

Machine Learning Engineer Nanodegree

Capstone Project

Jiawei Jin
December 31st, 2018

I. Definition

Project Overview

Rossmann operates 3,000 pharmacies in seven countries in Europe. Currently, managers at the Rossmann store are required to predict their daily sales for the next six weeks. Store sales are subject to many factors, including promotions, competition, school holidays and legal holidays. As thousands of operators predict sales based on their unique circumstances, the accuracy of the results can vary widely. In this project, it will predict the daily sales of 1,115 Rossmann stores in Germany for 6 weeks. Reliable sales forecasts will allow store operators to increase productivity and enthusiasm to create more efficient staffing arrangements. By helping Rossmann create a strong predictive model, I'll help operators stay focused on what matters to them: their customers and their teams.

Basically, this is a data mining challenge and data mining is the process of discovering patterns in large data sets involving methods of machine learning (ACM SIGKDD., 2006). Therefore, I want to use machine learning methods to solve this problem.

Problem Statement

I want to set up machine learning on old sales records and then predict the daily sales after six weeks. The specific steps and possible problems are as follows:

- Clean and organize the data. Possible problems are the handling of outliers and the handling of missing values.
- Perform a single variable analysis on the data, analyze multiple variables, and extract high-correlation variables that affect the results. I may encounter problems with how to extract highly correlated eigenvalues.
- Modeling with XGBoost, how to adjust the data of the training model to suit the input requirements of the XGBoost model is the challenge.
- How to find the most suitable super ginseng.
- Predict and evaluate the results. Using RMSPE as the evaluation which requires a visualization of the results and the need to supplement relevant knowledge.

Metrics

I will use RMSPE as a validation function, because there are many stores which can vary in sales level and Using RMSPE can eliminate this bias make every store

equally weighted in lost fuction. The lower the value, the smaller the difference. It is a measure of the difference between the predicted value of the model and the actual observed value.

$$\text{RMSPE} = \sqrt{\frac{1}{n} \sum_{i=1}^n \left(\frac{y_i - \hat{y}_i}{y_i} \right)^2}$$

y is the actual sales data and y_hat is the forecast data and any sales data of 0 will not be included in this evaluation. The advantage of using this evaluation function is that the problem is a linear regression problem. The predicted result is a specific numerical vector. The relevant prediction data can only be one that cannot be accurately matched, so the difference between the predicted value and the actual value is calculated. The squared sum of the average results of the final re-opening greatly reduces the numerically matching requirement between the predicted and actual values compared to the difference between the direct actual value and the predicted value.

II. Analysis

Data Exploration

Input data set as follows:

- Train.csv - historical data including sales training (Contains "Store", "DayOfWeek", "Date", "Sales", "Customers", "Open", "Promo", "StateHoliday", "SchoolHoliday" fields).
- Test.csv - historical data excluding Sales including historical data for sales (Contains "Id", "Store", "DayOfWeek", "Date", "Open", "Promo", "StateHoliday", "SchoolHoliday" fields).
- Sample_submission.csv - sample data format (Contains the "Id", "Sales" fields).
- Store.csv - additional information about the store (Contains "Store", "StoreType", "Assortment", "CompetitionDistance", "CompetitionOpenSinceMonth", "CompetitionOpenSinceYear", "Promo2", "Promo2SinceWeek", "Promo2SinceYear", "PromoInterval" fields).

Most of the fields are self-explanatory. The following are descriptions for those that aren't:

- Id - an Id that represents a (Store, Date) duple within the test set
- Store - a unique Id for each store
- Sales - the turnover for any given day (this is what you are predicting)
- Customers - the number of customers on a given day
- Open - an indicator for whether the store was open: 0 = closed, 1 = open

- StateHoliday - indicates a state holiday. Normally all stores, with few exceptions, are closed on state holidays. Note that all schools are closed on public holidays and weekends. a = public holiday, b = Easter holiday, c = Christmas, 0 = None
- SchoolHoliday - indicates if the (Store, Date) was affected by the closure of public schools
- StoreType - differentiates between 4 different store models: a, b, c, d
- Assortment - describes an assortment level: a = basic, b = extra, c = extended
- CompetitionDistance - distance in meters to the nearest competitor store
- CompetitionOpenSince[Month/Year] - gives the approximate year and month of the time the nearest competitor was opened
- Promo - indicates whether a store is running a promo on that day
- Promo2 - Promo2 is a continuing and consecutive promotion for some stores: 0 = store is not participating, 1 = store is participating
- Promo2Since[Year/Week] - describes the year and calendar week when the store started participating in Promo2
- PromoInterval - describes the consecutive intervals Promo2 is started, naming the months the promotion is started anew. E.g. "Feb,May,Aug,Nov" means each round starts in February, May, August, November of any given year for that store

Relevant preliminary statistical analysis is as follows:

- Test.csv has 41088 data. The missing data is missing 11 features on the feature Open. According to the missing Date, StateHoliday and SchoolHoliday, the estimated Open is 1 and filled in. The following are the first five output examples:

	Id	Store	DayOfWeek	Date	Open	Promo	StateHoliday	SchoolHoliday
0	1	1	4	2015-09-17	1.000	1	0	0
1	2	3	4	2015-09-17	1.000	1	0	0
2	3	7	4	2015-09-17	1.000	1	0	0
3	4	8	4	2015-09-17	1.000	1	0	0
4	5	9	4	2015-09-17	1.000	1	0	0

- Store.csv has 1115 data. The missing data features are CompetitionDistance, CompetitionOpenSinceMonth, CompetitionOpenSinceYear, Promo2SinceWeek, Promo2SinceYear and PromoInterval. There are 3 missing CompetitionDistance. I guess the three stores have no extra competitors with a very large value in the effective distance. The situation of CompetitionOpenSinceMonth and CompetitionOpenSinceYear has been missing. I guess it was long ago or before the train data. This competitor I gave a default time before 2010, Promo2SinceWeek, Promo2SinceYear and PromoInterval. These three characteristics are indeed the same, that is, all three items that are not participating in Promo2 are empty, then I set the time to a future time of 2030, PromoInterval is filled with "0,0,0,0". The following are the first five data samples:

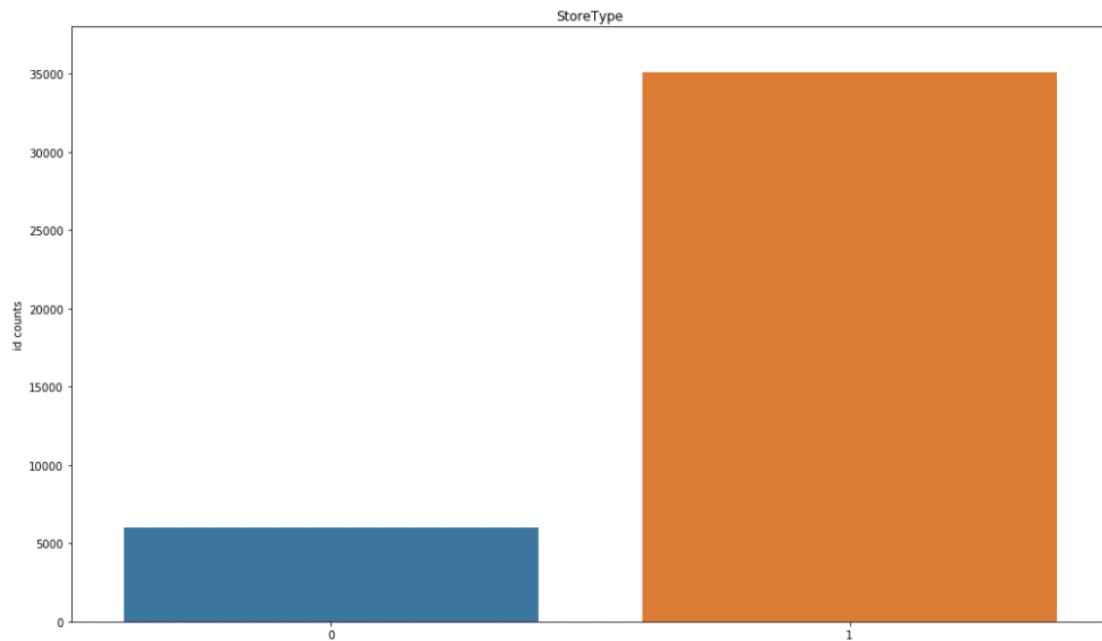
	Store	StoreType	Assortment	CompetitionDistance	CompetitionOpenSinceMonth	CompetitionOpenSinceYear	Promo2	Promo2SinceWeek	Promo2SinceYear	PromoInterval
0	1	c	a	1270	9	2008	0	0	2030	0,0,0,0
1	2	a	a	570	11	2007	1	13	2010	0,3,6,9
2	3	a	a	14130	12	2006	1	14	2011	0,3,6,9
3	4	c	c	620	9	2009	0	0	2030	0,0,0,0
4	5	a	a	29910	4	2015	0	0	2030	0,0,0,0

- Train.csv has 1017209 data, no data missing, and all the data provided generally have no abnormal values. The sample is as follows:

	Store	DayOfWeek	Date	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday
0	1	5	2015-07-31	5263	555	1	1	0	1
1	2	5	2015-07-31	6064	625	1	1	0	1
2	3	5	2015-07-31	8314	821	1	1	0	1
3	4	5	2015-07-31	13995	1498	1	1	0	1
4	5	5	2015-07-31	4822	559	1	1	0	1

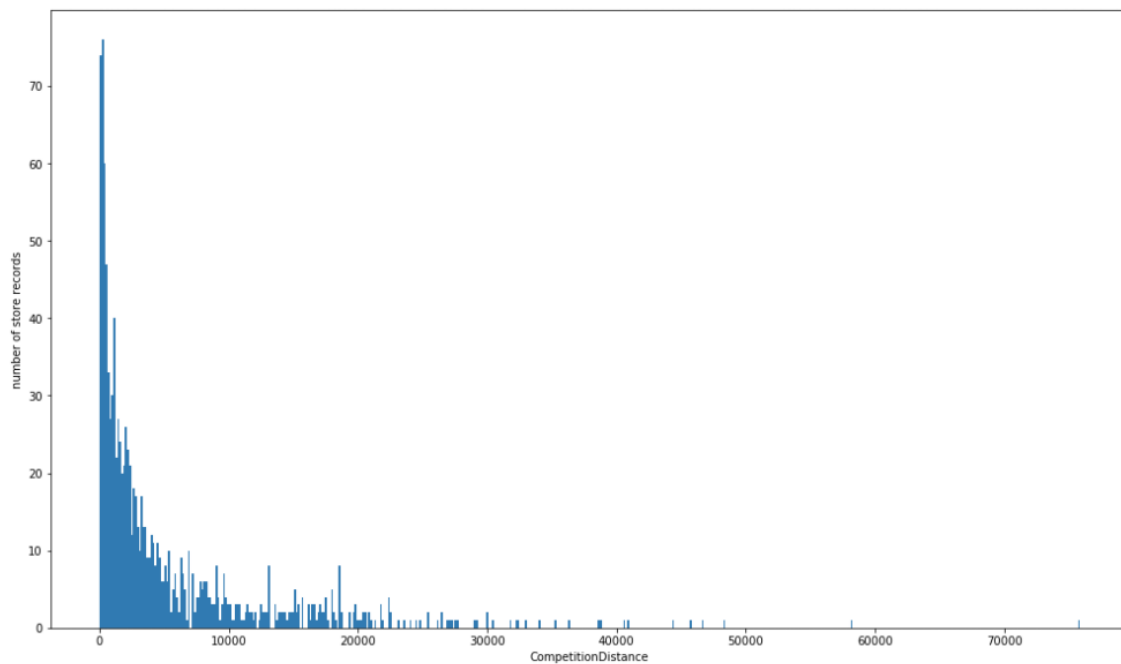
Exploratory Visualization

Visualization of feature open for test dataset



There are 5984 Open in the test dataset with 0, and the predicted value is directly set to 0.

View the distribution of stores at various distances for the store dataset view feature ComparisonDistance.



And view the details of the feature `CompetitionDistance`.

```
store['CompetitionDistance'].describe()
```

```
count    1112.000
mean      5404.901
std       7663.175
min        20.000
25%       717.500
50%      2325.000
75%      6882.500
max     75860.000
Name: CompetitionDistance, dtype: float64
```

The median is at 2325m, and some are larger than 20km. For the three missing values, I also do a maximum of 99999 to fill, indicating that there is no competitor within the effective distance. More than 20km, I will mark it as no competitor afterwards.

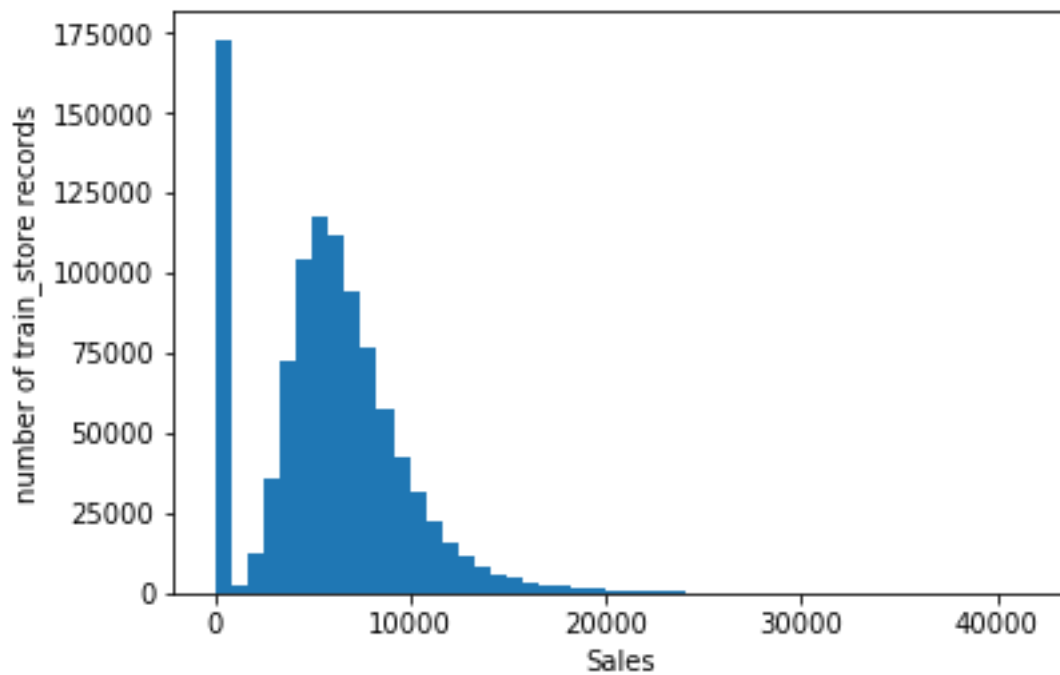
See the competitor's opening time by viewing feature `CompetitionOpenSinceYear` in the store data set.

```
store['CompetitionOpenSinceYear'].describe()
```

```
count      764.000
mean       2008.753
std         6.326
min        1900.000
25%        2006.000
50%        2010.000
75%        2013.000
max        2030.000
Name: CompetitionOpenSinceYear, dtype: float64
```

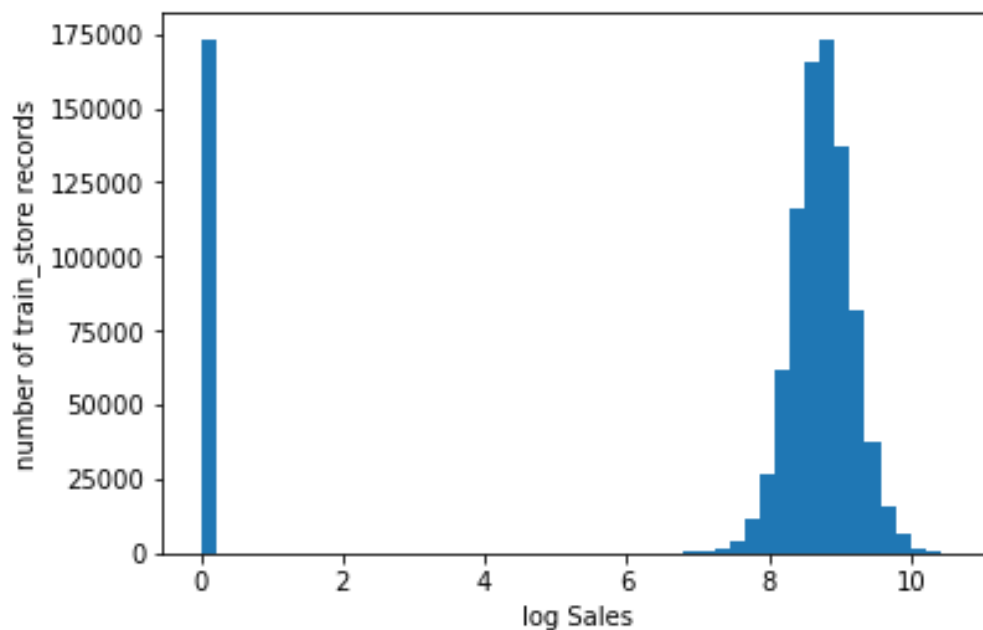
Missing values are filled in with the median 2010.

Visualize Sales for the train dataset.



Found that there are a large number of 0 data which is meaningless for training, we will only use data whose Sales is not 0 in training.

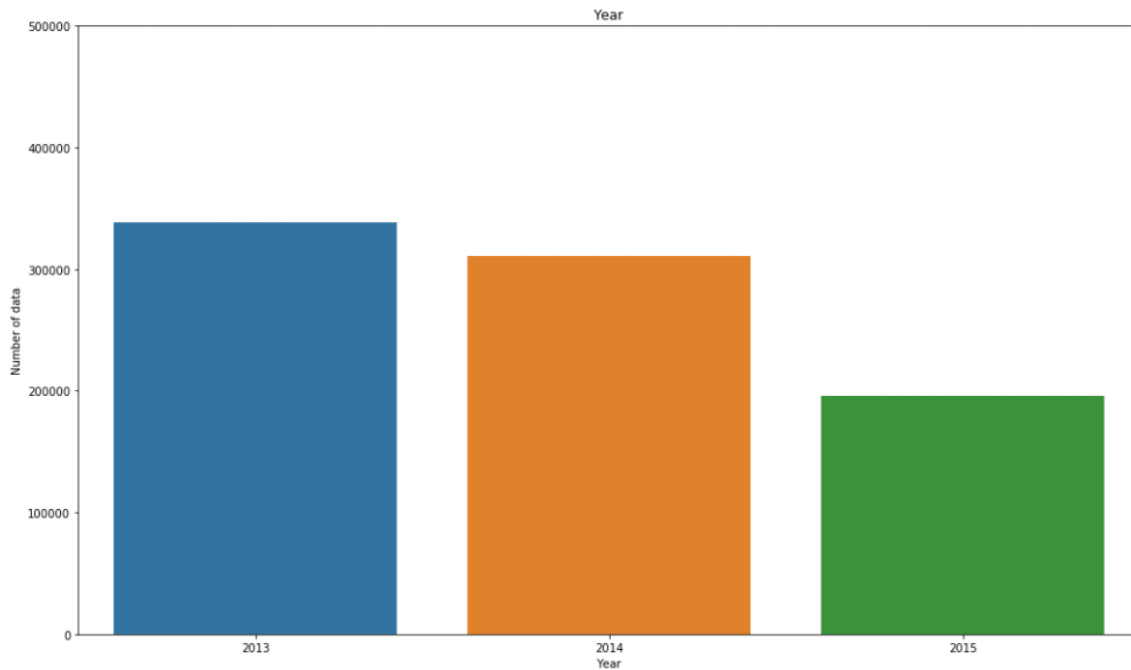
In addition, I will only train data with Open equal to 1, because open equal 0 is the same as sales equal to 0. The result set is an oblique distribution set which is a better training set. It will predict the result for better precision. I will convert it to a normal distribution graph as follows:



It can be seen from the figure that the prediction results are mainly concentrated between 8-10.

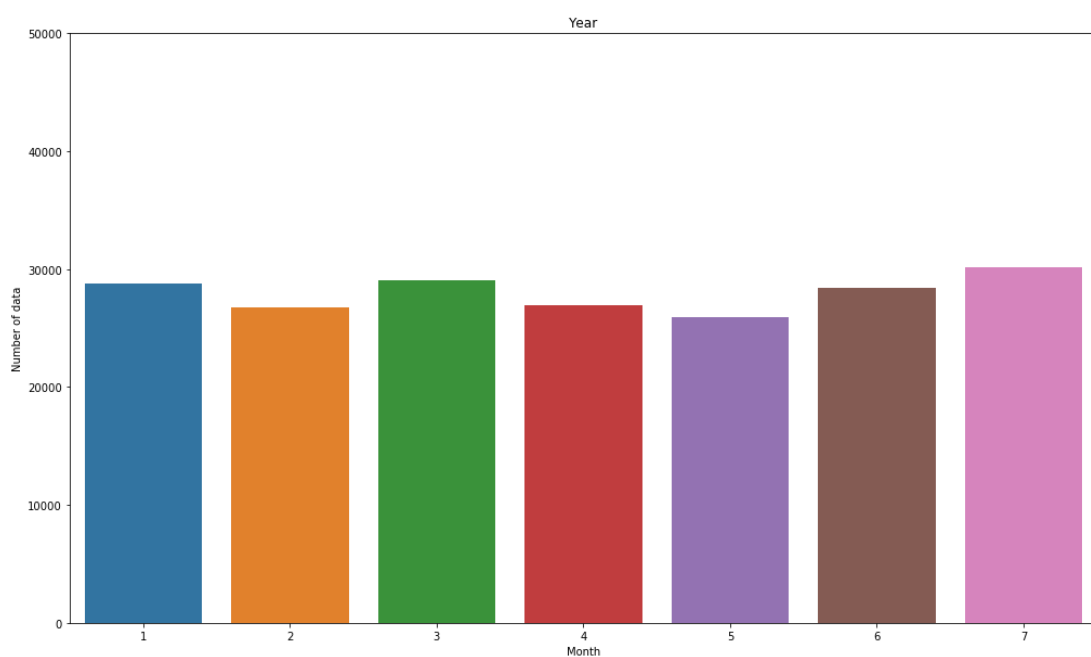
Visualize the Year in the date of the train.

```
Year
2013    337943
2014    310417
2015    196032
Name: Open, dtype: int64
```



I use 2013, 2014 for training, and the last month of 2015 as test verification data can guarantee the order of time from the past to the future.

The picture below is the month involved in 2015.



July is used as verification data. The verification data accounted for 3.5751% in training data set.

Algorithms and Techniques

This problem belongs to the regression problem in supervised learning, that is, the problem of predicting the test set based on the existing data set modeling.

For the features, the null value is filled according to the relevant business scenario by preprocessing, the abnormal value is processed to avoid overfitting the data, the date is converted, and then one-hot is performed for some classification features such as Assortment, StoreType and StateHoliday, Let train and test be connected to the store dataset, and finally only consider Open training data.

XGBoost is very suitable for this field in model selection. The data size of the first raining is not large, deep learning requires a larger amount of data sets, the second data is mainly in the form of tables, and deep learning is mainly for images, audio and others, while the traditional statistical-based methods are more Suitable, the third in the machine learning method, xgboost is fast, high precision, and the interpretation of the tree-based model model is high.


The interpretation of this algorithm begins with GB, starting with many weak models and then adding the pre-results of each weak model. The first model is the model after F0, for example, F1 which is based on this model F0 plus $h(x)$. This h actually is the representation of the residual. It is necessary to learn to generate this h through the negative gradient of the function space (the residual is known, $F(X)$ is known, the real result y is known), and then it can continue by deriving h to update the model.

The next step is to introduce GBDT. In fact, the weak model in GB is DT (decision tree) mainly refers to the regression tree, and this DT is a weak model. The leaf node does not exceed 10, the depth does not exceed 5, and the learning rate is less than 0.1. Cross-validation methods to select the optimal parameters.

Finally, returning to Xgboost is an efficient implementation of the GB algorithm. Its weak learner can be either gbtrees or gblinears.

Specific advances refine the regularization term in the objective function, which means that it can be better generalized to avoid overfitting. The regularization term is determined by the number of leaves and the value of each leaf.

$$Obj = \sum_{i=1}^n l(y_i, \hat{y}_i) + \sum_{k=1}^K \Omega(f_k)$$


Training loss Complexity of the Trees

The first and second derivatives are used in the loss function part (the former optimizes the empirical error, the latter controls the generalization error), while the GB only uses the first derivative.

$$\Omega(f_t) = \underbrace{\gamma T}_{\text{Number of leaves}} + \underbrace{\frac{1}{2} \lambda \sum_{j=1}^T w_j^2}_{\text{L2 norm of leaf scores}}$$

In the choice of segmentation points, GBDT uses the GINI coefficients to find a greedy algorithm that minimizes the mean square error, while xgboost uses an optimized approximation algorithm that maximizes, lamda, gama and is associated with regularity to improve accuracy and efficiency.

There are other optimizations in the calculation process that increase the amount of calculation and increase the calculation speed.

Because it is a linear regression problem of supervised learning, I will also use the Ridge regression model and the Lasso regression model to calculate the results as a benchmark regression model.

Ridge Regression is a technique for analyzing multiple regression data that suffer from multicollinearity. When multicollinearity occurs, least squares estimates are unbiased, but their variances are large so they may be far from the true value. By adding a degree of bias to the regression estimates, ridge regression reduces the standard errors. It is hoped that the net effect will be to give estimates that are more reliable. Another biased regression technique, principal components regression, is also available in NCSS. Ridge regression is the more popular of the two methods.

Following the usual notation, suppose our regression equation is written in matrix form as:

$$\underline{\mathbf{Y}} = \underline{\mathbf{X}}\underline{\mathbf{B}} + \underline{\mathbf{e}}$$

Y is the dependent variable,

X represents the independent variables,

B is the regression coefficients to be estimated, and

e represents the errors are residuals

Lasso regression analysis is a shrinkage and variable selection method for linear regression models. The goal of lasso regression is to obtain the subset of predictors that minimizes prediction error for a quantitative response variable. The lasso does this by imposing a constraint on the model parameters that causes regression coefficients for some variables to shrink toward zero. Variables with a regression coefficient equal to zero after the shrinkage

process are excluded from the model. Variables with non-zero regression coefficients are most strongly associated with the response variable. Explanatory variables can be either quantitative, categorical or both. Having a larger pool of predictors to test will maximize your experience with lasso regression analysis. The lasso regression analysis will help you determine which of your predictors are most important. Note also that if you are working with a relatively small data set, you do not need to split your data into training and test data sets. The cross-validation method you apply is designed to eliminate the need to split your data when you have a limited number of observations.

The loss function of Lasso is in the form:

$$L = \sum (\hat{Y}_i - Y_i)^2 + \lambda \sum |\beta|$$

Benchmark

Since it is a sales forecasting model, I think that the error rate of data prediction of 20% is acceptable. My choice is XGBoost as my prediction model. The way to measure the result is RMSPE.

Using the simplest mean benchmark model to train with the full amount of test data, the score is 0.45019, so the model result I selected should at least exceed this value.

III. Methodology

(approx. 3-5 pages)

Data Preprocessing

To sort out the missing data of the test dataset, only the Open feature guesses the missing ones by judging the date, and performs data conversion for DATE, which becomes Year, Month, Day, and One-Hot for StateHoliday.

Organize the missing data of the store dataset, handle the CompetitionOpenSinceMonth, and the CompetitionOpenSinceYear is filled by taking the median. The missing of Promo2SinceWeek, Promo2SinceYear and PromoInterval is set to 0, which is much larger than the current time of 2030, (0,0,0,0). In order to connect to train and test to set up without participating in Promo2 activity, do the same data conversion for Date, replace the string in PromoInterval with numbers, and do One-Hot for StoreType.

Organize the train dataset, convert the feature Date, do One-Hot for StateHoliday, but only train Open=1.

Train and test are connected to the store data through the store feature to get train_data and test_data. For the same processing of these two data sets, the combination of CompetitionOpenSinceMonth, CompetitionOpenSinceYear and a feature CompetitionMonths indicates the duration of the competitor's opening. Combining Promo2SinceWeek

Promo2SinceYear PromoInterval into a feature IsPromo2 indicates whether Promo2 is participating in the current time.

For the result value, since Sales is a positively skewed distribution, in order to unify the data to an order of magnitude and convert it into a normal distribution, it is generally logarithmic, which can speed up the gradient to find the optimal solution. Can improve the accuracy of the calculation. Therefore, the predicted result obtained must be further inversed before the evaluation score can be submitted.

Implementation

- First establish the test function rmspe

```
Rmspe = np.sqrt(np.mean(w * (y - yhat)**2))
```

Used to evaluate the accuracy of the model.

- Modeling with the Ridge algorithm

Using Ridge modeling in Sklearn's linear_model,

```
Clf = Ridge(alpha=1)
```

- Modeling with the Lasso algorithm

Using Lasso modeling in Sklearn's linear_model,

```
Clf = Lasso (alpha=1)
```

- Modeling with the Xgboost algorithm

I used the python interface by recompiling the xgboost library on the windows platform to use the CPU version I chose.

```
Gbm = xgb.train(params, dtrain, num_trees, evals=watchlist,  
early_stopping_rounds=100, feval=rmspe_xg, verbose_eval=True)
```

When I want to from sklearn.feature_selection import SelectKBest, there is an ImportError: cannot import name check_build. But it worked for me after installing scipy.

Refinement

For the Ridge algorithm model, RidgeCV is used to select the parameters.

```
Clf = RidgeCV(alphas=[0.1, 0.5, 1.0, 10.0], cv=10,  
fit_intercept=True, scoring=rmpse_estimator)
```

The result of the optimization is not much changed at 0.370419.

LassoCV is used to optimize parameters for the Lasso model.

Clf = LassoCV(cv=20)

The optimized result is 0.471350.

IV. Results

Model Evaluation and Validation

Ridge modeling uses the RidgeCV method to train the dataset size to 844392

Training time is 27.753535 seconds

The test_data dataset size that needs to be predicted is 35104.

The forecasting time is 0.008522

Predictive results rating is 0.41700

Lasso modeling uses the LassoCV method to train the data set size to 844392

Training time is 13.514766 seconds

The test_data data set size that needs to be predicted is 35104.

The forecasting time is 0.009526

Predictive results rating is 0.43365

XGBoost modeling uses the xgb.train method to train the dataset size to 844392

Training time is 11 minutes and 8 seconds

The test_data data set size that needs to be predicted is 35104.

The forecasting time is 1.973393

Predictive results rating is 0.11572

Small changes in the dataset will not significantly change the result, the new score is 0.41512, 0.43433, 0.11778 respectively.

Justification

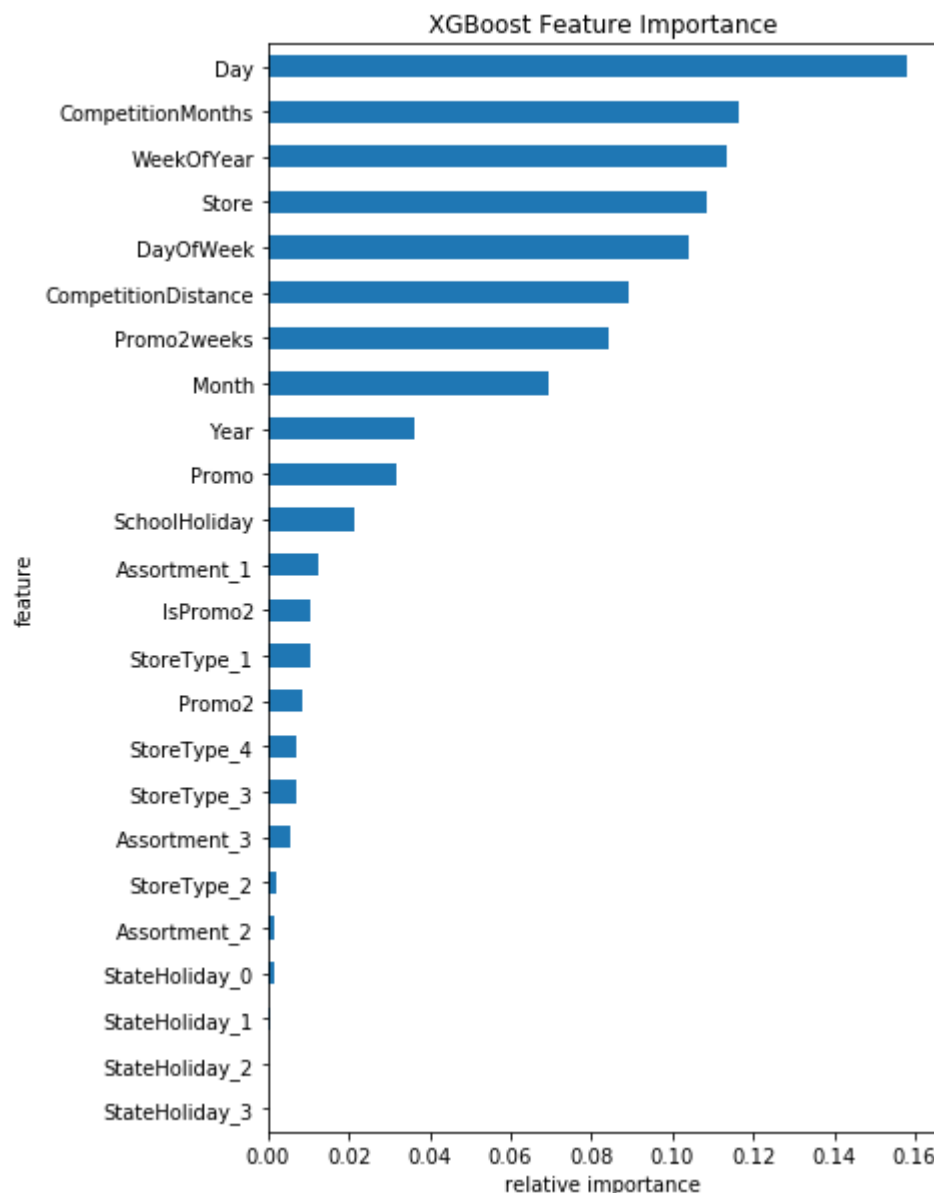
The lower cost of the Ridge and Lasso models is mainly due to the short training time and the inability to fit the complex input data. The punishment method also makes it difficult to calibrate. According to the input characteristics of the project, it is about 0.4-0.45. The result is not unexpected.

XGBoost training time is relatively long, especially the depth increases, the number of trees increases, but the model's prediction performance is very good, has met the initial goal of 0.2, after initial setup and parameter optimization reached 0.11572, which It is inseparable from its own characteristics. It adds a regular term to the cost function to control the complexity. As a tree model, it can also reduce the branch from the bottom to the top to avoid falling into local optimum, so it can be compared. Take this score easily.

V. Conclusion

Free-Form Visualization

Demonstrate the ordering of feature importance during Xgboost training.



Comment on the top five relevance features:

Time is the first important feature, because for example, holidays, competitors' opening hours, school holidays, and economic development are all related to time, so it is highly correlated with sales.

The situation of competitors is also a very important feature, after all, it will lead to the diversion of customers affecting sales.

The store's different expected results are also different, so the store is also very important.

Weekofyear and Dayofweek are also associated with time, so the correlation is high and normal.

Reflection

The project should be divided into three parts: data sorting, model selection, evaluation and optimization. The latter two parts can be combined and influenced each other.

The most important thing is in data sorting. This is also the most interesting and difficult place. You need to understand the background of the problem and the meaning of each feature. Add the missing values according to the actual situation and common sense of each feature, and then find out which ones are close. The characteristics of the actual connection are transformed into new features that are more intuitive and more suitable for the algorithm model.

In terms of model selection, the project is a regression problem of supervised learning. The mainstream supervised learning regression algorithm is applicable. Try to use multiple regression models by setting a benchmark score to compare and select the best performance. Model training time is costly when modeling, but modeling costs do not affect modeling when it is successfully used for forecasting.

The evaluation and optimization of the model is an eternal theme. The time of training modeling restricts the number of attempts to train, and the best parameters can be found by cv. But the best optimization is in the first step of data collation. The data collation of a project determines the upper limit of the model. Even the best model and optimization are just the upper limit of infinite approximation.

Improvement

In the data collation analysis stage, I can further try to deal with the feature of CompetitionDistance, discretize it in segments, and better regularize the model. The feature of CompetitionMonths can also consider segmentation discretization, and they are all ranked by importance. The characteristics of the former, in-depth research should have a good upgrade.