

Instacart customer's next basket prediction

Capstone 3 Project Summary

Robert Ciesielski

Mentor: Dipanjan Sarkar

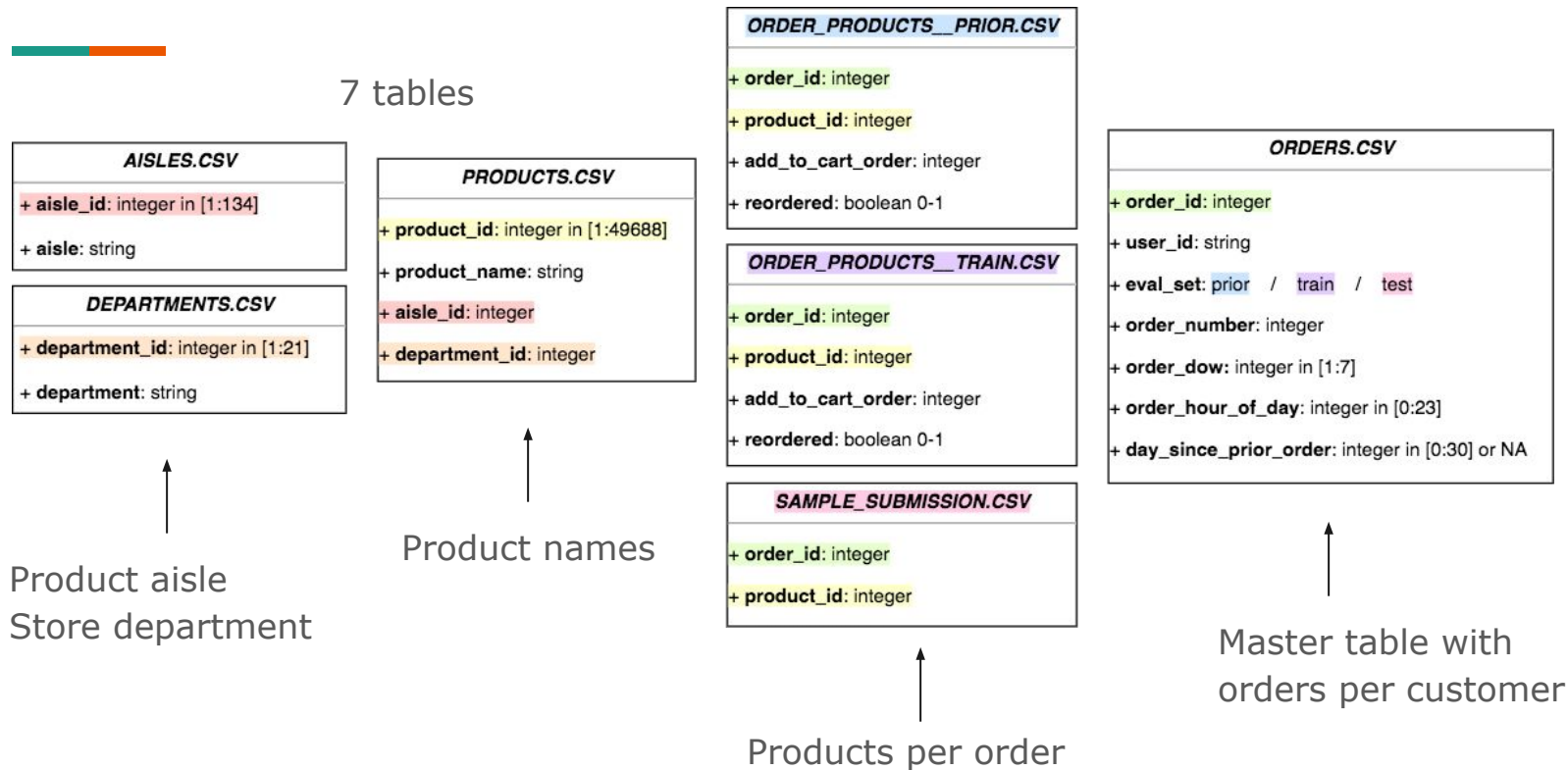
September 2021

Introduction

- In 2017, Instacart, an online grocery shopping company, released to the Kaggle community a dataset with ~3 million grocery orders of ~200,000 customers.
- Kaggle challenge: Predict which previously purchased products will be present in a customer's next order.
- The most common Kaggle solutions: customer-product pair classification using gradient boosting models (XGBoost, LighGBM, etc.) and a few most recent customer's orders.
- Deep Learning approaches not so popular - only two documented analyses.
- **Our goal: predict customer's next basket using Recurrent Neural Network (RNN) models convoluted with Dense Dayer (DL) classifier.**

RNN = Gated Recurrent Unit (GRU) or Long Short-Term Memory (LSTM) cell

Dataset

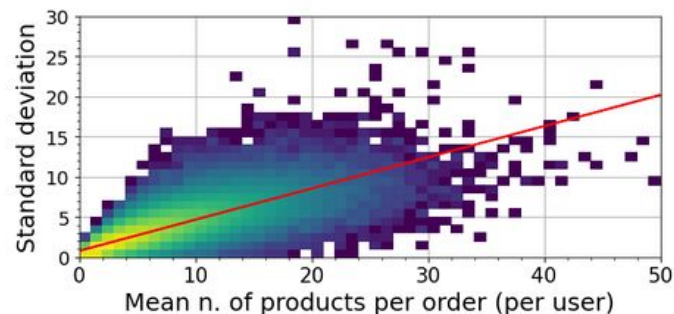
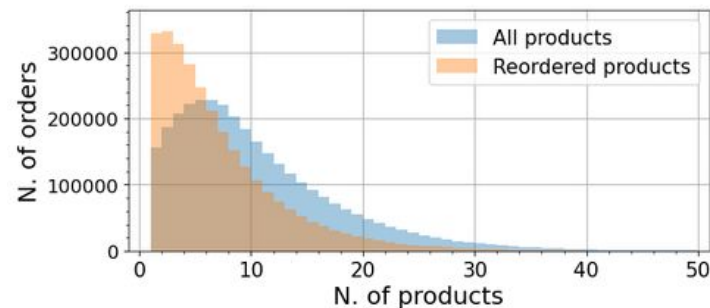
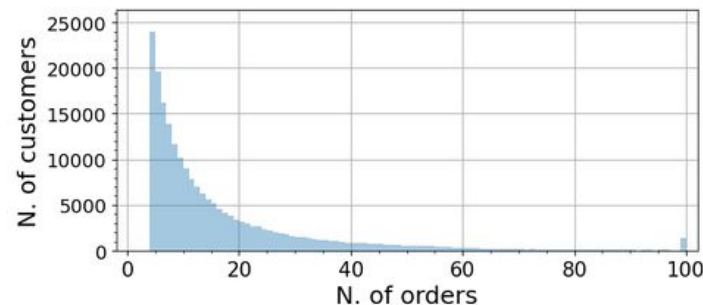


Fields with the same colors show links between tables

Exploratory Data Analysis (1)

- Number of orders per customer
- Number of products and reordered products per order
- How number of products changes between customer's orders

2-dim histogram: Std vs Mean

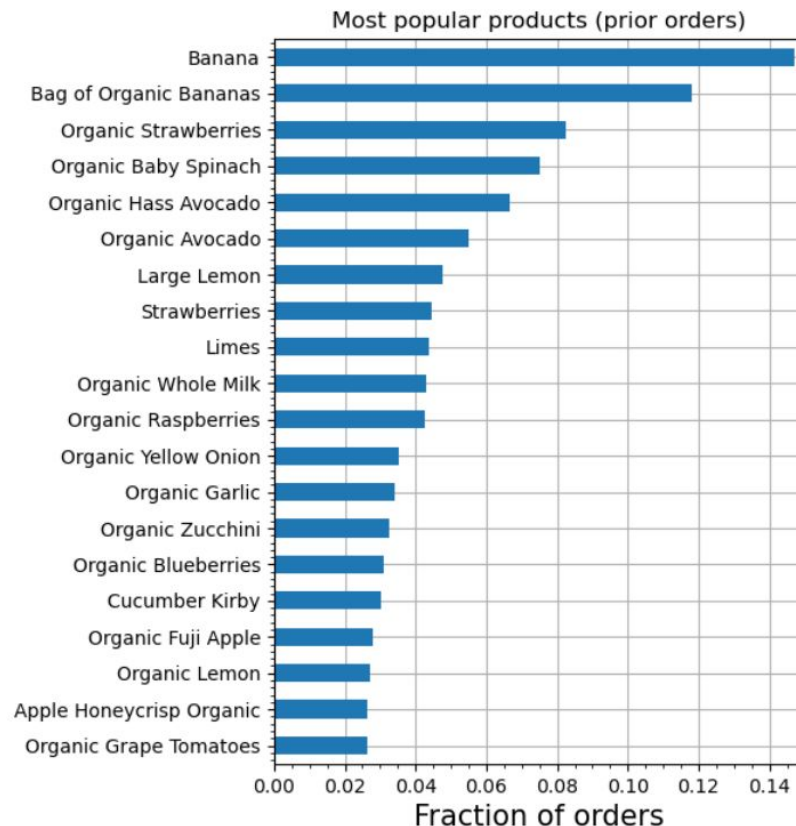


Exploratory Data Analysis (2)


- What are the most popular products?

Bananas!

- Bananas and organic bananas are present in 25% of orders
- 14 out of 20 most frequently both products are organic
- Generally, healthy-diet fruits and vegetables are most popular



Model evaluation metrics, mean f1 score



A suggested metric for the model evaluation is the mean f1 score, defined as an arythmetic average of the individual user f1 scores. Each user's f1 score is calculated as a harmonic mean of precision, P, and recall, R, using the standard formula:

$$f_1 = \frac{2}{P^{-1} + R^{-1}}. \quad (1)$$

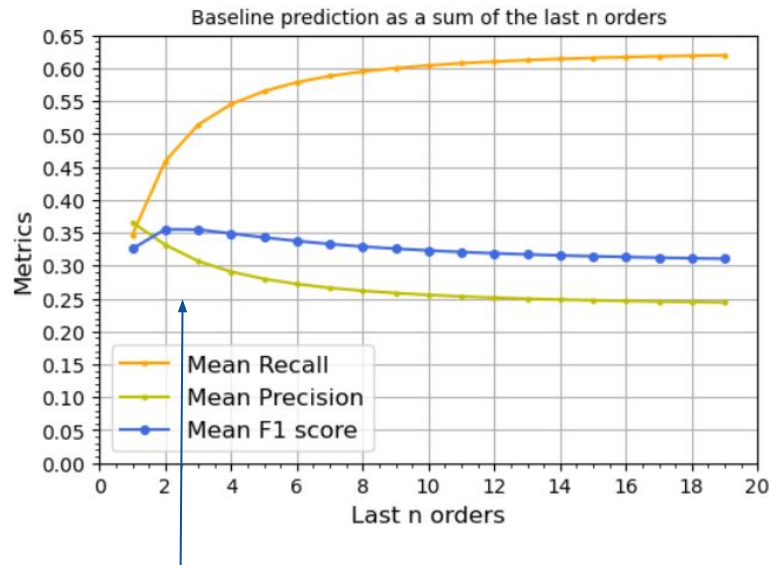
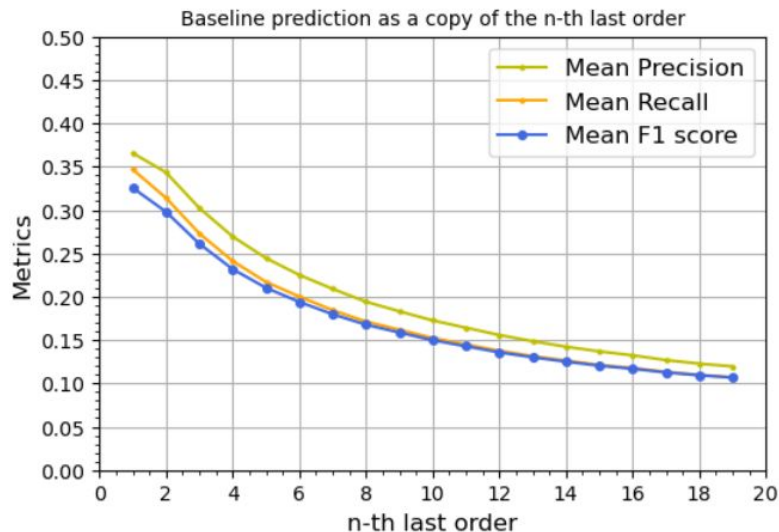
Given two lists of products, a true and a predicted one, P and R are defined as:

$$P = \frac{N^{PT}}{N^P}, \quad R = \frac{N^{PT}}{N^T} \quad (2)$$

where N^P and N^T are the number of products in the predicted and true lists, respectively, and N^{PT} is the number of common products in both lists. For two empty lists $P = R = f_1 = 1$, if one of the two lists is empty $P = R = f_1 = 0$, while non-empty lists receive $P, R, f_1 \in [0, 1]$

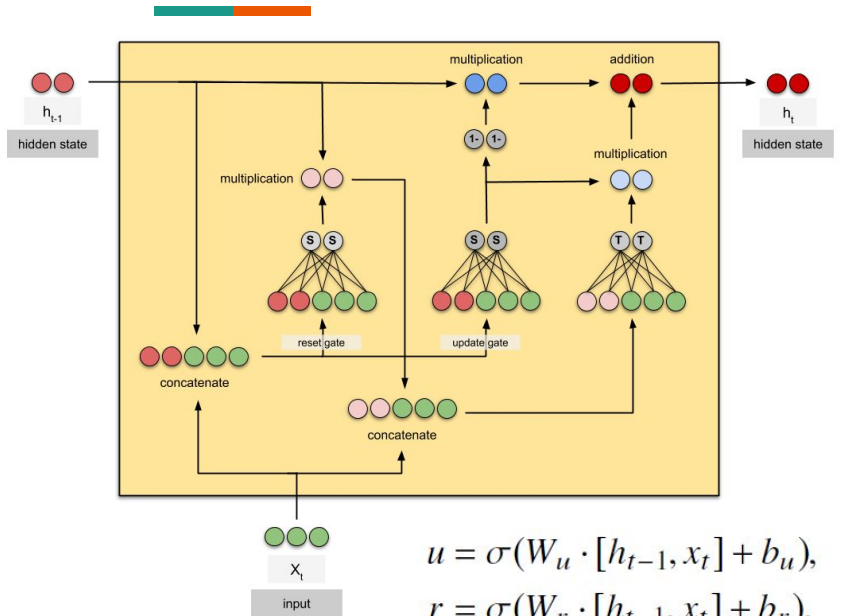
Baseline model

No ML involved. Copy customer's previous orders.



Copy of customer's last two orders gives the highest mean f1 score of 0355. It's our baseline model.

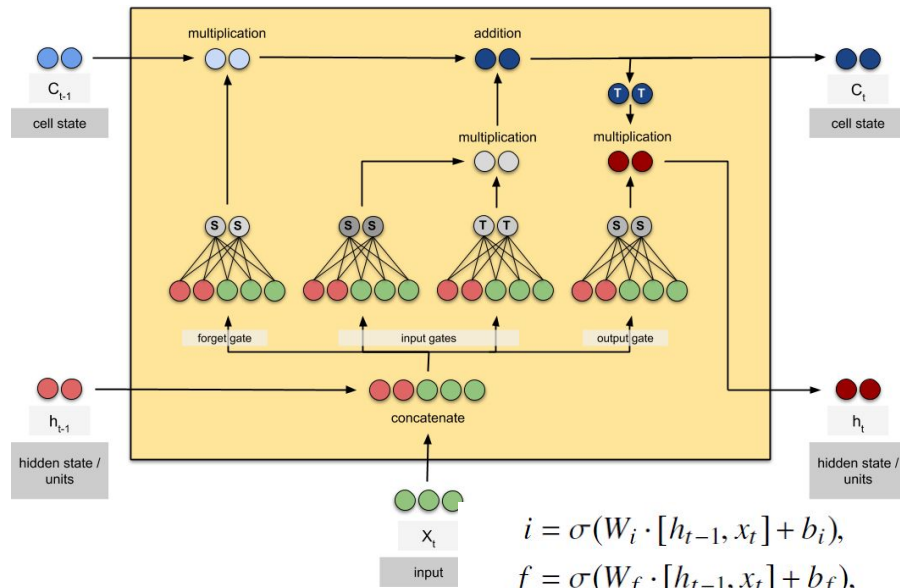
RNN models



GRU

(Gated Recurrent Unit)

$$\begin{aligned}
 u &= \sigma(W_u \cdot [h_{t-1}, x_t] + b_u), \\
 r &= \sigma(W_r \cdot [h_{t-1}, x_t] + b_r), \\
 \tilde{h}_t &= \tanh(W_h \cdot [r \cdot h_{t-1}, x_t] + b_h), \\
 h_t &= u \cdot h_{t-1} + (1 - u) \cdot \tilde{h}_t.
 \end{aligned}$$



LSTM:

(Long Short-Term Memory)

$$\begin{aligned}
 i &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i), \\
 f &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f), \\
 o &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o), \\
 \tilde{h}_t &= \tanh(W_h \cdot [h_{t-1}, x_t] + b_h), \\
 C_t &= i \circ \tilde{C}_t + f \circ C_{t-1}, \\
 h_t &= o \cdot \tanh(C_t).
 \end{aligned}$$

Modeling - two approaches

1) One model for all customers

- Data preprocessing:

$$X = [u, \text{orders}_{20}, \text{OHE}_{all}],$$

$$y = [u, \text{OHE}_{all}^P],$$

- Features: one-hot encoded product IDs

OHE_{all} - 42,730 unique products!!!

Output: OHE^P - product probabilities of being included in next order

- Train/test split:

Standard 75/25 parallel split

2) Many models, per customer

$$X = [1, \text{orders}_u, \text{OHE}_u],$$

$$y = [1, \text{OHE}_u^P],$$

OHE_u - less than 150 products per user

Sliding window method (see next slide)

Modeling (2)

Example data format for one customer model

Products
Orders



196 10258 12427 13032 13176 25133 26088 26405 38928 39657 46149 49235

order_number													
Train	2	1	0	1	0	0	0	1	0	0	0	0	0
	3	1	1	1	0	0	0	0	0	0	0	0	0
	4	1	1	1	0	0	1	0	1	0	0	0	0
	5	1	1	1	0	1	1	0	0	0	0	0	0
	6	1	1	1	0	0	1	0	0	0	0	0	0
	7	1	1	1	1	0	1	0	0	0	0	0	0
	8	1	1	1	0	0	1	0	0	0	0	0	0
	9	1	1	1	0	0	1	0	0	0	0	1	1
	10	1	1	1	1	0	1	0	0	0	0	1	0
	11	1	1	0	1	0	1	1	1	1	1	1	1

Train

Test

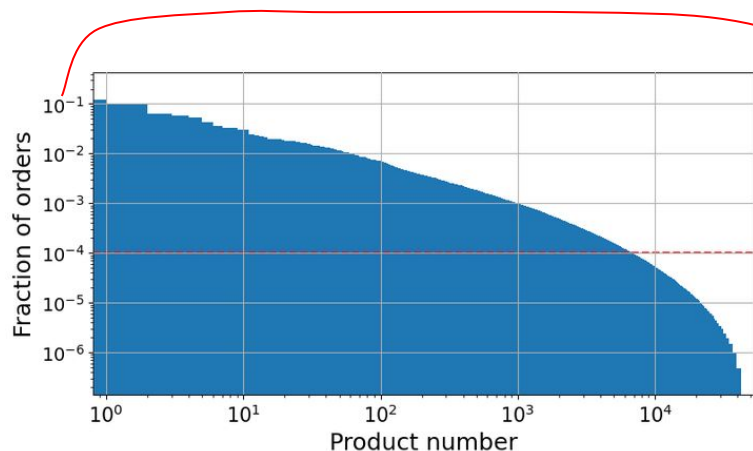
Predict

Next basket:

All-customer model

42,730 unique products are way too many to be input to the model.

GRU+DL model evaluated for a few reduced lists of most frequent products:



f_{min}	(n_{min})	OHE _{all}	n.pars	time	thres.	mean f1
0.003	(6139)	271	123k	7 m	0.18	0.222
0.002	(4094)	450	339k	11 m	0.15	0.242
0.001	(2047)	994	1.6M	47 m	0.13	0.267
0.0004	(818)	2,390	9.5M	4.5 h	0.12	0.288
0.0002	(409)	4,231	29.8M	9 h	0.11	0.287
0.0001	(204)	6,867	78.6M	memory limit		

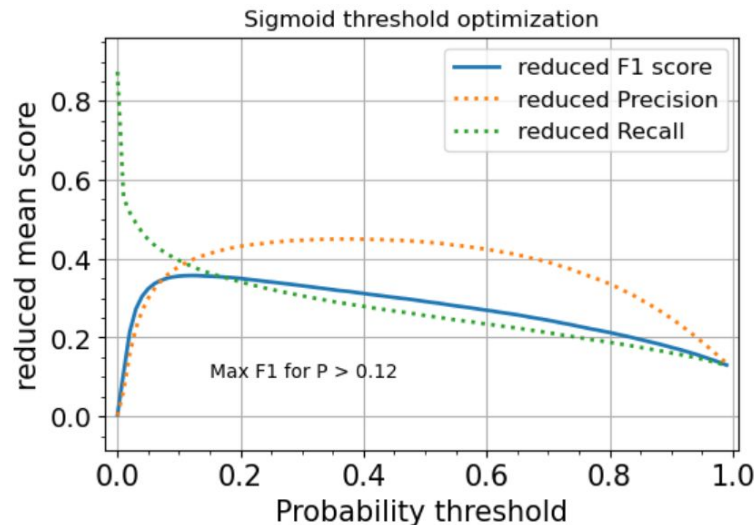
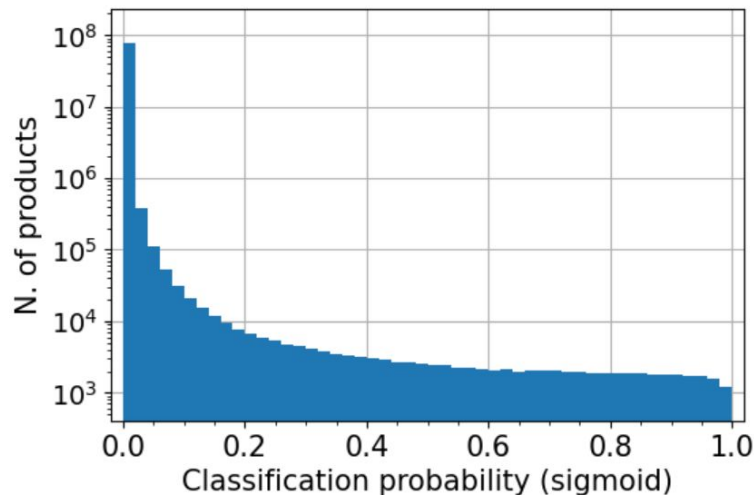
Product frequency plot - total number of product counts divided by total number of (~2 million) orders

The best performance for the model with 2,390 most frequent products.

Mean f1 = 0.288, worse performance than the baseline model (mean f1 = 0.355)

All-customer model (2)

Output probabilities and threshold optimization.



Peak at 0 indicates sparse data.

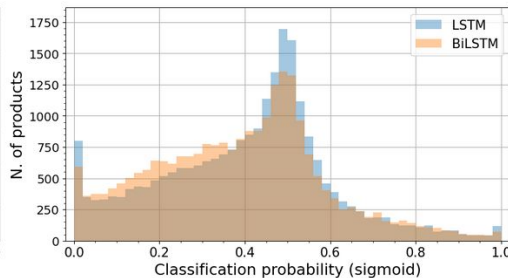
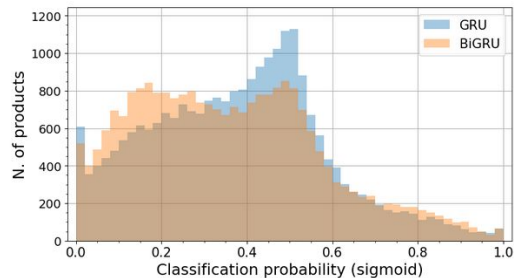
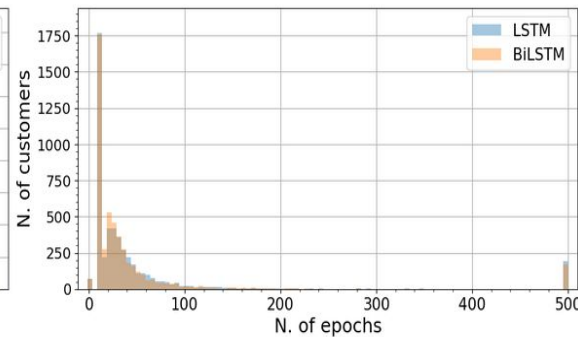
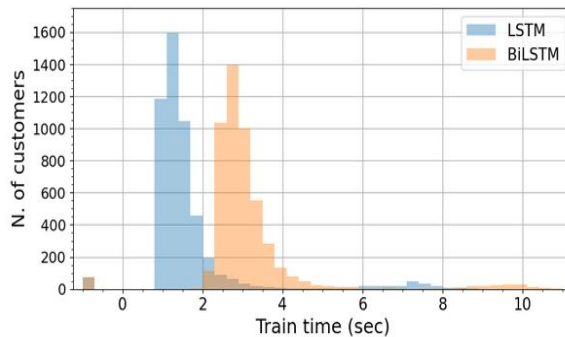
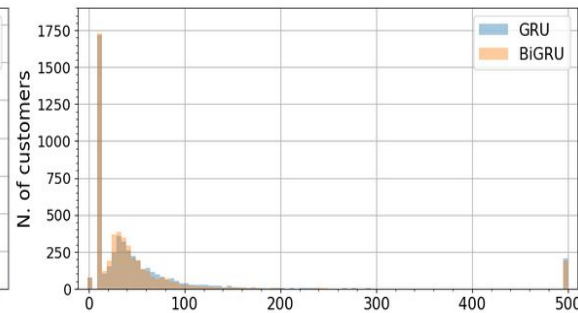
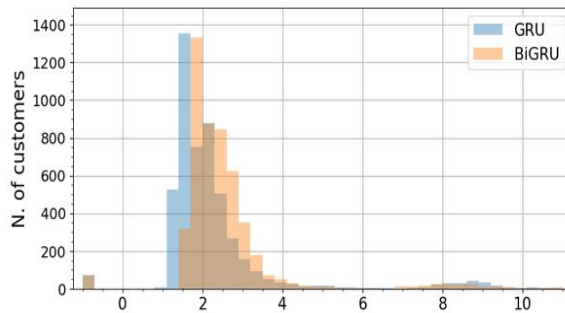
Optimal threshold := a threshold that maximizes the f1 score when comparing predicted order with actual future order.

Per-customer models



- GRU
- LSTM,
- Bidirectional GRU
- Bidirectional LSTM

+ DL.



- Training time
- Number of epochs until convergence
- Output product probabilities (sigmoid)

Per-customer models (2)

Model evaluation (mean f1 score) for 3 different methods of threshold estimation

Ranking method	GRU	LSTM	BiGRU	BiLSTM
FUT	0.463	0.456	0.466	0.462
TES	0.315	0.311	0.319	0.318
DTA	0.321	0.317	0.326	0.321

1. *FUT - thresholds estimated from comparison of actual future order and future prediction. Data leakage present, so cannot be used for deployed models. Minimizes uncertainties related to DL, evaluates the RNN performance only!*
2. TES - thresholds estimated from test data and applied to the future data. OK for deployment.
3. DTA - Decision-Theoretic Approach described in <https://arxiv.org/pdf/1206.4625.pdf>, Uses only predicted probabilities and analytical formula that optimizes f1 score.

- FUT approach significantly outperforms baseline model (mean f1 of ~0.46 compared to 0.36) - RNN part of the model performs remarkably well.
- TES and DTA give lower score than baseline - threshold optimization/ranking method is a bottleneck of modeling.
- BiGRU gives slightly better score than other RNN units.

Summary and Future Work



- The customer's next basket prediction was modeled using Instacart data and the RNN+DL neural network architecture, with RNN = GRU, LSTM, BiGRU, or BiLSTM.
- One model for all customers, as well as separate models for each customer were studied.
- Per-customer models perform better than the all-customer model.
- However, RNN+DL models perform worse than the baseline model, defined as the copy of the customer's last two orders.
- The bottleneck of modeling may be attributed to the way a probability threshold is estimated; this threshold is needed to convert DL classifier probabilities into the future product list.
- Future work should focus on improving techniques for threshold estimation, e.g., by predicting it with a dedicated ML model, such as LightGBM or feed-forward NN.
- The performance of RNN models could be further improved by adding data features with more information on customers' orders and product buying patterns.