Analysis of Municipal Waste Data-set

Predicting cost per capita

I. Sukhija

This research paper presents a study on the use of linear regression models to predict the relationship between the dependent variable (cost per capita) and the independent variables (other columns) of the Municipal Waste management dataset from Kaggle.[1] The objective of the research was to analyze the dataset and make prediction model and optimize it. This paper also includes the discussion of the various data pre-processing techniques like data cleaning, outlier analysis, log transformation, standardization and feature selection. The prediction models used are Linear Regression Model, Random Forest Regression Mode and the Histogram Gradient Boosting Regression Model, which have been discussed in the detail to make it easy to understand. The metrics use to check the prediction quality of these models are R2- score, MSE and RMSE. This study demonstrates the value of linear regression models in predictive modeling and provides insights into the best practices for using these models in real-world scenarios.

I. Dataset Inspection

The cost per capita of municipal waste management was predicted using the Municipal Waste Management dataset[1] from Kaggle. This waste is collected by the municipalities. The waste includes waste from houses, businesses and industry, office buildings, institutions, as well as yard and garden waste, street sweeping, litter containers, and markets, waste from municipal sewage networks and treatment, as well as waste from construction and demolition activities, are not included in the definition. The waste is expressed as kilograms per capita and thousand tonnes.

The dataset is named as "public_data_waste_fee.csv". To read the contents of the file in Python we'll be importing pandas library and then using the same for csv (comma-separated-values)file reading, the following code was used.

```
import pandas as pd

data = pd.read_csv("public_data_waste_fee.csv")
data.info()
```

The dataset contains 39 columns which are of either datatype 'object', 'float64' or 'int64'. The number of records in the dataset are 4341. The dataset includes null values for various columns represented by empty space for that particular entry space. The dataset contains both categorical and numerical type of data. The categorical data includes the strings (e.g. in name column) and has datatype = 'object'. The numerical data is further of two types in the dataset, continuous (e.g. in cres column) and discrete (e.g. in geo column). There are four categorical data columns and 35 numerical data columns in total.

Table 1: Overview of Dataset

S.No	Column Name	Non-Null Values	Datatype
1	region	4341	object
2	province	4341	object
3	name	4335	object
4	tc	4341	float64
5	cres	4289	float64
6	csor	4274	float64
7	istat	4341	int64
8	area	4335	float64
9	pop	4341	int64
10	alt	4335	float64
11	isle	4335	float64
12	sea	4335	float64
13	pden	4335	float64
14	wden	4335	float64
15	urb	4335	float64
16	fee	4341	object
17	d_fee	4341	int64
18	sample	4341	int64
19	organic	3829	float64
20	paper	4316	float64
21	glass	4308	float64
22	wood	3246	float64
23	metal	4095	float64
24	plastic	4302	float64
25	raee	4027	float64
26	texile	3328	float64
27	other	4205	float64
28	msw_so	4341	float64
29	msw_un	4341	int64
30	msw	4341	int64
31	sor	4341	float64
32	geo	4056	float64
33	roads	3898	float64
34	s_wteregio	4056	float64
35	s_landfill	4056	float64
36	gdp	3955	float64
37	proads	3898	float64
38	wage	4056	float64
39	finance	3955	float64

II. Method

II.A. Data Pre-processing

As real world data are tend to be incomplete, noisy and inconsistent, pre-processing is important to clean the data to be used for prediction. As the data contained some missing values and some outliers which were important to handle, data pre-processing has to be done to improve the prediction quality.

Data cleaning: This technique is applied to identify the missing values, fill in the missing values, remove noise and for correcting the inconsistency in the data. In the dataset provided, the missing values for numerical data was replaced by the median of the respective columns. For missing values in categorical data, the records were dropped. The code for the data cleaning is shown below.

```
data=data.fillna(data.median(numeric_only=True))
data.drop('name',axis=1,inplace=True)
data.dropna(inplace=True)
```

The name column was dropped as shown in the code above and was not considered for the prediction of cost per capita as the name consisted of very large number of unique categorical values, which could lead to over-fitting, where model fits to the training data too closely and fails to generalize well to new data, thus having a poor prediction capacity. The target variable ('tc') is separated from the dataset as Y and X as the remaining dataset. The data columns in X are then further divided as numerical and categorical columns (num_cols and cat_cols respectively). The numerical columns are then further divided as discrete and continuous columns (dis_cols and con_cols respectively). The criteria for choosing a discrete column was that if the data column has less than 10 unique values. The code for the above stated implementation is shown below.

```
# Separating dependent variables from target variables
target = 'tc'
X = data.drop(target,axis=1)
Y = data[target]

# Separating type of attributes in X (Numerical (Continuous and Discrete) & Categorical)
num_cols = X.select_dtypes(exclude='object').columns
cat_cols = X.select_dtypes(include='object').columns
dis_cols = [col for col in num_cols if X[col].unique().sum()<10]
con_cols = [col for col in num_cols if col not in dis_cols]
```

Data Transformation: After seeing the distribution curves of the continuous numerical data columns, it was analysed out that the data was highly skewed. To reduce the skewness (asymmetric nature) of the data, log transformation is used. Log transformation helps in normalizing the data distribution, making it more symmetric and reducing the impact of the outliers, which leads to improvement in the precision of the prediction. The following code is used

to apply log transformation to the data.

```
for col in num_cols:
    if 0 in X[col].unique():
        X[col] = np.log1p(X[col])
    else:
        X[col] = np.log(abs(X[col]))
```

Outlier Analysis: are data points that differ significantly from the other data points in a given dataset. The causes of outliers can be measurement errors, data entry errors, or even legitimate but extreme observations. The decision to remove outliers from a dataset should be made with caution because outliers can potentially carry important information and removing them may alter the overall distribution and characteristics of the data as there are chances that the outliers may be the result of legitimate data points that are outside the norm, and removing them may result in wrong results. Removal of outliers can also lead to over-fitting of data, resulting in poor general performance of the prediction models. For the given dataset, the data points with the distance more than that of 2.7 times the standard deviation from the mean are considered as outliers. The outliers are changed to null values (using np.nan), and then these records containing null values are dropped. The following is the code implementation of the stated procedure.

```
index = list(X.index)
for col in con_cols:
    mean = X[col].mean()
    std = X[col].std()
    lb = mean-2.7*std
    ub = mean+2.7*std
    for j in index:
        if X[col][j]>ub or X[col][j]<lb:
            X[col][j] = np.nan
X.dropna(inplace=True)
Y = Y.filter(items=X.index)
```

Variable Selection: It is an essential machine learning process, as it can enhance model performance, lessen over-fitting, and make the model easier to understand. For the dataset, filter method of variable selection is used. This method involves checking the correlation of the columns with the target variable (in this case only numeric columns are used) and then dropping the columns which are almost uncorrelated to the target variable i.e. having correlation value almost equal to 0. The correlation value tells how the two variables are related to each other. Positive correlation value between attributes A and B means the values of A increases as the values of B increases or vice versa. Similarly, negative correlation value between attributes A and B means the values of A increases as the value of B decreases or vice versa. If the correlation value of the attributes A and B is 0, it means that A and B have no correlation between them (maybe independent). Higher the magnitude, the stronger is the correlation. The threshold value used in the code is 0.05. All the data columns with the correlation magnitude less that the threshold are dropped from the data for target prediction. The columns that were dropped included 'pop' , 'wden', 'urb' and 'plastic' . All these correlation values are with respect to the target variable 'tc'. The following code block shows the variable selection process.

```
# Finding correlation of num_cols with target
↪  column
for col in num_cols:
    print(f"Correlation of {col} with tc is:
    ↪  {np.round(X[col].corr(Y),4)}")


# As the data columns wden, pop, alt and plastic
↪  have correlation value (<0.05)
# Therefore, dropping these X columns.
drop = ['alt','pden','wden','plastic']
X = X.drop(columns=drop,axis=1)
num_cols =
↪  X.select_dtypes(exclude='object').columns
```

The correlation values from the above code have been tabulated below.

Table 2: Correlation of Data Columns with 'tc' column

S.No	Column Name	Correlation Value
1	cres	0.5554
2	csor	0.364
3	istat	0.1089
4	area	0.2044
5	pop	0.0333
6	alt	-0.0942
7	isle	0.1677
8	sea	0.3796
9	pden	-0.1395
10	wden	-0.0327
11	urb	0.0366
12	d_{fee}	-0.1255
13	sample	-0.2769
14	organic	-0.2263
15	paper	-0.1473
16	glass	-0.1431
17	wood	-0.1475
18	metal	-0.159
19	plastic	-0.0395
20	raee	-0.1165
21	textile	-0.1165
22	other	-0.1834
23	msw_so	0.0838
24	msw_un	0.2399
25	msw	0.1414
26	sor	-0.2484
27	geo	-0.1932
28	roads	0.1516
29	s_wteregio	-0.2607
30	s_landfill	0.2406
31	gdp	0.417
32	proads	-0.1165
33	wage	-0.1162
34	finance	0.417

Data Standardization: also known as Z-score normalization, is a technique to transform the features of data to have zero mean and one standard deviation. This ensures that each feature has equal importance in the model, as the features with very large variance or scale might have dis-proportionally large influence on the model, which can result unbiased or incorrect predictions. The formula for the standardization is:

$$z = \frac{(x - \mu)}{\sigma} \tag{1}$$

where, x is the original feature, μ is the mean of the feature, σ is the standard deviation of the feature, and z is the standardized feature.The following can be achieved directly by the StandardScaler class of sklearn.preprocessing module in Python. The standardization has to be done on the numerical data columns of the dataset other than the target column. The code implementation of the procedure is given below.

```
# Standardizing the numerical data
scaler = StandardScaler()
scaler.fit_transform(X[num_cols])
```

Encoding: It is the process of converting the categorical data into numerical values that can be used for various machine learning algorithms. There are many ways to encode categorical values. The mostly used are One-Hot Encoding, Label Encoding, Ordinal Encoding etc. The choice of the encoding method depends on the nature of the data and the requirement of the machine learning algorithm. For the given dataset, target encoder is used. It labels the categorical values according to the mean value of the group of the categories. The code implementation of the TargetEncoder is given below.

```
# TargetEncoder for categorical data values
for col in cat_cols:
    labels_ordered =
    ↪  data.groupby([col])['tc'].mean()
    ↪  .sort_values().index
    labels_ordered={k:i for i,k in
    ↪  enumerate(labels_ordered,0)}
    X[col]=X[col].map(labels_ordered)
```

II.B. Prediction

Regression Models with the pre-processed data as inputs can be used to predict the cost per capita for the municipal waste dataset.

Regression analysis is a statistical method used to identify and model the relationship between the independent variable(s) and a dependent variable. The goal of the regression analysis is to find the parameters of the model based on the values of the independent variables that best fit the data. It is a supervised learning technique. There are two phases in regression analysis: The training phase, where we estimate the parameters of the chosen regression

model and the testing phase, where we test the model against some unseen values and evaluate the model. The training phase aims to make the model good for general cases, i.e., the ability to make precise predictions on new, unseen data. In this paper, three of the mostly used regression models will be discussed in the later portion.

Linear regression: In statistics, linear regression is a linear approach used for modelling the relationship between a scalar response and one or more predictor variables. The scalar response is the dependent variable, and the predictor variable is the independent variable. The relationships in the linear regression models are modelled using linear predictor functions whose model parameters are estimated from the data. The relationship is modelled through a disturbance term or error variable $\boldsymbol{\epsilon}$ - an unobserved random variable that adds noise to the linear relationship between the dependent variable and the predictor variables. The model takes the form:

$$\boldsymbol{y_i = \beta_0 + \beta_1.x_{i1} + ... + \beta_p.x_{ip} + \epsilon_i} \qquad (2)$$

where $\{\boldsymbol{y_i, x_{i1}, ..., x_{ip}}\}$ is a given dataset of n statistical units for i=1,...,n. $\boldsymbol{\beta}$ is a (p+1) dimensional parameter vector, where $\boldsymbol{\beta_0}$ is the intercept term. its elements are known as regression coefficients. $\boldsymbol{\epsilon}$ is a vector of values $\boldsymbol{\epsilon_i}$. This part of the model is called the error term or noise. This variable captures all other factors influencing the dependent variable y other than the regressors $\boldsymbol{x}$.

Random Forest Regression:[?] is a supervised learning regression method that uses an ensemble learning algorithm that combines multiple decision trees to make predictions as a result, it makes a more accurate prediction than a single model. It can handle many independent variables and even complex non-linear relationships. An example of an applied ensemble model is bootstrapping. Bootstrapping is the process of randomly sampling subsets of a dataset over a given number of iterations and a given number of variables. In bootstrapping random forest algorithms, multiple decision trees are randomly drawn, and averaging the results to output a new result often leads to strong prediction. The expectation for ensemble learning is that the errors of each decision tree are distinct and independent from one another. The structure of a Random Forest consists of multiple decision trees that run in parallel with no interaction among them. The algorithm builds a decision tree associated with a portion of the data points it randomly chooses during training. The final forecast is created by averaging the predicted values from all the trees after repeating this process for a predetermined number of trees.
The Random Forest method has the advantage of handling non-linear correlations between features, which makes it applicable to a wide range of issues. However, the model has significant drawbacks, such as a lack of interpretability, which makes it difficult to comprehend how the model functions from the inside out. Another frequent problem with Random Forest is over-fitting, which must be prevented by carefully tuning the parameter values. In general, Random Forest is a strong and adaptable method that can produce high accuracy on a variety of data sets.

Histogram Gradient Boosting Regression[?] is a regression method that uses a combination of gradient-boosting algorithms and histograms to predict continuous variables. The histograms are used to discretize the continuous input variables into bins, and then these binned variables are used to build decision trees. Until the final leaf nodes are reached, each tree is constructed by recursively dividing the data into progressively smaller subsets based on the values of the input variables. Each tree's output is the prediction for a specific instance. The HGBR method employs a gradient-boosting strategy to create an ensemble of decision trees. In gradient boosting, a loss function is created to quantify the discrepancy between the dependent variable's true values and its predicted values. By including decision trees into the model iteratively, the objective is to reduce this loss function. Because of the use of histograms, HBGR is faster and more accurate than some other traditional gradient boosting algorithms as it reduces the number of operations for each split and the variance in the estimates of gradients.

II.C. Error Analysis

An essential component of assessing a regression model's performance is error analysis.[?] It includes evaluating discrepancies between the dependent variable's actual values and those predicted by the regression model. For evaluating a regression model, two metrics are considered: $\boldsymbol{R^2}$ score and Mean Square Error (MSE).

$\boldsymbol{R^2}$, or the coefficient of determination, is a statistical indicator that shows how much of the variance in the dependent variable can be accounted for by the model's independent variables. It is frequently used to evaluate a regression model. The value of $\boldsymbol{R^2}$ score ranges from 0 to 1, 0 indicating that the model explains none of the variability in the dependent variable, while the value of 1 indicates that the model explains all the variability in the dependent variable. The formula for calculating the $\boldsymbol{R^2}$ value for a model is:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \quad (3)$$

where $\hat{y}_i$ is the ith predicted value by the model, y_i is the ith target value and $\bar{y}$ is the mean of the y attribute

MSE is an error metric used to evaluate the performance of a regression model by measuring the average squared difference between the predicted and actual values of the dependent variable. MSE is calculated as:

$$MSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n} \quad (4)$$

where n is the no. of data points, y_i is the actual dependent variable for the ith- data point and $\hat{y}_i$ is the predicted value of the dependent variable for the ith data point. MSE is not the best metric to use in cases like large ranges as the MSE value may be large even for small errors; thus it becomes difficult to interpret the results by using MSE in such cases.

RMSE is another error metric used to determine the precision and error rate of a regression model. It measures the average of the squared differences between the actual and predicted values of the dependent variable. RMSE is the square root of the Mean Square Error (MSE). The formula for calculating the RMSE value is:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y})^2}{n}} \quad (5)$$

where n is the no. of data points, y_i is the actual dependent variable for the ith- data point and $\hat{y}_i$ is the predicted value of the dependent variable for the ith data point. Since RMSE is expressed in the same units as the dependent variable, it is simpler to interpret.

Thus, R^2 evaluates the prediction of the model, and RMSE provides the average deviation of the predicted values from the actual values made by the model.

III. Results and Inferences

The following were the error metrics obtained for the three models built for the prediction of the cost per capita for the Municipal waste dataset.

Table 3: Analysis of Regression Models

Error Metric Model Name	Linear Regressor	Random Forest Regressor	Histogram Gradient Boosting Regressor
R^2 socre	0.717	0.795	0.822
RMSE	36.309	30.905	28.763
MSE	1318.403	955.160	827.37

- From the table it can be seen that the Histogram Gradient Boosting Regressor Model was able to achieve the R^2 metric of 0.82 which is higher than that of the other two models.
- It can be seen that the higher the R^2 value, the lesser the RMSE and MSE values.
- The run time of the Random Forest Regressor is high as compared to the other two models.

IV. Conclusion and Recommendations

- Data Pre-processing techniques can improve the quality of data used for prediction.
- Drop irrelevant data columns for higher prediction quality as it lessens over-fitting and makes it easier to understand the model.
- It is better to standardize the data with a large number of outliers as compared to normalization as normalization will compress the outliers to a smaller range.
- Null values should be handled properly at the beginning, either by dropping them or by replacing them with their median otherwise these can cause computation errors.
- The choice of model should be made after carefully analysing and pre-processing the data and the requirements.
- Care must be taken in the case of data overfitting as the model might be working with high-quality predictions in the case of training data but not for testing data, or some new, unseen data.

V. Acknowledgements

Firstly, I would like to thank Prof. Dr Markus Ludwig, Professor of Statistical Learning in Economics at TU Braunschweig for providing me with the necessary resources and assistance required to conduct this research. I also thank Hanh My Le and Dr Steinkraus Arne for the Statistical Learning in Economics course at TU Braunschweig for their insightful comments and recommendations on my methodology. In addition to it, I would like to express my gratitude to each and every one who was directly or indirectly supporting me in completing this research paper.

References

[1]Tiwari, S., "Municipal Waste Management Cost Prediction," 2022.