Project 1 - Building & Evaluating ML Algorithms

Yuyang Zhang

Electrical and Computer Engineering, University of Florida

EEL 5934 Applied Machine Learning

Gainesville, U.S.

yuyangzhang@ufl.edu

*Abstract*— This article employs machine learning techniques to analyze and predict key attributes, including gross income, unit price, gender, customer type, and day of purchase, within a supermarket sales dataset. Leveraging the developed training model, we extend our capability to make predictions regarding supermarket data, thereby furnishing a solid foundation for informed business decision-making. We applied multiple linear regression, logistic regression, and random forest to predict these key attributes, and experimental results show that all methods can effectively predict the results.

# Introduction

In this article, we use machine learning techniques to analyze and predict gross income, unit price, gender, customer type, and day of purchase in the supermarket sales dataset. By applying the training model, we can make predictions for supermarket data and provide a basis for business decisions.

# Methodology

In this experiment, we used machine learning methods including multiple linear regression (with/without regularization), logistic regression, and random forest. Linear regression can be used to predict a continuous outcome variable based on one or more predictor variables. Logistic regression is a model for classification tasks that uses a logistic curve to predict the probability of the event occurring. Random forests can be applied to both regression and classification tasks. In this case, we use it on the classification task. It combines multiple decision trees to make predictions.[1]

# implementation steps

In this article, we need to complete the following tasks:

1. Data preprocessing
2. Train a multiple linear regression with and without Lasso regularization to predict gross income and unit price, study the relationship between attributes, and analyze its R-square 95% confidence interval.
3. Train a logistic regression to classify gender and customer type, study the relationship between attributes, and analyze its Accuracy 95% confidence interval.
4. Train a logistic regression classifier and a random forest classifier to predict the day of purchase. Choose a better model based on comparing Accuracy 95% confidence intervals.

For each task, we create and analyze the model using the following process:

1. Distribute data into a categorical and numerical list based on observations, and create a transformer containing different encoders.
2. Split the data set according to a certain proportion into feature training set, target training set, feature validation set, target validation set, feature test set, and target test set.
3. Select an appropriate model, merge it with the preprocessor (transformer) into a pipeline, and use the training set to train to obtain the final model. Depending on the task requirements, you may need to use a validation set to find the best hyperparameters for the model and retrain the model after applying the hyperparameters.
4. After obtaining the final model, import it into the test file for testing. Output the 95% confidence interval based on the tested performance indicators and analyze

# Experiment

## Data preprocessing

In this task, we import the file supermarket\_sales.csv in the data frame format of the Pandas library. By observation, the data contains useless features, we drop them first:

a. Invoice id: This column contains computer-generated sales slip invoice identification numbers. It is an ID number to distinguish each sale, and cannot provide characteristics for any task.

b. Gross margin percentage: The data contained in this column is the same in every sale (4.761904762), so it cannot provide features for any task.

After dropping these two data, we still need to encode the 'Date' and 'Time' in the data frame, because their format does not support the use of normal data preprocessing pipelines:

a. 'Date': We use the function 'to\_datetime' from the Pandas package to convert this data to the default date format (YYYY-MM-DD), and then apply the function 'day\_of\_week' to convert to the numbers '0, 1, 2, 3, 4, 5, 6'. These numbers correspond in order to ‘Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday’.

b. 'Time': We also use the function 'to\_datetime' to convert its data to the default time format, and then use the 'strftime' function to extract the hour and compare it according to different time periods (Morning (10:00 - 11:59 ), Afternoon (12:00 - 16:59), Evening (17:00 - 18:59), Night (19:00 - 23:59)). And we assigned time periods to the corresponding numbers '1, 2, 3, 4'.

For any numerical data, we select them out through observation and output their histogram(Fig. 1). Through histogram observation, we found that there is no outlier in the numerical data as a whole, and it conforms to Gaussian distribution to a certain extent. Therefore, we choose the preset ‘StandardScaler’ function to implement the preprocessing pipeline of numerical data in all tasks.

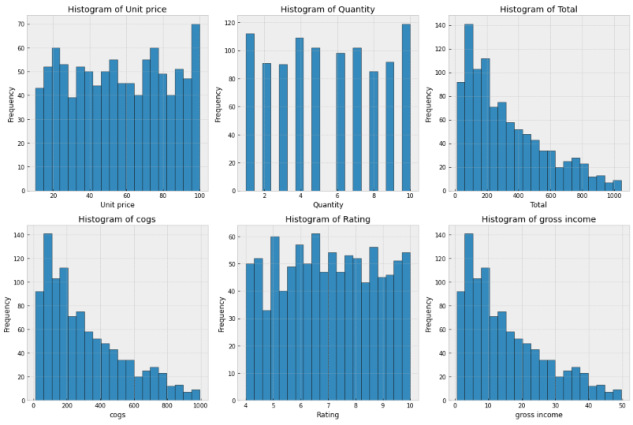


Fig. 1 Histograms of six sets of numerical data

We would choose different preprocessing pipelines for categorical data according to different tasks.

## Common steps

### Data Split

In each task, we first split the data into 4 sets: the training set of features, the test set of features, the training set of the target, and the test set of the target. The ratio of applications is 8:2. Based on the training set, we need to further divide it into 4 parts: a smaller training set for features, a validation set for features, a smaller training set for the target, and a validation set for the target. The ratio of applications is 7:3. The selection of the proportion is adjusted based on the proportion commonly used in the community, and the random number is set to 42 to obtain the same partition for reuse.

### Load Model And Test Sets

For each trained model and test set, we load it to the test file by using the packages ‘joblib’ and ‘pickle’.

### 95% confidence interval

In the test file, we customize a function to output the 95% confidence interval of the model. This metric is used to evaluate the model performance. The performance scores under different metrics are calculated through 10 times cross-validation and converted into 95% confidence intervals. This function accepts 4 parameters: model (the training model), X (test set of features), y (test set of target), and score (performance evaluation method). The function outputs the scores of 10 cross-validations and their 95% confidence intervals.

## Regression on gross income and unit price

### Predict gross income without regularization

Since the task requires predicting gross income, we distinguish features into categorical and numerical. We also move gross income out as the target that does not participate in features. For the categorical data in this task, we use the default 'OneHotEncoder' function to encode. Because it prevents unexpected correlations in the converted data compared to ordinary encoding. On the other hand, we use the ‘StandardScaler’ function as the encoder of numerical features. After that, we create a 'ColumnTransformer' pipeline to combine the encoders.

Then we need to create a pipeline that includes the preprocessor and regression model without regularization. So we choose the default model 'LinearRegression', then combine it with the preprocessor we created before through the pipeline to create our linear regression model. After training with the training set, we import the model into the test file.

By outputting the non-zero coefficients of the model, we find that gross income is affected by all features. However, according to the value of each coefficient, the gross income is hardly affected by Unit Price (-1.614834e-15) and Quantity (-1.222877e-15) because the coefficients of both of them are very low. The feature Total has a high positive coefficient (6.261186e+00) which means it highly affects Gross Income. The rest of the variables have a certain coefficient value, so they all have impacts on Gross Income in a general linear regression model.

We use R-square as the score of the regression model because it can be a good measure of the degree to which the model explains the variability of the target variable. By running the model in the test file and applying test sets, we obtained its 10 R-square-based scores: [1. 1. 1. 1. 1. 1. 1. 1. 1. 1.] and 95 % confidence interval: (nan, nan). It can be seen from the 10 ratings that the model perfectly fits the test set. The 95% confidence interval cannot be displayed because all 10 cross-validations gave a score of 1, resulting in no interval generation.

### Predict gross income with regularization

To create a model with Lasso regression, we replace the ‘LinearRegression’ model in the previous model pipeline with ‘Lasso’ and create a new pipeline that also combines the previous preprocessor. After applying the training set to generate a basic lasso regression, we also need to set its hyperparameter - the regularization strength parameter (i.e. λ) to obtain the best model. We create a grid search on it and apply validation sets to find the best-fit λ. In the grid setting, we choose [0.0001, 0.001, 0.01, 0.1, 1] as the interval of λ. This range selection comes from the results of multiple validations, and the interval is modified to confirm that it contains the optimal value. Applying this interval to the validation sets, we obtain a best λ value of 0.001. We apply λ to the lasso regression model and use both the training set and validation set to train the final model.

Since lasso selects some coefficients to become 0 based on the size of the regularization hyperparameter λ, the non-zero coefficients of the model are only Unit price (0.002865), Quantity (0.003360), and Total (11.577620). Other features ('Branch', 'City', 'Customer type', 'Gender', 'Product line', 'Date', 'Time', 'Payment', 'cogs', 'Rating') are all excluded.

By outputting the 95% confidence interval, we found that the R-squares given by the lasso regression model in 10 cross-validations were extremely close to 1, causing its interval to be between two values very close to 1. This indicates that the model predicts the test sets perfectly.

### Predict Unit Price without regularization

In this task, our target is Unit Price, so we create a new numerical list and replace Unit Price with Gross Income. Similarly, when splitting datasets, we use Unit Price as the new target, and the new features include 'Gross Income. Then we apply the new numerical list to create a preprocessor for this task, apply it to the same transformer as before, and merge it with ‘LinearRegression’ to create a linear model without the regularization.

Likewise, we output non-zero coefficients. Like the previous task, every feature has coefficients. The complete opposite is that, except for Quantity and Rating, other features have extremely high coefficients. Gross income (3.513463e+12) has a huge coefficient, while Quantity (-2.295876e+01) has a relatively low coefficient. At the 95% confidence interval, we found that the 10 R-square values were very different, which resulted in an interval of about 0.66-0.85. This indicates that the model can predict Unit Price well but is not good in reliability.

### Predict Unit Price with regularization

Similar to the Gross Income task, we use lasso regression instead of linear regression add it to our pipeline, and train through a training set to obtain the basic model. When creating the grid search, we used a regularization hyperparameter interval [0.001, 0.01, 0.1, 1, 10]. Through grid search, we found the optimal λ to be 1, applied it to re-train the lasso regression model, and then loaded it into the test file.

We output the coefficients of lasso regression and find that only Quantity (-19.653822) and Total (29.130228) are retained, and the remaining features ('Branch', 'City', 'Customer type', 'Gender', 'Product line', 'Date', 'Time', 'Payment', 'cogs', 'gross income', 'Rating' ) are excluded. The 95% confidence interval of this task is (0.67, 0.86), which is not much different from linear regression. Similar to the linear regression model without regularization, the model can predict unit price well but is not good in reliability.

## Classification on gender and customer type

### Predict gender

In this task, we need to create a logistic regression model to predict Gender (female or male) based on Product Line, Payment, and Gross Income in Brand C. Since Gender is a categorical target, we need to modify our transformer, numerical, and categorical lists. We only care about the impact of Product line, Payment, and Gross Income on gender, so we assign Product line and Payment to categorical lists, and Gross Income to numerical lists. Since we also need to apply interaction attribution of degree 2 to learn relationships, we cannot use the one-hot encoding because this would cause the feature column to expand. So in this task, we choose ordinal encoding for any categorical features. Therefore, the transformer consists of ‘OrdinalEncoder’ and ‘StandardScaler’.

Before splitting the dataset, we need to encode the target into a number. We also use the ordinal encoder to ensure consistency. We select the relevant columns in Brand C using the same split method as in the previous task and load the required sets into the test file. We can implement interaction attribution of degree 2 by defining the 'PolynomialFeatures' and setting its hyperparameters degree and interaction\_only to 2 and True. In addition, we also use the default 'LogisticRegression()' and set its hyperparameters 'tol' and 'fit\_intercept' to '2' and 'True' to correspond to the selection of 'PolynomialFeatures'. Finally, we merged the preprocessor (the same one used before, but applied the lists for this task), polynomial features, and logistic regression into a pipeline and used the train set to train a base model.

Similar to the regression task with lasso, we use grid search to find the best lambda for the logistic regression model. We use accuracy as the performance metric due to classification. To find the best λ, we set interval [0.0001, 0.001, 0.01, 0.1, 1, 10]. It outputs the best lambda is 0.0001. Next, we apply it to logistic regression to re-train the model and import it into the test file.

We first output all the interaction attribute coefficients of degree 2(Fig. 2). Among the 7 attributes, 'Product line + gross income' has the highest coefficient (approximately -0.00256), which is the most informative. On the other hand, we are also interested in the interaction attributes coefficients of degree 2 for male customers. We plotted a graph to express the coefficient of each attribute. According to observation, similarly, ‘Product line + gross income’ is also the most informative attribute.

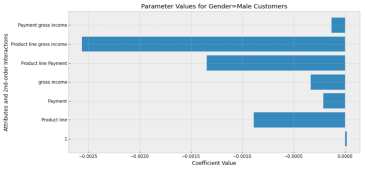


Fig. 2 In a logistic regression model to predict gender, output all the interaction attribute coefficients of degree 2

We also use the 95% confidence interval to evaluate the performance of logistic regression, but we need to change the scoring method to accuracy to match the classification task. The 95% confidence interval of this model is approximately 0.52-0.57 (0 is female, 1 is male). This means that the model would tend to predict more male customers when predicting customer gender.

### Predict Customer type

In this task, we need to create a logistic regression model to predict Customer Type (Member or Normal) based on Gender, Day of Week, and Time Slot in Brand C. So we changed the target to Customer Type and the features to Gender, Date, and Time. Since all features are categorical features, we do not need ‘StandardScaler’ in the transformer. For categorical features and targets, we also use the ordinal encoder to convert them into numbers. Then we split the datasets and loaded the test sets into the test file. Similarly, in order to apply interaction attribution of degree 2, we need to set the same hyperparameters as classification on Gender for polynomial features and logistic regression in the pipeline. After using the training set to train the basic model, we apply grid search. This task’s hyperparameter interval is [0.0001, 0.001, 0.01, 0.1, 1, 10]. By validation, the best λ it finds is 0.0001. After applying λ to the logistic regression model and retraining using the training set and validation set, we re-train the model and load it into the test file.

For this task, we need to output all the interaction attribute coefficients of degree 2 (Fig. 3). By outputting the coefficients of 7 attributes, we found that Gender + Date (approximately 0.00085) is the most informative. Then we filter out normal customers and plot their attributes coefficient graph. By observing, Gender + Date is also the most informative.

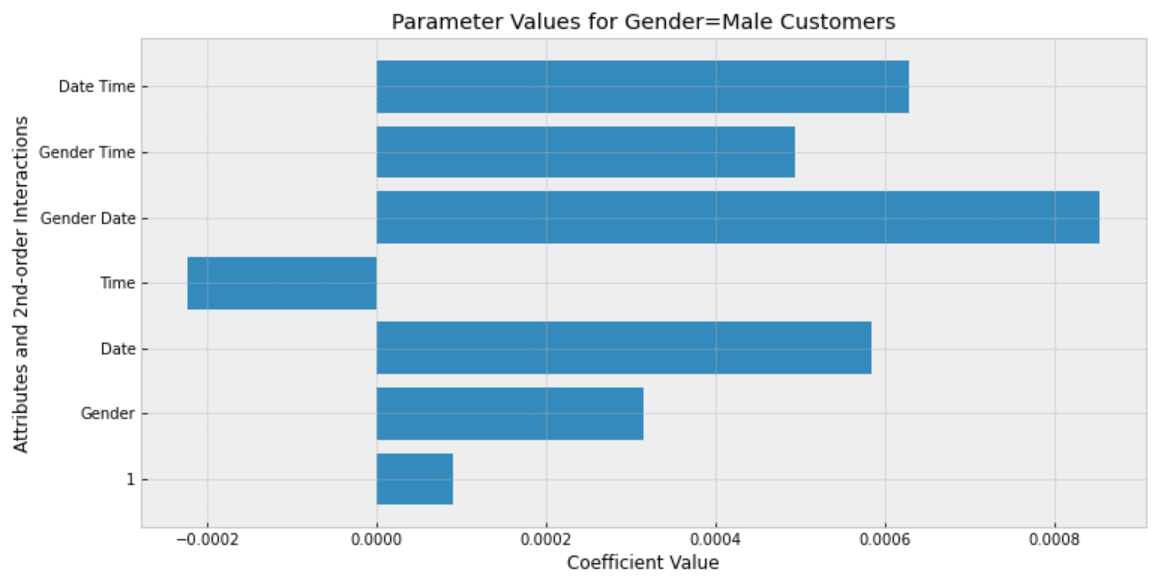


Fig. 3 In a logistic regression model to predict Customer Type , output all the interaction attribute coefficients of degree 2

By outputting the 95% confidence interval, we find that the interval of this model is approximately 0.55-0.63 (0 is member, 1 is normal). This means that the model would tend to predict more normal members when predicting customer type.

## Classification on day of purchase (Date)

We use logistic regression and random forest to create two different classification models to predict the Day of Purchase.

1. Logistic Regression

The logistic regression model is similar to tasks 4 and 5. We replace the target with Date, but the features include all columns except Date. In this task, since we do not need to provide interaction attribute coefficients of degree 2, we use the one-hot encoder to prevent unexpected correlation. Therefore, in this task, we use the transformer composed of ‘StandardScaler’ and ‘OneHotEncoder’, and merge it with ‘LogisticRegression’ to form the base model.

We use validation sets to perform a grid search on the base model to find the best regularization hyperparameter λ and the maximum number of iterations, max\_iter. This is added to prevent the logistic regression from being unable to converge, and its improvement in the model is far less important than λ. The range of super parameter λ is [0.001, 0.01, 0.1, 1, 10], and the range of max\_iter is [250, 500, 1000]. After grid search, the best lambda is 0.001 and the best maximum number of iterations is 250. We apply it to the logistic regression model retrain it using training and validation sets, and load the trained model into the test file.

1. Random Forest

We merge the preprocessor (same as in logistic regression) with the default ‘RandomForestClassifier’ to create a basic random forest model using the training set. When using grid search to select hyperparameters, we consider more hyperparameter selections because the performance of random forests depends on more hyperparameters. We choose the number of trees (n\_estimators), the criterion for node splitting (criterion), the maximum depth of each tree (max\_depth), the minimum number of samples required for node splitting (min\_samples\_split), and the minimum number of leaf nodes required. Number of samples (min\_samples\_leaf). After applying grid search to the validation set, we found the optimal hyperparameter sets: 150, entropy, None, 2, and 2. After applying these optimal hyperparameters we re-train the random forest model using the training and validation sets and load it into the test file.

By outputting the 95% confidence interval of both the logistic regression model and the random forest model, we find that the interval of the logistic regression model is approximately 0.13-0.16, while the interval of the random forest model is approximately 0.11-0.20. Since the interval of the logistic regression model has a smaller gap, this means that it can predict the day of purchase more reliably, which also means that we will give priority to the logistic regression model.

# conclusion

Based on the data obtained in the regression experiment, we found that in the task of predicting gross income, the linear regression model we created demonstrated a high degree of fit with or without lasso regularization and performed perfectly in performance analysis. According to its attributes coefficient, we know that Total is the most informative feature. When predicting unit price, our linear regression model is not highly fitted, and according to performance analysis, we know that it has a wide confidence interval, which means it is not reliable.

Based on the data obtained in the classification experiment, we found that the most informative attribute for gender in the logistic regression model we trained in the task of predicting gender is ‘Product line + gross income’. And in performance analysis, it gives excellent probabilities and relatively narrow confidence intervals, which means that it performs very well in terms of reliability. In the task of predicting customer type, we found that the most informative attribute for customer type is 'Gender + Date'. Likewise, the logistic regression model we trained performs well in performance analysis, and it also gives excellent probabilities and reliable confidence intervals.

In the task of predicting the day of purchase, we compare the confidence intervals of the logistic regression model and the random forest model to choose a better classification model. Based on the results, we found that the logistic regression model has a narrower interval, which means that it can provide more reliable predictions compared to random forest, so we think the logistic regression model is better.

##### References

1. A. Géron, “Hands-On Machine Learning with Scikit-Learn and TensorFlow,” 1st ed. O’Reilly Media, 2017.