Kaggle InClass Competition Report for Coupon Recommendation

CompSci 671

Danyang Chen, Kaggle username: Birfied

December 10, 2021

Coupon recommendations are widely used in online platforms to attract customers. It is challenging for the business owners to find potential customers to provide coupons, so as to reduce advertising cost and maximize profits. Machine learning can be used to develop targeted coupon recommendstions for this purpose. In this report, I will analyze the Coupon Recommendation Dataset (Link) to decide whether customers will accept a coupon based on their profile.

1 Exploratory Analysis

The training dataset has 10184 entries and 21 features. 12 of the features are objects, 4 are integers and 5 are floats. In this dataset, 57% of people accept the coupon recommendation. First, some entries are missing values for some features, for example, Bar, Coffeehouse, Carryaway. Luckily, only less than 2% of entries are missing either one of these features. There is no need to remove these features, so the missing ones are just filled with the average value of this feature. Then, the features with object values are replaced with integers. For the features Driving_to, Passenger, Weather, Time, Coupon, Coupon validity, Gender, Maritalstatus, Occupation, the features can be replaced by a integer indicating a category. For the features Age, Education and Income, they should be sequentially maped to the categories, for example, lowest income correspoinds to 0 and highest income corresponds to 1. All the features values are normalized to between 0 and 1 using the MinMaxScaler in sklearn to make sure all the features contributes equally to the fitting of the model, which is required for some of machine learning models.

2 Methods

2.1 Models

The models I choose for training are decision tree, random forest and SVC because they are well implemented and documented in the sklearn package. They have only a few number of tunable parameters, which are pretty easy to train. They are much faster than training neural networks.

2.2 Training

The decision tree model is trained by choosing the best split feature based on the gini index or entropy gain, until the data is completely classified or it reaches the maximum depth. In sciket-learn package, the CART algorisim is implemented.

The random forest model is trained by first obtaining a random decision tree on a subset of training data, and then find the best splitting features in a subset of all the features. The prediction will be the average vote of all the randomly generated trees.

The support vector machine model split the data by finding the hyper-plane that maximize the distance from the decision boundary to the nearest training observations.

The cross validation is done by first splitting the dataset into k folds, then using one fold for testing and all other folds for training. Iterate over all folds for testing and average the accuracy over k folds.

The CPU time estimation for a 10-fold cross validation of these models on the dataset with 10184 entries are

- Decision tree: 0.31s
- Random forest: 6.83s
- Support Vector Machine: 27.3s

2.3 Hyper-parameter Selection

For decision tree model, the hyper parameters in consideration are criterion (splitting criterion, gini or entropy), splitter (best or random), max_depth (maximum depth depth of the tree), min_samples_leaf (minimum numer of samples for splitting) and min_samples_split (minimum number samples required to be at a leaf node). The best parameters are

- criterion: entropy
- max_depth: 10
- min_samples_leaf: 1
- min_samples_split: 10
- splitter: random

For random forest model, the hyper parameters in consideration are n_estimator (number of trees), criterion (splitting criterion, gini or entropy), max_depth (maximum depth depth of the tree). The best parameters are

- n_estimator: 600
- criterion: entropy
- max_depth: 30

For support vector machine model, the hyper parameters in consideration are C (regularization parameter), kernel ('linear', 'poly', 'rbf' or 'sigmoid'). The best parameters are

- C: 4
- kernel: rbf

3 Results

The accuracy in these models are

- Decision tree: 0.771
- Random forest: 0.829
- Support Vector Machine: 0.815

Precision:

- Decision tree: 0.815
- Random forest: 0.763
- Support Vector Machine: 0.730

Recall:

- Decision tree: 0.719
- Random forest: 0.768
- Support Vector Machine: 0.739

The random forest model predicts the best accuracy.

Citations

Sklearn (scikit-learn.org)

Code

```
Put your code here.
```

Include necessary comments so that the grader can understand what is going on. Points will be taken off if it is not clear. Your code should be executable with the installed libraries (with citations) and with only minor modifications.