Predicting the productivity of garment factories

3804ICT Group 7

Group List:
Sebastian Perry (S5132483)
Josh Pearson (S5177636)
Bennett Taylor (S5095512)

Introduction

The garment industry is one of the most dominating industries in today’s globalized society. It plays a key role in the growth of a country's economy by generating employment and trade. The ever increasing global demand drives a need for companies to innovate and maximise their productivity. This report looks into finding ways to predict productivity using a variety of data mining techniques, as a way to help guide the hand of project managers. In this report, a dataset collated by a garment factory will be used to train several different data mining algorithms, these algorithms will then be tested in order to ascertain which one is most effective at predicting productivity in the given circumstance.

Solutions

K Nearest Neighbours

Brief Description

(here there will be a brief description of the solution)
K Nearest Neighbours (otherwise known as k-NN) is a classification and regression algorithm commonly used in machine learning and statistics for its effectiveness at classification problems. Unlike other classification models, no training is needed; for each case, the algorithm compares the datapoint to the training set.

Data Preprocessing

In order to fairly test each implementation the following steps were done to the dataset before it was given to all the implementations.

The “actual_productivity” was binned to 10 bins with an equal width to allow the implementation of a classification KNN. This was also done in WEKA.

All the numerical values were scaled using the standard_scaler from sklearn, which standardised by removing the man and scaling with unit variance. This is necessary because it makes all values weighted equally. This method was replicated within WEKA.

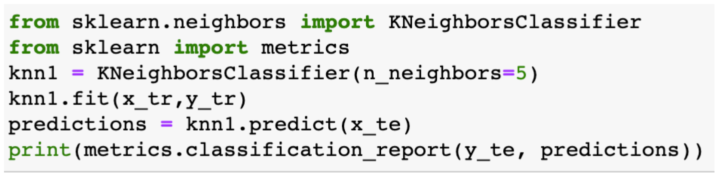
All the categorical values were split using onehotencoder. This splits categorical data into separate columns dedicated to each value, and if that value is present, it’s set to 1. This allows for easier processing of KNN as it compares integers, rather than of strings. This step was not taken in WEKA as it has inbuilt functionality to handle categorical data.

The evaluation-set was set at 30% with the test-set set to 70%. This was replicated within WEKA too.

The final step of preprocessing was with the ‘wip’ column. This column contained some missing values, so they were replaced with zero. This was done for ease of use for the KNN algorithm.

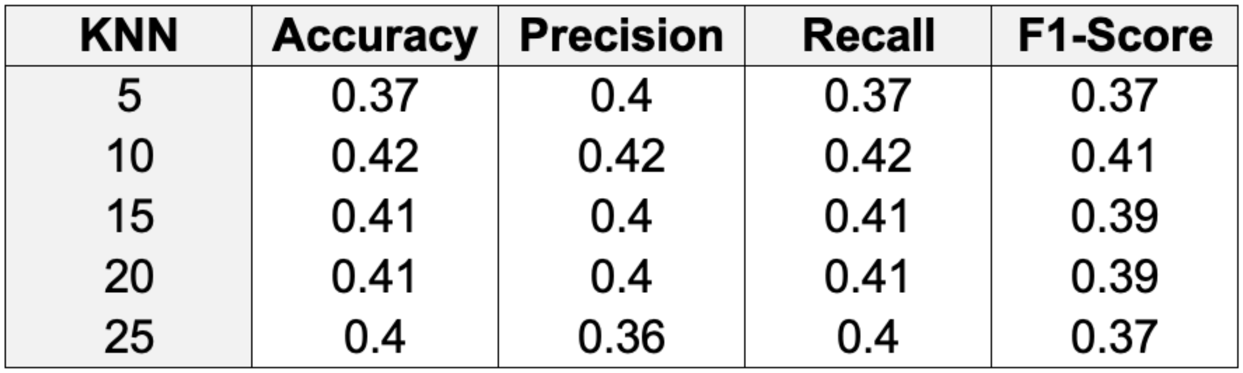
Sklearn KNN

This implementation was fairly straightforward. With the pre-processed data, the training set was fit to the KNN algorithm for faster processing. It was then tested using neighbours of 5, 10, 15, 20, 25 to find the optimal configuration. The code is as below:



```
from sklearn.neighbors import KNeighborsClassifier
from sklearn import metrics
knn1 = KNeighborsClassifier(n_neighbors=5)
knn1.fit(x_tr,y_tr)
predictions = knn1.predict(x_te)
print(metrics.classification_report(y_te, predictions))
```

The results for the algorithm were quite similar. The following is a table showing the accuracy along with the weighted averages of precision, recall and F-1 score.



KNN	Accuracy	Precision	Recall	F1-Score
5	0.37	0.4	0.37	0.37
10	0.42	0.42	0.42	0.41
15	0.41	0.4	0.41	0.39
20	0.41	0.4	0.41	0.39
25	0.4	0.36	0.4	0.37

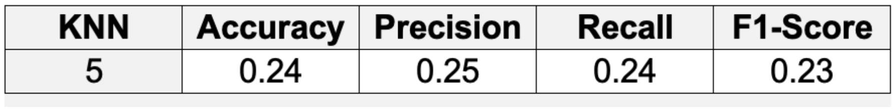
Overall, the optimal neighbour value taking in account all evaluation metrics was 10 neighbours. This will be used to compare to the other implementations.

WEKA IBk

IBk, known as instance based learner, is the WEKA equivalent of a KNN algorithm.
The steps to obtain results were:

1. Pre-process as mentioned in the preprocessing section above
2. Click on the classify tab and choose IBK
3. Set percentage split 70% (in line with all the other methods)
4. Click on the IBk field and set KNN to 5
5. Click start
6. Repeat but set KNN to 10, 15, 20, 25

When doing this method, WEKA was not able to compute the evaluation metrics for many of the KNN values because there was not enough data in certain class categories (that was binned), therefore the only KN value that worked was 5. The results were as follows:



KNN	Accuracy	Precision	Recall	F1-Score
5	0.24	0.25	0.24	0.23

Own Implementation

My implementation of KNN consisted of two main functions.
The first function is KNN_prediction, where a datapoint is compared to the test set to predict that particular datapoint's classification. It evaluates K nearest neighbours, and picks the most frequent classification and in the event where there are an equal number of classifications, it picks one at random.
The second function Evaluate_KNN outputs an array of predicted values of the evaluation set. This then can be compared with the actual values, and this is done using sklearn's metric library.
The code is as follows:

```
# Returns the k prediction
def KNN_prediction(data_point, training_set, training_results, k, Distance_function):
    neighbours = []

    # Line refers to index of the answer
    for line in range(training_set.shape[0]):
        neighbours.append([Distance_function(training_set[line], data_point), line])

    #stores the answers
    class_type = {}

    neighbours.sort(key=Distance_sort)

    for i in range(k):
        response = training_results[neighbours[i][1]] # the second value in neighbours is it's index
        if response in class_type:
            class_type[response] += 1
        else:
            class_type[response] = 1


    return max(class_type, key=class_type.get)
```

```
from sklearn import metrics

def Evaluate_KNN(training_set, training_results, evaluation_set, evaluation_results, k, Distance_function):
    predictions = []
    for data in evaluation_set:
        predictions.append(KNN_prediction(data, training_set, training_results, k, Distance_function))

    return predictions
```

An example of evaluating a particular K value is:

```
bespoke_knn = Evaluate_KNN(x_tr, y_tr, x_te, y_te, 5, Euclidean_distance)
print(metrics.classification_report(y_te, bespoke_knn))
```

```
              precision    recall  f1-score   support

           0       0.11      0.11      0.11         9
           1       0.28      0.26      0.27        19
           2       0.00      0.00      0.00        11
           3       0.22      0.07      0.11        29
           4       0.25      0.21      0.23        47
           5       0.42      0.44      0.43        82
           6       0.43      0.57      0.49        87
           7       0.31      0.39      0.35        36
           8       0.56      0.43      0.48        35
           9       0.50      0.20      0.29         5

    accuracy                           0.37       360
   macro avg       0.31      0.27      0.28       360
weighted avg       0.36      0.37      0.36       360
```

The results for my KNN algorithm is as follows:

KNN	Accuracy	Precision	Recall	F1-Score
5	0.37	0.38	0.37	0.36
10	0.42	0.43	0.43	0.42
15	0.42	0.4	0.42	0.4
20	0.41	0.39	0.41	0.38
25	0.4	0.39	0.4	0.37

The optimal configuration was with 10 neighbours.

Bayes Classification

Bayes classification is a probabilistic prediction algorithm. It is based on the Bayes theorem, and is known to have comparable performance with decision trees and neural networks. Based on posterior probability, it calculates the probability that the dimension results in a certain classification based on prior known knowledge. The continuous data variables were binned so the implementation of bayes that will be used will be the multinomial naive bayes.
In order to properly.

Data Preprocessing

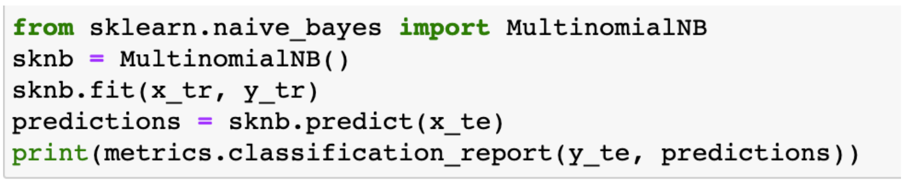
Similar to other implementations, actual productivity was binned with equal width with 10 bins.

Because multinomial bayes' was used, all numerical columns with more than 15 unique values were binned, and all the numerical values that had less than 15 unique values were turned into categorical data.

The training set size was 0.7 of all values and 0.3 of all values were for training.

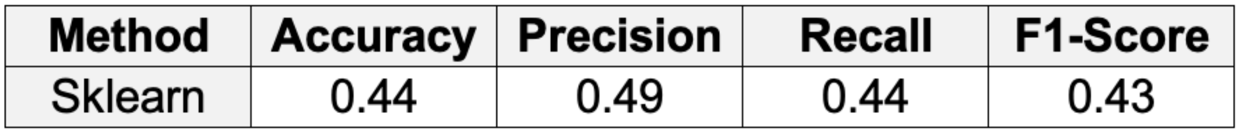
Sklearn multinomialNB

Similar to the sklearn implementation of the K nearest neighbour algorithm, it was fairly straightforward once the data was preprocessed. The model was trained, and then tested. Below is an image of the implementation



```
from sklearn.naive_bayes import MultinomialNB
sknb = MultinomialNB()
sknb.fit(x_tr, y_tr)
predictions = sknb.predict(x_te)
print(metrics.classification_report(y_te, predictions))
```

The results were as follows:



Method	Accuracy	Precision	Recall	F1-Score
Sklearn	0.44	0.49	0.44	0.43

WEKA Naive Bayes

1. Pre-process by binning the actual productivity and standardising the numerical values(naive bayes was used for WEKA, and this helped with the implementation)
2. Click on the classify tab and choose Naive Bayes
3. Set percentage split 70% (in line with all the other methods)
4. Click start

Method	Accuracy	Precision	Recall	F1-Score
WEKA	0.35	0.371	0.35	0.33

Own Implementation

I found this implementation quite hard, and realised that my methods were not very efficient due to using interaction rather than vectorisation. Nonetheless, it was implemented and it worked. The main components of calculating a multinomial naive bayes with multiple features is calculating the prior (the probability of the data given the class) and calculating the probability of the class. This is because the formula to predict the classification is as follows:

$$P(c \mid X) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

P(c | X) is the posterior probability of the class, *P(xn | c)* is the likelihood of each variable and *P(c)* is the class prior probability.

The code to calculate the probabilities are as follows:

```python
def Calculate_column_probabilities(dataframe):

    probabilities = {}

    for column in dataframe:
        # this calculates the probabilities of each unique data value in each column
        probabilities[column] = dataframe[column].value_counts(normalize=True).to_frame()

    return probabilities
```

The code to calculate the priors is as follows:

```python
def Calculate_priors(training):

    prior = {}
    all_classes = training["actual_productivity"].unique()
    for classes in all_classes:
        prior[classes] = {}
        for dimensions in clean_df.columns[:-1]:
            # find the counts of each value of the column for that classification)
            store = training.loc[training["actual_productivity"] == classes][dimensions].value_counts().to_frame()
            # Alleviate the zero sum problem by adding one to all values
            store += 1
            store_sum = store.sum()
            # calculate the probabilities after the zero sum problem was solved
            prior[classes][dimensions] = store/store_sum

    return prior
```

As you can see in the code, to overcome the zero-sum problem, I added one to each count of the unique value before calculating the probabilities, avoiding the situation where a probability is equal to 0. Vectorisation could have been done here so that each column was calculated simultaneously

The code used to predict based on the model is as follows:

```
def Predict(probabilities, prior, data_point):
    class_prob = {}
    for i in probabilities["actual_productivity"].index:

        prob_class = probabilities["actual_productivity"].loc[i][0]

        total = prob_class

        for j in data_point.index[:-1]:
            prior_data_given_class = prior[i][j].loc[data_point[j]][0]
            total = total * prior_data_given_class

        class_prob[i] = total
    return max(class_prob, key = class_prob.get)
```

Essentially, the algorithm calculates the probabilities of each classification, and returns the classification with the highest probability. It follows the formula stated above.

To get the predictions, the following code was used:

```
def Get_predictions(probabilities, prior, evaluation_set):
    predictions = []
    num = 0
    #iterate through each datapoint to calculate predictions
    for index, row in evaluation_set.iterrows():
        prediction = Predict(probabilities, prior, row)
        predictions.append(prediction)
    return predictions
```

Again, vectorisation could have been used here to calculate the many data points simultaneously. Thankfully, this dataset is quite small with <1500 points, therefore it did not take too long.

Finally, this was the code used to evaluate the algorithm:

```
all_predictions = Get_predictions(nb_probabilities, nb_priors, test)
all_results=test['actual_productivity'].values
```

```
from sklearn import metrics
print(metrics.classification_report(all_results, all_predictions))
```

The results are as follows:

Method	Accuracy	Precision	Recall	F1-Score
Josh's	0.45	0.42	0.45	0.41

Support Vector Machine

Support Vector Machines (SVM's) are a machine learning algorithms used for classification and regression of both linear and non-linear data. It is highly preferred in many situations due to it's high accuracy to low computational power ratio. SVM's transform the training data into a N-dimensional space in order to search for a linear optimal separating hyperplane that distinctly classifies the data.

Data Preprocessing

Similar to other implementations, actual productivity was binned with equal width with 10 bins.

The nominal attributes (such as 'date', 'quarter', 'department', 'day', 'team) were removed from both the training and testing set. This is because the Support Vector Machine only uses numerical values since it works with a graph/matrix. The training set was set to 70% of all values and 30% of all values for testing.

Popular Implementation 1: SKLearn

SKLearn is a very useful and robust Python library for Machine Learning algorithms. It provides a selection for algorithms and statistical modeling. The library is aimed to streamline code, converting what is usually a whole file into a few lines of code (as seen below).

```python
from sklearn.model_selection import train_test_split
x_training_data, x_test_data, y_training_data, y_test_data = train_test_split(x, y, test_size = 0.3)

from sklearn.svm import SVC
model = SVC()
model.fit(x_training_data, y_training_data)
predictions1 = model.predict(x_test_data)

from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
print(classification_report(y_test_data, predictions1))
```

The results were as follows:

```
              precision    recall  f1-score   support

 0.23 - 0.38       0.00      0.00      0.00        31
 0.38 - 0.54       0.00      0.00      0.00        24
 0.54 - 0.69       0.00      0.00      0.00        66
 0.69 - 0.84       0.48      0.76      0.59       137
    0.84 - 1       0.43      0.61      0.50       102

    accuracy                           0.46       360
   macro avg       0.18      0.27      0.22       360
weighted avg       0.31      0.46      0.37       360
```

Confusion Matrix:

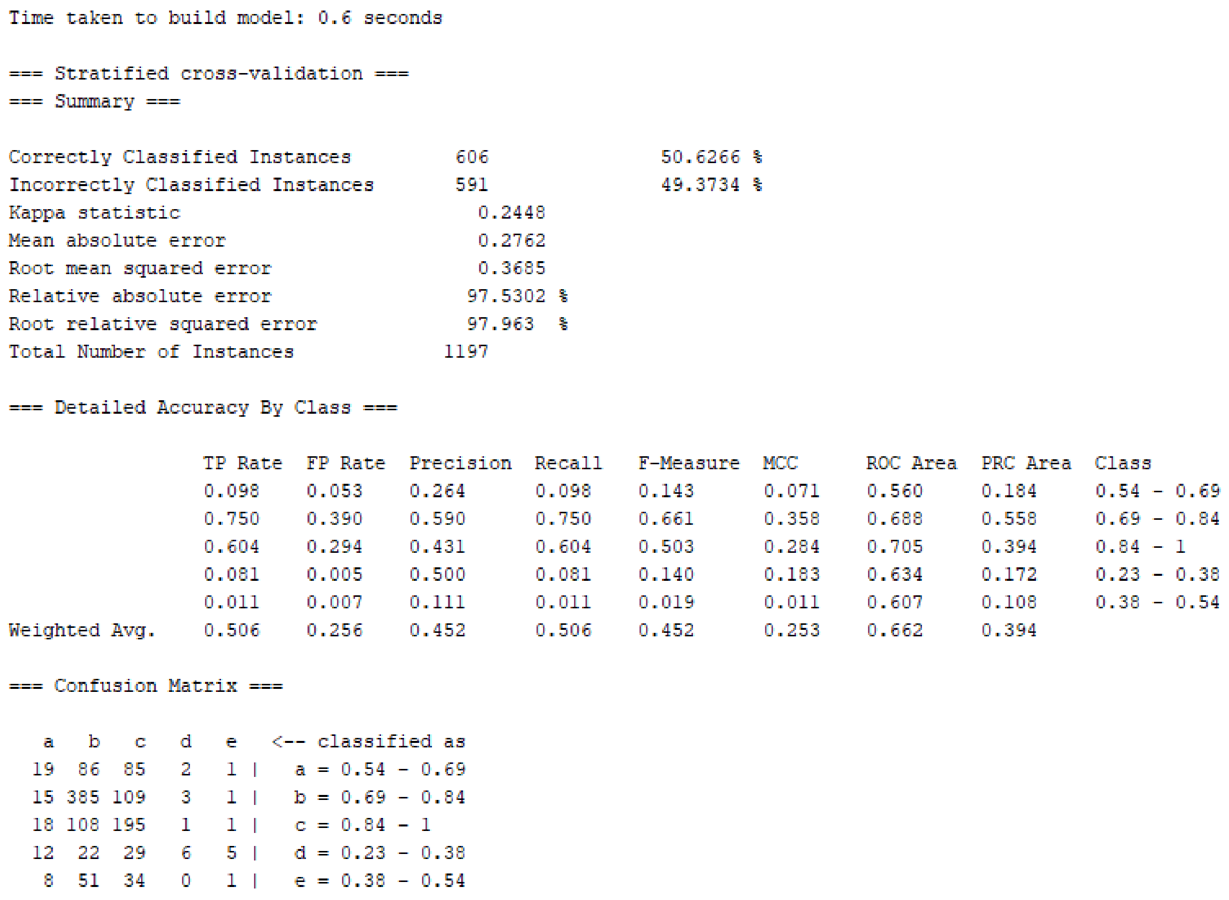
```
[[  0   0   0  18  13]
 [  0   0   0  16   8]
 [  0   0   0  38  28]
 [  0   0   0 104  33]
 [  0   0   0  40  62]]
```

Popular Implementation 2 : WEKA SMO

WEKA is another popular machine learning tool used by data scientists. It provides an extensive library of machine learning algorithms to solve various data mining problems. It includes many different panels for manipulating data sets given a certain problem. These panels include a preprocessing panel, a classify panel, associate panel, a cluster panel and visualization panel. It is the most straightforward and easy to use implementation of the three.

1. The data was imported into WEKA using the preprocess panel
2. The Algorithm SMO was then chosen in the classify panel
3. The split percentage was set to 70% to be inline with other implementations and algorithms
4. The start button was then pressed in the classify panel

The results were as follows:

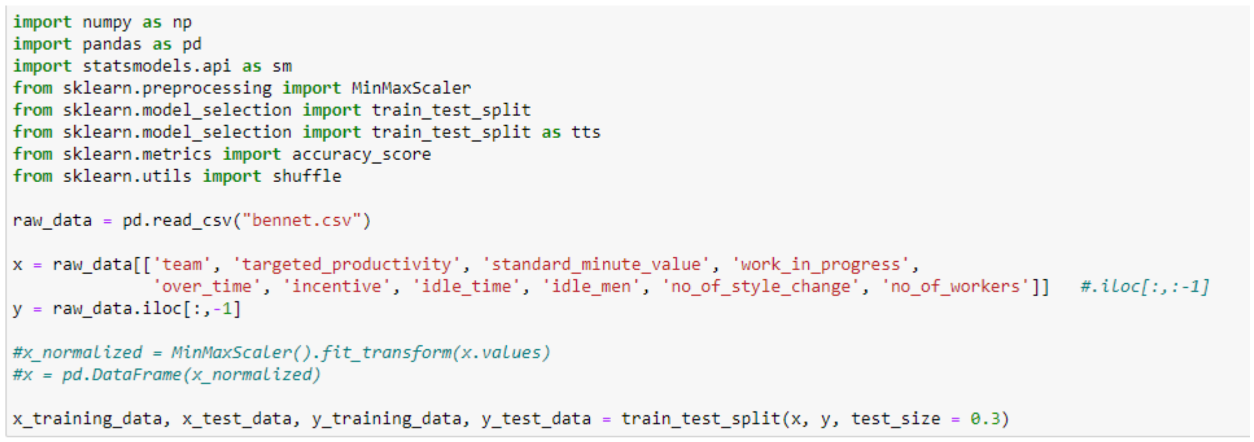


```
Time taken to build model: 0.6 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         606               50.6266 %
Incorrectly Classified Instances       591               49.3734 %
Kappa statistic                          0.2448
Mean absolute error                      0.2762
Root mean squared error                  0.3685
Relative absolute error                 97.5302 %
Root relative squared error             97.963  %
Total Number of Instances             1197

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.098    0.053    0.264      0.098    0.143      0.071    0.560     0.184     0.54 - 0.69
                 0.750    0.390    0.590      0.750    0.661      0.358    0.688     0.558     0.69 - 0.84
                 0.604    0.294    0.431      0.604    0.503      0.284    0.705     0.394     0.84 - 1
                 0.081    0.005    0.500      0.081    0.140      0.183    0.634     0.172     0.23 - 0.38
                 0.011    0.007    0.111      0.011    0.019      0.011    0.607     0.108     0.38 - 0.54
Weighted Avg.    0.506    0.256    0.452      0.506    0.452      0.253    0.662     0.394

=== Confusion Matrix ===

   a   b   c   d   e   <-- classified as
  19  86  85   2   1 |   a = 0.54 - 0.69
  15 385 109   3   1 |   b = 0.69 - 0.84
  18 108 195   1   1 |   c = 0.84 - 1
  12  22  29   6   5 |   d = 0.23 - 0.38
   8  51  34   0   1 |   e = 0.38 - 0.54
```

Own Implementation

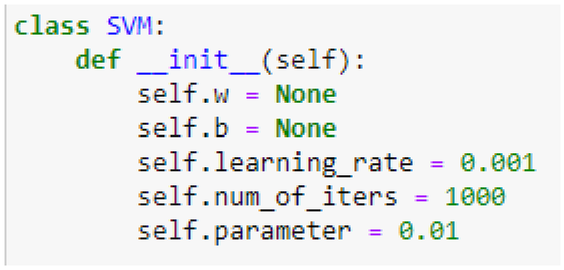
The 3rd implementation was written in Python to create a linear support vector machine model. The aim of this implementation was to only use built in Python modules, essentially creating a support vector machine from scratch.

The necessary libraries and data was first read in. Then some data preprocessing was done to remove the nominal attributes and split the data into a training and testing set.



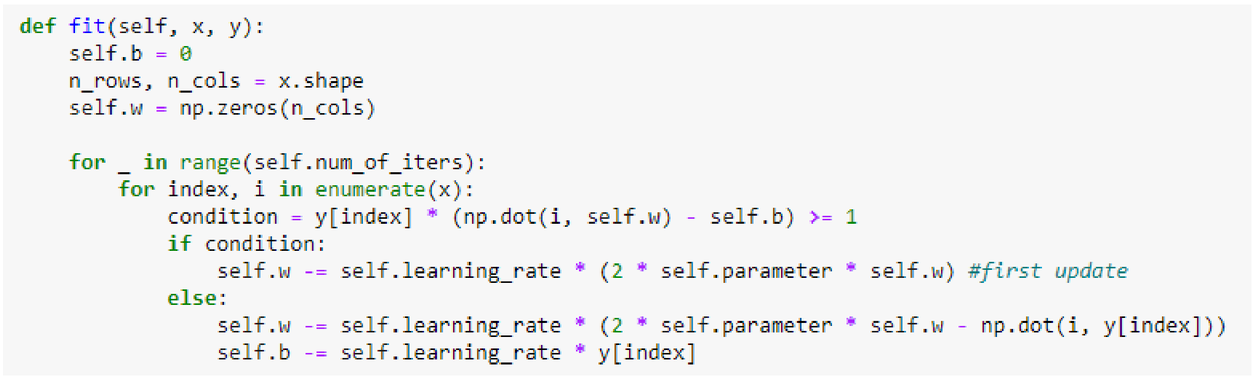
```python
import numpy as np
import pandas as pd
import statsmodels.api as sm
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split
from sklearn.model_selection import train_test_split as tts
from sklearn.metrics import accuracy_score
from sklearn.utils import shuffle

raw_data = pd.read_csv("bennet.csv")

x = raw_data[['team', 'targeted_productivity', 'standard_minute_value', 'work_in_progress',
              'over_time', 'incentive', 'idle_time', 'idle_men', 'no_of_style_change', 'no_of_workers']]   #.iloc[:,:-1]
y = raw_data.iloc[:,-1]

#x_normalized = MinMaxScaler().fit_transform(x.values)
#x = pd.DataFrame(x_normalized)

x_training_data, x_test_data, y_training_data, y_test_data = train_test_split(x, y, test_size = 0.3)
```

A class was then made to contain all the necessary Support Vector Machine functions. The algorithm follows the same structure as the SKLearn algorithm by splitting the algorithm up into three main sections; initialization of the support vector machine, fitting the data to the model and prediction.



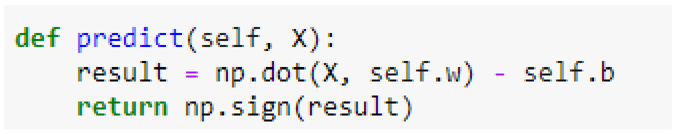
```python
class SVM:
    def __init__(self):
        self.w = None
        self.b = None
        self.learning_rate = 0.001
        self.num_of_iters = 1000
        self.parameter = 0.01
```

The fit method creates the hyperplane used to classify the data points on the support vector machine graph and classifies them based on their positions.



```python
def fit(self, x, y):
    self.b = 0
    n_rows, n_cols = x.shape
    self.w = np.zeros(n_cols)

    for _ in range(self.num_of_iters):
        for index, i in enumerate(x):
            condition = y[index] * (np.dot(i, self.w) - self.b) >= 1
            if condition:
                self.w -= self.learning_rate * (2 * self.parameter * self.w) #first update
            else:
                self.w -= self.learning_rate * (2 * self.parameter * self.w - np.dot(i, y[index]))
                self.b -= self.learning_rate * y[index]
```

The predict method predicts the labels of each sample in the test data set by applying the linear model formula to the algorithm.



```python
def predict(self, X):
    result = np.dot(X, self.w) - self.b
    return np.sign(result)
```

$$f(x) = sign\left(\mathbf{w}^{*} \cdot x + \mathbf{b}^{*}\right)$$

Decision Tree

Decision trees are classification algorithms which are easy to interpret and inexpensive to construct. Training data is used to construct a tree, in which each branching point holds a question, which if run through to the end leads to a leaf which classifies the data. The following consists decision trees which branch on either side of found thresholds, as such only continuous training data was used (this is described as a step in the data preprocessing stage)

Data Preprocessing

In order to fairly test each implementation the following steps were done to the dataset before it was given to all the implementations as the training data csv file caller 'continuousv2.csv'. The steps taken and the reasons behind them will be listed below:

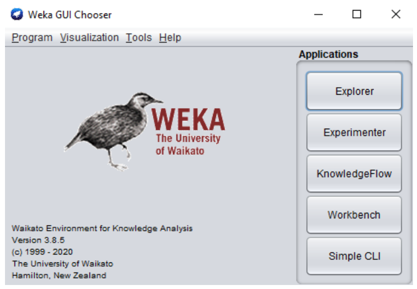
- The spelling of the word sewing was corrected for aesthetic reasons.
- The following nominal attributes were removed so that the training data is usable by all implementations of the decision tree on a level playing field: ('date', 'quarter', 'department', 'day', 'team').
- The empty instances of the 'work_in_progress' attribute were autofilled with the value 0 to avoid errors in the implementations.
- The 'no_of_workers' attribute had all of it's decimal point instances rounded down to the nearest whole number as in the given circumstances these values were unrealistic and deemed as errors, as they would seem to imply terms contained half people.
- The 'actual_productivity' attribute had all of it's top instances moved back to a maximum of 1, as in the given circumstance it is unrealistic for productivity to be above 100%.
- Equal width binning was used to assign the instances of the 'actual_productivity' data attribute in 5 bins to allow the decision trees to classify data.

A small sample of the resulting dataset it shown below

targeted_ productivity	standard_ minute_ value	work_in_ progress	over_time	incentive	idle_time	idle_men	no_of_style_ change	no_of_workers	actual_productivity
0.7	4.15	0	2700	0	0	0	0	15	0.23 - 0.38
0.75	4.15	0	2400	0	0	0	0	20	0.23 - 0.38
0.7	24.26	1400	6720	0	0	0	0	56	0.38 - 0.54
0.5	22.4	947	3390	23	0	0	0	56	0.38 - 0.54
0.8	4.15	0	1440	0	0	0	0	8	0.54 - 0.69
0.65	3.94	0	1440	0	0	0	0	8	0.54 - 0.69
0.8	4.15	0	1440	0	0	0	0	8	0.69 - 0.84
0.8	2.9	0	1440	0	0	0	0	8	0.69 - 0.84
0.75	2.9	0	1080	0	0	0	0	9	0.84 - 1.00
0.8	4.6	0	1080	0	0	0	0	9	0.84 - 1.00

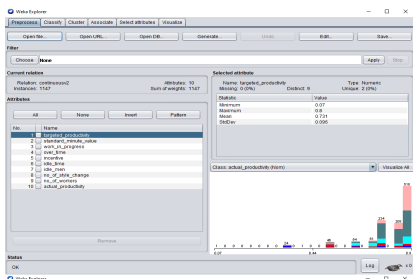
Test data

In order to fairly test each implementation a set of 10 instances was gathered for each class in the training data, these instances were omitted from the training data and used for testing purposes as the testing data csv file caller ‘continuousv2.csv’.



Popular Implementation 1: Weka REPTREE

Weka was the first popular decision tree implementation chosen. It featured a tree which could receive both nominal and continuous data attributes to create its trees (however as mentioned above, it was given data only consisting of continuous attributes)



Filter: None

Current relation
Relation: continuousv2
Instances: 1147
Attributes: 10
Sum of weights: 1147

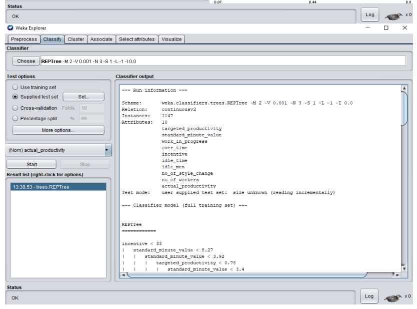
No.	Name
1	targeted_productivity
2	standard_minute_value
3	work_in_progress
4	over_time
5	incentive
6	idle_time
7	idle_men
8	no_of_style_change
9	no_of_workers
10	actual_productivity

Selected attribute
Name: targeted_productivity Type: Numeric
Missing: 0 (0%) Distinct: 9 Unique: 2 (0%)

Statistic	Value
Minimum	0.07
Maximum	0.8
Mean	0.731
StdDev	0.096

Class: actual_productivity (Nom)

Status: OK



REPTree -M 2 -V 0.001 -N 3 -S 1 -L -1 -I 0.0

Test options: Use training set [✓ Supplied test set] Cross-validation (Folds 10) Percentage split (% 66)

(Nom) actual_productivity

13:38:53 - trees.REPTree

Status: OK

```
=== Run information ===

Scheme:       weka.classifiers.trees.REPTree -M 2 -V 0.001 -N 3 -S 1 -L -1 -I 0.0
Relation:     continuousv2
Instances:    1147
Attributes:   10
              targeted_productivity
              standard_minute_value
              work_in_progress
              over_time
              incentive
              idle_time
              idle_men
              no_of_style_change
              no_of_workers
              actual_productivity
Test mode:    user supplied test set:  size unknown (reading incrementally)

=== Classifier model (full training set) ===


REPTree
============

incentive < 33
|   standard_minute_value < 8.27
|   |   standard_minute_value < 3.92
|   |   |   targeted_productivity < 0.78
|   |   |   |   standard_minute_value < 3.4
```

Method

1. Upon opening the program, select explorer
2. Open file was selected and the training data was chosen
3. Navigate to the Classify tab on the top of the window
4. The REPTREE was chosen for the classifier
5. The supplied test set option was selected and a file containing the test instances were given
6. The start button was pressed

Results

Weka automatically calculated the following results:

Class	Accuracy	Precision	Recall	F1-value
0.23 - 0.38	0.487	0.250	0.100	0.143
0.38 - 0.54	0.500	0.333	0.100	0.154
0.54 - 0.69	0.663	0.417	0.500	0.455
0.69 - 0.84	0.375	0.000	0.000	0.000
0.84 - 1	0.800	0.429	0.900	0.581
Average	0.565	0.286	0.320	0.267

The tree generated

```
incentive < 33
|  standard_minute_value < 8.27
| |  standard_minute_value < 3.92
| | |  targeted_productivity < 0.78
| | | |  standard_minute_value < 3.4
| | | | |  targeted_productivity < 0.65 : 0.23 - 0.38 (9/6) [2/1]
| | | | |  targeted_productivity >= 0.65
| | | | | |  over_time < 2580
| | | | | | |  over_time < 1080 : 0.38 - 0.54 (17/8) [4/1]
| | | | | | |  over_time >= 1080 : 0.69 - 0.84 (3/1) [4/1]
| | | | | |  over_time >= 2580
| | | | | | |  targeted_productivity < 0.72 : 0.38 - 0.54 (3/1) [2/1]
| | | | | | |  targeted_productivity >= 0.72 : 0.23 - 0.38 (2/0) [2/1]
| | | |  standard_minute_value >= 3.4 : 0.54 - 0.69 (12/5) [5/3]
| | |  targeted_productivity >= 0.78 : 0.54 - 0.69 (33/16) [15/5]
| |  standard_minute_value >= 3.92
| | |  no_of_workers < 9.5
| | | |  over_time < 3030 : 0.84 - 1.00 (111/65) [66/35]
| | | |  over_time >= 3030 : 0.23 - 0.38 (12/10) [7/3]
| | |  no_of_workers >= 9.5
| | | |  over_time < 13800
| | | | |  targeted_productivity < 0.72 : 0.84 - 1.00 (27/12) [15/4]
| | | | |  targeted_productivity >= 0.72
| | | | | |  over_time < 6150
| | | | | | |  no_of_workers < 11.5
| | | | | | | |  standard_minute_value < 4.04
| | | | | | | | |  over_time < 1560 : 0.84 - 1.00 (13/2) [4/2]
| | | | | | | | |  over_time >= 1560
| | | | | | | | | |  targeted_productivity < 0.78 : 0.84 - 1.00 (2/0) [1/0]
| | | | | | | | | |  targeted_productivity >= 0.78 : 0.69 - 0.84 (7/4) [1/0]
| | | | | | | |  standard_minute_value >= 4.04 : 0.84 - 1.00 (7/1) [2/1]
| | | | | | |  no_of_workers >= 11.5
| | | | | | | |  no_of_workers < 19.5
| | | | | | | | |  no_of_workers < 15.5
| | | | | | | | | |  over_time < 2820
| | | | | | | | | | |  standard_minute_value < 4.87
| | | | | | | | | | | |  no_of_workers < 13.5
| | | | | | | | | | | | |  targeted_productivity < 0.78 : 0.84 - 1.00 (3/1) [3/1]
| | | | | | | | | | | | |  targeted_productivity >= 0.78 : 0.69 - 0.84 (10/6) [4/2]
| | | | | | | | | | | |  no_of_workers >= 13.5 : 0.84 - 1.00 (7/3) [0/0]
| | | | | | | | | | |  standard_minute_value >= 4.87 : 0.54 - 0.69 (2/1) [0/0]
| | | | | | | | | |  over_time >= 2820 : 0.69 - 0.84 (3/1) [0/0]
| | | | | | | | |  no_of_workers >= 15.5 : 0.84 - 1.00 (9/3) [6/1]
| | | | | | | |  no_of_workers >= 19.5 : 0.84 - 1.00 (9/2) [5/0]
| | | | | |  over_time >= 6150 : 0.84 - 1.00 (9/1) [4/1]
| | | |  over_time >= 13800 : 0.54 - 0.69 (2/0) [0/0]
|  standard_minute_value >= 8.27
| |  targeted_productivity < 0.68
| | |  targeted_productivity < 0.55
| | | |  incentive < 26.5
| | | | |  over_time < 5790 : 0.38 - 0.54 (6/0) [3/1]
| | | | |  over_time >= 5790
| | | | | |  no_of_workers < 57.5 : 0.23 - 0.38 (11/2) [3/2]
| | | | | |  no_of_workers >= 57.5
| | | | | | |  targeted_productivity < 0.42 : 0.23 - 0.38 (2/1) [2/0]
| | | | | | |  targeted_productivity >= 0.42 : 0.38 - 0.54 (2/0) [2/0]
| | | |  incentive >= 26.5 : 0.38 - 0.54 (7/1) [3/3]
| | |  targeted_productivity >= 0.55
| | | |  incentive < 11.5
| | | | |  standard_minute_value < 23.04 : 0.54 - 0.69 (7/2) [0/0]
| | | | |  standard_minute_value >= 23.04
| | | | | |  over_time < 6900 : 0.23 - 0.38 (6/4) [3/1]
| | | | | |  over_time >= 6900 : 0.38 - 0.54 (4/0) [0/0]
| | | |  incentive >= 11.5 : 0.54 - 0.69 (23/1) [10/2]
| |  targeted_productivity >= 0.68 : 0.69 - 0.84 (66/29) [38/20]
incentive >= 33
|  incentive < 69.5
| |  incentive < 58
| | |  targeted_productivity < 0.68
| | | |  standard_minute_value < 24.59 : 0.54 - 0.69 (5/0) [9/3]
| | | |  standard_minute_value >= 24.59 : 0.69 - 0.84 (4/1) [1/0]
| | |  targeted_productivity >= 0.68 : 0.69 - 0.84 (165/10) [93/6]
| |  incentive >= 58
| | |  incentive < 61
| | | |  targeted_productivity < 0.75 : 0.69 - 0.84 (8/0) [1/0]
| | | |  targeted_productivity >= 0.75 : 0.84 - 1.00 (16/0) [3/0]
| | |  incentive >= 61
| | | |  incentive < 64 : 0.69 - 0.84 (42/0) [20/0]
| | | |  incentive >= 64
| | | | |  over_time < 8550 : 0.84 - 1.00 (4/1) [2/0]
| | | | |  over_time >= 8550 : 0.69 - 0.84 (6/0) [0/0]
|  incentive >= 69.5 : 0.84 - 1.00 (68/8) [38/5]
```

Popular Implementation 2: sklearn DecisionTreeClassifier class

sklearn’s DecisionTreeClassifier class was used as the second popular tool. It consists of a package for the python programming language, which could use supplied data to construct classification decision trees. As of this report there is a bug in the package such that it is unable to read in both continuous and nominal attributes while still treating the nominal attributes correctly; it is for this reason that the training data had all nominal attributes removed for all implementations (and furthermore why the 3rd implementation was made to only deal with continuous attributes).

Method

The code used to drive this package will be displayed below (though please not that the definition of the testTree function will not be shown as it is virtually identical to the class method holding the same name in implementation 3 except for the method in which it runs the test cases through the tree)

```
 # sklearn decision tree implementation
from sklearn.tree import DecisionTreeClassifier
from matplotlib import pyplot as plt
from sklearn import datasets
from sklearn import tree

# variable definition
in_file = "continuousv2"
in_test = "continuousv2test"

# read in the dataset
data = pd.read_csv(in_file+".csv")

# split the dataset into independent and independant
X = data[data.columns[:-1]].values
Y = data['actual_productivity'].values

# initialise tree object
dt = DecisionTreeClassifier()

# build tree
dt.fit(X,Y)

# test the tree
testTree(dt, in_test)

# print the tree
text_representation = tree.export_text(dt, max_depth=99)
print(text_representation)
```

Results

The testTree tree evaluation function found the following results:

Class	Accuracy	Precision	Recall	F1-value
0.23 - 0.38	0.800	0.500	0.300	0.375
0.38 - 0.54	0.840	0.667	0.400	0.500
0.54 - 0.69	0.720	0.400	0.800	0.533
0.69 - 0.84	0.660	0.000	0.000	0.000
0.84 - 1	0.780	0.455	0.500	0.476
Average	0.760	0.404	0.400	0.377

The tree generated

```
- incentive <= 29.50
 |- work_in_progress <= 3.50
 | |- standard_minute_value <= 3.92
 | | |- no_of_workers <= 8.50
 | | | |- targeted_productivity <= 0.78
 | | | | |- standard_minute_value <= 3.40
 | | | | | |- targeted_productivity <= 0.65
 | | | | | | |- targeted_productivity <= 0.55
 | | | | | | | |- actual_productivity --> 0.23 - 0.38
 | | | | | | |- targeted_productivity >  0.55
 | | | | | | | |- actual_productivity --> 0.23 - 0.38
 | | | | | |- targeted_productivity >  0.65
 | | | | | | |- targeted_productivity <= 0.72
 | | | | | | | |- over_time <= 2160.00
 | | | | | | | | |- no_of_workers <= 7.00
 | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
 | | | | | | | | |- no_of_workers >  7.00
 | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
 | | | | | | | |- over_time >  2160.00
 | | | | | | | | |- actual_productivity --> 0.38 - 0.54
 | | | | | | |- targeted_productivity >  0.72
 | | | | | | | |- over_time <= 2160.00
 | | | | | | | | |- actual_productivity --> 0.38 - 0.54
 | | | | | | | |- over_time >  2160.00
 | | | | | | | | |- actual_productivity --> 0.23 - 0.38
 | | | | |- standard_minute_value >  3.40
 | | | | | |- over_time <= 1440.00
 | | | | | | |- targeted_productivity <= 0.55
 | | | | | | | |- actual_productivity --> 0.54 - 0.69
 | | | | | | |- targeted_productivity >  0.55
 | | | | | | | |- targeted_productivity <= 0.62
 | | | | | | | | |- actual_productivity --> 0.38 - 0.54
 | | | | | | | |- targeted_productivity >  0.62
 | | | | | | | | |- targeted_productivity <= 0.67
 | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
 | | | | | | | | |- targeted_productivity >  0.67
 | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
 | | | | | |- over_time >  1440.00
 | | | | | | |- actual_productivity --> 0.38 - 0.54
 | | | |- targeted_productivity >  0.78
 | | | | |- over_time <= 1200.00
 | | | | | |- over_time <= 930.00
 | | | | | | |- over_time <= 810.00
 | | | | | | | |- actual_productivity --> 0.23 - 0.38
 | | | | | | |- over_time >  810.00
 | | | | | | | |- actual_productivity --> 0.38 - 0.54
 | | | | | |- over_time >  930.00
 | | | | | | |- actual_productivity --> 0.54 - 0.69
 | | | | |- over_time >  1200.00
 | | | | | |- over_time <= 3600.00
 | | | | | | |- over_time <= 1920.00
 | | | | | | | |- actual_productivity --> 0.54 - 0.69
 | | | | | | |- over_time >  1920.00
 | | | | | | | |- actual_productivity --> 0.54 - 0.69
 | | | | | |- over_time >  3600.00
 | | | | | | |- actual_productivity --> 0.38 - 0.54
 | | |- no_of_workers >  8.50
 | | | |- over_time <= 5970.00
 | | | | |- targeted_productivity <= 0.55
 | | | | | |- actual_productivity --> 0.84 - 1.00
 | | | | |- targeted_productivity >  0.55
 | | | | | |- targeted_productivity <= 0.65
 | | | | | | |- no_of_workers <= 11.00
 | | | | | | | |- actual_productivity --> 0.23 - 0.38
 | | | | | | |- no_of_workers >  11.00
 | | | | | | | |- actual_productivity --> 0.69 - 0.84
 | | | | | |- targeted_productivity >  0.65
 | | | | | | |- standard_minute_value <= 3.40
 | | | | | | | |- targeted_productivity <= 0.78
 | | | | | | | | |- over_time <= 3720.00
 | | | | | | | | | |- over_time <= 1500.00
 | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
 | | | | | | | | | |- over_time >  1500.00
 | | | | | | | | | | |- targeted_productivity <= 0.72
 | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
 | | | | | | | | | | |- targeted_productivity >  0.72
 | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
 | | | | | | | | |- over_time >  3720.00
 | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
 | | | | | | | |- targeted_productivity >  0.78
 | | | | | | | | |- over_time <= 1710.00
 | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
 | | | | | | | | |- over_time >  1710.00
 | | | | | | | | | |- over_time <= 2100.00
 | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
 | | | | | | | | | |- over_time >  2100.00
 | | | | | | | | | | |- over_time <= 3360.00
 | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
 | | | | | | | | | | |- over_time >  3360.00
 | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
 | | | | | | |- standard_minute_value >  3.40
 | | | | | | | |- actual_productivity --> 0.84 - 1.00
 | | | |- over_time >  5970.00
 | | | | |- actual_productivity --> 0.54 - 0.69
 | |- standard_minute_value >  3.92
 | | |- no_of_workers <= 9.50
 | | | |- over_time <= 3030.00
 | | | | |- targeted_productivity <= 0.78
 | | | | | |- standard_minute_value <= 4.45
 | | | | | | |- no_of_workers <= 8.50
 | | | | | | | |- targeted_productivity <= 0.72
 | | | | | | | | |- over_time <= 1200.00
 | | | | | | | | | |- no_of_workers <= 6.50
 | | | | | | | | | | |- standard_minute_value <= 4.05
 | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
 | | | | | | | | | | |- standard_minute_value >  4.05
 | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
 | | | | | | | | | |- no_of_workers >  6.50
 | | | | | | | | | | |- targeted_productivity <= 0.67
 | | | | | | | | | | | |- targeted_productivity <= 0.42
 | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
 | | | | | | | | | | | |- targeted_productivity >  0.42
 | | | | | | | | | | | | |- standard_minute_value <= 4.05
 | | | | | | | | | | | | | |- targeted_productivity <= 0.62
 | | | | | | | | | | | | | | |- targeted_productivity <= 0.55
 | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
 | | | | | | | | | | | | | | |- targeted_productivity >  0.55
 | | | | | | | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
 | | | | | | | | | | | | | |- targeted_productivity >  0.62
 | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
 | | | | | | | | | | | | |- standard_minute_value >  4.05
 | | | | | | | | | | | | | |- targeted_productivity <= 0.57
 | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
 | | | | | | | | | | | | | |- targeted_productivity >  0.57
 | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
 | | | | | | | | | | |- targeted_productivity >  0.67
 | | | | | | | | | | | |- standard_minute_value <= 4.05
 | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
 | | | | | | | | | | | |- standard_minute_value >  4.05
 | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
 | | | | | | | | |- over_time >  1200.00
 | | | | | | | | | |- standard_minute_value <= 4.05
 | | | | | | | | | | |- targeted_productivity <= 0.62
 | | | | | | | | | | | |- actual_productivity --> 0.23 - 0.38
 | | | | | | | | | | |- targeted_productivity >  0.62
 | | | | | | | | | | | |- over_time <= 1920.00
 | | | | | | | | | | | | |- actual_productivity --> 0.23 - 0.38
 | | | | | | | | | | | |- over_time >  1920.00
 | | | | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
 | | | | | | | | | |- standard_minute_value >  4.05
 | | | | | | | | | | |- targeted_productivity <= 0.55
 | | | | | | | | | | | |- standard_minute_value <= 4.23
 | | | | | | | | | | | | |- actual_productivity --> 0.23 - 0.38
 | | | | | | | | | | | |- standard_minute_value >  4.23
 | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
 | | | | | | | | | | |- targeted_productivity >  0.55
 | | | | | | | | | | | |- over_time <= 1920.00
 | | | | | | | | | | | | |- targeted_productivity <= 0.67
 | | | | | | | | | | | | | |- targeted_productivity <= 0.62
 | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
 | | | | | | | | | | | | | |- targeted_productivity >  0.62
 | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
 | | | | | | | | | | | | |- targeted_productivity >  0.67
 | | | | | | | | | | | | | |- actual_productivity --> 0.23 - 0.38
 | | | | | | | | | | | |- over_time >  1920.00
 | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
 | | | | | | | |- targeted_productivity >  0.72
 | | | | | | | | |- over_time <= 1740.00
 | | | | | | | | | |- over_time <= 1500.00
 | | | | | | | | | | |- standard_minute_value <= 4.05
 | | | | | | | | | | | |- over_time <= 1200.00
 | | | | | | | | | | | | |- over_time <= 600.00
 | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
 | | | | | | | | | | | | |- over_time >  600.00
 | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
 | | | | | | | | | | | |- over_time >  1200.00
 | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
 | | | | | | | | | | |- standard_minute_value >  4.05
 | | | | | | | | | | | |- no_of_workers <= 5.00
 | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
 | | | | | | | | | | | |- no_of_workers >  5.00
 | | | | | | | | | | | | |- over_time <= 1200.00
 | | | | | | | | | | | | | |- actual_productivity --> 0.23 - 0.38
 | | | | | | | | | | | | |- over_time >  1200.00
 | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
 | | | | | | | | | |- over_time >  1500.00
 | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
 | | | | | | | | |- over_time >  1740.00
 | | | | | | | | | |- standard_minute_value <= 4.05
 | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
 | | | | | | | | | |- standard_minute_value >  4.05
 | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
 | | | | | | |- no_of_workers >  8.50
 | | | | | | | |- over_time <= 2160.00
 | | | | | | | | |- over_time <= 1350.00
 | | | | | | | | | |- standard_minute_value <= 4.12
 | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
 | | | | | | | | | |- standard_minute_value >  4.12
 | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
 | | | | | | | | |- over_time >  1350.00
 | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
 | | | | | | | |- over_time >  2160.00
| | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | |- standard_minute_value >  4.45
| | | | | | |- over_time <= 1440.00
| | | | | | | |- standard_minute_value <= 4.87
| | | | | | | | |- targeted_productivity <= 0.72
| | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | |- targeted_productivity >  0.72
| | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | |- standard_minute_value >  4.87
| | | | | | | | |- targeted_productivity <= 0.72
| | | | | | | | | |- targeted_productivity <= 0.60
| | | | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | | |- targeted_productivity >  0.60
| | | | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | |- targeted_productivity >  0.72
| | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | |- over_time >  1440.00
| | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | |- targeted_productivity >  0.78
| | | | | |- over_time <= 1020.00
| | | | | | |- standard_minute_value <= 4.87
| | | | | | | |- standard_minute_value <= 4.05
| | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | |- standard_minute_value >  4.05
| | | | | | | | |- no_of_workers <= 6.50
| | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | | |- no_of_workers >  6.50
| | | | | | | | | |- standard_minute_value <= 4.23
| | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | |- standard_minute_value >  4.23
| | | | | | | | | | |- standard_minute_value <= 4.45
| | | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | | | | |- standard_minute_value >  4.45
| | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | |- standard_minute_value >  4.87
| | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | |- over_time >  1020.00
| | | | | | |- standard_minute_value <= 4.01
| | | | | | | |- over_time <= 1350.00
| | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | |- over_time >  1350.00
| | | | | | | | |- over_time <= 1680.00
| | | | | | | | | |- no_of_workers <= 5.00
| | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | |- no_of_workers >  5.00
| | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | |- over_time >  1680.00
| | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | |- standard_minute_value >  4.01
| | | | | | | |- no_of_workers <= 8.50
| | | | | | | | |- over_time <= 1350.00
| | | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | |- over_time >  1350.00
| | | | | | | | | |- over_time <= 1950.00
| | | | | | | | | | |- over_time <= 1680.00
| | | | | | | | | | | |- no_of_workers <= 7.50
| | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | |- no_of_workers >  7.50
| | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | |- over_time >  1680.00
| | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | |- over_time >  1950.00
| | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | |- no_of_workers >  8.50
| | | | | | | | |- standard_minute_value <= 4.12
| | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | |- standard_minute_value >  4.12
| | | | | | | | | |- over_time <= 1890.00
| | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | |- over_time >  1890.00
```

```
| | | | | | | | | | |- standard_minute_value <= 4.87
| | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | |- standard_minute_value >  4.87
| | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | |- over_time >  3030.00
| | | | |- no_of_workers <= 8.50
| | | | | |- standard_minute_value <= 4.27
| | | | | | |- targeted_productivity <= 0.55
| | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | |- targeted_productivity >  0.55
| | | | | | | |- targeted_productivity <= 0.67
| | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | |- targeted_productivity >  0.67
| | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | |- standard_minute_value >  4.27
| | | | | | |- targeted_productivity <= 0.78
| | | | | | | |- targeted_productivity <= 0.72
| | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | |- targeted_productivity >  0.72
| | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | |- targeted_productivity >  0.78
| | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | |- no_of_workers >  8.50
| | | | | |- standard_minute_value <= 4.34
| | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | |- standard_minute_value >  4.34
| | | | | | |- actual_productivity --> 0.54 - 0.69
| | |- no_of_workers >  9.50
| | | |- no_of_workers <= 15.50
| | | | |- over_time <= 1380.00
| | | | | |- targeted_productivity <= 0.78
| | | | | | |- standard_minute_value <= 4.05
| | | | | | | |- no_of_workers <= 10.50
| | | | | | | | |- targeted_productivity <= 0.65
| | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | |- targeted_productivity >  0.65
| | | | | | | | | |- targeted_productivity <= 0.72
| | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | |- targeted_productivity >  0.72
| | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | |- no_of_workers >  10.50
| | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | |- standard_minute_value >  4.05
| | | | | | | |- standard_minute_value <= 4.64
| | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | |- standard_minute_value >  4.64
| | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | |- targeted_productivity >  0.78
| | | | | | |- standard_minute_value <= 4.12
| | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | |- standard_minute_value >  4.12
| | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | |- over_time >  1380.00
| | | | | |- over_time <= 8700.00
| | | | | | |- over_time <= 3900.00
| | | | | | | |- over_time <= 2820.00
| | | | | | | | |- targeted_productivity <= 0.78
| | | | | | | | | |- targeted_productivity <= 0.42
| | | | | | | | | | |- standard_minute_value <= 4.23
| | | | | | | | | | | |- over_time <= 1620.00
| | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | |- over_time >  1620.00
| | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | |- standard_minute_value >  4.23
| | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | |- targeted_productivity >  0.42
| | | | | | | | | | |- no_of_workers <= 11.00
| | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | |- no_of_workers >  11.00
| | | | | | | | | | | |- over_time <= 1620.00
```

```
| | | | | | | | | | | | |- targeted_productivity <= 0.67
| | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | |- targeted_productivity >  0.67
| | | | | | | | | | | | | |- targeted_productivity <= 0.72
| | | | | | | | | | | | | | |- standard_minute_value <= 4.05
| | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | |- standard_minute_value >  4.05
| | | | | | | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | | | | | | | |- targeted_productivity >  0.72
| | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | |- over_time >  1620.00
| | | | | | | | | | | | |- over_time <= 2610.00
| | | | | | | | | | | | | |- over_time <= 2340.00
| | | | | | | | | | | | | | |- standard_minute_value <= 4.05
| | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | |- standard_minute_value >  4.05
| | | | | | | | | | | | | | | |- standard_minute_value <= 4.23
| | | | | | | | | | | | | | | | |- no_of_workers <= 13.50
| | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | |- no_of_workers >  13.50
| | | | | | | | | | | | | | | | | |- targeted_productivity <= 0.72
| | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | | |- targeted_productivity >  0.72
| | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | |- standard_minute_value >  4.23
| | | | | | | | | | | | | | | | |- targeted_productivity <= 0.62
| | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | |- targeted_productivity >  0.62
| | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | |- over_time >  2340.00
| | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | |- over_time >  2610.00
| | | | | | | | | | | | | |- targeted_productivity <= 0.72
| | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | |- targeted_productivity >  0.72
| | | | | | | | | | | | | | |- over_time <= 2730.00
| | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | |- over_time >  2730.00
| | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | |- targeted_productivity >  0.78
| | | | | | | | | |- over_time <= 2550.00
| | | | | | | | | | |- over_time <= 2100.00
| | | | | | | | | | | |- standard_minute_value <= 4.87
| | | | | | | | | | | | |- standard_minute_value <= 4.23
| | | | | | | | | | | | | |- no_of_workers <= 13.50
| | | | | | | | | | | | | | |- standard_minute_value <= 4.05
| | | | | | | | | | | | | | | |- no_of_workers <= 11.00
| | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | |- no_of_workers >  11.00
| | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | |- standard_minute_value >  4.05
| | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | |- no_of_workers >  13.50
| | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | |- standard_minute_value >  4.23
| | | | | | | | | | | | | |- standard_minute_value <= 4.45
| | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | |- standard_minute_value >  4.45
| | | | | | | | | | | | | | |- no_of_workers <= 13.50
| | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | |- no_of_workers >  13.50
| | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | |- standard_minute_value >  4.87
| | | | | | | | | | | | |- no_of_workers <= 13.50
| | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | |- no_of_workers >  13.50
| | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | |- over_time >  2100.00
| | | | | | | | | | | |- standard_minute_value <= 4.54
| | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | |- standard_minute_value >  4.54
| | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | |- over_time >  2550.00
| | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | |- over_time >  2820.00
| | | | | | | | |- standard_minute_value <= 4.87
| | | | | | | | | |- over_time <= 3240.00
| | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | |- over_time >  3240.00
| | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | |- standard_minute_value >  4.87
| | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | |- over_time >  3900.00
| | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | |- over_time >  8700.00
| | | | | | |- targeted_productivity <= 0.72
| | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | |- targeted_productivity >  0.72
| | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | |- no_of_workers >  15.50
| | | | |- over_time <= 13800.00
| | | | | |- over_time <= 4080.00
| | | | | | |- over_time <= 2340.00
| | | | | | | |- over_time <= 2220.00
| | | | | | | | |- targeted_productivity <= 0.78
| | | | | | | | | |- targeted_productivity <= 0.72
| | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | |- targeted_productivity >  0.72
| | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | |- targeted_productivity >  0.78
| | | | | | | | | |- standard_minute_value <= 4.54
| | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | |- standard_minute_value >  4.54
| | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | |- over_time >  2220.00
| | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | |- over_time >  2340.00
| | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | |- over_time >  4080.00
| | | | | | |- over_time <= 4920.00
| | | | | | | |- over_time <= 4380.00
| | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | |- over_time >  4380.00
| | | | | | | | |- over_time <= 4620.00
| | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | |- over_time >  4620.00
| | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | |- over_time >  4920.00
| | | | | | | |- over_time <= 11250.00
| | | | | | | | |- standard_minute_value <= 4.45
| | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | |- standard_minute_value >  4.45
| | | | | | | | | |- over_time <= 6780.00
| | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | |- over_time >  6780.00
| | | | | | | | | | |- no_of_workers <= 19.00
| | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | |- no_of_workers >  19.00
| | | | | | | | | | | |- no_of_workers <= 22.00
| | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | |- no_of_workers >  22.00
| | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | |- over_time >  11250.00
| | | | | | | | |- targeted_productivity <= 0.75
| | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | | |- targeted_productivity >  0.75
| | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | |- over_time >  13800.00
| | | | | |- actual_productivity --> 0.54 - 0.69
|- work_in_progress >  3.50
| |- targeted_productivity <= 0.55
| | |- targeted_productivity <= 0.42
| | | |- over_time <= 4650.00
| | | | |- actual_productivity --> 0.38 - 0.54
| | | |- over_time >  4650.00
| | | | |- targeted_productivity <= 0.21
| | | | | |- actual_productivity --> 0.38 - 0.54
| | | | |- targeted_productivity >  0.21
| | | | | |- actual_productivity --> 0.23 - 0.38
| | |- targeted_productivity >  0.42
| | | |- standard_minute_value <= 30.22
| | | | |- work_in_progress <= 1622.50
| | | | | |- work_in_progress <= 1094.00
| | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | |- work_in_progress >  1094.00
| | | | | | |- work_in_progress <= 1248.00
| | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | |- work_in_progress >  1248.00
| | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | |- work_in_progress >  1622.50
| | | | | |- actual_productivity --> 0.23 - 0.38
| | | |- standard_minute_value >  30.22
| | | | |- no_of_style_change <= 0.50
| | | | | |- actual_productivity --> 0.23 - 0.38
| | | | |- no_of_style_change >  0.50
| | | | | |- work_in_progress <= 999.50
| | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | |- work_in_progress >  999.50
| | | | | | |- actual_productivity --> 0.38 - 0.54
| |- targeted_productivity >  0.55
| | |- incentive <= 22.00
| | | |- idle_men <= 22.50
| | | | |- over_time <= 5910.00
| | | | | |- targeted_productivity <= 0.67
| | | | | | |- no_of_workers <= 52.50
| | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | |- no_of_workers >  52.50
| | | | | | | |- no_of_style_change <= 0.50
| | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | |- no_of_style_change >  0.50
| | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | |- targeted_productivity >  0.67
| | | | | | |- work_in_progress <= 452.50
| | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | |- work_in_progress >  452.50
| | | | | | | |- over_time <= 3420.00
| | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | |- over_time >  3420.00
| | | | | | | | |- standard_minute_value <= 24.45
| | | | | | | | | |- work_in_progress <= 864.50
| | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | |- work_in_progress >  864.50
| | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | |- standard_minute_value >  24.45
| | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | |- over_time >  5910.00
| | | | | |- no_of_workers <= 54.00
| | | | | | |- no_of_style_change <= 1.50
| | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | |- no_of_style_change >  1.50
| | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | |- no_of_workers >  54.00
| | | | | | |- targeted_productivity <= 0.67
| | | | | | | |- standard_minute_value <= 30.22
| | | | | | | | |- over_time <= 7830.00
| | | | | | | | | |- work_in_progress <= 1315.50
| | | | | | | | | | |- standard_minute_value <= 23.62
| | | | | | | | | | | |- work_in_progress <= 894.00
| | | | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | | | | | |- work_in_progress >  894.00
| | | | | | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | | | |- standard_minute_value >  23.62
| | | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | | | |- work_in_progress >  1315.50
| | | | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | |- over_time >  7830.00
| | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | |- standard_minute_value >  30.22
| | | | | | | | |- work_in_progress <= 854.50
| | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | |- work_in_progress >  854.50
| | | | | | | | | |- standard_minute_value <= 30.40
| | | | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | | |- standard_minute_value >  30.40
| | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | |- targeted_productivity >  0.67
| | | | | | | |- work_in_progress <= 272.50
| | | | | | | | |- targeted_productivity <= 0.75
| | | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | |- targeted_productivity >  0.75
| | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | |- work_in_progress >  272.50
| | | | | | | | |- standard_minute_value <= 35.76
| | | | | | | | | |- work_in_progress <= 2168.50
| | | | | | | | | | |- idle_men <= 17.50
| | | | | | | | | | | |- over_time <= 8700.00
| | | | | | | | | | | | |- no_of_workers <= 56.50
| | | | | | | | | | | | | |- standard_minute_value <= 23.90
| | | | | | | | | | | | | | |- targeted_productivity <= 0.75
| | | | | | | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | | | | | | | | |- targeted_productivity >  0.75
| | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | |- standard_minute_value >  23.90
| | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | |- no_of_workers >  56.50
| | | | | | | | | | | | | |- standard_minute_value <= 29.75
| | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | |- standard_minute_value >  29.75
| | | | | | | | | | | | | | |- work_in_progress <= 915.50
| | | | | | | | | | | | | | | |- work_in_progress <= 716.50
| | | | | | | | | | | | | | | | |- targeted_productivity <= 0.72
| | | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | | |- targeted_productivity >  0.72
| | | | | | | | | | | | | | | | | |- work_in_progress <= 679.50
| | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | |- work_in_progress >  679.50
| | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | |- work_in_progress >  716.50
| | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | |- work_in_progress >  915.50
| | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | |- over_time >  8700.00
| | | | | | | | | | | | |- work_in_progress <= 1119.00
| | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | |- work_in_progress >  1119.00
| | | | | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | | | | |- idle_men >  17.50
| | | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | | | |- work_in_progress >  2168.50
| | | | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | |- standard_minute_value >  35.76
| | | | | | | | | |- no_of_workers <= 57.50
| | | | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | | |- no_of_workers >  57.50
| | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | |- idle_men >  22.50
| | | | |- idle_men <= 42.50
| | | | | |- actual_productivity --> 0.23 - 0.38
| | | | |- idle_men >  42.50
| | | | | |- actual_productivity --> 0.54 - 0.69
| | |- incentive >  22.00
| | | |- standard_minute_value <= 48.34
| | | | |- targeted_productivity <= 0.67
| | | | | |- work_in_progress <= 868.50
| | | | | | |- work_in_progress <= 847.00
| | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | |- work_in_progress >  847.00
| | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | |- work_in_progress >  868.50
| | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | |- targeted_productivity >  0.67
| | | | | |- standard_minute_value <= 28.79
| | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | |- standard_minute_value >  28.79
| | | | | | |- actual_productivity --> 0.54 - 0.69
| | | |- standard_minute_value >  48.34
| | | | |- over_time <= 4200.00
| | | | | |- actual_productivity --> 0.38 - 0.54
| | | | |- over_time >  4200.00
| | | | | |- actual_productivity --> 0.23 - 0.38
- incentive >  29.50
|- incentive <= 69.50
| |- targeted_productivity <= 0.67
| | |- standard_minute_value <= 22.73
| | | |- targeted_productivity <= 0.55
| | | | |- over_time <= 18045.00
| | | | | |- work_in_progress <= 1257.50
| | | | | | |- standard_minute_value <= 13.71
| | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | |- standard_minute_value >  13.71
| | | | | | | |- work_in_progress <= 1009.50
| | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | |- work_in_progress >  1009.50
| | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | |- work_in_progress >  1257.50
| | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | |- over_time >  18045.00
| | | | | |- actual_productivity --> 0.23 - 0.38
| | | |- targeted_productivity >  0.55
| | | | |- actual_productivity --> 0.54 - 0.69
| | |- standard_minute_value >  22.73
| | | |- incentive <= 32.00
| | | | |- over_time <= 6900.00
| | | | | |- no_of_workers <= 57.50
| | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | |- no_of_workers >  57.50
| | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | |- over_time >  6900.00
| | | | | |- work_in_progress <= 1471.00
| | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | |- work_in_progress >  1471.00
| | | | | | |- actual_productivity --> 0.54 - 0.69
| | | |- incentive >  32.00
| | | | |- no_of_style_change <= 0.50
| | | | | |- actual_productivity --> 0.69 - 0.84
| | | | |- no_of_style_change >  0.50
| | | | | |- over_time <= 2580.00
| | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | |- over_time >  2580.00
| | | | | | |- actual_productivity --> 0.54 - 0.69
| |- targeted_productivity >  0.67
| | |- incentive <= 58.00
| | | |- standard_minute_value <= 30.36
| | | | |- idle_time <= 3.00
| | | | | |- work_in_progress <= 1595.00
| | | | | | |- work_in_progress <= 22.00
| | | | | | | |- no_of_style_change <= 0.50
| | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | |- no_of_style_change >  0.50
| | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | |- work_in_progress >  22.00
| | | | | | | |- incentive <= 51.50
| | | | | | | | |- no_of_style_change <= 1.50
| | | | | | | | | |- work_in_progress <= 540.00
| | | | | | | | | | |- work_in_progress <= 521.50
| | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | |- work_in_progress >  521.50
| | | | | | | | | | | |- over_time <= 8257.50
| | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | |- over_time >  8257.50
| | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | |- work_in_progress >  540.00
| | | | | | | | | | |- over_time <= 6750.00
| | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | |- over_time >  6750.00
| | | | | | | | | | | |- over_time <= 6810.00
| | | | | | | | | | | | |- work_in_progress <= 1043.00
| | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | |- work_in_progress >  1043.00
| | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | |- over_time >  6810.00
| | | | | | | | | | | | |- no_of_workers <= 56.50
| | | | | | | | | | | | | |- standard_minute_value <= 24.42
| | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | |- standard_minute_value >  24.42
| | | | | | | | | | | | | | |- work_in_progress <= 1023.50
| | | | | | | | | | | | | | | |- over_time <= 8820.00
| | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | |- over_time >  8820.00
| | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | |- work_in_progress >  1023.50
| | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | |- no_of_workers >  56.50
| | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | |- no_of_style_change >  1.50
| | | | | | | | | |- work_in_progress <= 545.00
| | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | | | |- work_in_progress >  545.00
| | | | | | | | | | |- no_of_workers <= 55.00
| | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | |- no_of_workers >  55.00
| | | | | | | | | | | |- over_time <= 7620.00
| | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | |- over_time >  7620.00
| | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | |- incentive >  51.50
| | | | | | | | |- no_of_workers <= 32.00
| | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | |- no_of_workers >  32.00
| | | | | | | | | |- incentive <= 54.00
| | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | |- incentive >  54.00
| | | | | | | | | | |- no_of_workers <= 53.00
| | | | | | | | | | | |- no_of_workers <= 42.50
| | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | |- no_of_workers >  42.50
| | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | |- no_of_workers >  53.00
| | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | |- work_in_progress >  1595.00
| | | | | | |- no_of_style_change <= 0.50
| | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | |- no_of_style_change >  0.50
| | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | |- idle_time >  3.00
| | | | | |- actual_productivity --> 0.54 - 0.69
| | | |- standard_minute_value >  30.36
| | | | |- work_in_progress <= 423.00
| | | | | |- actual_productivity --> 0.54 - 0.69
| | | | |- work_in_progress >  423.00
| | | | | |- over_time <= 10215.00
| | | | | | |- no_of_workers <= 55.50
| | | | | | | |- work_in_progress <= 516.50
| | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | |- work_in_progress >  516.50
| | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | |- no_of_workers >  55.50
| | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | |- over_time >  10215.00
| | | | | | |- work_in_progress <= 1161.00
| | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | |- work_in_progress >  1161.00
| | | | | | | |- actual_productivity --> 0.69 - 0.84
| | |- incentive >  58.00
| | | |- incentive <= 61.00
| | | | |- targeted_productivity <= 0.75
| | | | | |- actual_productivity --> 0.69 - 0.84
| | | | |- targeted_productivity >  0.75
| | | | | |- actual_productivity --> 0.84 - 1.00
| | | |- incentive >  61.00
| | | | |- incentive <= 64.00
| | | | | |- actual_productivity --> 0.69 - 0.84
| | | | |- incentive >  64.00
| | | | | |- incentive <= 67.00
| | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | |- incentive >  67.00
| | | | | | |- actual_productivity --> 0.69 - 0.84
|- incentive >  69.50
| |- targeted_productivity <= 0.72
| | |- work_in_progress <= 507.50
| | | |- targeted_productivity <= 0.62
| | | | |- actual_productivity --> 0.84 - 1.00
| | | |- targeted_productivity >  0.62
| | | | |- standard_minute_value <= 3.40
| | | | | |- actual_productivity --> 0.38 - 0.54
| | | | |- standard_minute_value >  3.40
| | | | | |- no_of_workers <= 9.00
| | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | |- no_of_workers >  9.00
| | | | | | |- actual_productivity --> 0.54 - 0.69
| | |- work_in_progress >  507.50
| | | |- incentive <= 72.50
| | | | |- actual_productivity --> 0.84 - 1.00
| | | |- incentive >  72.50
| | | | |- actual_productivity --> 0.69 - 0.84
| |- targeted_productivity >  0.72
| | |- standard_minute_value <= 8.10
| | | |- targeted_productivity <= 0.78
| | | | |- actual_productivity --> 0.69 - 0.84
| | | |- targeted_productivity >  0.78
| | | | |- no_of_workers <= 10.50
| | | | | |- actual_productivity --> 0.84 - 1.00
| | | | |- no_of_workers >  10.50
| | | | | |- actual_productivity --> 0.69 - 0.84
| | |- standard_minute_value >  8.10
| | | |- actual_productivity --> 0.84 - 1.00
```

Own Implementation

For the 3rd implementation code was written in python to use the training data in the construction of a decision tree. The code consists of two classes (‘DecisionTreeMaker’ and ‘treeNode’). Because of the data being trained on, it has been implemented to only train on continuous. I build the trees through a recursive function which will take a given dataset, find the attribute and value which would best divide the dependent classes, and then make this divide recalling itself on the two new halves.

Results

The testTree tree evaluation method found the following results:

Class	Accuracy	Precision	Recall	F1-value
0.23 - 0.38	0.760	0.400	0.400	0.400
0.38 - 0.54	0.800	0.500	0.500	0.500
0.54 - 0.69	0.880	0.667	0.800	0.727
0.69 - 0.84	0.700	0.000	0.000	0.000
0.84 - 1	0.780	0.462	0.600	0.522
Average	0.784	0.406	0.460	0.430

Method

The code used to drive this package will be displayed below (though please not that the definition of the testTree function will not be shown as it is virtually identical to the class method holding the same name in implementation 3)

```
# imports
import numpy as np
import pandas as pd

# class definitions
class TreeNode:

    # constructor
    def __init__(self, i = None, v = None, l = None, r = None, d = None):

        # initialise variables
        self.attribute_index = i
        self.value = v
        self.left_child = l
        self.right_child = r
        self.dependant = d

    # run input through the node
    def runInput(self, x):

        # if this is a leaf node
        if (self.dependant != None):
            return self.dependant

        if (x[self.attribute_index] <= self.value):
            return self.left_child.runInput(x)

        else:
            return self.right_child.runInput(x)

    # prints the nodes information
    def printNode(self, names, indent=0):

        # make indent
        prefix = ("| " * indent) + "|-"

        # detection for leaf
        if (self.dependant == None):

            # print this node
            print(prefix, names[self.attribute_index], " : ", self.value)

            # if applicable print the children
            if (self.left_child != None):
                self.left_child.printNode(names, indent+1)

            if (self.left_child != None):
                self.right_child.printNode(names, indent+1)

        else:
            print(prefix, "actual_productivity -->" ,self.dependant)

# the decision tree maker
class DecisionTreeMaker:

    # constructor
    def __init__(self, in_file_name):

        # initialise variables
        self.attribute_names = None
        self.dataset = None
        self.tree_root = None
        self.branch_count = 0
        self.node_count = 0

        print("reading in input file...")

        # read in the data from the csv
        csv_filename = in_file_name + ".csv"
        data = pd.read_csv(csv_filename)

        # read in the attribute names
        self.attribute_names = data.columns.values

        # split the data into independant and independant
        x_train = data.iloc[:, :-1].values
        y_train = data.iloc[:, -1].values.reshape(-1,1)

        # create dataset
        self.dataset = np.concatenate((x_train, y_train), axis=1)

        # report attribute count
        instances, attributes = x_train.shape

        # announce progress
        print("succesfully read in data set with ", instances, " instances and ", (attributes+1), "attributes")

    # main functions

    # starts off the process of building the tree
    def buildTree(self):
        print("beginning to build the tree...")
        self.tree_root = self.buildBranch(self.dataset)
        print("tree build complete making a tree containing", self.branch_count, " branches and ", self.node_count, " nodes")

    # runs given test data through the tree returning the results
    def runThroughTree(self, x):
        # initialise answer holder
        y = []
        for input in x:
            y.append(self.tree_root.runInput(input))
        return y

    # runs tests on the accuracy of the tree using the test data
    def testTree(self, in_file_name):

        csv_filename = in_file_name + ".csv"
        data = pd.read_csv(csv_filename)

        x_test = data.iloc[:, :-1].values
        y_test = data.iloc[:, -1].values.reshape(-1,1)

        # run the test data through the tree
        y_result = self.runThroughTree(x_test)

        # display the results for each class
        for label in np.unique(y_test):
            # inform which class the tests are on
            print("the following results are on the ", label, " class")

            # perform and report class accuracy
            correct = 0
            for x in range(len(y_test)):
                # increment the correct count if the class is correctly
                # classified with regards to the given class
                if (((y_test[x] == label) and (y_result[x] == label)) or ((y_test[x] != label) and (y_result[x] != label))):
                    correct+=1

            print("accuracy_score: ", correct/len(y_test))

            # perform and report precision test
            tp = 0
            fp = 0
            for x in range(len(y_test)):
                # if correctly classed as label, increment
                if ((y_test[x] == label) and (y_result[x] == label)):
                    tp+=1
                # if classed as label when it shouldn't, increment
                if ((y_test[x] != label) and (y_result[x] == label)):
                    fp+=1

            precision = (tp/(tp+fp))
            print("precision: ", precision)

            # perform and report sklearn recall test
            tp = 0
            fn = 0
            for x in range(len(y_test)):
                # if correctly classed as label, increment
                if ((y_test[x] == label) and (y_result[x] == label)):
                    tp+=1
                # if not classed as label when it should, increment
                if ((y_test[x] == label) and (y_result[x] != label)):
                    fn+=1

            recall = (tp/(tp+fn))
            print("recall: ", recall)

            # perform and report sklearn f1 test
            if (precision + recall != 0):
                f1_score = ((2 * precision * recall)/(precision + recall))
            else:
                f1_score = 0

            print("F1: ", f1_score)

    # recursively prints the tree
    def printTree(self):
        self.tree_root.printNode(self.attribute_names)

    # supporting functions

    # a recurrsive function which builds out the tree from a given node position with using
    # the given data
    def buildBranch(self, dataset):

        # split the dataset into independant and dependant variables
        x, y = dataset[:,:-1],dataset[:,-1]

        # get instance count
        instances, attributes = np.shape(x)

        # increase node count
        self.node_count+=1

        # if there are is enough instances to split and the attributes have not run out
        if ((instances >= 4) and (attributes >=2)):

            # find the best split
            info, attribute_index, value, left_dataset, right_dataset = self.findBestSplit(dataset, attributes)

            # if this split results in an information gain
            # then perform the split
            if (info > 0):

                # increment branch count
                self.branch_count+=2

                # create child trees
                left_tree = self.buildBranch(left_dataset)
                right_tree = self.buildBranch(right_dataset)

                # return root node
                return TreeNode(attribute_index, value, left_tree, right_tree)

        # else compute leaf value (by finding the most frequently occuring value) and send it to the node
        dependant_variables = list(y)
        return TreeNode(d=max(dependant_variables, key=dependant_variables.count))

    # finds where the if would be best to split the dataset in such a way to better distingish between output
    def findBestSplit(self, dataset, attribute_count):

        # initialise
        out_info = 0
        out_attribute_index = None
        out_value = None
        out_left = None
        out_right = None

        # iterate through the attributes looking for one which
        # contains an instance holding the best split
        for attribute_index in range(attribute_count):

            # seperate instances, and unique instances
            instances = dataset[:, attribute_index]
            unique_instances = np.unique(instances)

            # loop through the unique instance values for the given attribute
            # looking for a good split
            for value in unique_instances:

                # make current split
                left_split = np.array([row for row in dataset if row[attribute_index]<=value])
                right_split = np.array([row for row in dataset if row[attribute_index]>value])

                # check splits aren't null
                if ((len(left_split) > 0) and (len(right_split) > 0)):

                    # measure info gain
                    tmp_info = self.measureInfoGain(dataset[:, -1], left_split[:, -1], right_split[:, -1])

                    # if the info gain is higher
                    if (tmp_info > out_info):

                        # replace all of the output values
                        out_attribute_index = attribute_index
                        out_value = value
                        out_left = left_split
                        out_right = right_split
                        out_info = tmp_info

        return out_info, out_attribute_index, out_value, out_left, out_right

    # measure the info gain of the given split over the given parent through the use of
    # the entropy formula
    def measureInfoGain(self, parent, new_left, new_right):

        # initialise variables
        left_entropy = 0
        right_entropy = 0
        parent_entropy = 0

        # calculate parent entropy
        for instance in np.unique(parent):
            probability = len(parent[parent==instance]) / len(parent)
            parent_entropy += -probability * np.log2(probability)

        # calculate child entropies
        for instance in np.unique(new_left):
            probability = len(new_left[new_left==instance]) / len(new_left)
            left_entropy += -probability * np.log2(probability)

        left_entropy = left_entropy * (len(new_left) / len(parent))

        for instance in np.unique(new_right):
            probability = len(new_right[new_right==instance]) / len(new_right)
            right_entropy += -probability * np.log(probability)

        right_entropy = right_entropy * (len(new_right) / len(parent))

        # return the gain
        return (parent_entropy - (left_entropy + right_entropy))

# beginning of program

# variable definition
in_file = "continuousv2"
in_test_files = "continuousv2test"

# construct tree maker object
tree_maker = DecisionTreeMaker(in_file)

# build the tree
tree_maker.buildTree()

print("performing tests")

# test the tree
tree_maker.testTree(in_test_files)

# print the tree
tree_maker.printTree()
```

The tree generated

```
|- targeted_productivity  :  0.35
| |- standard_minute_value  :  4.3
| | |- standard_minute_value  :  4.15
| | | |- standard_minute_value  :  2.9
| | | | |- actual_productivity --> 0.84 - 1.00
| | | | |- no_of_workers  :  10.0
| | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | |- actual_productivity --> 0.69 - 0.84
| | | |- actual_productivity --> 0.84 - 1.00
| | |- over_time  :  3240.0
| | | |- actual_productivity --> 0.38 - 0.54
| | | |- work_in_progress  :  1448.0
| | | | |- actual_productivity --> 0.23 - 0.38
| | | | |- actual_productivity --> 0.38 - 0.54
| |- targeted_productivity  :  0.5
| | |- standard_minute_value  :  5.13
| | | |- targeted_productivity  :  0.4
| | | | |- actual_productivity --> 0.54 - 0.69
| | | | |- standard_minute_value  :  2.9
| | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | |- standard_minute_value  :  3.9
| | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | |- standard_minute_value  :  3.94
| | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | |- no_of_workers  :  8.0
| | | | | | | | |- over_time  :  960.0
| | | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | |- standard_minute_value  :  4.15
| | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | |- incentive  :  23.0
| | | | |- no_of_workers  :  53.0
| | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | |- over_time  :  6720.0
| | | | | | |- no_of_workers  :  54.0
| | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | |- work_in_progress  :  749.0
| | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | |- standard_minute_value  :  20.79
| | | | | | |- actual_productivity --> 0.38 - 0.54
```

```
| | | | | | |- standard_minute_value  :  26.66
| | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | |- actual_productivity --> 0.38 - 0.54
| | |- no_of_workers  :  2.0
| | | |- standard_minute_value  :  3.94
| | | | |- actual_productivity --> 0.84 - 1.00
| | | | |- actual_productivity --> 0.54 - 0.69
| | | |- no_of_workers  :  6.0
| | | | |- targeted_productivity  :  0.65
| | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | |- no_of_workers  :  4.0
| | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | |- no_of_workers  :  7.0
| | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | |- targeted_productivity  :  0.6
| | | | | | |- standard_minute_value  :  2.9
| | | | | | | |- over_time  :  960.0
| | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | |- over_time  :  1200.0
| | | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | |- standard_minute_value  :  4.15
| | | | | | | | |- over_time  :  0.0
| | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | |- over_time  :  960.0
| | | | | | | | | | |- standard_minute_value  :  3.9
| | | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | | | | |- no_of_workers  :  9.0
| | | | | | | | | | | |- standard_minute_value  :  4.08
| | | | | | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | |- incentive  :  0.0
| | | | | | | | | |- no_of_style_change  :  0.0
| | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | | | | |- standard_minute_value  :  15.28
| | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | | |- standard_minute_value  :  20.4
| | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | |- incentive  :  25.0
```

```
| | | | | | | | | | | |- no_of_workers  :  56.0
| | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | |- work_in_progress  :  486.0
| | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | |- work_in_progress  :  1069.0
| | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | |- targeted_productivity  :  0.65
| | | | | | | |- over_time  :  0.0
| | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | |- standard_minute_value  :  4.15
| | | | | | | | | |- standard_minute_value  :  3.9
| | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | |- standard_minute_value  :  3.94
| | | | | | | | | | | |- over_time  :  2160.0
| | | | | | | | | | | | |- over_time  :  960.0
| | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | | | | | |- over_time  :  960.0
| | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | |- no_of_workers  :  12.0
| | | | | | | | | | | | | |- over_time  :  1440.0
| | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | |- standard_minute_value  :  22.53
| | | | | | | | | | |- incentive  :  49.0
| | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | |- incentive  :  0.0
| | | | | | | | | | | |- work_in_progress  :  841.0
| | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | |- over_time  :  6600.0
| | | | | | | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | | | | | |- incentive  :  26.0
| | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | |- no_of_style_change  :  0.0
| | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | |- standard_minute_value  :  3.9
| | | | | | | | |- over_time  :  0.0
| | | | | | | | | |- targeted_productivity  :  0.7
| | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | |- targeted_productivity  :  0.7
| | | | | | | | | | |- no_of_workers  :  8.0
| | | | | | | | | | | |- standard_minute_value  :  2.9
| | | | | | | | | | | | |- over_time  :  960.0
| | | | | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | |- standard_minute_value  :  2.9
| | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | |- targeted_productivity  :  0.75
| | | | | | | | | | | |- over_time  :  1800.0
| | | | | | | | | | | | |- over_time  :  960.0
| | | | | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | | | | | | | |- over_time  :  1200.0
| | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | |- over_time  :  3360.0
| | | | | | | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | |- over_time  :  960.0
| | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | |- over_time  :  1200.0
| | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | |- no_of_workers  :  8.0
| | | | | | | | | | | | | | |- over_time  :  2400.0
| | | | | | | | | | | | | | | |- over_time  :  1440.0
| | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | | | | | | | | |- over_time  :  1620.0
| | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | |- over_time  :  1800.0
| | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | |- over_time  :  360.0
| | | | | | | | | |- incentive  :  50.0
| | | | | | | | | | |- idle_time  :  0.0
| | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | |- incentive  :  138.0
| | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | |- targeted_productivity  :  0.75
| | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | |- over_time  :  840.0
| | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | |- over_time  :  960.0
| | | | | | | | | | | |- targeted_productivity  :  0.7
| | | | | | | | | | | | |- standard_minute_value  :  3.94
| | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | |- standard_minute_value  :  4.6
| | | | | | | | | | | | | | |- standard_minute_value  :  4.15
| | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | | | | | |- targeted_productivity  :  0.75
| | | | | | | | | | | | | |- standard_minute_value  :  3.94
| | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | | | | | | |- standard_minute_value  :  3.94
| | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | |- standard_minute_value  :  4.15
| | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | | | | |- over_time  :  1080.0
| | | | | | | | | | | | |- targeted_productivity  :  0.75
| | | | | | | | | | | | | |- targeted_productivity  :  0.7
| | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | |- standard_minute_value  :  4.08
| | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | |- standard_minute_value  :  5.13
| | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | |- over_time  :  1320.0
| | | | | | | | | | | | | |- no_of_workers  :  8.0
| | | | | | | | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | | | | | | | |- standard_minute_value  :  3.94
| | | | | | | | | | | | | | | |- targeted_productivity  :  0.7
| | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | |- targeted_productivity  :  0.75
| | | | | | | | | | | | | | | | | |- over_time  :  1200.0
| | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | |- standard_minute_value  :  5.13
| | | | | | | | | | | | | | | | |- standard_minute_value  :  4.15
| | | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | | | | | | |- standard_minute_value  :  3.94
| | | | | | | | | | | | | | |- targeted_productivity  :  0.7
| | | | | | | | | | | | | | | |- no_of_workers  :  8.0
| | | | | | | | | | | | | | | | |- over_time  :  1440.0
| | | | | | | | | | | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | | |- over_time  :  5100.0
| | | | | | | | | | | | | | | | | |- over_time  :  1440.0
| | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | |- over_time  :  2160.0
| | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | |- targeted_productivity  :  0.75
| | | | | | | | | | | | | | | | |- no_of_workers  :  8.0
| | | | | | | | | | | | | | | | | |- over_time  :  1560.0
| | | | | | | | | | | | | | | | | | |- over_time  :  1440.0
| | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | |- no_of_workers  :  15.0
| | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | | | |- over_time  :  2160.0
| | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | | | | | |- over_time  :  2280.0
| | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | |- no_of_workers  :  8.0
| | | | | | | | | | | | | | | | | |- over_time  :  1440.0
| | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | | |- no_of_workers  :  12.0
| | | | | | | | | | | | | | | | | | |- over_time  :  1440.0
| | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | | | | | |- over_time  :  1800.0
| | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | |- over_time  :  2400.0
| | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | |- no_of_workers  :  16.0
| | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | | | | |- over_time  :  2160.0
| | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | | | | | |- no_of_workers  :  18.0
| | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | |- no_of_workers  :  9.0
| | | | | | | | | | | | | | | |- standard_minute_value  :  4.08
| | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | | |- over_time  :  1620.0
| | | | | | | | | | | | | | | | | |- targeted_productivity  :  0.7
| | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | | | | | | | | | | | |- targeted_productivity  :  0.75
| | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | | | |- standard_minute_value  :  4.15
| | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | | | | | | | | | | | | |- over_time  :  2160.0
| | | | | | | | | | | | | | | | | | | |- standard_minute_value  :  4.6
| | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | | | | | |- over_time  :  3360.0
| | | | | | | | | | | | | | | | | | | | |- targeted_productivity  :  0.7
| | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | | | | | | | | | | | | | | |- targeted_productivity  :  0.75
| | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | |- no_of_workers  :  10.0
| | | | | | | | | | | | | | | | |- standard_minute_value  :  4.6
| | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | |- standard_minute_value  :  4.15
| | | | | | | | | | | | | | | | | |- over_time  :  1440.0
| | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | | | | | | | | | | | | |- no_of_workers  :  12.0
| | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | |- over_time  :  1800.0
| | | | | | | | | | | | | | | | | | | | |- targeted_productivity  :  0.7
| | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | |- over_time  :  6600.0
| | | | | | | | | | | | | | | | | | | | | |- targeted_productivity  :  0.7
| | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | | | | | | | |- over_time  :  2400.0
| | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | |- incentive  :  29.0
| | | | | | | | | | | | | | | | | | |- standard_minute_value  :  5.13
| | | | | | | | | | | | | | | | | | | |- targeted_productivity  :  0.75
| | | | | | | | | | | | | | | | | | | | |- targeted_productivity  :  0.7
| | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | | | | | | |- over_time  :  6000.0
| | | | | | | | | | | | | | | | | | | | | | |- over_time  :  2160.0
| | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | | | | | | |- standard_minute_value  :  4.3
| | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | |- over_time  :  1440.0
| | | | | | | | | | | | | | | | | | | | | | |- standard_minute_value  :  4.6
| | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | | | | | | | |- over_time  :  1800.0
| | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | | | | | | | | | |- over_time  :  2640.0
| | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | | | | | | | | | |- over_time  :  6000.0
| | | | | | | | | | | | | | | | | | | | | | | | | |- standard_minute_value  :  4.6
| | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | | | | | |- over_time  :  7560.0
| | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | | | | |- work_in_progress  :  14.0
| | | | | | | | | | | | | | | | | | | | |- targeted_productivity  :  0.7
| | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | | | | | | | | | | | | | | |- work_in_progress  :  154.0
| | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | | | | | | | | | | | | | | |- over_time  :  3360.0
| | | | | | | | | | | | | | | | | | | | | | |- standard_minute_value  :  21.82
| | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | | | | | | | | |- standard_minute_value  :  18.79
| | | | | | | | | | | | | | | | | | | | | | | |- work_in_progress  :  834.0
| | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | | | |- no_of_workers  :  53.0
| | | | | | | | | | | | | | | | | | | | | | | | |- no_of_style_change  :  1.0
| | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | | | | |- over_time  :  4260.0
| | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | | | | | | | | | | | | | | | | | | |- over_time  :  6240.0
| | | | | | | | | | | | | | | | | | | | | | | | | | |- targeted_productivity  :  0.7
| | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | | | | | | | | | | | | | | | | | | | |- over_time  :  6660.0
| | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | | | | | | | | | | | | | |- standard_minute_value  :  23.69
| | | | | | | | | | | | | | | | | | | | | | | | | | | | |- work_in_progress  :  965.0
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | | | | | | | | | | | | | | | | | | | | | | |- no_of_workers  :  57.0
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- work_in_progress  :  724.0
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- work_in_progress  :  1248.0
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.23 - 0.38
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- work_in_progress  :  419.0
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- work_in_progress  :  481.0
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- work_in_progress  :  541.0
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- work_in_progress  :  557.0
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- work_in_progress  :  627.0
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- work_in_progress  :  658.0
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- standard_minute_value  :  24.26
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- over_time  :  6960.0
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- over_time  :  7080.0
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- work_in_progress  :  679.0
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- standard_minute_value  :  30.1
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity > 0.54 - 0.69
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity-> 0.38 - 0.54
| | | | | | | | | | | | | | | | | | |- incentive  :  56.0
| | | | | | | | | | | | | | | | | | | |- no_of_workers  :  30.0
| | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | |- over_time  :  2040.0
| | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | |- over_time  :  2160.0
| | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | | |- standard_minute_value  :  16.1
| | | | | | | | | | | | | | | | | | | | | | | |- over_time  :  6120.0
| | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | | | | |- work_in_progress  :  539.0
| | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | | | |- no_of_workers  :  32.0
| | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | | | | | | | | | | |- no_of_workers  :  38.0
| | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | | | | | |- work_in_progress  :  247.0
| | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | | | | | | |- work_in_progress  :  282.0
| | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | | | | | | | | | | | | | |- work_in_progress  :  568.0
| | | | | | | | | | | | | | | | | | | | | | | | | | | | |- over_time  :  6840.0
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- standard_minute_value  :  25.9
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- targeted_productivity  :  0.75
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- work_in_progress  :  381.0
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- work_in_progress  :  437.0
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- work_in_progress  :  541.0
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.38 - 0.54
| | | | | | | | | | | | | | | | | | | | | | | | | | | | |- work_in_progress  :  964.0
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- incentive  :  33.0
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- standard_minute_value  :  29.4
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- work_in_progress  :  1587.0
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.54 - 0.69
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- no_of_workers  :  50.0
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- incentive  :  40.0
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- work_in_progress  :  1039.0
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- targeted_productivity  :  0.75
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- incentive  :  50.0
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- work_in_progress  :  1170.0
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | |- incentive  :  69.0
| | | | | | | | | | | | | | | | | | | | |- incentive  :  60.0
| | | | | | | | | | | | | | | | | | | | | |- targeted_productivity  :  0.7
| | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | | | | | | |- incentive  :  63.0
| | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | | |- over_time  :  7140.0
| | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | |- targeted_productivity  :  0.7
| | | | | | | | | | | | | | | | | | | | | |- standard_minute_value  :  22.94
| | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.69 - 0.84
| | | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
| | | | | | | | | | | | | | | | | | | | | |- actual_productivity --> 0.84 - 1.00
```

1. Key Results and Metrics of the algorithms

As a quick recap it was decided that performance of the algorithms will be measured by the values found through the following equations:

First let:

	Classified Positive	**Classified Negative**
Actual Positive	True Positive(TP)	False Negative(FN)
Actual Negative	False Positive (FP)	True Negative(TN)

Accuracy (a)

$$a = \frac{TP + TN}{TP + TN + FP + FN}$$

Recall (r)

$$r = \frac{TP}{TP + FN}$$

Precision (p)

$$p = \frac{TP}{TP + FP}$$

F1-value (f1)

$$F_1 = \frac{2pr}{p + r}$$

K Nearest Neighbours

Metric	Accuracy	Precision	Recall	F1-value
sklearn	0.44	0.45	0.44	0.41
WEKA	0.24	0.25	0.24	0.23
Own Implementation	0.44	0.42	0.44	0.41

Data Summary

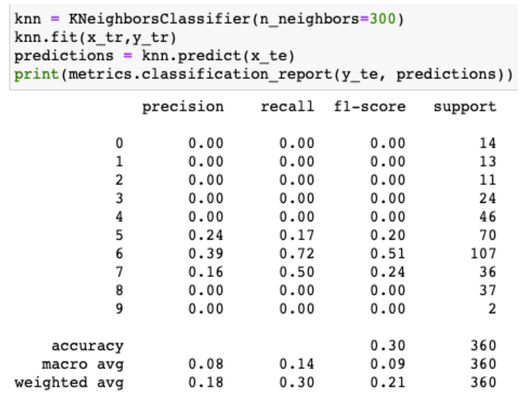
SKlearn and my implementation had similar accuracy, precision and recall score, whereas weka was worse in every metric.
A reason why weka may not have worked as well could be in the preprocessing of the data. For sklearn and my own implementation, the data was preprocessed and split at the start into a training and test set, so both implementations used the same data. However with WEKA I reproduced the steps in preprocessing the data as opposed to using the same data.

Weka's low precision means that the classification of a class is more likely to result in a false positive than sklearn or my own algorithm. The low recall means that there is much more occurrence of false negatives. The combination of precision and recall in the F1 value is much lower than sklearn and my implementation, making it the worst implementation of KNN.

The similarity between sklearn and my own implementation's results is to be expected. This is because sklearn uses euclidean distance (sklearn.neighbors.KNeighborsClassifier, 2021) as does my implementation. Had I used a different distance measure, the evaluation metrics would have been different.

The main point of difference between sklean and my implementation was the time it took to run. To run sklearn at each number of neighbour took ~<0.5 seconds, whereas my implementation too ~80 seconds to run. While KNN is notorious for being slow because there is no model generated and each test data point needs to be compared to the training dataset, my implementation took much longer than it should have. Sklearn has a fit() method which creates data structures that are more organised, making it faster to access, whereas my implementation uses a brute force method that is line by line in the training set. This could be the reason why my implementation is so much slower.

What was surprising about the data was that as the number of neighbours increased within sklearn and my algorithm, the accuracy, precisions, recall and f1 value did not change much. I expected that with higher neighbour numbers the more underfitting and eventual decline in all the evaluation metrics, but instead it stayed roughly the same. I wanted to explore this further so I tested the algorithm with 300 neighbours, which is roughly 40% of all the data, and the results were



```
knn = KNeighborsClassifier(n_neighbors=300)
knn.fit(x_tr,y_tr)
predictions = knn.predict(x_te)
print(metrics.classification_report(y_te, predictions))
```

```
              precision    recall  f1-score   support

           0       0.00      0.00      0.00        14
           1       0.00      0.00      0.00        13
           2       0.00      0.00      0.00        11
           3       0.00      0.00      0.00        24
           4       0.00      0.00      0.00        46
           5       0.24      0.17      0.20        70
           6       0.39      0.72      0.51       107
           7       0.16      0.50      0.24        36
           8       0.00      0.00      0.00        37
           9       0.00      0.00      0.00         2

    accuracy                           0.30       360
   macro avg       0.08      0.14      0.09       360
weighted avg       0.18      0.30      0.21       360
```

The accuracy was still relatively high, but the precision and recall was low. This indicates to that the actual_productivity value is skewed to one particular bin.

Bayes Classification

Metric	Accuracy	Precision	Recall	F1-value
sklearn	0.44	0.49	0.44	0.43
WEKA	0.35	0.371	0.35	0.33
Own Implementation	0.45	0.42	0.45	0.41

Data Summary

Similar to KNN, SKlearn and my implementation yielded similar results, but weka was worse in every metric.

Again, the culprit could be in the way I preprocessed the data, as sklearn and my implementation used the same test and train set but with weka I recreated the preprocessing steps within the Weka explorer.

The similar F1 values with sklearn and my implementation indicates that the recall and precision of both the implementations are similar.

Looking closer, the precision of sklearn is higher and recall is lower. If the cost of acting upon the classification is higher, such as if one was to invest in the factory and expects a certain productivity to yield a higher return on investment, then sklearn would be the better implementation as it's precision is higher.

If the cost of acting upon the classification is low but the opportunity cost, i.e. the cost of inaction, is high, then my implementation is only slightly better. The context where that would be useful would may be if a national body were to conduct an audit on factories with certain productivity, the cost of inaction meaning not auditing a particular factory with malpractice.

The accuracy is similar between my implementation and sklearns. This is to be expected too, as multinomial naive bayes follow the same formula whether it's in my implementation or sklearns.

While the time to build the model was similar between my implementation and sklearn, the time to run the model on training data was longer for my implementation at ~6 seconds whereas for sklearn it took consistently <0.1 seconds. To test the model, I sequentially go through each datapoint in the test set, which is inefficient use of the data structures of pandas (pandas speeds things up with methods such as vectorisation), whereas I am sure that sklearn makes use of different techniques to speed up algorithms.

Support Vector Machine

Metric	Accuracy	Precision	Recall	F1-value
Implementation 1	0.46	0.31	0.46	0.37
Implementation 2	0.5	0.45	0.51	0.45
Own Implementation				

Data Summary:

The data for my own implementation couldnt be shown because the algorithm was not in a working state at the time of finish.

One of the biggest differences between the Weka implementation and the sklearn implementation is the precision. This represents how precise/accurate the model is out of the predicted positive. This difference could be due to an overfitting of the data.

Decision Tree

Metric	Accuracy	Precision	Recall	F1-value
Implementation 1	0.565	0.286	0.320	0.267
Implementation 2	0.760	0.404	0.400	0.377
Own Implementation	0.784	0.406	0.460	0.430

Data Summary

Upon observing the results gathered from each implementation it would appear that the self made implementation performed the best in every regard. Weka came in last and sklearn came second, in fact oddly enough this hierarchy was preserved across all measured metrics. Now although this news may at first seem very positive, there are two very important facts which must be acknowledged as it is possible they are artificially skewing results.

Firstly due to the pre-processing method it must be remembered that several of the datasets attributes were discarded due to the incompatibility with nominal attributes that two of the implementations sported.

Secondly, as there was no max depth control being used to cut off the tree for the second and third implementations, it is very possible that both of these models have become massively overfit. This idea is sadly further backed up by the fact that tree size follows the same hierarchy as the recorded performance of the implementations (E.g. the trees have greatly increased in size with each implementation to the point where the tree produced by the last implementation spans three triple columned pages).

2. Overall Result Summary

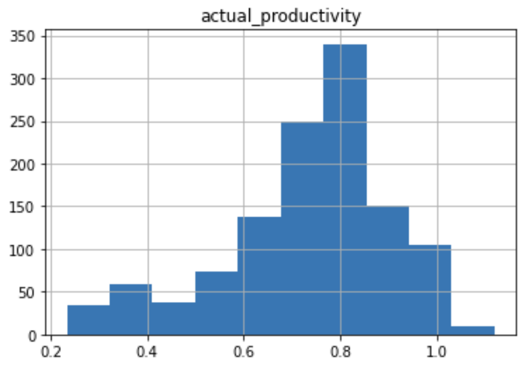
Accuracy

The most accurate algorithm to predict the class is the decision tree, which is to be expected as decision trees weigh each variable differently. Decision trees utilise information gain, meaning that exploring certain variables may yield a higher chance of classifying, allowing the algorithm to disregard unimportant variables and even classifying without considering all the available variables.
K nearest neighbours, support vector machines and multinomial naive bayes classification weigh each variable equally, allowing the noise of unimportant variables to affect the ultimate classification. An information gain component could be added to enhance the other algorithms.

F-1 value

The optimum F1-value is similar across all algorithms, meaning they all have similar precision and recall. In events where precision and recall supersede the importance of accuracy, choosing any of the above classification models would suffice. This was a surprise as it was expected to have some variance amongst the different implementations, as was with the accuracy, however the consistent number may indicate that the data may be skewed to a particular classification.

Distribution of classes



Upon further investigation, it was the case that the data was skewed to one bin, with roughly 28% of all values falling into one bit at the 0.8 productivity. If one were to consistently choose that one bin, for each test case, the resultant accuracy would be ~30%. This positive news, as it showcases that our algorithms perform better than random chance

3. <u>References / Bibliography</u>

Al Imran, A., 2021. *UCI Machine Learning Repository: Productivity Prediction of Garment Employees Data Set*. [online] Archive.ics.uci.edu.
Available at:
<https://archive.ics.uci.edu/ml/datasets/Productivity+Prediction+of+Garment+Employees>
[Accessed 2 September 2021].

Archive.ics.uci.edu. 2021. *UCI Machine Learning Repository*. [online]
Available at:
<https://archive.ics.uci.edu/ml/index.php>
[Accessed 2 September 2021].

scikit-learn. 2021. sklearn.neighbors.KNeighborsClassifier. [online]
Available at:
<https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html>
[Accessed 10 October 2021].

Appendix 1: self - contributions

Bennett
I implemented the Support Vector machine and discussed its findings.

Sebastian
I wrote the introduction for the final report, in addition to being the creator of all implementations and investigations on the decision trees.

Josh
I implemented the KNN and Naive Bayes algorithms and wrote about the implementation, findings and data summary for those algorithms. I also wrote the overall result summary