January 2022

Predicting Future Sales

CHARLES FRANZNICK

Charles Franznick
cfranznick@gmail.com
https://www.linkedin.com/in/charles-franznick/

Introduction

Why?

Predicting sales data is fundamentally helpful to any business. Knowing what to expect for monthly revenue, inventory, and other features helps the business plan, prepare, and meet customers' needs and expectations.

Audience

This specific project was built using data provided by one of the largest Russian software firms - 1C Company. All predictions, locations, and items are therefore going to be directly applicable only to 1C Company. While this project's target audience is the 1C Company, the methodology used is applicable to any business. Additionally, insights from this project can be very helpful for other future sales prediction projects.

Data Source and Scoring

The data comes from a Kaggle competition titled, "Predict Future Sales" and is intended for use in a Coursera course about this data specifically.

The data has daily sales data from January 2013 through October 2015; the goal is to predict the sales for November 2015. The source data has 22, 170 unique items and 60 unique shops, resulting in a total of 214,200 item-shop pairings to predict. Other data include item categories, shop names, and item prices.

The competition has a hidden test set to score against, which I will use as a score for the quality of my prediction. I also validated my methods by using data from January 2013 through September 2015 as training data, using the real, known values from October 2015 as my target to predict, then scoring the real data against my prediction using a number of prediction statistics.

Data

About the data

This data is daily sales records provided by 1C Company. Some of the data include canceled transactions and returns, which can complicate summary statistics and add challenges to aggregating the data from daily into the monthly prediction we need. The list of shops and products slightly changes every month; part of the challenge is to build a model that can withstand those changes.

There were 5 csv files that provided data and 1 sample submission csv for output. Taken from the Kaggle competition:

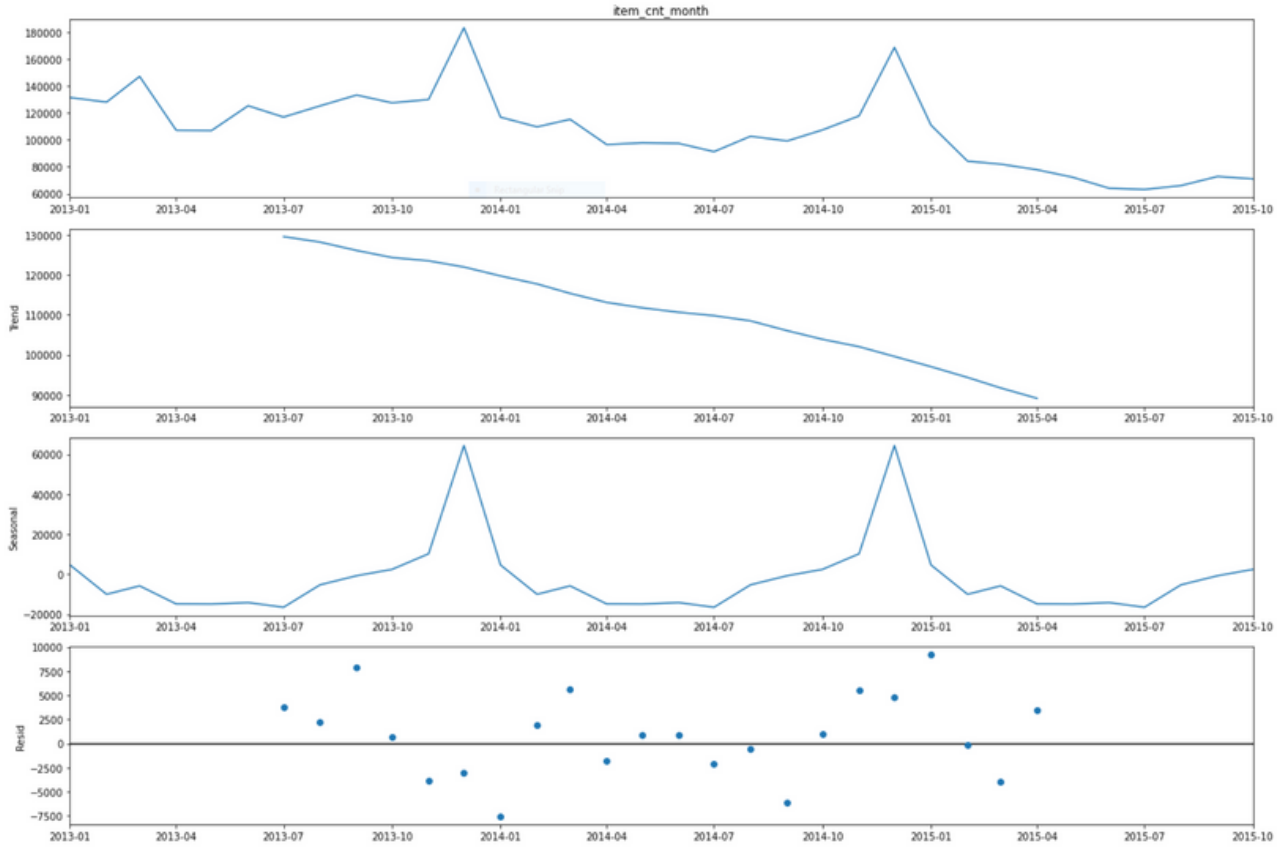
- sales_train.csv - the training set. Daily historical data from January 2013 to October 2015.
- test.csv - the test set. You need to forecast the sales for these shops and products for November 2015.
- items.csv - supplemental information about the items/products.
- item_categories.csv - supplemental information about the items categories.
- shops.csv- supplemental information about the shops.
- sample_submission.csv - a sample submission file in the correct format.

Cleaning the data

Cleaning the data involved compensating for any null values present, making sure the data types were all appropriate for analysis, checking for values that are present in the sales_train data that are not in the other data, and getting rid of any duplicated values.

Exploritory Data Analysis

In the EDA report, I was able to find features and trends that benefit the final prediction algorithm.



Findings

There is a clear trend to the is daily sales records. The top chart shows the daily sales data; the second graph shows the general trend line of sales over time. The third graph shows the seasonality of the data, i.e. when sales are changing predictably every calendar year.

Analyzing the store and item sales numbers, I found that stores located in Moscow or online retailers sold best. The most popular items were AAA video games, and the most popular category was games. Since the games were spread across many categories, this led me to make groups as a feature later on.

Feature Engineering

I developed 28 features for this data.

The data was provided in a daily format, meaning any monthly insights I wanted had to be calculated. I found the number of holidays per month, the number of weekend days per month, and the monthly revenue by first finding the info per day, then aggregating the data into monthly data.

I also calculated the following information:

- Days per month - how many days the month had
- Group - larger categories, i.e. merging video games for all systems into a Games category
- First month sold - identifying what month the item first sold, if any
- Number of months since last sale - how long it has been since the item sold

Lag Features

Finally, I added lag features. Lag features track data from a prior time. I made a function to calculate the following for 1, 2, 3, and 12 months prior:

- Item average sales
- Shop average sales
- Item category sales
- Item average sales per shop
- Item category sales per shop

Using lag features allows us to get very helpful features; however, it does mean that we do not have those features for the first months. Since I looked up to 12 months prior, I lose those features for the first 12 months. Since I have enough data, I am comfortable simply not using the first 12 months of data.

Modeling

Models

Model	Scaled	Unscaled
Linear Regression	0.964	0.964
LASSO	1.182	1.060
ElasticNet	1.182	1.027
CART	0.080	0.0753
XGB	0.273	0.273

Lower score is better

I initially tested 5 different machine learning algorithms before fine-tuning the 2 best performers: Decision Tree Regression (CART) and Extreme Gradient Boosting Regression (XGB). These two algorithms are both based on decision trees; the former is a single tree, while the latter takes multiple, weaker decision trees and combines them together for the final output.

Scaling the data resulted in no change or a slightly worse score for all algorithms and increased the time the algorithms took, so deciding against scaling the data was an easy decision.

Training

I used cross-validation to train these models. Traditional cross-validation does not work on time-series data, so I used a time-series split, ensuring we don't use future data to determine past sales.

Tuning and Predictions

Tuning

The two algorithms have two different hyperparameters to tune. I used a grid search to iterate over 3-4 values per hyperparameter. Using cross-validation, I found that the best hyperparameters were as follows:

- CART: max_depth: 6, max_features: 0.8, min_samples_leaf: 0.04
- XGB: max_depth: 4, max_features: log2, min_samples_leaf: 0.01, n_estimators: 200

Accuracy

I used RMSE as the accuracy metric for two reasons: first, the errors are squared before they are averaged which is desirable when large errors are particularly undesirable. Second, the final scoring metric on the Kaggle competition uses RMSE, so this would allow me to compare my algorithm directly with others who submit to the competition. The best scores I got for the training data were as follows (lower is better):

- CART: 0.890
- XGB: 0.306

Based on these scores, I would choose the XGB model.

Predictions

I made predictions with both models and made a third prediction that averaged the two sale prediction numbers. The final Kaggle competition scores were as follows (lower is better):

- CART: 1.25011
- XGB: 1.24991
- Averaged: 1.24313