Final Project: M5 Forecasting - Accuracy

Xuzheng Lu
G34363475

1 Problem Introduction

1.1 Topic (competition) selected

Kaggle competition: M5 Forecasting - Accuracy
Link: https://www.kaggle.com/c/m5-forecasting-accuracy/

1.2 Background and application introduce

In this competition, we can use hierarchical sales data from Walmart, the world's largest company by revenue, to forecast daily sales for the next 28 days. If successful, the methods used can be applied in various business areas, such as setting up appropriate inventory or service levels.

1.3 Dataset description

The dataset covers stores in three US States (California, Texas, and Wisconsin) and includes item level, department, product categories, and store details. In addition, it has explanatory variables such as price, promotions, day of the week, and special events.

1.4 Evaluation method

The accuracy of the point forecasts will be evaluated using the **Root Mean Squared Scaled Error** (**RMSSE**), which is a variant of the well-known Mean Absolute Scaled Error (MASE) proposed by Hyndman and Koehler (2006). The measure is calculated for each series as follows:

$$\boldsymbol{RMSSE} = \sqrt{\frac{1}{h}\frac{\sum_{t=n+1}^{n+h}\left(Y_t - \widehat{Y_t}\right)^2}{\frac{1}{n-1}\sum_{t=2}^{n}(Y_t - Y_{t-1})^2}},$$

where Y_t is the actual future value of the examined time series at point *t*, $\widehat{Y_t}$ the generated forecast, *n* the length of the training sample (number of historical observations), and *h* the forecasting horizon.

2 Files description

```
m5-forecasting-accuracy/
├── M5-Competitors-Guide.docx        // the guidelines the of competition
├── doc.docx                         // this documentation
├── README.md                        // README file
├── data/                            // data folder
│   ├── calendar.csv
│   ├── sales_train_validation.csv
│   ├── sample_submission.csv
│   └── sell_prices.csv
└── src/                             // source code folder
    ├── eda_notebook.ipynb           // jupyter notebook of EDA
    ├── xgboost_notebook.ipynb       // jupyter notebook of XGBoost
    ├── lstm_notebook.ipynb          // jupyter notebook of LSTM
    ├── xgboost_main.py              // python script of XGBoost
    ├── lstm_main.py                 // python script of LSTM
    ├── eval_func.py                 // evaluation functions
    ├── utils.py                     // utilities functions
```

```
├── cv_results.csv          // results of Grid Search
└── future/                 // codes of future plans (code from other projects of mine)
```

3 Exploratory Data Analysis

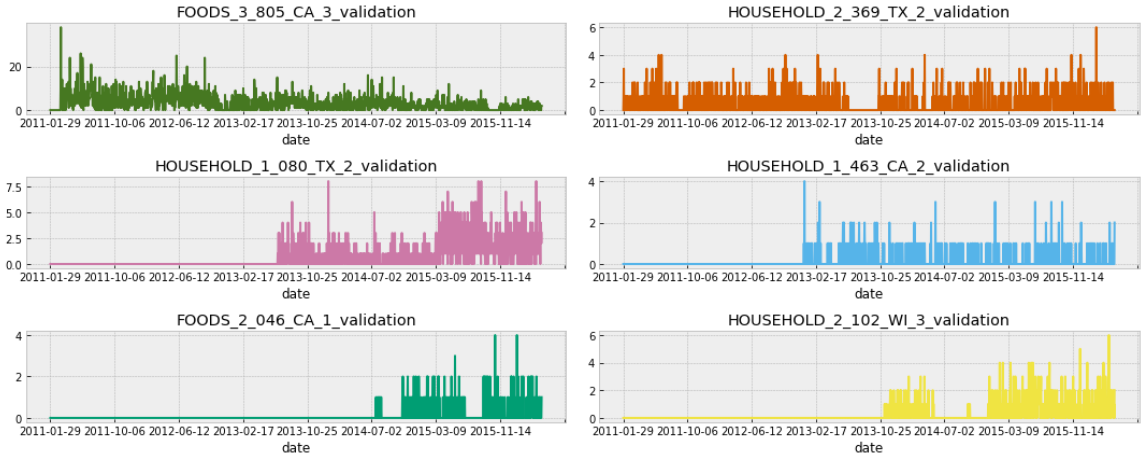
To have a quick overview of the competition and understand the dataset, I did the EDA (Exploratory Data Analysis) at first.

3.1 Get a glance at dataset files

1) **calendar.csv** - Contains information about the dates on which the products are sold.
2) **sales_train_validation.csv** - Contains the historical daily unit sales data per product and store [d_1 - d_1913]
3) **sample_submission.csv** - The correct format for submissions. Reference the Evaluation tab for more info.
4) **sell_prices.csv** - Contains information about the price of the products sold per store and date.

3.2 Data for different items

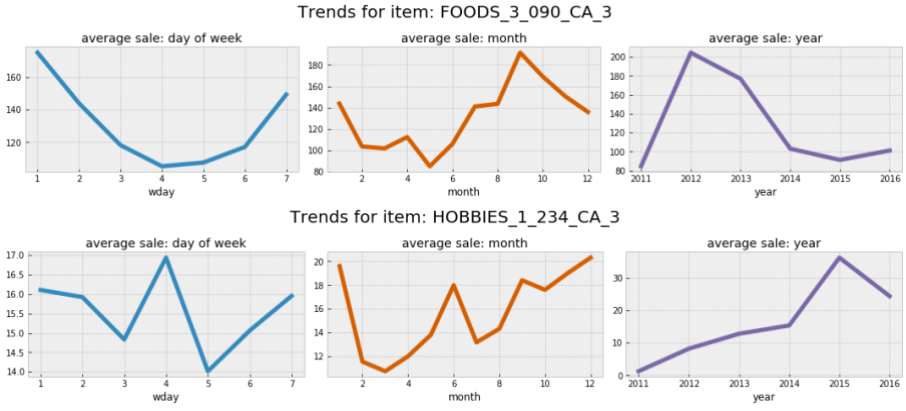
The following figures show data of different items.



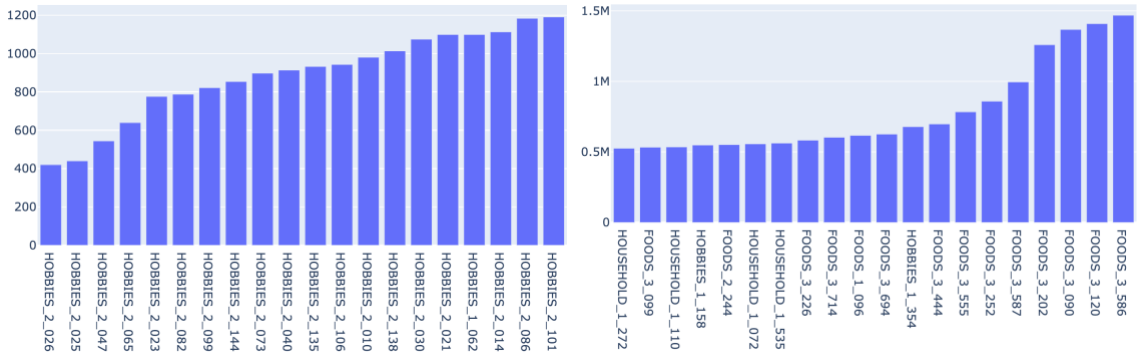
Observations:

- There are unavailable periods for some items, and it is normal.
- The start time of different items may be different.
- The sales increased at some certain time point, and that may be caused by events or sales.

The following figures show data of different items on two different time scales.

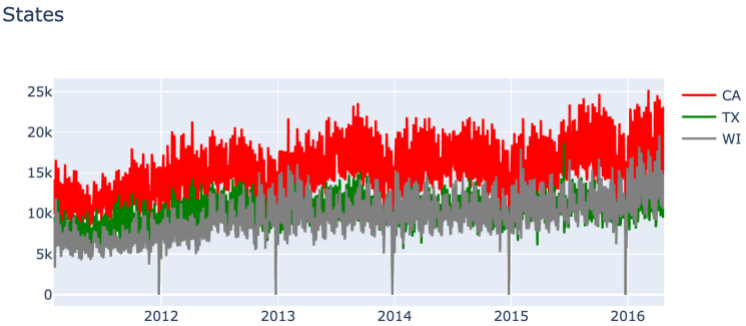


The following figure shows the sales distribution of the bottom and top 20 items:



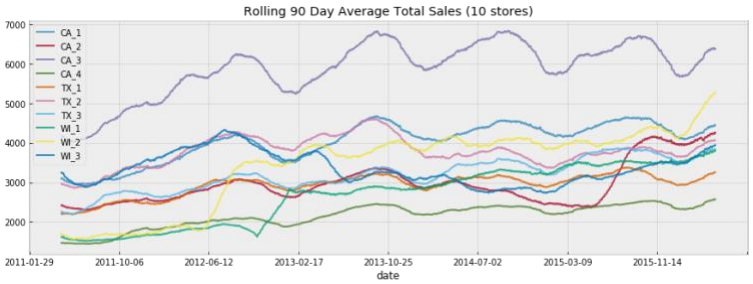
3.3 Relationship between sales and stores

The following figure shows the sales grouped by states.



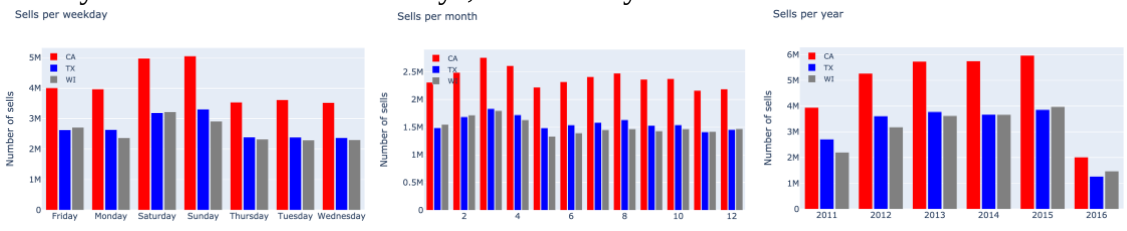
As shown above, stores in California sales more than the other two states.

Rolling 90 Day Average Total Sales on 10 different stores:



3.4 Relationship between sales and time

Sales by states on different weekdays, moths and years

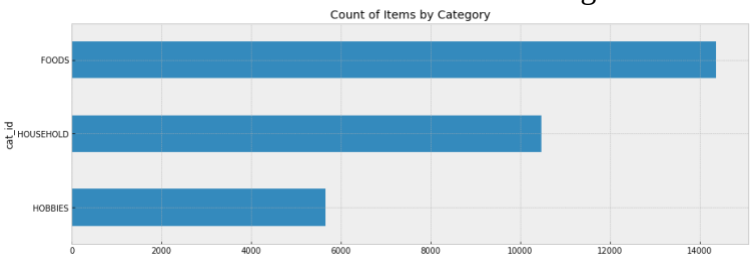


Observations

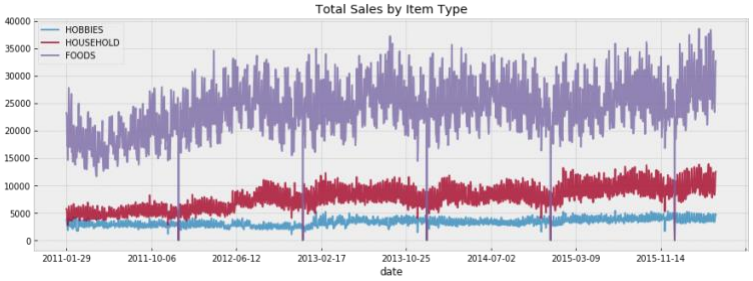
- The sales increased significantly on weekend days.
- The sales increased around March and August.
- There is a fall of sales in 2016.

3.5 Relationship between sales and categories

The following figure shows the number of items of different categories.



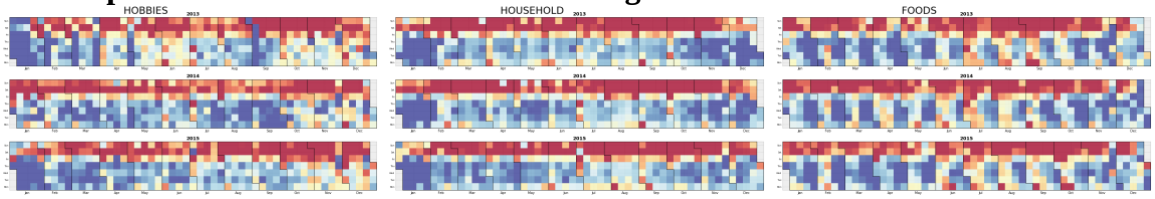
The following figure shows the total sales of different categories.



Observation:

- Categories type ordered by number of items are:
  Hobbies < Household < Foods
- Categories type ordered by sales are:
  Hobbies < Household < Foods

3.6 Heatmap of sales on calendar of different categories



Observations:

- Weekends have more sales.
- Household and hobby items are not popular in January.

4 Modeling and training.

I used two models – XGBoost and LSTM.

4.1 XGBoost

1) Introduction

XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable. It implements machine learning algorithms under the Gradient Boosting framework. XGBoost provides a parallel tree boosting (also known as GBDT, GBM) that solve many data science problems in a fast and accurate way. The same code runs on major distributed environment (Hadoop, SGE, MPI) and can solve problems beyond billions of examples.

2) Modeling

I used the default parameter to train the model.

3) Training

I used RMSE as the objective function, and set the early stopping rounds to 10.

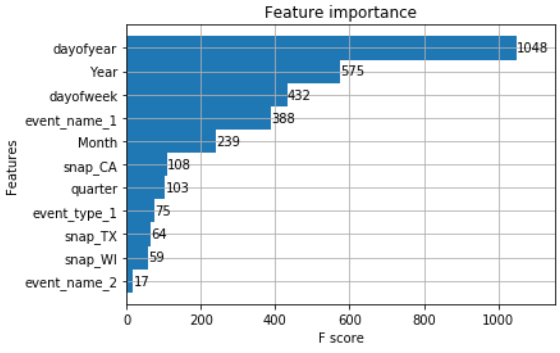
4) Evaluation

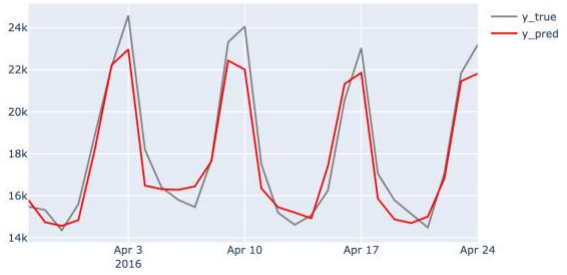
- **California**
  - Train Error: 872.0211 RMSE
  - Error: 1182.8644 RMSE
- **Texas**
  - Train Error: 757.1770 RMSE
  - Valid Error: 652.5682 RMSE
- **Wisconsin**
  - Train Error: 813.6269 RMSE
  - Valid Error: 1369.8346 RMSE

5) Parameter tuning

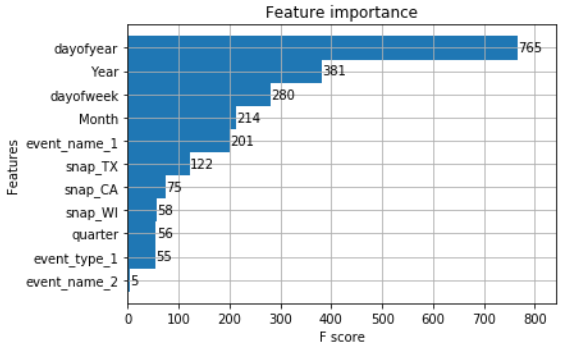
I used Grid Search with 10-fold cross-validation for model selection and parameter tuning. The results for grid searching are as follows:

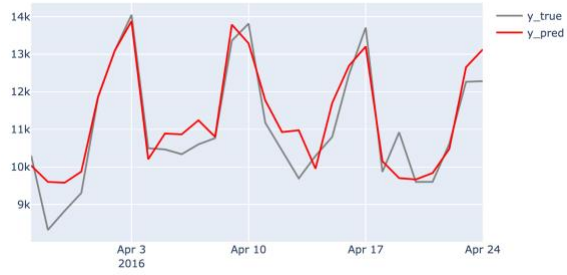
- **California**
  - Best parameters: {'learning_rate': 0.1, 'max_depth': 5, 'n_estimators': 200}
  - Train Error: 590.1588 RMSE
  - Valid Error: 1045.6394 RMSE
  - Fearure importance and Prediction:



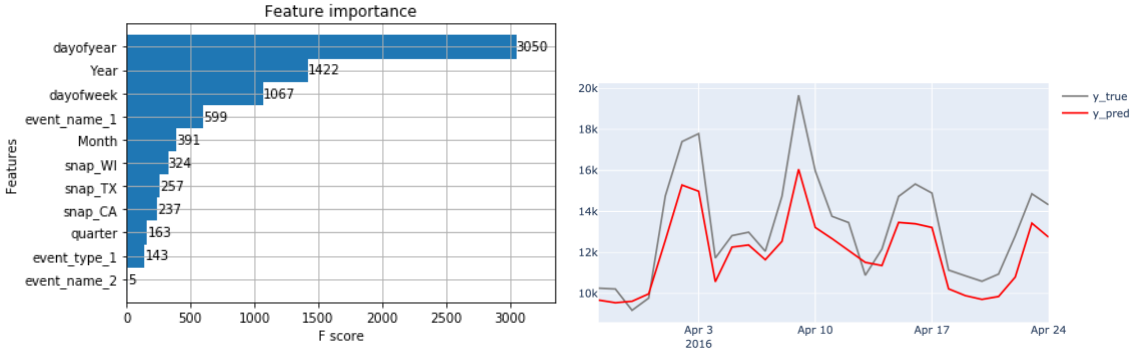


- **Texas**
  - Best parameters: {'learning_rate': 0.1, 'max_depth': 5, 'n_estimators': 200}
  - Train Error: 757.1770 RMSE
  - Valid Error: 652.5682 RMSE
  - Fearure importance:





- **Wisconsin**
    - Best parameters: {'learning_rate': 0.05, 'max_depth': 5, 'n_estimators': 500}
    - Train Error: 517.0896 RMSE
    - Valid Error: 1545.4203 RMSE
    - Fearure importance and Prediction:



The results of every grid-search attempt are stored in 'cv_results.csv'.

4.2 LSTM

1) Introduction

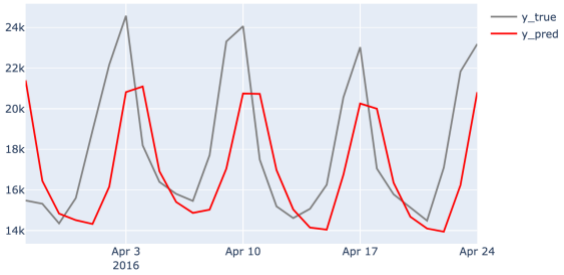
Long short-term memory (LSTM) is an artificial recurrent neural network (RNN) architecture used in the field of deep learning. Unlike standard feedforward neural networks, LSTM has feedback connections. It can not only process single data points (such as images), but also entire sequences of data (such as speech or video). For example, LSTM is applicable to tasks such as unsegmented, connected handwriting recognition, speech recognitionand anomaly detection in network traffic or IDS's (intrusion detection systems).

2) Modeling

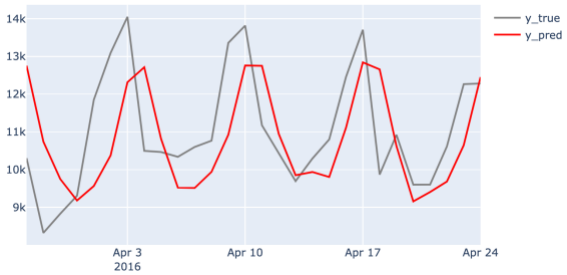
I uesd Keras building the model.

3) Evaluation

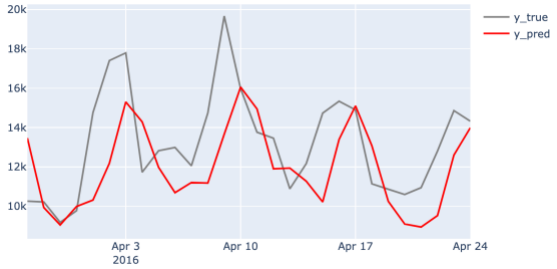
- **California**
    - Train Error: 1473.67 RMSE
    - Valid Error: 1621.02 RMSE
    - Prediction:



- **Texas**
    - Train Error: 1039.18 RMSE
    - Valid Error: 861.66 RMSE
    - Prediction:



- **Wisconsin**
  - Train Error: 1194.97 RMSE
  - Valid Error: 1777.53 RMSE
  - Prediction:



5 Conclusion

This is my first time dealing with time-series tasks. It looks like regression problems but very different in detail. I tried XGBoost, the most popular model used in data science competitions, and LSTM, a model base on Deep Learning technology.

I also spent tons of time on EDA (Exploratory Data Analysis), which let me understand the dataset thoroughly. In the end, XGBoost got a better result than LSTM, and I think that's because the architecture and hyper-parameters of LSTM are not good enough, and I did a lot of work on the parameter tuning process of XGBoost.

I will continue this project this summer and try to get a good place in the competition, so there are still tons of works I need to do.

6 Future plans

In the future, I will try to do the following approaches to reduce the loss further and try to get a better score in this competition.

6.1 Feature engineering

The first thing I can try is to add more features such as rolling time features and moving average features. I can also use clustering algorithms, such as t-sne and PCA to generate more features.The second is to reduce the number of features. I can use the feature importances to select the features to impact the model most and delete the others.

6.2 Advanced parameter tuning

The grid search and model selection process I implemented in this project is too simple, because I just tried a few parameters combinations. In the future, I will implement a more advanced parameter tuning architecture.

6.3 Try more models

In this project, my main work is doing the EDA and just tried two models. I will more models and algorithms.Also, I will try more time series models, such as HMM and ARIMA.

The models I will use are as follows

Simple models	Ensemble models	Time series models	Deep learning models
Linear Regression	Random Forest	Moving Average	DNN
Logistic Regression	GBM	Exponential	RNN/LSTM
KNN	GBDT/GBRT	Smoothing	1-Dimensional CNN
Decision Tree	AdaBoost	Markov Model	
Extra Tree	XGBoost	HMM	
SVM	LightGBM	ARIMA	

6.4 Ensemble

The ensemble is a technology to increase the stability and generalization of the model. In the future, I will use technologies such as voting, blending, bagging, and staking, to improve the accuracy.

7 References

[1] Chen, Tianqi, and Carlos Guestrin. "Xgboost: A scalable tree boosting system." Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. 2016.
[2] Ke, Guolin, et al. "Lightgbm: A highly efficient gradient boosting decision tree." Advances in neural information processing systems. 2017.
[3] Tang, Jian, et al. "Visualizing large-scale and high-dimensional data." Proceedings of the 25th international conference on world wide web. 2016.
[4] Wang, Liwei, et al. "On the margin explanation of boosting algorithms." COLT. 2008.
[5] A Tour of Machine Learning Algorithms (https://machinelearningmastery.com/a-tour-of-machine-learning-algorithms/)
[6] Tree Based Algorithms: A Complete Tutorial from Scratch (https://www.analyticsvidhya.com/blog/2016/04/tree-based-algorithms-complete-tutorial-scratch-in-python/)
[7] https://www.kaggle.com/tarunpaparaju/m5-competition-eda-models
[8] https://www.kaggle.com/robikscube/m5-forecasting-starter-data-exploration
[9] https://xgboost.readthedocs.io/en/latest/
[10] https://www.wikiwand.com/en/Long_short-term_memory