Applied Statistics and Machine Learning

Assignment Title: Statistical Regression Analysis

Kaggle Dataset link: Food wastage data in restaurant (kaggle.com)

1. Data Preparation (What steps would you take to prepare your data?

Data Reading: In first step, we import read.csv from pandas so we can read our

"**FOOD WASTAGE DATA**" dataset. Later on, I run Data Frame, Get Dummies and Series by using the code data1.info () to display data on the screen. After running the code, we find out we have 1782 rows (0 to 1781) and 11 columns. All columns have a non-null count with 1782 rows, which means there is no missing data. It means there is no need for data cleaning. While we get the D-type of each column, which shows there are 8 columns with categorical data and have "object" type, 3 columns with numerical data, have "int64" type, as shown in the below table.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1782 entries, 0 to 1781
Data columns (total 11 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   Type of Food           1782 non-null   int64
 1   Number of Guests       1782 non-null   int64
 2   Event Type             1782 non-null   int64
 3   Quantity of Food       1782 non-null   int64
 4   Storage Conditions     1782 non-null   int64
 5   Purchase History       1782 non-null   int64
 6   Seasonality            1782 non-null   int64
 7   Preparation Method     1782 non-null   int64
 8   Geographical Location  1782 non-null   int64
 9   Pricing                1782 non-null   int64
 10  Wastage Food Amount    1782 non-null   int64
dtypes: int64(11)
memory usage: 153.3 KB
```

Data Encoding:

We encode to convert categorical data into numerical values. After reading the data, we find out that there is a need for encoding so we can convert all the columns that are the categorical type "object" into numerical data "integer".

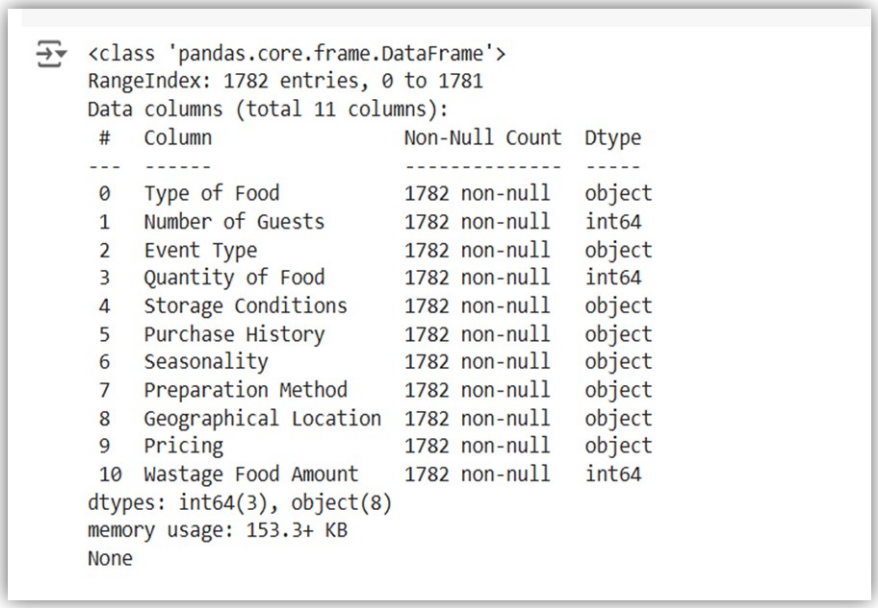
For encoding, we used Hot encoding method:

we used Get_Dummies - when one variable has more than two categories and if we use mapping method, we convert original variable into list of sub categories and assign assumed numerical value to each sub-category which may have low effect on each category. This approach avoids suggesting erroneous ordinal correlations, maintains categorical data, and improves interpretability and compatibility with different algorithms.

i.e. Type of Food, Event Type, Purchase History, Seasonality, Preparation Method

Geographical location, pricing, and storage conditions further divide into encoding columns by using get-dummies code. Now we have 1782 rows and 11 columns, as shown in the below table.

Use Code data2.info () to verify that all the columns Dtype converted to “int64”.



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1782 entries, 0 to 1781
Data columns (total 11 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Type of Food          1782 non-null   object
 1   Number of Guests      1782 non-null   int64
 2   Event Type            1782 non-null   object
 3   Quantity of Food      1782 non-null   int64
 4   Storage Conditions    1782 non-null   object
 5   Purchase History      1782 non-null   object
 6   Seasonality           1782 non-null   object
 7   Preparation Method    1782 non-null   object
 8   Geographical Location 1782 non-null   object
 9   Pricing               1782 non-null   object
 10  Wastage Food Amount   1782 non-null   int64
dtypes: int64(3), object(8)
memory usage: 153.3+ KB
None
```

Featuring and labeling:

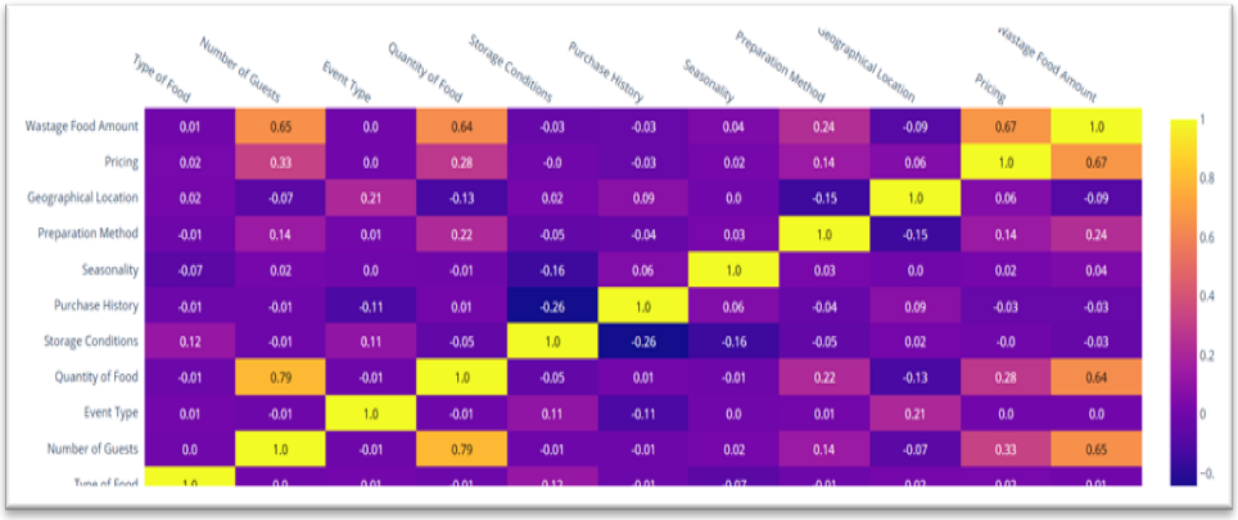
We divided the dataset into label (X) and feature (Y) using ‘Wastage Food Amount’ as a target variable to ensure the feature has a similar scale using StandardScaler as part of the preprocessing step.

Data scaling:

Data scaling guarantees that all features contribute equally to the model's performance in the context of the provided dataset, which involves predicting the waste food amount based on various features like type of food, number of guests, event type, and others. It also accelerates and increases the accuracy of the algorithms' convergence.

Heat Map:

A heat map is a kind of graphical depiction of a dataset that uses a structured matrix or rows and columns to display the volumes within the dataset. They are helpful in addressing multicollinearity, detecting anomalies, and determining correlations between multiple numerical variables.



This segment includes the heat map depicting correlation between different features for our dataset.

1. Heatmap creation:

· A heatmap is a type of data visualization that uses colour to represent a phenomenon's magnitude in two dimensions. The heatmap displays the correlation coefficients between various features and the target variable (waste food amount) in the context of this dataset.

· As per understanding of the above heatmap, which goes from yellow to blue,. Purple denotes no correlation; blue denotes a strong negative correlation; and yellow suggests a significant positive correlation.

· The correlation ranges from -1 to 1, where 1 means a strong positive correlation, -1 means a strong negative correlation, and 0 means no correlation.

Conclusion: "To sum up, the heatmap has shown how strongly variables like "number of guests," "quantity of food," and "pricing" are correlated with "stage food amount." These observations are helpful in formulating plans to reduce food waste. In the future, we can use this data to develop more focused interventions and raise the effectiveness of food service at events."

Data Splitting:

In this step, we split data into two parts, i.e. Training Set and Testing Set. We do this step because it evaluates the model's performance, which prevents overfitting and gives strong predictions. So, we split the data into two data sets, one set is used for training and the other part is for testing.

In my case, I split the data into 80:20 percentages. Here 80% is the Training Set size that we are using as a Training Set to minimize the error in predicting waste of food issues which helps in identifying how changes in features affect the amount of food wasted, which is sufficient to get a good model result. While 20% is Test size which is enough to evaluate the model's performance accurately.

Data balancing:

In the dataset Wastage Food Amount, balancing the dataset is essential. While balance is typically linked to classification tasks, it also plays a crucial role in preventing the model from being biased toward values of the target variable that occur more frequently in regression tasks.

By correcting any discrepancies in the distribution of waste food amounts, balancing aids in the prediction of food waste. For example, the model may become biased towards values that are over-represented in the dataset if particular ranges of food waste are, and this could result in false predictions for under-represented ranges. These biases can be lessened by balancing methods, including resampling, data manipulation, or the creation of synthetic data.

Specifically, this code uses grid search for hyperparameter tuning and concentrates on L1 regularization (Lasso) for a linear regression model. Regression problems and

classification problems require balancing to guarantee that ranges of values in the former and classes in the latter are represented equally to minimize bias.

2. Impact of L1, L2, and elastic net regularization on linear regression coefficients, performance, and interpretability.

In order to stop overfitting, regularization approaches in linear regression add a penalty to the model coefficients. The model's coefficients, performance, and interpretability are all impacted differently by the three popular regularization techniques: L1 (Lasso), L2 (Ridge), and Elastic Net.

Linear Regression:

To comprehend a linear relationship between the input and output variables, one sort of regression technique is called linear regression. One of the most effective machine learning methods for forecasting variable values is this one.
predicated on inputs and outputs.
With this approach, the optimal fit is assessed by minimizing the difference between the predicted values and the actual values in the dataset.

In linear regression, we try to find the optimal value of beta (β) associated with each independent variable and the line, plane or hyperplane that will help us get optimal beta values will be considered the best fit.

Equation:

$y = \beta 0+\beta 1x1+\beta 2x2+\ldots \ldots\ldots\ldots \beta nxn$

Here,

y = Dependent variable β0,β1,βn =

Intercepts

x1,x2,xn = Independent variables

Linear Regression Without Regularization:

In this step we have used two parameters in SGD Regressor: **random_state** as **1** and **Penalty** as **None**.

After this, we have used 'eta0' (learning rate) and **'max_iter'** (maximum number of iterations) as hyperparameters and passed the suitable values to them.

The next step we have followed is by passing the above-mentioned parameters in GridSearchCV.

To score our regression model we have used the scoring as r^2 and CV=5

which implies 5-fold cross validation.

```
from sklearn import linear_model
from plotly import figure_factory

# Linear Regression (LR)

LinearRegression1 = linear_model.SGDRegressor(random_state = 1, penalty = None) # building
Hparameter1 = {'eta0': [.0001, .001, .01, .1, 1], 'max_iter':[10000, 20000, 30000, 40000]}
grid_search1 = GridSearchCV(estimator=LinearRegression1, param_grid=Hparameter1, scoring='r2', cv=5)
grid_search1.fit(X_,Y)

# results = DataFrame.from_dict(grid_search1.cv_results_)
# print("Cross-validation results:\n", results)
best_parameters = grid_search1.best_params_
print("Best parameters: ", best_parameters)
best_result = grid_search1.best_score_
print("Best result: ", best_result)
best_model = grid_search1.best_estimator_
print("Intercept β0: ", best_model.intercept_)
print(DataFrame(zip(X.columns, best_model.coef_), columns=['Features','Coefficients']))
#print(DataFrame(zip(X.columns, best_model.coef_), columns=['Features','Coefficients']).sort_values(by=['Coefficients'],ascending=False))
```

OUTPUT After implementing the above code is follows:

Best Result r2: 0.6958723648840179

Modified r2: 0.6958723648840179

eta0: 0.001

Max_iter: 10000

With an R-squared value of roughly 0.696, the model offered a comparatively decent match. Food waste factors are revealed by the coefficients, which show how each aspect influences the Wastage Food Amount.

Feature and Coefficients without regularization

	Features	Coefficients
0	Type of Food	0.002133
1	Number of Guests	2.755072
2	Event Type	0.185064
3	Quantity of Food	2.782535
4	Storage Conditions	-0.122920
5	Purchase History	-0.097904
6	Seasonality	0.287960
7	Preparation Method	0.634506
8	Geographical Location	-0.604342
9	Pricing	5.268051

NOTE: The hyperparameter values we obtained in the previous phase will be taken into account when applying regularization techniques, and our model will be tuned accordingly.

Linear Regression with Regularization:

To evaluate the impact on our model.

we have used all the 3 regularizations techniques:

L1, L2 and elastic net.

1. **L1(Lasso):** In order to reduce the complexity of the model, Lasso uses a penalty term to minimize the coefficients of less significant independent variables to zero on training data.

These parameters show the ideal values for the best-performing model's learning rate **(eta0)**, regularization strength **(alpha)**, and iteration count **(max_iter)**.

Best Result r2: -3.765006609454037e+16.

The model performs poorly than a basic horizontal line model (mean model) on the training data, as indicated by the negative R-squared score.

This is a strong indication that the model is not adequately fitting the data, which could be caused by a variety of factors including poor model selection, problems with feature scaling, or problems with the quality of the data.

Feature and Coefficients L1:

	Features	Coefficients
8	Geographical Location	9.708523e+08
4	Storage Conditions	5.699270e+08
9	Pricing	5.228840e+07
1	Number of Guests	4.197307e+06
3	Quantity of Food	-2.822518e+06
7	Preparation Method	-1.563458e+07
0	Type of Food	-1.675082e+08
5	Purchase History	-2.156113e+08
2	Event Type	-2.964088e+08
6	Seasonality	-3.445643e+08

2. L2(Ridge):

Ridge used to prevent overfitting in linear regression with multiple independent variables.

The hyperparameters we have used here are the same as L1

Below are the results: r2: -4470382539047.401

This is a strong indication that the model is not adequately fitting the data, which could be caused by a variety of factors including poor model selection, problems with feature scaling, or problems with the quality of the data.

Feature and Coefficients L2:

	Features	Coefficients
8	Geographical Location	9.708523e+08
4	Storage Conditions	5.699270e+08
9	Pricing	5.228840e+07
1	Number of Guests	4.197307e+06
3	Quantity of Food	-2.822518e+06
7	Preparation Method	-1.563458e+07
0	Type of Food	-1.675082e+08
5	Purchase History	-2.156113e+08
2	Event Type	-2.964088e+08
6	Seasonality	-3.445643e+08

3. Elastic net:

In order to estimate food waste based on several variables, we used an Elastic Net regression model in our research. L1 (Lasso) and L2 (Ridge) penalties are combined in Elastic Net regularization in an effort to maximize the advantages of both methods. GridSearchCV was used to refine the model and determine the ideal hyperparameters. The optimal parameters that were determined were {'alpha': 10, 'eta0': 0.0001, 'l1_ratio': 0.1,'max_iter': 10000}.

Even yet, the model performed poorly as evidenced by its negative R-squared score of -7049136942166.159.

Given the negative number, it was discovered that the model's intercept was 5.5021081e+08, an extremely high figure that suggested a high baseline forecast. The characteristics that showed favorable contributions were pricing, number of guests, and purchase history; features that showed negative influences included quantity of food, event kind, and geographic location. The coefficients varied greatly.

Feature and Coefficients Elastic net :

	Features	Coefficients
9	Pricing	7.960896e+05
1	Number of Guests	4.417826e+05
5	Purchase History	3.365711e+05
7	Preparation Method	8.807592e+04
4	Storage Conditions	-8.562962e+04
6	Seasonality	-1.227606e+06
3	Quantity of Food	-1.770281e+06
2	Event Type	-1.798496e+06
8	Geographical Location	-1.989730e+06
0	Type of Food	-2.527359e+06

Conclusion:

Modified_r2 = 1 - (1 - best_result) * (1 - (4/5*r) / (4/5*r - c - 1)

When dealing with high-dimensional data, this computation modifies the R2 score to take sample size and feature count into consideration, yielding a more precise performance indicator.

The updated R2 for your final Elastic Net regularized model

was 0.8186777975080494, which helps to make sure that, considering the number of predictors, the model's performance evaluation is not too optimistic.

The use of L1, L2, and Elastic Net regularization techniques in linear regression models to forecast food waste has brought to light important issues with model performance. Regularization techniques helped reduce overfitting and revealed information about the significance of features, but they were unable to fully capture the complex relationships present in the data.

Hence, after analysis we can say that our model was not overfitted but it is strong indication that the model is not adequately fitting the data.

3. Impact of L2 regularization on support vector regression performance and interpretability

Support Vector Regression as the name suggests is a regression algorithm that supports both linear and non-linear regressions.

In order to penalize the size of the coefficients and reduce overfitting, L2 regularization (Ridge) is performed to SVR. This improves the model's generalization.

In SVR, we have used 2 hyperparameters: Regularization Parameter ‘C’ and ‘Kernel’. Their values we have chosen in code as:

```
Best parameters:  {'C': 110, 'kernel': 'rbf'}
Best result:  0.8191869150217148
```

Kernel: ['linear', 'poly', 'rbf', 'sigmoid']

'C': [100,1000,10000]

After, implementing the SVR with the above mentioned hyperparameters we have received the following output:

Best Parameters: {‘C’: 110, ‘Kernel’: Linear}

r2: 0.8191869150217148

The performance and interpretability of the model are greatly affected by the use of L2 regularization to Support Vector Regression.

It enhances preventing overfitting and results in more stable and reliable predictions.

The enhanced R-squared and RMSE values show that a better balance between variance and bias produced higher prediction accuracy. Additionally, the model's predictions became easier to understand and apply as a result of the shrinking of coefficients brought about by L2 regularization, which also revealed additional details about the relative relevance of each attribute.

4. If you were to implement random forest regression, then its comparative performance and interpretability would differ with respect to regularized linear regression and regularized support vector regression models.

Random Forest Regression creates several decision trees and then combines them to provide a forecast that is more reliable and accurate.

It is capable of efficiently managing non-linear connections and feature interactions.

In SVR, the use of a support vector is used to define a linear regression. Continuous linear regression enhances traditional linear regression by incorporating regularization methods such as L1 (Lasso) and L2 (Ridge). The hyperparameters that will be tuned are n_estimators and the metric for the evaluation is r2.

The hyperparameters that will be tuned are n_estimators and the metric for the evaluation is r2.

```
Best parameters:  {'n_estimators': 100}
best_score:  0.8984843382943215
modified_r2:  0.8977667102602082
Pricing                   0.410652
Number of Guests          0.275515
Quantity of Food          0.160414
Preparation Method        0.067877
Event Type                0.020238
Type of Food              0.020004
Geographical Location     0.017023
Seasonality               0.015686
Storage Conditions        0.007261
Purchase History          0.005330
dtype: float64
```

Random forest after Selecting features with higher significance and redefining feature set

```
Best parameters:  {'n_estimators': 200}
r2:  0.9254838637852796
modified_r2:  0.9249570990728894
```

Output values for random forest:

When we implement the random forest method,

we get the following results.

Best parameters: {'n_estimators': 100}

r2: 0.8984843382943215

After implementing feature selection for the random forest regression, when the first 5 features with higher significance are taken into consideration then the score increases to **0.8984843382943215 from 0.9249570990728894**.

Output: Best parameters: {'n_estimators': 200}

r2: 0.9249570990728894

In terms of predictive accuracy for the food waste dataset, Random Forest Regression performed better than L2-regularized Support Vector Regression (SVR) and L2-regularized linear regression (Ridge). Higher R-squared and lower RMSE values were obtained, which suggests improved generalization and more precise predictions.

Interpretability-wise, Random Forest offers insightful information about the significance of features by identifying non-linear correlations and interactions that regularized linear models could overlook. However, in situations where linear relationships are adequate, Ridge regression's simpler form makes it easier to understand.

Random Forest Regressor is the best fitting model which gives a good score, being more interpretable and less complex.

5. **Performing a prediction for one of your models using new data**After carefully performing and fine tuning the various regression models like Linear Regression, Random Forest Regression and Support Vector Regression on dataset FOOD_WASTAGE_DATA.

We come to the conclusion that for this dataset Random Forest Regressor is the best fitting model, giving a good score, being more interpretable.

· We used a methodical approach to show our Random Forest Regression model's prediction power on fresh data. The new dataset, new_data.csv, which has the same features as our training dataset, was first loaded. In order to ensure consistency with the training data, we next pre-processed the data by translating categorical variables to numerical values. We used StandardScaler to normalize the features following preprocessing.

· Using the prepared data, we projected the amount of food waste using our Random Forest model that had been trained. For ease of comprehension, the predictions were incorporated back into the updated dataset. The input features were taken into consideration when analysing the expected values. For instance, the model estimated that 30 food units would be wasted for a brand-new event with 50 attendees and 200 food units.

· This procedure not only confirms that the model can be applied to new data, but it also offers useful information on anticipated food waste, which helps us organize resources more efficiently and cut down on waste.

· When it came to forecasting food waste on fresh data, the Random Forest Regression model performed admirably. This sample demonstrates how the model can be used in practice by outlining the stages involved in data preparation, prediction, and result interpretation. By taking these precautions, we can make sure that our forecasts are accurate and significant, which will ultimately help with improved resource allocation decision-making.