Predicting Employee Productivity Using Tree Models

Kardelen E.

Contents

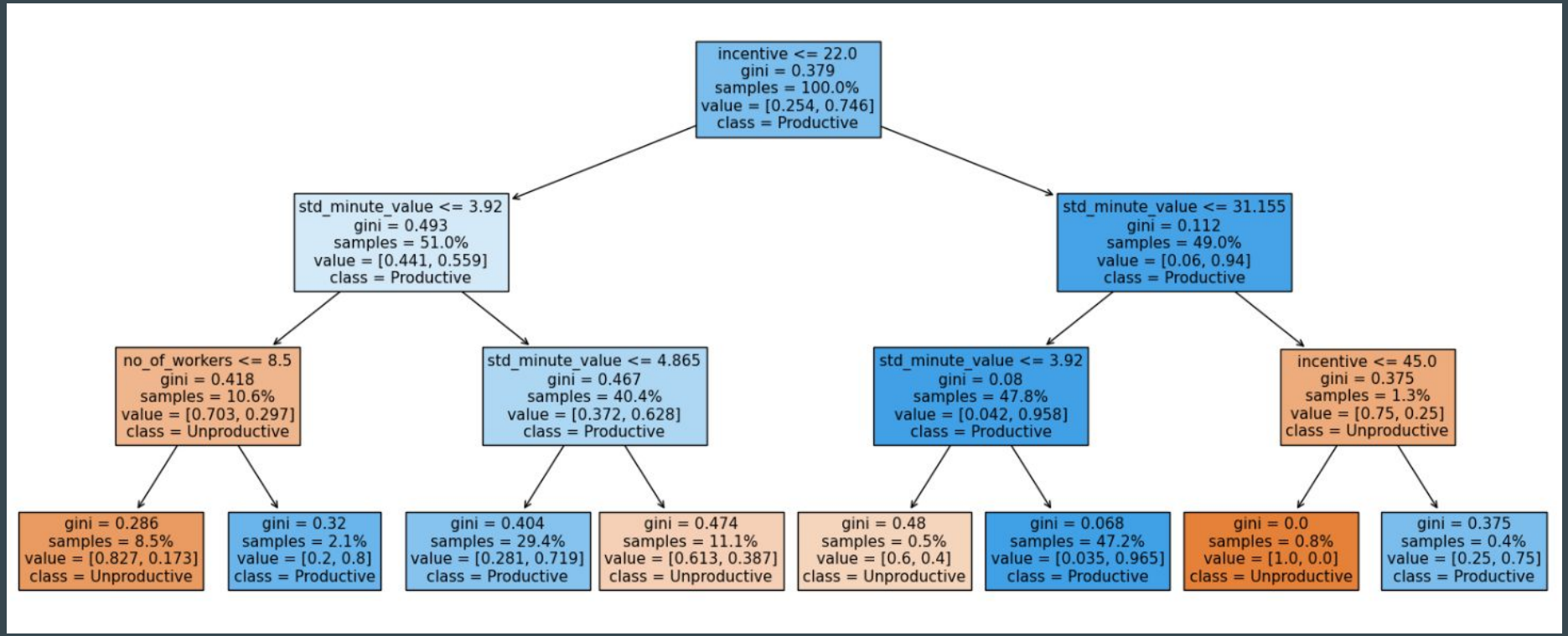
1. Introduction

Decision trees is a type of machine learning that is useful and easy to explain to non-techy audiences. In this project, we'll be introducing the dataset Productivity Prediction of Garment Employees. The original dataset is in the UCI Machine Learning Repository.

The garment industry is a highly labour-intensive industry. Thus, it is crucial for the decision-makes in the industry to track, analyze, and predict the productivity performance of the workers.

	date	quarter	department	day	team	targeted_productivity	smv	wip	over_time	incentive	idle_time	idle_men	no_of_style_change	no_of_workers	actual_productivity
0	1/1/2015	Quarter1	sweing	Thursday	8	0.80	26.16	1108.0	7080	98	0.0	0	0	59.0	0.940725
1	1/1/2015	Quarter1	finishing	Thursday	1	0.75	3.94	NaN	960	0	0.0	0	0	8.0	0.886500
2	1/1/2015	Quarter1	sweing	Thursday	11	0.80	11.41	968.0	3660	50	0.0	0	0	30.5	0.800570
3	1/1/2015	Quarter1	sweing	Thursday	12	0.80	11.41	968.0	3660	50	0.0	0	0	30.5	0.800570
4	1/1/2015	Quarter1	sweing	Thursday	6	0.80	25.90	1170.0	1920	50	0.0	0	0	56.0	0.800382

2. Tree Model


incentive <= 22.0
gini = 0.379
samples = 100.0%
value = [0.254, 0.746]
class = Productive
std_minute_value <= 3.92
gini = 0.493
samples = 51.0%
value = [0.441, 0.559]
class = Productive
std_minute_value <= 31.155
gini = 0.112
samples = 49.0%
value = [0.06, 0.94]
class = Productive
no_of_workers <= 8.5
gini = 0.418
samples = 10.6%
value = [0.703, 0.297]
class = Unproductive
std_minute_value <= 4.865
gini = 0.467
samples = 40.4%
value = [0.372, 0.628]
class = Productive
std_minute_value <= 3.92
gini = 0.08
samples = 47.8%
value = [0.042, 0.958]
class = Productive
incentive <= 45.0
gini = 0.375
samples = 1.3%
value = [0.75, 0.25]
class = Unproductive
gini = 0.286
samples = 8.5%
value = [0.827, 0.173]
class = Unproductive
gini = 0.32
samples = 2.1%
value = [0.2, 0.8]
class = Productive
gini = 0.404
samples = 29.4%
value = [0.281, 0.719]
class = Productive
gini = 0.474
samples = 11.1%
value = [0.613, 0.387]
class = Unproductive
gini = 0.48
samples = 0.5%
value = [0.6, 0.4]
class = Unproductive
gini = 0.068
samples = 47.2%
value = [0.035, 0.965]
class = Productive
gini = 0.0
samples = 0.8%
value = [1.0, 0.0]
class = Unproductive
gini = 0.375
samples = 0.4%
value = [0.25, 0.75]
class = Productive

3. Explanation of the Model

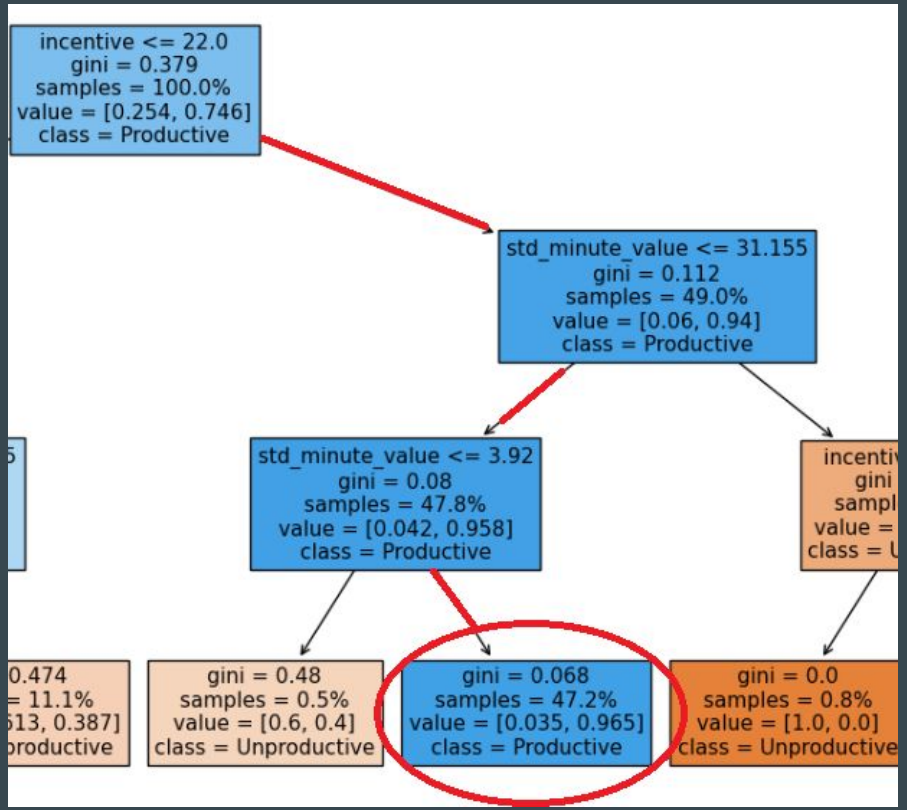
Our model seems to have come to the conclusion that the most important features are **incentive** and **standard minute value** (the column "smv" in the original dataset). Keep in mind that if we hadn't restricted the model into having only 3 branches, we would've seen other columns being used as well.

The best way to explain what we're looking at is to give example scenarios.

4.1 Example Scenarios

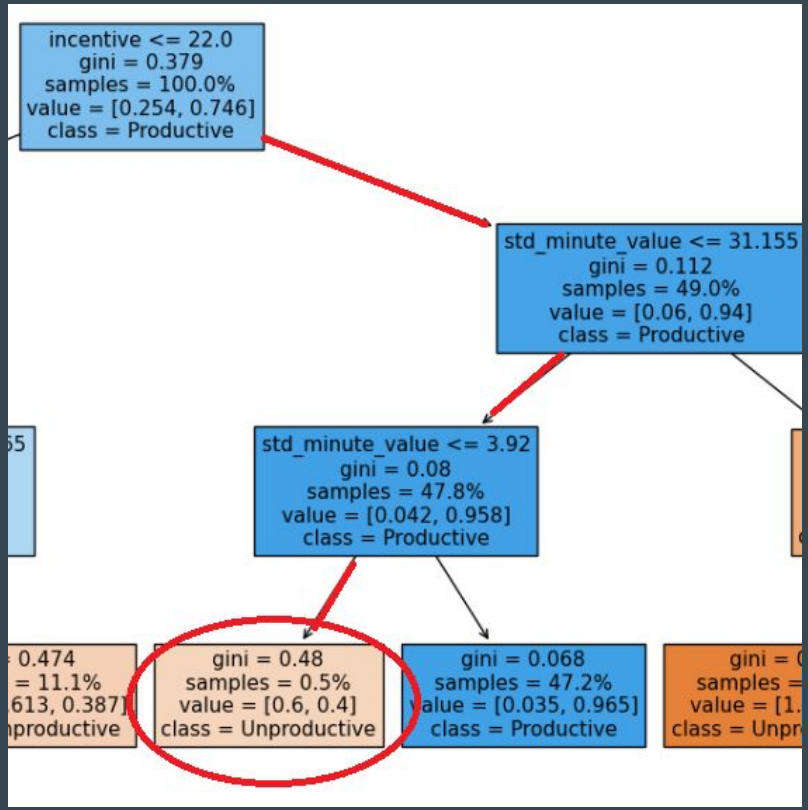
Let's assume that there is an observation which has an "incentive" of 24 and a "standard minute value" of 5. Would our model predict that the productivity goal of the day is going to be reached?

Since the "incentive" is greater than 22, "24 <= 22" evaluates to False at the root node and so we follow the path to the right (**False splits always go to the right.**). Since the "standard minute value" is less than 31.155, "5 <= 31.155" evaluates to True at the child node below the root and so we follow the path to the left (**True splits always go to the left.**). In the final split, since the "standard minute value" is greater than 3.92, "5 <= 3.92" evaluates to False and so we follow the path to the right where we reach the final node, which has a "Productive" class.)



4.2 Example Scenarios

Let's try another scenario. This time, incentive will be 24.0 and standard minute value will be 3.0. The incentive is again greater than 22, so ""incentive <= 22.0" evaluates to false, we'll thus continue to the right. The standard minute value being 3.0 makes "std_minute_value <= 31.155" true, so we'll continue to the left. This time, our condition is "std_minute_value <= 3.92" and our scenario makes it true again. We continue left again and end up with an "Unproductive" class. From our tree, we can see that 0.5% of the samples shares these truth conditions and ends up here.



5. Quality of the Model

Let's check out some statistics about our model to see its quality.:

- **Precision** , or positive productive value (PPV) is calculated as follows:

$$\mathrm{PPV} = \frac{\#\ \text{True Positives}}{\#\ \text{True Positives} + \#\ \text{False Positives}}$$

Precision of our model is 88%.

- **Recall** , or negative predictive value (NPV) is calculated the same way as precision, but for negative values. For our data, it's 93%.
- The **F-score** or F-measure is calculated from the precision and recall of the model. Here, it's 90%.

6. Conclusion

We received data from a garment business, cleaned the said data and created a decision tree to predict whether an instance will be from a day when the productivity goal is reached or not.

From our tests, we saw that the accuracy of our model is quite good. We hypothesized certain features for imaginary days and ran them through our tree to see if they would be productive or non-productive days.