SIT307 Machine Learning
Garment Industry Employee Productivity

CHANPUTHI TITH
219498222
ctith@deakin.edu.au

I.) Background

Garment Industry is significantly boosting the nation's economy and provide enormous jobs to the citizen. This industry highly requires skill and experience labour worker. Additionally, it is important to ensure the employee have high productivity and love their work.

In this report, we will implement three supervised machine learning models to make a prediction on actual productivity of the employee. First of all, we will do pre-processing and split the dataset into training set and testing set. After that, we will optimise the hyper-parameters for each machine learning model and make a recommendation based on the performance of the machine learning model. Finally, we will analyse the important of the features for predicting actual productivity of employee with statistical reasons.

II.) Explore Dataset

In this section, we will explore the "Garment Industry Employee Productivity" dataset by introducing the dataset's details, explaining how to deal with missing values, and demonstrating the splitting data technique.

i.) Dataset Details

The "Garment Industry Employee Productivity" dataset consists of 1197 rows (or instances), and 15 columns including 14 features and 1 target variable. The information of this dataset is shown below:

Column Number	Column Name	Data Type
0	date	object
1	quarter	object
2	department	object
3	day	object
4	team	integer
5	targeted productivity	float
6	smv	float

7	wip	float
8	over time	integer
9	incentive	integer
10	idle time	float
11	idle men	integer
12	number of style change	integer
13	number of workers	float
14	actual productivity	float

Note:
smv: standard minute value
wip: work in progress

ii.) Dealing with Missing Values

During the pre-processing stage, we have identified that “wip” (or Work in Progress) attribute has 506 missing values.

To deal with this, we have plotted the histogram for “wip” to decides whether to apply mean or median to the missing data.

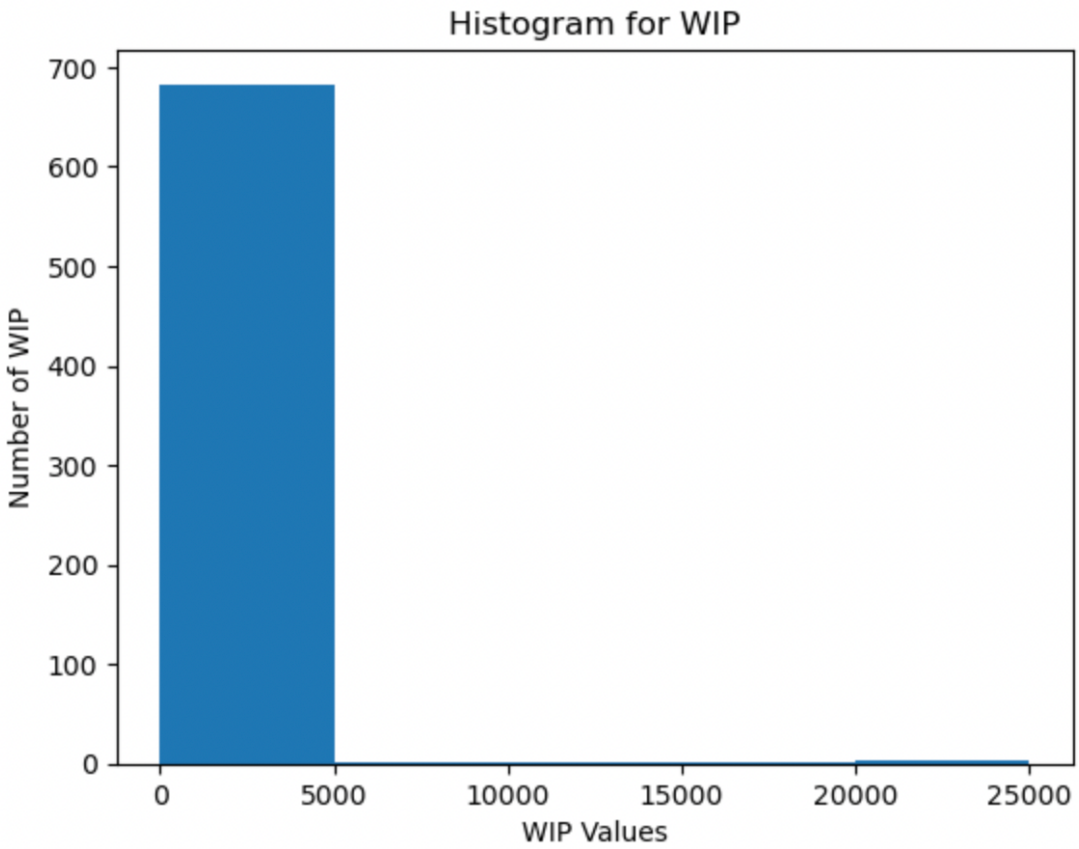

Figure 1: Right-Skewed WIP Histogram

As shown from the visualization above, this histogram is right skewed between 0 and 5000 wip values. Therefore, we will apply the median value of the wip attribute to each of its missing data with the value of 1039.

After dealing with the missing value of wip with its median value, there are no more missing value within this dataset.

Note:
The minimum value of wip is 7.
The maximum value of wip is 23122.
The mean value of wip is 1190.465.
The median value of wip is 1039.

iii.) Data Encoding

In the dataset, there are four attributes with data type as objects. Those four attributes are date, quarter, department, and day. However, most machine learning model can only be fit with numerical data. Therefore, we will encode these categorical variables into numerical variables.

Among these four features, date is not relevant to fit with the machine learning model because each instance (or row) has different date. With a total 1197 instances, there are at least 100 different dates. Thus, this attribute is less likely to have any impact on the performance of the machine learning model.

After deciding which features to encode, it is now time to pick a categorical data encoding technique. Among different categorical data encoding technique, one hot encoding and label encoding are two of the most popular and relevant techniques for these three attributes.

- One Hot Encoding: encode the categorical (and nominal) feature into a binary variable with a value of either 0 (absence) or 1 (presence).
- Label Encoding: encode the categorical (and ordinal) feature into a numerical variable with integer value which represent each categorical data.

Quarter and day attributes are ordinal features while department could be either nominal or ordinal (example, department id). Thus, label encoding is the most suitable categorical data encoding technique to encode these three categorical variables into numerical variable.

Note:
In the Python code, we have encoded the categorical variables' name quarter, department, and day, into the numerical variables' name quarter_int, department_int, and day_int.

iv.) Splitting Dataset

In this last pre-processing stage, we will split the dataset into training data and testing data.

With a large amount of data (1197 instances), random splitting is the most suitable method to split the dataset. Although, cross-validation method is the best one but it's computationally expensive for this large dataset.

After randomly splitting the dataset with 70 percentage of training size and 30 percentage of testing size, we have 837 instances for training data and 360 instances for testing data.

Note:
Training Shape: (837, 13)
Testing Shape: (360, 13)

III.) Supervised Machine Learning Models for Prediction

In this section, we will choose three suitable machine learning models to make a prediction on employee actual productivity within this dataset. Then, we will illustrate the performance score of each machine learning algorithms based on different performance metrics. In addition, we will discuss each chosen supervised machine learning models and reason to apply it for this dataset. Moreover, we will demonstrate the improvement of machine learning model by applying hyper-parameter optimization. Finally, we will make a recommendation on the best performance machine learning model for this study.

i.) Prediction Process

There are many machine learning algorithms which have different objectives and goals.

For the purpose of predicting actual productivity of Garment Industry's employees, we will use three supervised machine learning models including linear regression, support vector machine regression, and decision tree regression.

First of all, we will create three variables represent each machine learning model. Then, we will fit the training data X and the training data y into each variable. Finally, we will make predictions on the testing data X. (One of) The coding example is provided below.

```
linear_regressor = LinearRegression()
linear_regressor.fit(X_train, y_train)
lr_y_preds = linear_regressor.predict(X_test)
```

ii.) Performance Score

To understand the performance of these machine learning models, it is significant to evaluating and reporting it with the performance metrics. There are five performance metrics which can report the performance of these regression models including mean absolute error, mean squared error, root mean squared error, r squared, and explained variance.

To make it simple and efficient, I have created a function “regression_performance(y_data, y_pred)” to show the results of each performance metrics with a single line of code. This function takes y_data and y_pred as input. The code of this function is shown below.

```
def regression_performance(y_data, y_pred):
    mae = mean_absolute_error(y_data, y_pred)
    mse = mean_squared_error(y_data, y_pred)
    rmse = mean_squared_error(y_data, y_pred, squared=False)
    r2 = r2_score(y_data, y_pred)
    ev = explained_variance_score(y_data, y_pred)

    print(f'MAE: {mae:.2f}')
    print(f'MSE: {mse:.2f}')
    print(f'RMSE: {rmse:.2f}')
    print(f'R2: {r2:.2f}')
    print(f'EV: {ev:.2f}')
```

After that, we will use this function to show the performance metrics of each regression model as shown in code below.

Linear Regression model

```
regression_performance(y_test, lr_y_preds)
```

Support Vector Machine Regression model

```
regression_performance(y_test, svr_y_preds)
```

Decision Tree Regression model

```
regression_performance(y_test, dtr_y_preds)
```

The results of each performance metrics for each machine learning models on the testing data are shown in the table below.

Model/Metric	MAE	MSE	RMSE	R2	EV
Linear Regression	0.11	0.02	0.15	0.21	0.21
Support Vector Machine Regression	0.13	0.03	0.17	0.05	0.05
Decision Tree Regression	0.09	0.02	0.15	0.21	0.21

Table 1: Performance of ML models on Testing data

As shown from the result table above, each machine learning models have performed well on most performance metrics with less error.

To understand whether these results have high bias (underfitting) or high variance (overfitting), we need to apply these performance metrics on training data as well.

To apply these performance metrics on training data, we use the same code as shown above on the testing data by changing the input to the function from y_test to y_train, and lr_y_preds to lr_y_train_preds. (Therefore, we don't need to explain the code again here.)

The results of each performance metrics for each machine learning models on the training data are shown in the table below.

Model/Metric	MAE	MSE	RMSE	R2	EV
Linear Regression	0.11	0.02	0.15	0.31	0.31
Support Vector Machine Regression	0.13	0.03	0.17	0.06	0.06
Decision Tree Regression	0.00	0.00	0.00	1.00	1.00

Table 2: Performance of ML models on Training data

By comparing the results from both tables, we have identified that mean absolute error, mean squared error, and root mean squared error show the same (or similar) results on both training data and testing data for linear regression and support vector machine regression models. Thus, this means that both machine learning models have fit training and testing set pretty well (or generalization).

In contrast, the performance metrics for decision tree regression has 0 score (the lowest possible) on training data, and higher than 0 score on testing data. This indicate that this decision tree regression model could be bias (or underfitting). However, the performance score of this model on testing data are also low as well, therefore, we cannot make a conclusion yet.

iii.) Design Decisions

In this study, we applied three supervised machine learning models to make prediction on the actual productivity of Garment Industry's employees. These three regression models are linear regression, support vector (machine) regressor, and decision tree regressor.

Linear regression is the most commonly used supervised machine learning algorithm for prediction. It is a linear model which describe the relationship between the predictor variables X (such as day, team, wip) and the target variable y (actual productivity).

Support vector (machine) regressor is a non-linear model that capture more complex relationship between the (X) predictor variables and the (y) target variable y. It uses similar principle as support vector machine by finding the best fit line which is the hyperplane that maximally separates the data points.

Decision tree regressor is a tree-based model that capture the relationship between the (X) predictor variables and the (y) target variable by recursively partitioning the data into subsets based on the values of the predictor variables. As each subset will have its own prediction, this regressor will make overall predictions based on the mean value of those subset.

To conclude, these three supervised machine learning models have different capabilities and limitations to make a prediction. It is important to implement these algorithms and decide based on its performance.

Nonetheless, there are also a few machine learning models that could make better predictions on this dataset as well such as random forest regressor (ensemble learning) and neural network. However, these algorithms will not be covered in this study.

iv.) Hyper-parameters Optimization

To receive a better performance score, it is significant to choose the right (or the best) hyper-parameters for each model. This could be done by using "GridSeachCV" from model selection function from scikit learn library. The following content will explain each parameter for each machine learning model.

- Linear Regression with optimised hyper-parameters

```
param_grid = {'fit_intercept': [True, False], 'copy_X': [True, False], 'positive': [True, False], 'n_jobs': [-1, 1]}
```

For linear regression model, there are four important parameters including “fit intercept”, “copy X”, “positive”, and “n jobs”.

Based on the implementation, the best parameters for linear regression in this study is:

```
Best Parameters: {'copy_X': True, 'fit_intercept': False, 'n_jobs': -1, 'positive': False}
```

- Support Vector Machine Regressor with optimised hyper-parameters

```
param_grid = {'kernel': ['poly', 'rbf']}
```

For support vector (machine) regressor model, there are three important parameters including “kernel”, “C”, and “epsilon”. However, we are unable to run these parameters due to computational speed because SVM with Linear kernel is time-consuming on this non-linear data and our dataset is larger and complexity as well.

Based on the implementation, the best parameters for support vector machine regressor in this study is:

```
Best Parameters: {'kernel': 'poly'}
```

- Decision Tree Regressor with optimised hyper-parameters

```
param_grid = {
    'max_depth': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
    'min_samples_split': [2, 4, 6, 8],
    'min_samples_leaf': [1, 2, 3],
}
```

For decision tree regressor model, there are many parameters. However, to make it simple, we only used three parameters including max depth, min sample split, and min sample leaf while max depth is the most important one.

Based on the implementation, the best parameters for decision tree regressor in this study is:

```
Best parameters: {'max_depth': 7, 'min_samples_leaf': 3, 'min_samples_split': 8}
```

After optimised the hyper-parameters for each machine learning model, we have applied the best parameters from each model to make predictions again.

The following table show the performance comparison between non-optimised hyper-parameter and optimised hyper-parameter for each machine learning model. To make it simple, we will compare it based on two performance metrics including mean average error and mean squared error.

Model/Metric	MAE		MSE	
	Non-Optimised	Optimised	Non-Optimised	Optimised
Linear Regression	0.11	0.11	0.02	0.02
Support Vector Machine Regression	0.13	0.12	0.03	0.03
Decision Tree Regression	0.09	0.09	0.02	0.02

Table 3: Performance Comparison of ML models Optimisation on Testing data

Model/Metric	MAE		MSE	
	Non-Optimised	Optimised	Non-Optimised	Optimised
Linear Regression	0.11	0.10	0.02	0.02
Support Vector Machine Regression	0.13	0.13	0.03	0.03
Decision Tree Regression	0.00	0.07	0.00	0.01

Table 4: Performance Comparison of ML models Optimisation on Training data

Based on the result from both table 3 and table 4, it illustrated that the performance score of each machine learning model is slightly change between “before optimisation” and “after optimisation”. It does not change much because these regression model have better performance score already.

As shown from the table above, mean square error for linear regression, support vector regressor, and decision tree regressor have only 0.02, 0.03, and 0.02 respectively. These are some of the best possible performance score for these algorithms on testing data.

Additionally, mean square error for linear regression, support vector machine regressor, and decision tree regressor on training data also show the best possible score as well with 0.02, 0.03, and 0.01. Although, the performance score for decision tree algorithm is increasing, it might not have a high impact on the prediction.

Thus, the supervised machine learning models that we have trained have made a great prediction with or without optimisation.

v.) Recommendation

Based on the result table, decision tree regressor algorithm has the best performance score for both mean squared error and mean average error on both training data and testing data. However, this model might be overfitting (high variance) because the training data has better performance than the testing data. As both training and testing data has lowest mean squared error and lowest mean average error, it could be alright to use this model with further optimization in further study.

Therefore, it is recommended to apply decision tree algorithm to make a prediction on actual productivity of Garment Industry's employee.

IV.) Important of the features

It is essential to know some of the most feature which have a high impact on the prediction of actual productivity. In this study, we used two different approach including linear regression feature coefficients and decision tree regressor feature importances. The following figure visualize some of the most importance feature for both approaches.

Linear Regression approach with Coefficient

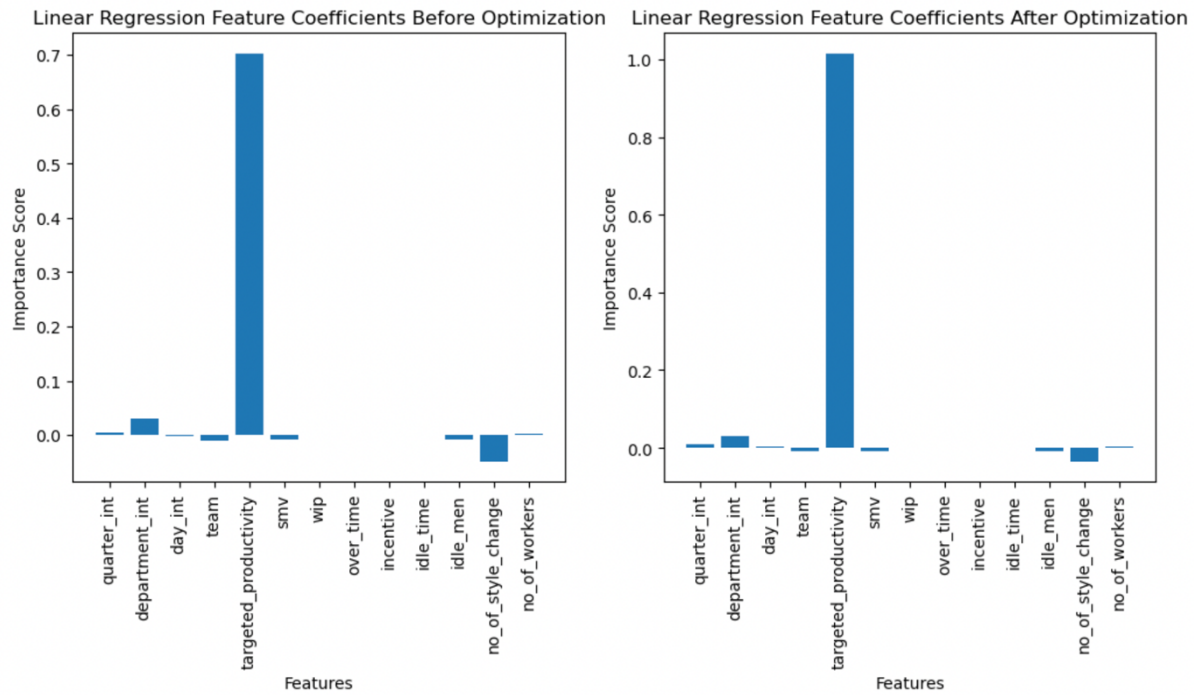

Figure 2: Important Features with Linear Regression approach

Decision Tree Regressor approach with Feature Importance

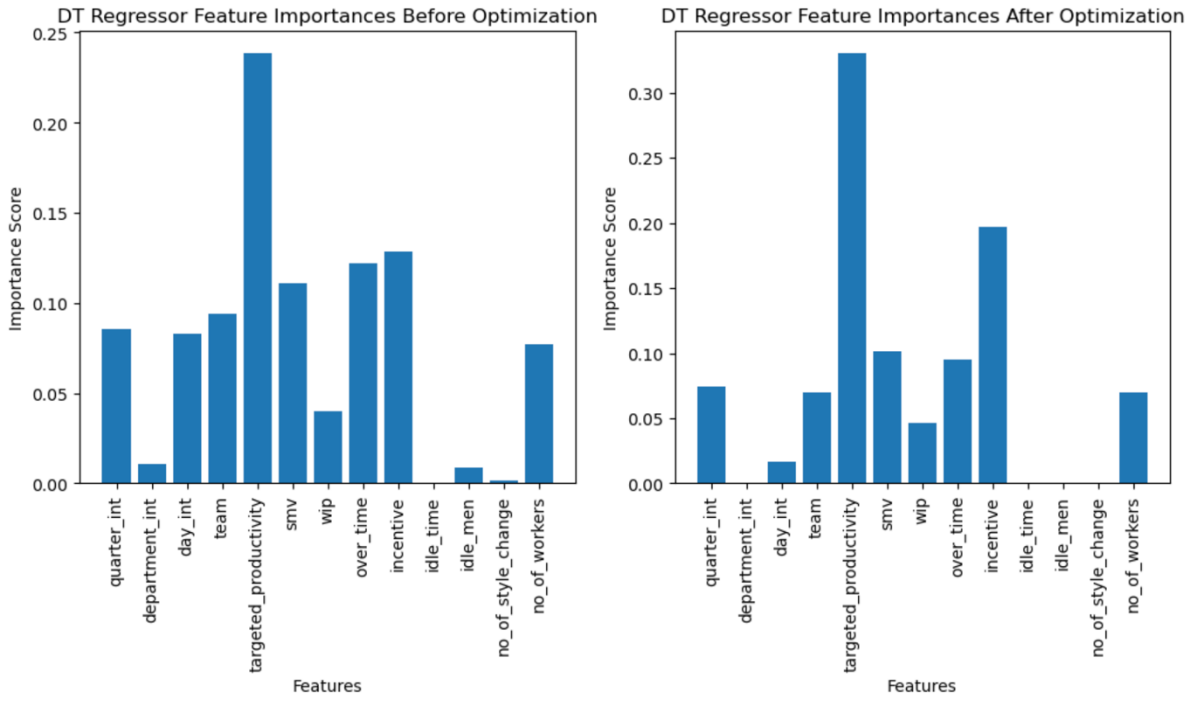

Figure 3: Importance Feature with Decision Tree Regressor Approach

As shown from figure 2 and figure 3 above, both approaches show different importance score before and after optimization and have different importance features as well.

The tables below show the top 3 importance feature from each approach.

	Top 3 Importance Features				
Linear Regression	Targeted Productivity	Number of Style Change	Department		
Decision Tree Regressor	Targeted Productivity			Incentive	Standard Minute Value (SMV)

Table 5: Top 5 Importance Feature

V.) Conclusion

In conclusion, this study has found that targeted productivity is the most important feature to predict actual productivity of employee from Garment Industry. It followed by number of style change, department, incentive, and standard minute values.

The best model among the three implemented model is decision tree regressor model with the mean squared error of 0.0.1 followed by linear regression model (0.02) and support vector (machine) regressor model (0.03).