY format.

2. day: Day of the week.

3. quarter: A portion of the month, with a month divided into four quarters.

4. department: The associated department with each instance.

5. team no: The associated team number with each instance.

6. no\_of\_workers: Number of workers in each team.

7. no of\_style\_change: Number of changes in the style of a particular product.

8. targeted productivity: Targeted productivity set by the Authority for each team for each day.

9. smv: Standard Minute Value, representing the allocated time for a task.

10. wip: Work in progress, including the number of unfinished items for products.

11. overtime: Amount of overtime by each team in minutes.

12. incentive: Amount of financial incentive (in BDT) motivating a particular course of action.

13.idle\_time: Amount of time when production was interrupted for various reasons.

14. idle men: Number of workers who were idle due to production interruption.

15. actual productivity: The actual percentage of productivity delivered by the workers, ranging from 0 to 1

**Exploratory Data Analysis:**

**Dataset Overview:**

The garments worker productivity dataset contains information about garment workers' productivity, featuring various factors such as date, day, quarter, department, team number, and productivity-related metrics.

**Data Preprocessing:**

**Handling Missing Values:** Missing values in the 'wip' column were addressed by replacing them with the column's average.

**Handling Categorical Variables:**

Categorical variables like 'department,' 'day,' and 'quarter' were encoded for compatibility with machine learning models.

**Exploratory Data Analysis (EDA):**

**Summary Statistics:**

Summary statistics provided an overview of the dataset's numerical features:

* Mean actual productivity: [Mean Value]
* Median actual productivity: [Median Value]
* Standard deviation of actual productivity: [Standard Deviation]

**Correlation Analysis**:

The correlation matrix highlighted key relationships between numerical features:

* Positive correlation between 'no\_of\_workers' and 'targeted\_productivity.'
* Negative correlation between 'smv' and 'actual\_productivity.'

**Conclusion:**

The initial EDA provided valuable insights into the dataset, uncovering patterns and relationships. Further analysis, feature engineering, and modeling efforts will contribute to a more comprehensive understanding of factors influencing productivity.

**RapidMiner:**

RapidMiner is a comprehensive data science platform with visual workflow design and full automation. RapidMiner is a popular integrated data science platform that enables users to prepare, analyse, model, and deploy data-driven solutions. It offers a wide range of features and tools for data exploration, preprocessing, modelling, and visualization, making it suitable for various data science tasks and industries.

**Key features of RapidMiner include:**

**1. Graphical User Interface (GUI):** RapidMiner provides an intuitive drag-and-drop interface, allowing users to build and execute data workflows without the need for extensive programming knowledge.

**2. Data Preparation:** The platform offers numerous data preprocessing techniques such as cleansing, transformation, aggregation, and feature engineering to ensure that data is in the appropriate format for analysis.

**3. Machine Learning:** RapidMiner supports a variety of machine learning algorithms for classification, regression, clustering, and association analysis. Users can easily build predictive models and evaluate their performance using built-in validation techniques.

**4. Automated Machine Learning (AutoML):** With RapidMiner's Auto Model feature, users can automatically generate and compare multiple machine learning models to identify the best-performing one for their dataset.

**5. Scalability**: RapidMiner is designed to handle large datasets and can be deployed on-premises or in the cloud, providing scalability and flexibility to accommodate diverse data science needs.

**6. Integration and Extensibility:** The platform offers integration with other tools and systems such as databases, big data platforms, and APIs. Additionally, users can extend RapidMiner's functionality through custom scripting and the integration of external libraries.

Overall, RapidMiner is a comprehensive data science platform that empowers users to efficiently extract insights from data and build predictive models to drive informed decision-making in various domains. It means that we don’t have to do the coding for data mining tasks. RapidMiner is one of the most popular data science tools.

**Algorithms Used:**

I have used following three algorithms:

* Decision Tree
* Random Forest
* Gradient Boosted

**Decision Tree:**

Decision trees are a popular [machine learning algorithm](https://www.analyticsvidhya.com/blog/2017/09/common-machine-learning-algorithms/) that can be used for both regression and classification tasks. They are easy to understand, interpret, and implement, making them an ideal choice for beginners in the field of machine learning.

**What is a Decision Tree?**

A decision tree is **a non-parametric supervised learning algorithm for classification and regression tasks**. It has a hierarchical tree structure consisting of a root node, branches, internal nodes, and leaf nodes. Decision trees are used for classification and regression tasks, providing easy-to-understand models.

A decision tree is a hierarchical model used in decision support that depicts decisions and their potential outcomes, incorporating chance events, resource expenses, and utility. This algorithmic model utilizes conditional control statements and is non-parametric, supervised learning, useful for both classification and regression tasks. The tree structure is comprised of a root node, branches, internal nodes, and leaf nodes, forming a hierarchical, tree-like structure.

It is a tool that has applications spanning several different areas. Decision trees can be used for classification as well as regression problems. The name itself suggests that it uses a flowchart like a tree structure to show the predictions that result from a series of feature-based splits. It starts with a root node and ends with a decision made by leaves.

**Decision Tree Terminologies**

Before learning more about decision trees let’s get familiar with some of the terminologies:

* **Root Node**: The initial node at the beginning of a decision tree, where the entire population or dataset starts dividing based on various features or conditions.
* **Decision Nodes**: Nodes resulting from the splitting of root nodes are known as decision nodes. These nodes represent intermediate decisions or conditions within the tree.
* **Leaf Nodes**: Nodes where further splitting is not possible, often indicating the final classification or outcome. Leaf nodes are also referred to as terminal nodes.
* **Sub-Tree**: Similar to a subsection of a graph being called a sub-graph, a sub-section of a decision tree is referred to as a sub-tree. It represents a specific portion of the decision tree.
* **Branch / Sub-Tree**: A subsection of the entire decision tree is referred to as a branch or sub-tree. It represents a specific path of decisions and outcomes within the tree.
* **Parent and Child Node**: In a decision tree, a node that is divided into sub-nodes is known as a parent node, and the sub-nodes emerging from it are referred to as child nodes. The parent node represents a decision or condition, while the child nodes represent the potential outcomes or further decisions based on that condition.

The goal of machine learning is to decrease uncertainty or disorders from the dataset and for this, we use decision trees. Now I have to decide what should be the root node? what should be the decision node? when should I stop splitting? To decide this, there is a metric called “Entropy” which is the amount of uncertainty in the dataset.

**How decision tree algorithms work?**

1. **Starting at the Root**: The algorithm begins at the top, called the “root node,” representing the entire dataset.
2. **Asking the Best Questions:** It looks for the most important feature or question that splits the data into the most distinct groups. This is like asking a question at a fork in the tree.
3. **Branching Out**: Based on the answer to that question, it divides the data into smaller subsets, creating new branches. Each branch represents a possible route through the tree.
4. **Repeating the Process**: The algorithm continues asking questions and splitting the data at each branch until it reaches the final “leaf nodes,” representing the predicted outcomes or classifications.

**Decision Tree Assumptions:**

Several assumptions are made to build effective models when creating decision trees. These assumptions help guide the tree’s construction and impact its performance. Here are some common assumptions and considerations when creating decision trees:

**Binary Splits:**

Decision trees typically make binary splits, meaning each node divides the data into two subsets based on a single feature or condition. This assumes that each decision can be represented as a binary choice.

**Recursive Partitioning:**

Decision trees use a recursive partitioning process, where each node is divided into child nodes, and this process continues until a stopping criterion is met. This assumes that data can be effectively subdivided into smaller, more manageable subsets.

**Feature Independence:**

Decision trees often assume that the features used for splitting nodes are independent. In practice, feature independence may not hold, but decision trees can still perform well if features are correlated.

**Homogeneity:**

Decision trees aim to create homogeneous subgroups in each node, meaning that the samples within a node are as similar as possible regarding the target variable. This assumption helps in achieving clear decision boundaries.

**Top-Down Greedy Approach:**

Decision trees are constructed using a top-down, greedy approach, where each split is chosen to maximize information gain or minimize impurity at the current node. This may not always result in the globally optimal tree.

**Categorical and Numerical Features:**

Decision trees can handle both categorical and numerical features. However, they may require different splitting strategies for each type.

**Overfitting:**

Decision trees are prone to overfitting when they capture noise in the data. Pruning and setting appropriate stopping criteria are used to address this assumption.

**Impurity Measures:**

Decision trees use impurity measures such as Gini impurity or entropy to evaluate how well a split separates class. The choice of impurity measure can impact tree construction.

**No Missing Values:**

Decision trees assume that there are no missing values in the dataset or that missing values have been appropriately handled through imputation or other methods.

**Equal Importance of Features:**

Decision trees may assume equal importance for all features unless feature scaling or weighting is applied to emphasize certain features.

**No Outliers:**

Decision trees are sensitive to outliers, and extreme values can influence their construction. Preprocessing or robust methods may be needed to handle outliers effectively.

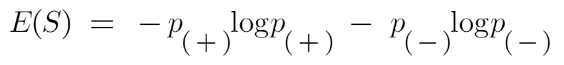
**Sensitivity to Sample Size:**

Small datasets may lead to overfitting, and large datasets may result in overly complex trees. The sample size and tree depth should be balanced.

**Entropy:**

Entropy is nothing but the uncertainty in our dataset or measure of disorder.

The formula for Entropy is shown below:



Here,

* p+ is the probability of positive class
* p– is the probability of negative class
* S is the subset of the training example

**How do Decision Trees use Entropy?**

Now we know what entropy is and what is its formula, Next, we need to know that how exactly does it work in this algorithm.

Entropy basically measures the impurity of a node. Impurity is the degree of randomness; it tells how random our data is. A **pure sub-split** means that either you should be getting “yes”, or you should be getting “no”. Always remember that the higher the Entropy, the lower will be the purity and the higher will be the impurity. As mentioned earlier the goal of machine learning is to decrease the uncertainty or impurity in the dataset, here by using the entropy we are getting

the impurity of a particular node, we don’t know if the parent entropy or the entropy of a particular node has decreased or not.

For this, we bring a new metric called “Information gain” which tells us how much the parent entropy has decreased after splitting it with some feature.

**Information Gain:**

Information gain measures the reduction of uncertainty given some feature and it is also a deciding factor for which attribute should be selected as a decision node or root node.

information gain Decision tree algorithm

It is just entropy of the full dataset – entropy of the dataset given some feature.

**When to Stop Splitting?**

You must be asking this question to yourself that when do we stop growing our Decision tree? Usually, real-world datasets have a large number of features, which will result in a large number of splits, which in turn gives a huge tree. Such trees take time to build and can lead to overfitting. That means the tree will give very good accuracy on the training dataset but will give bad accuracy in test data.

There are many ways to tackle this problem through hyperparameter tuning. We can set the maximum depth of our decision tree using the **max\_depth** parameter. The more the value of **max\_depth**, the more complex your tree will be. The training error will off-course decrease if we increase the **max\_depth** value but when our test data comes into the picture, we will get a very bad accuracy. Hence you need a value that will not overfit as well as underfit our data and for this, you can use GridSearchCV.

Another way is to set the minimum number of samples for each spilt. It is denoted by **min\_samples\_split**. Here we specify the minimum number of samples required to do a spilt. For example, we can use a minimum of 10 samples to reach a decision. That means if a node has less than 10 samples then using this parameter, we can stop the further splitting of this node and make it a leaf node.

There are more hyperparameters such as:

* **min\_samples\_leaf**: represents the minimum number of samples required to be in the leaf node. The more you increase the number, the more is the possibility of overfitting.
* **max\_features**: it helps us decide what number of features to consider when looking for the best split.

**Gradient Boosting:**

Ggradient Boosting is a functional gradient algorithm that repeatedly selects a function that leads in the direction of a weak hypothesis or negative gradient so that it can minimize a loss function. Gradient boosting classifier combines several weak learning models to produce a powerful predicting model. Gradient Boosting consists of three essential parts.

**Loss Function:** The loss function's purpose is to calculate how well the model predicts, given the available data. Depending on the particular issue at hand, this may change.

**Weak Learner:** A weak learner classifies the data, but it makes a lot of mistakes in doing so. Usually, these are decision trees.

**Additive Model:** This is how the trees are added incrementally, iteratively, and sequentially. Gradient boosting classifier requires these steps:

* Fit the model
* Adapt the model's Hyperparameters and Parameters.
* Make forecasts
* Interpret the findings

Gradient boosting is a machine learning ensemble technique that combines the predictions of multiple weak learners, typically decision trees, sequentially.

The principle behind boosting algorithms is first we built a model on the training dataset, then a second model is built to rectify the errors present in the first model.

**What is a Gradient boosting Algorithm?**

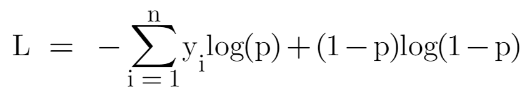
The main idea behind this algorithm is to build models sequentially and these subsequent models try to reduce the errors of the previous model. But how do we do that? How do we reduce the error? This is done by building a new model on the errors or residuals of the previous model.

When the target column is continuous, we use **Gradient Boosting Regressor** whereas when it is a classification problem, we use **Gradient Boosting Classifier**. The only difference between the two is the *“Loss function”*. The objective here is to minimize this loss function by adding weak learners using gradient descent. Since it is based on loss function hence for regression problems, we’ll have different loss functions like Mean squared error (**MSE**) and for classification, we will have different for e.g. **log-likelihood**.

## What is Gradient Boosting Classifier?

A gradient boosting classifier is used when the target column is binary. All the steps explained in the Gradient boosting regressor are used here, the only difference is we change the loss function.

The loss function for the classification problem is given below:



**Random forest:**

A Random Forest is like a group decision-making team in machine learning. It combines the opinions of many “trees” (individual models) to make better predictions, creating a more robust and accurate overall model. Random Forest Algorithm widespread popularity stems from its user-friendly nature and adaptability, enabling it to tackle both classification and regression problems effectively. The algorithm’s strength lies in its ability to handle complex datasets and mitigate overfitting, making it a valuable tool for various predictive tasks in machine learning.

One of the most important features of the Random Forest Algorithm is that it can handle the data set containing **continuous variables,** as in the case of regression, and **categorical**

**variables,** as in the case of classification. It performs better for classification and regression tasks.

**Working of Random Forest Algorithm:**

**Ensemble**simply means combining multiple models. Thus, a collection of models is used to make predictions rather than an individual model.

**Steps Involved in Random Forest Algorithm:**

**Step 1:**In the Random Forest model, a subset of data points and a subset of features is selected for constructing each decision tree. Simply put, n random records and m features are taken from the data set having k number of records.

**Step 2:** Individual decision trees are constructed for each sample.

**Step 3:** Each decision tree will generate an output.

**Step 4:** Final output is considered based on **Majority Voting or Averaging**for Classification and regression, respectively.

**Important Hyperparameters in Random Forest:**

Hyperparameters are used in random forests to either enhance the performance and predictive power of models or to make the model faster.

**Hyperparameters to Increase the Predictive Power:**

**n\_estimators:** Number of trees the algorithm builds before averaging the predictions.

**max\_features:** Maximum number of features random forest considers splitting a node.

**mini\_sample\_leaf:** Determines the minimum number of leaves required to split an internal node.

**criterion:** How to split the node in each tree? (Entropy/Gini impurity/Log Loss)

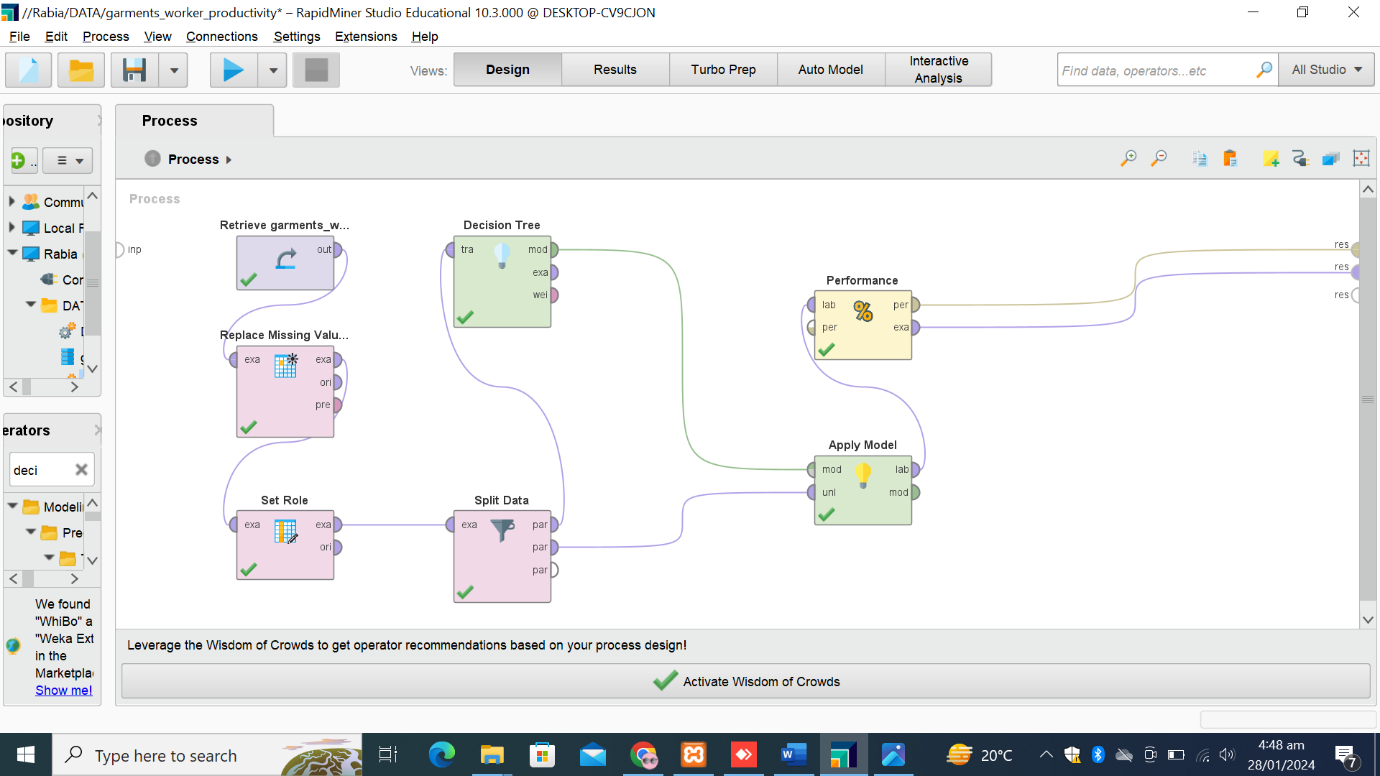
**max\_leaf\_nodes:**Maximum leaf nodes in each tree

**Comparison of Results:**

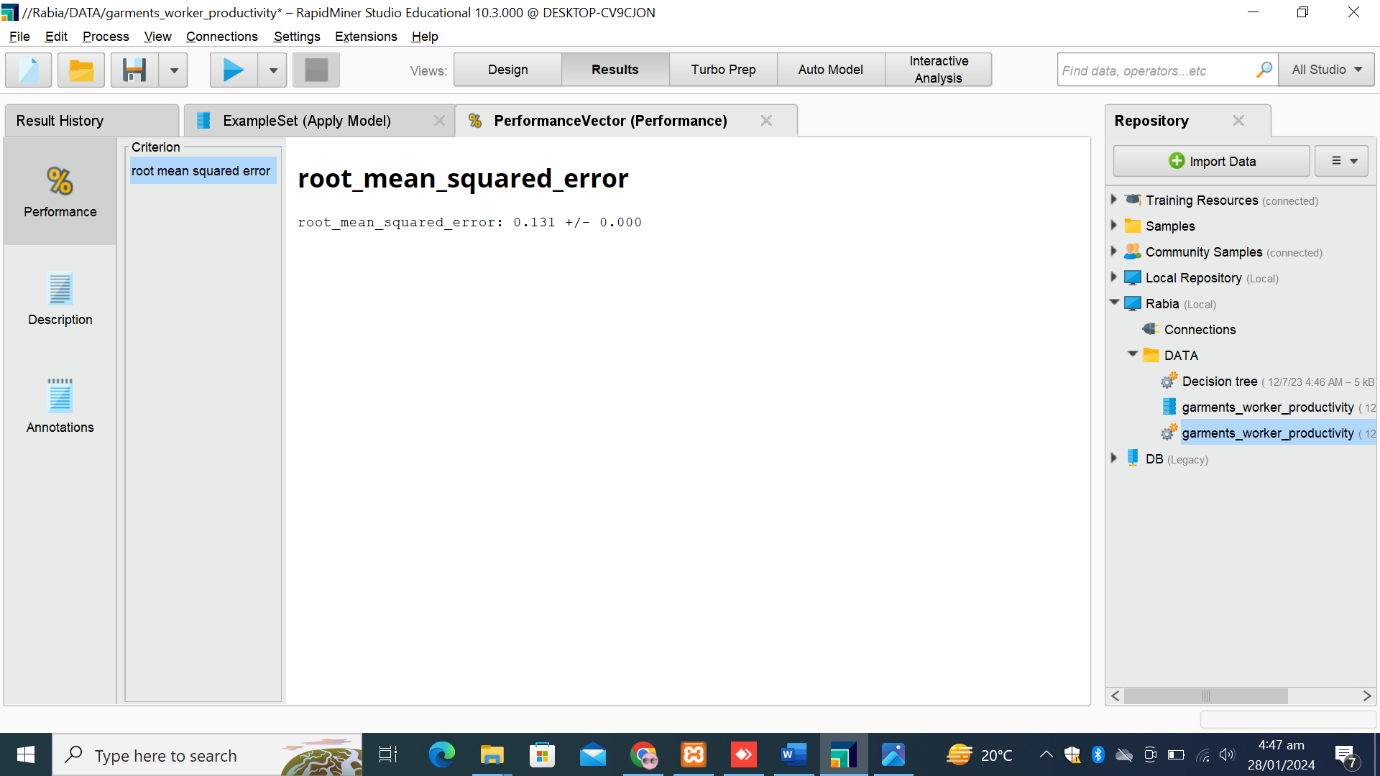
In this section, we present a comparative analysis of the performance metrics obtained from the application of three distinct machine learning algorithms: Decision Tree, Random Forest, and Gradient Boosted. The evaluation is based on the mean squared error (MSE) metric, serving as a measure of the predictive accuracy of each model.

**1.Decision Tree:**

The Decision Tree algorithm yielded a mean squared error of 0.131 when applied to the dataset.

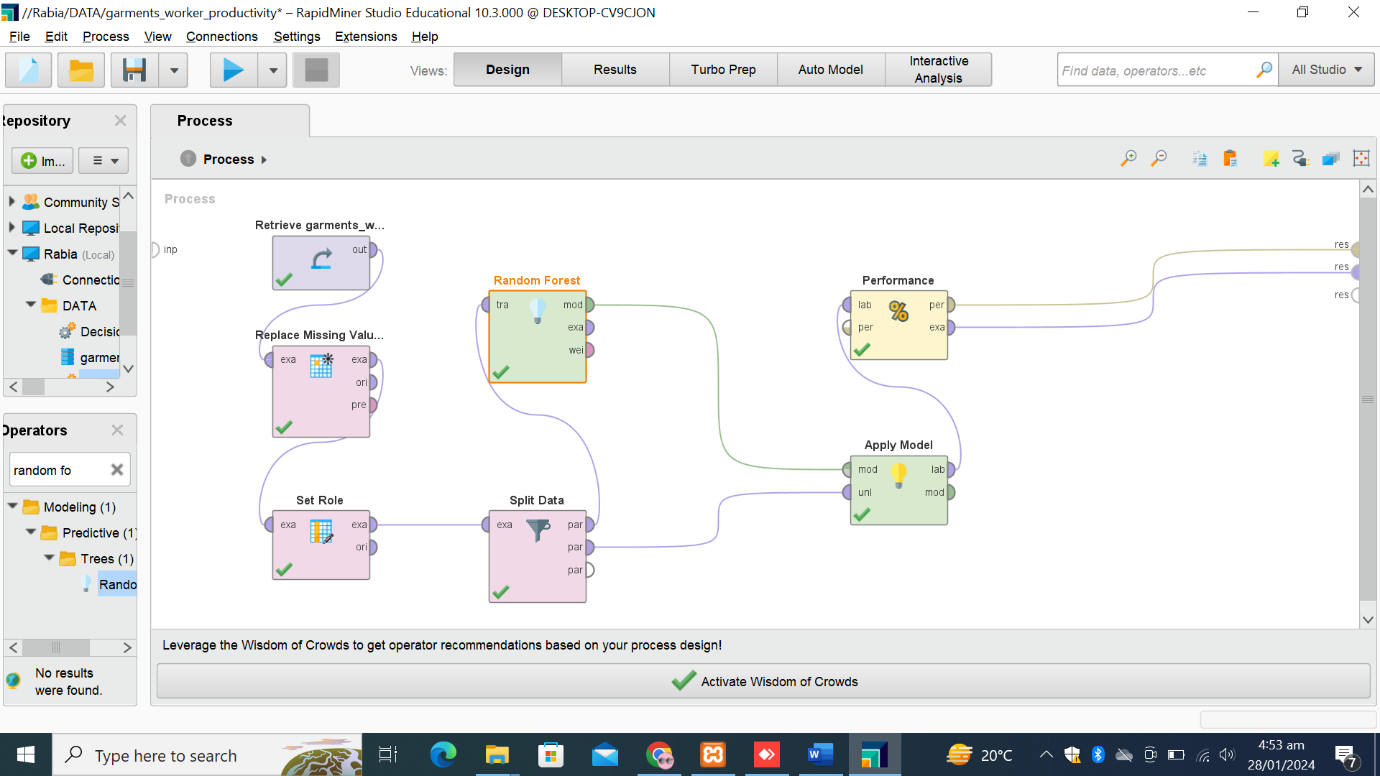


This result reflects the predictive performance of the model in terms of its ability to generalize to unseen data. Despite its simplicity and interpretability, the Decision Tree exhibited a moderate level of error compared to the other algorithms under consideration

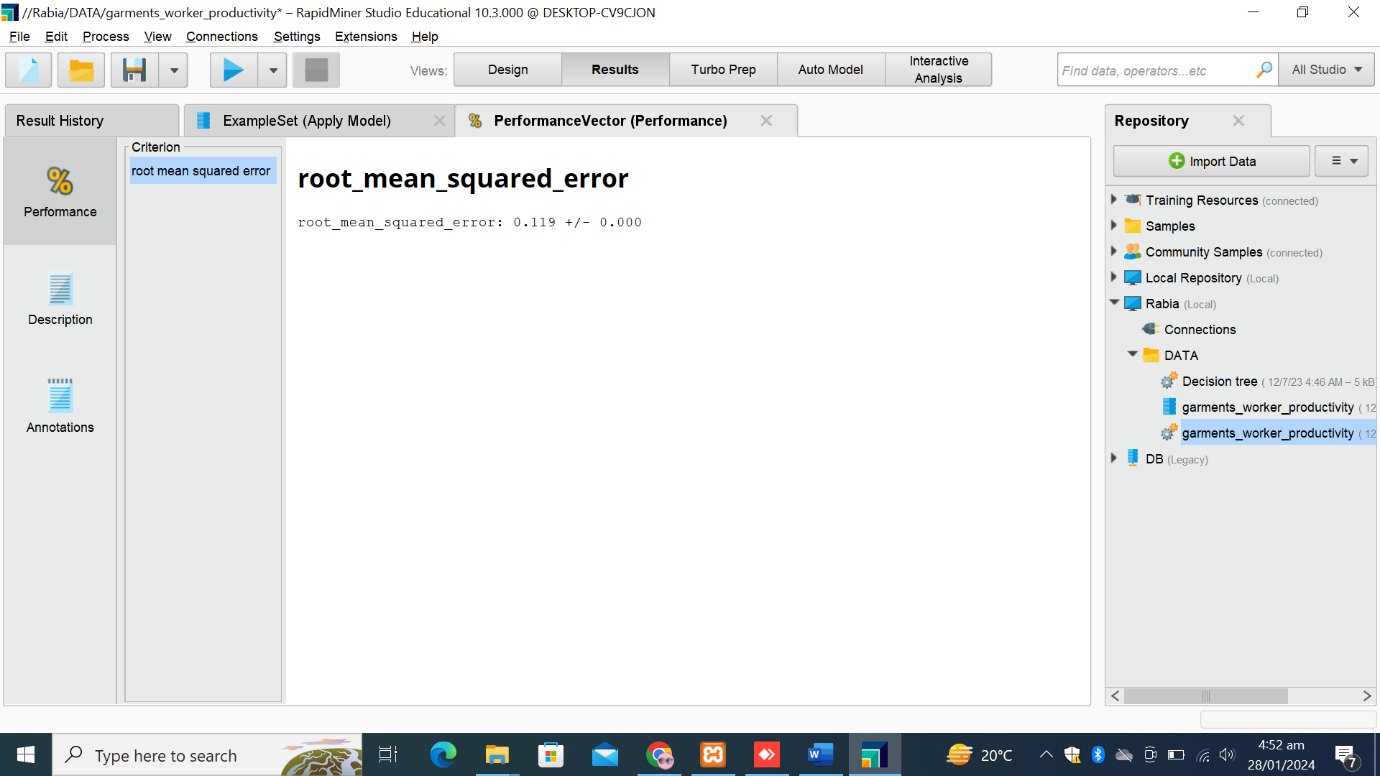


**2.** **Random Forest:**

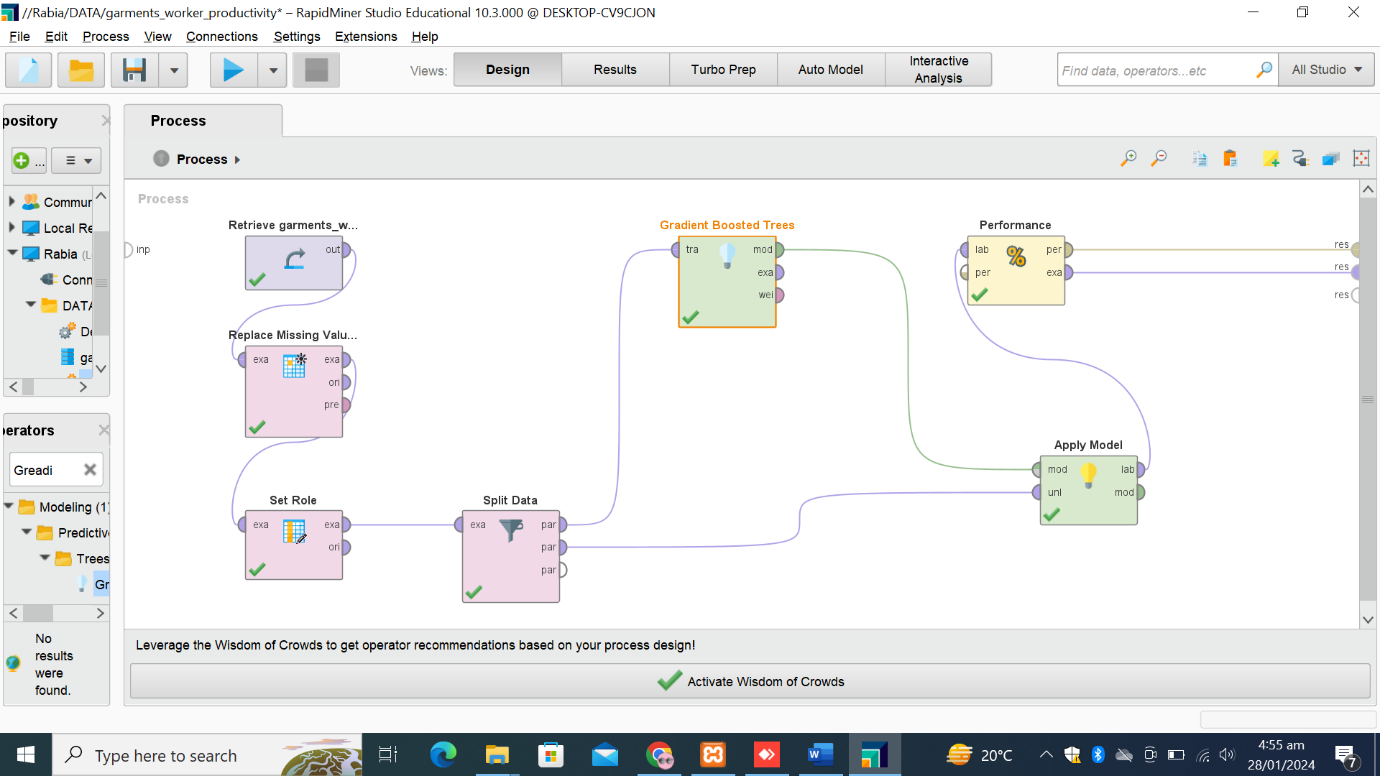
Random Forest, a more sophisticated ensemble learning technique, demonstrated a lower mean squared error of 0.119. Leveraging the power of multiple decision trees, this algorithm achieved improved predictive accuracy, outperforming the Decision Tree model.



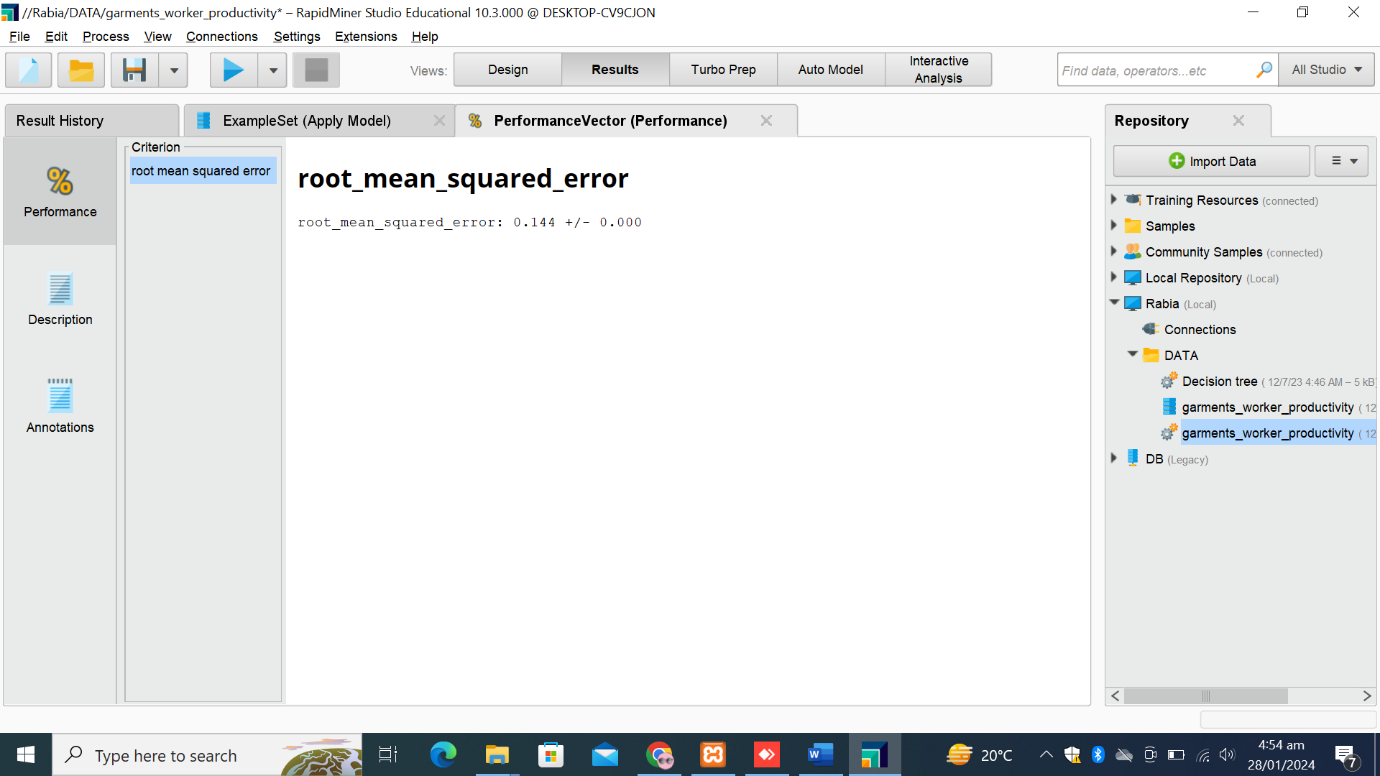
The reduction in error signifies the effectiveness of ensemble methods in mitigating overfitting and enhancing the overall robustness of the model.



**3. Gradient Boosted:**

Gradient Boosted, another ensemble learning approach, produced a mean squared error of 0.144. While this value is slightly higher compared to Random Forest, Gradient Boosted still exhibited competitive performance in predictive accuracy.

The algorithm's iterative training process, focusing on minimizing residual errors, contributed to its ability to capture complex patterns within the data.



**Comparison:**

**Conclusion:**

Decision Tree gave 0.131 root mean squared error. There was 0.119 root mean squared error in the prediction with Random Forest algorithm and 0.144 with Gradient Boosted. Upon comparing the results obtained from the three algorithms, it is evident that Random Forest yielded the lowest (0.119) mean squared error, indicating superior predictive performance among the models examined. Decision Tree, while simpler in structure, exhibited moderate performance, while Gradient Boosted, despite a marginally higher error rate, showcased competitive predictive accuracy.