Grocery Sales Forecasting with Embedding and Residual Networks

Alexia Xu
alexiaxu@stanford.edu

OVERVIEW

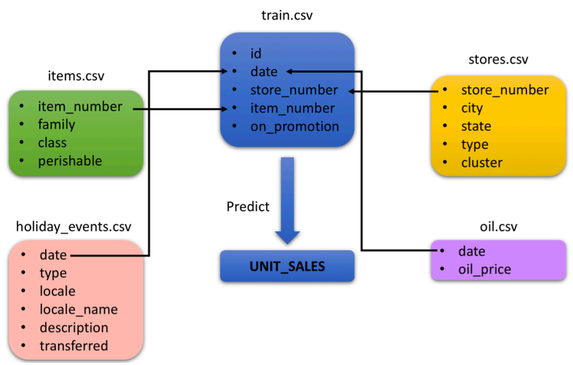
Motivation: Grocery stores rely heavily on accurate sales forecasting. Overestimation of the sales makes them stuck with overstocked, perishable goods, and underestimation leaves money on the table and customers fuming. In this project, we aim to predicting the unit sales amount of each item in stores based on historical data, item/store information, oil price, etc.

Models: We employed linear regression as the baseline, and build an XGBoost model and ResNet based DNN with embedded features.

Results: Our best model has a weighted MSE dev loss of 0.28, comparing to the 0.85 of the baseline without embedding and the 0.67 of linear regression with embedding.

DATA & FEATURES

Our data come from the on-going Kaggle competition: Corporacion Favorita Grocery Sales Forcasting (https://www.kaggle.com/c/favorita-grocery-sales-forecasting). The structure of the dataset is shown below:



LOSS FUNCTION

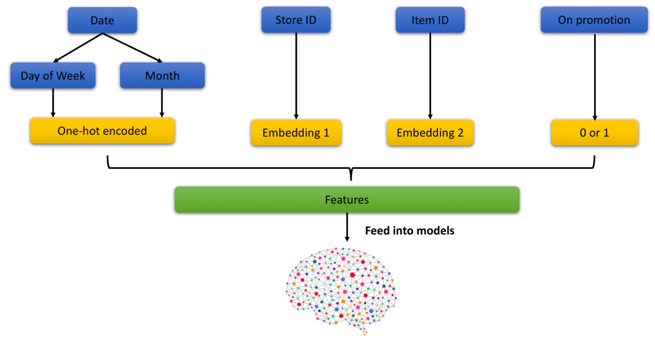
According to the requirement of the competition, we use the Normalized Weighted Mean Squared Logarithmic Error (NWMSLE) as the loss function, which is defined as follows:

$$J = \frac{\sum_{i=1}^{m} w_i \times (\log(\hat{y_i} + 1) - \log(y_i + 1))^2}{\sum_{i=1}^{m} w_i}$$

where $w_i = 1.25$ if item i is perishable, else $w_i = 1.0$

FEATURE EMBEDDING

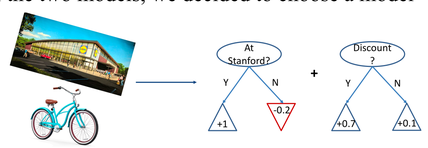
In analysis of the dataset, we noticed that store ids and item ids are categorical features. To handle the situation, we took inspiration from word embedding in natural language processing. We treated the store ids and the items ids as indices in two vocabularies, and trained a vector representation for each index (as shown below).

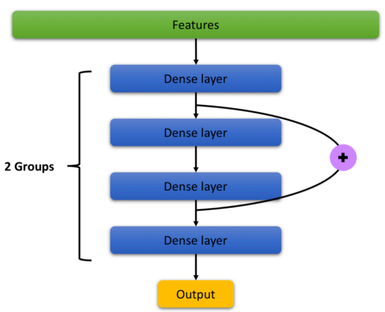


MODEL: RESIDUAL NETS

Based on the baseline, we first trained trained an XGBoost[1] Model and a simple NN model. Observing the underfitting of the two models, we decided to choose a model with a large hypothesis space

$$\text{obj} = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t)}) + \sum_{i=1}^{t} \Omega(f_i)$$





We also trained a simple neutral nets (NNs) and a simple residual nets [2,3] (as shown). The reason we chose the residual nets is not only because its great performance in computer vision, but also its resistance to the vanishing gradient problem as the nets get deeper. Since the original residual nets are convolutional neural nets (CNNs), we developed a variant by taking the idea of residuals.

Reference:
Tianqi Chen *et al.* XGBoost: A Scalable Tree Boosting System. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016
Kaiming He *et al.* Deep Residual Learning for Image Recognition. CVPR, 2015
Kaiming He *et al.* Identity Mappings in Deep Residual Networks. ECCV, 2016

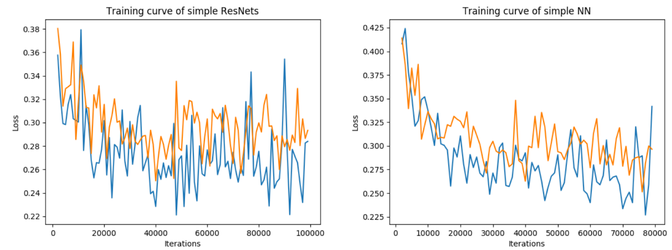
RESULTS

We have trained four models: linear regression, XGBoost, a simple NN, and a simple residual nets. We used the sales data from 01/01/2017 to 07/31/2017 as the training set and the data from 08/01/2017 to 08/16/2017 as the development set. There are 22,237,293 training examples and 1,570,968 development examples. We split the datasets in temporal order because the dev set needs to evaluate the performance of the model on predicting future sales.

Table 1. Performance of the models

Model	Train Error	Best Dev Error
Linear Regression	0.65	0.67
XGBoost	0.47	0.53
Simple NN	0.29	0.32
Simple ResNet	0.25	0.28

The training curves of the simple NN and the Residual NN are shown below



DISCUSSION AND FUTURE

Summary: Currently, we have built four models to predict the sales unit amount of each item in grocery stores. The residual nets model presented the best performance, with a dev loss of 0.28 (comparing to the 0.67 of the baseline model)

Discussion: Possible solutions for long term sequential data
In the analysis of the models, we realized that it seems difficult for the model to describe long term sequential data. We can think of the following solutions:

A) Using RNNs to capture the sequential info. Disadvantage: Sequences longer than 40 will face with the vanishing gradient problem
B) Using CNN to model sequential data. We could stack the sales in temporal order, with each day's sales as a channel (3rd dimension of the training data)
C) Taking inspiration from the "teacher forcing" mechanism in NLP, we may use the previous 15 days of sales as features directly

Future: I will try to implement the "teacher forcing" method (C in the discussion) hopefully before the end of the quarter. In long term, I hope to try LSTM to predict 15 days of sales sequentially with earlier sales as input in each time step