SoFi

Internship Showcase Presentation

Zhongruo Wang

Content

- Project Overview
- Data
- Model Architecture
- Performance

Content

- ❑ Project Overview
- ❑ Data
- ❑ Model Architecture
- ❑ Performance

Project Overview

- Internship Objective:
  - Extract and incorporate signals from our behavioral data for our risk models.
  - Investigated SoFi Money measurement and transactions data.

- Background and Context:
  - Our current implementation of customer risk model uses users' credit, socure and aggregated transactions features to predict the likelihood an account being risky.
  - It is built with LightGBM (gradient boosting tree), which takes in tabular data.
  - Assumption: May obtain a performance lift by extract sequential signals from time series data.

- Solution: Temporal Convolutional Network (TCN)
  - We built a TCN classifier that takes in sequential transactions data and predict a score reflecting the riskiness of an account. Then the score currently acts as an input feature for our current lgbm model.

Project Overview

- Deliverable:
  - A customer risk model by combining TCN and LightGBM.
  - A generalized TCN classifier API in PyTorch.

- Performance:
  - On OOT validation, for the top 5% of accounts which the models deem risky:
    - LGBM baseline captures 88.5% of risky accounts; false positive rate = 9.1%
    - TCN + LGBM captures 92.2% of risky accounts; false positive rate = 5.34%

Content

- Project Overview
- Data
- Model Architecture
- Performance

Data

- Two dataset:

dataframe	notes	neg : pos	length	n_borrowers
Account level	One row for each user for each day	14:1	2,096,968	24158
Transactions level	Past transactions made by users	19:1	817,174	23308

- Target definition: at the current time, True if an user has exhibited high risk behaviors in the past.
  - Has charge offs
  - Has check bounces
  - Negative account balance

- Targets are created on the account level, and we used the same target for transactions level.

Example

Account level data

borrower_id	reporting_date	fico_score	target
6855192	2019-04-08	769.0	False
6855192	2019-04-09	769.0	False
6855192	2019-04-10	769.0	False

transaction_datetime	nr_past_transactions	transaction_as_pct_of_balance	borrower_id
2019-03-18 05:50:09	23.0	-0.016029	6855192
2019-03-18 05:50:09	22.0	-0.076036	6855192
2019-03-26 22:27:14	24.0	0.012491	6855192
2019-04-02 05:47:17	26.0	-0.495789	6855192
2019-04-02 05:47:17	25.0	-0.387769	6855192
2019-04-02 18:01:04	27.0	-0.598306	6855192
2019-04-04 14:01:45	28.0	1.191564	6855192
2019-04-09 10:28:06	29.0	-0.906175	6855192

Transactions level data

Content

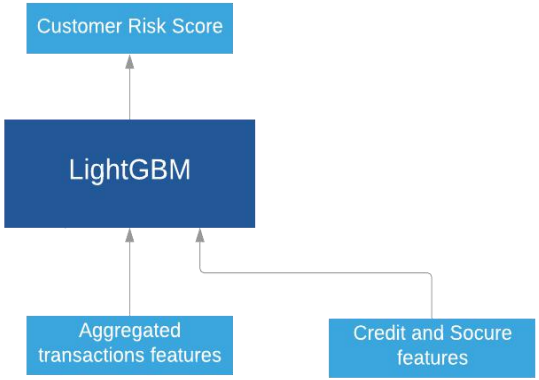
- ❑ Project Overview
- ❑ Data
- ❑ Model Architecture
- ❑ Performance

Model Architecture - TCN + LightGBM

- Separate data into three categories:
  - credit and socure features - account level
  - aggregated transactions features
    - account level
  - simple transactions feature
    - transactions level

- Calculate TCN risk score using simple transactions data

- Calculate customer risk score using
  - TCN risk score
  - aggregated transactions features
  - credit features



Example

Account level data

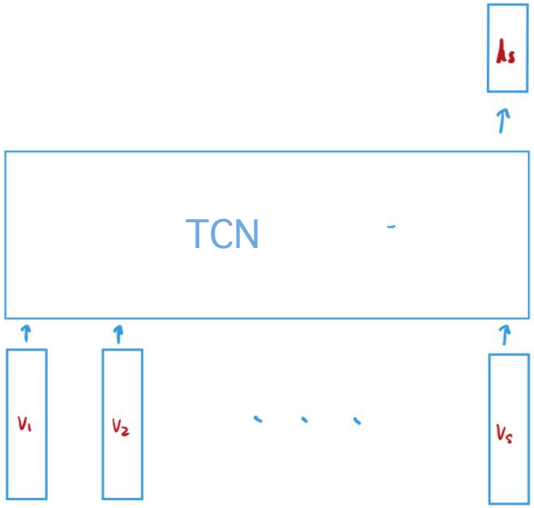
borrower_id	reporting_date	fico_score	target
6855192	2019-04-08	769.0	False
6855192	2019-04-09	769.0	False
6855192	2019-04-10	769.0	False

transaction_datetime	nr_past_transactions	transaction_as_pct_of_balance	borrower_id
2019-03-18 05:50:09	23.0	-0.016029	6855192
2019-03-18 05:50:09	22.0	-0.076036	6855192
2019-03-26 22:27:14	24.0	0.012491	6855192
2019-04-02 05:47:17	26.0	-0.495789	6855192
2019-04-02 05:47:17	25.0	-0.387769	6855192
2019-04-02 18:01:04	27.0	-0.598306	6855192
2019-04-04 14:01:45	28.0	1.191564	6855192
2019-04-09 10:28:06	29.0	-0.906175	6855192

Transactions level data

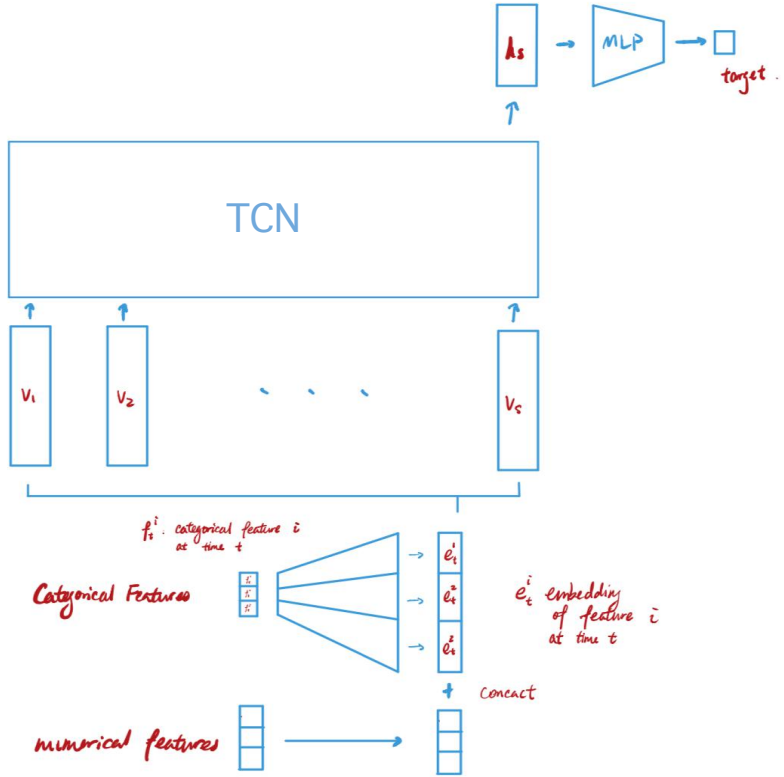
Model Architecture - TCN general information

- TCN: Temporal Convolutional Networks
- Autoregressive model works similar to RNN (Recurrent Neural Network)
- TCN vs. RNNs (GRU, LSTM)
  - Achieves better and faster performances on many tasks.
  - More parallelized: faster training.
  - Lower memory requirement.
  - Long-term dependencies.
- More detailed information for TCN can be found in the Appendix.



Model Architecture - My adaptations

- Each user has a sequence of transactions.
- For each user at each time step:
  - Encode categorical features into embeddings.
  - Concatenate embedded categorical features with numerical features.
- Feed the sequence of vectors into a sequential encoder: TCN
- Feed the output vector at the last time step into a multi-layer perceptron to predict target.
- Current TCN has effective history = 32



Content

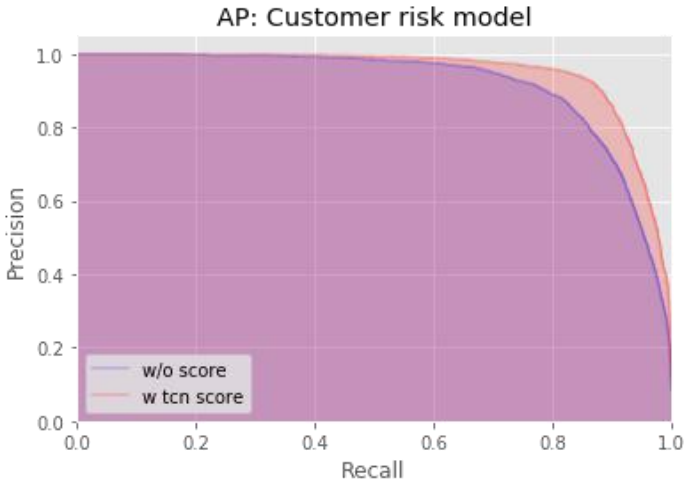
- Project Overview
- Data
- Model Architecture
- Performance

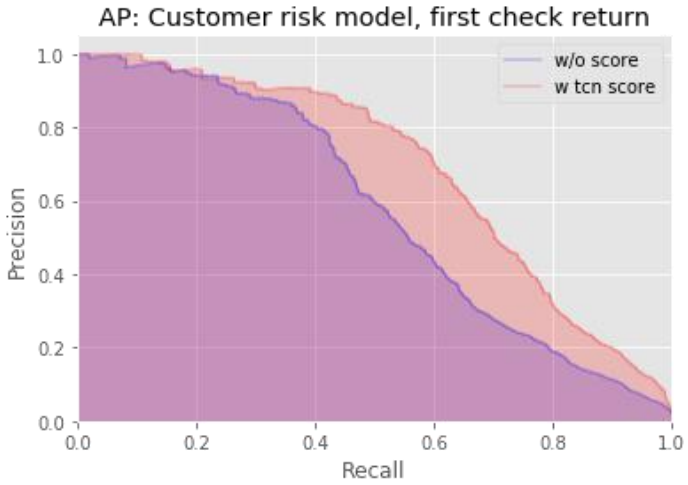
Performance - OOF - LGBM vs. TCN + LGBM

- Customer Risk Model - Out-of-fold and grouped by borrower_id
  - A borrower can only appear in either train or validation data, but not both.

- Average Precision: Area under the Precision-Recall curve
  - LGBM baseline: 0.893
  - TCN + LGBM: 0.927

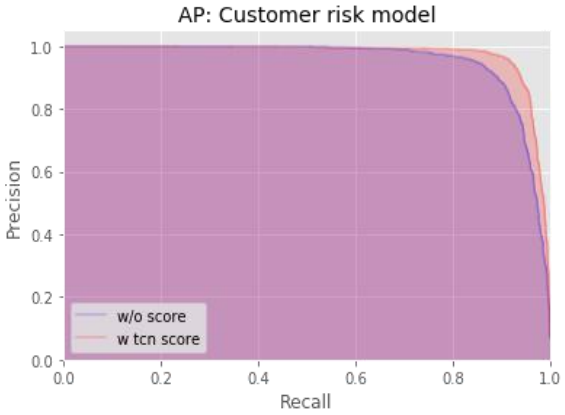
- TCN + LGBM outperforms when the nr_past_returns = 0, when catching initial fraud attempt.

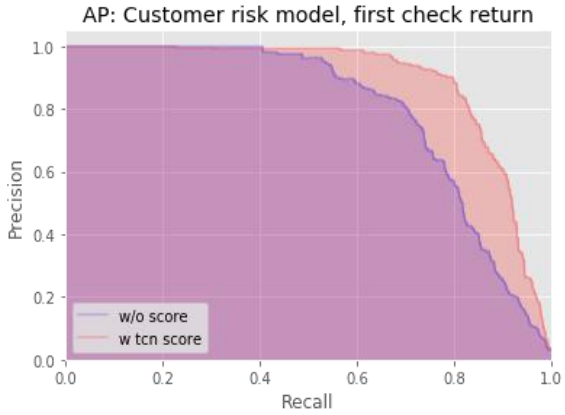


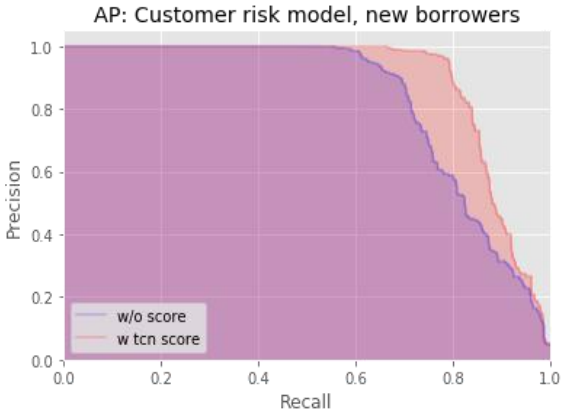


Performance - OOT - LGBM vs. TCN + LGBM

- Customer Risk Model - Out-of-time
- Average Precision: Area under the Precision-Recall curve
  - LGBM baseline: 0.986
  - TCN + LGBM: 0.993
- TCN+LGBM outperforms when capturing initial fraud, and on new borrowers.







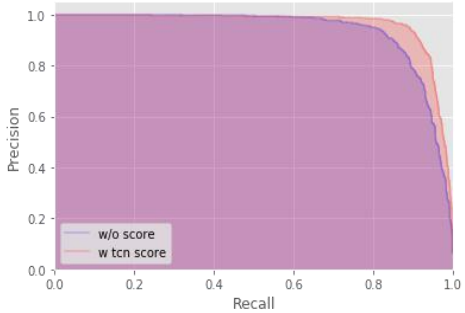
Performance - OOT - LGBM vs. TCN + LGBM

- Customer Risk Model - Out-of-time segmented by time - weekly
- Performances drops as time goes on -> need to refit periodically

AP: Customer Risk Validation over [2019-05-22, 2019-05-29)

AP: Customer Risk Validation over [2019-05-29, 2019-06-05)

AP: Customer Risk Validation over [2019-06-05, 2019-06-12)



AP: Customer Risk Validation over [2019-06-12, 2019-06-19)

AP: Customer Risk Validation over [2019-06-19, 2019-06-26)

Performance

- Performance on top 5 pctls
  - LGBM Baseline:

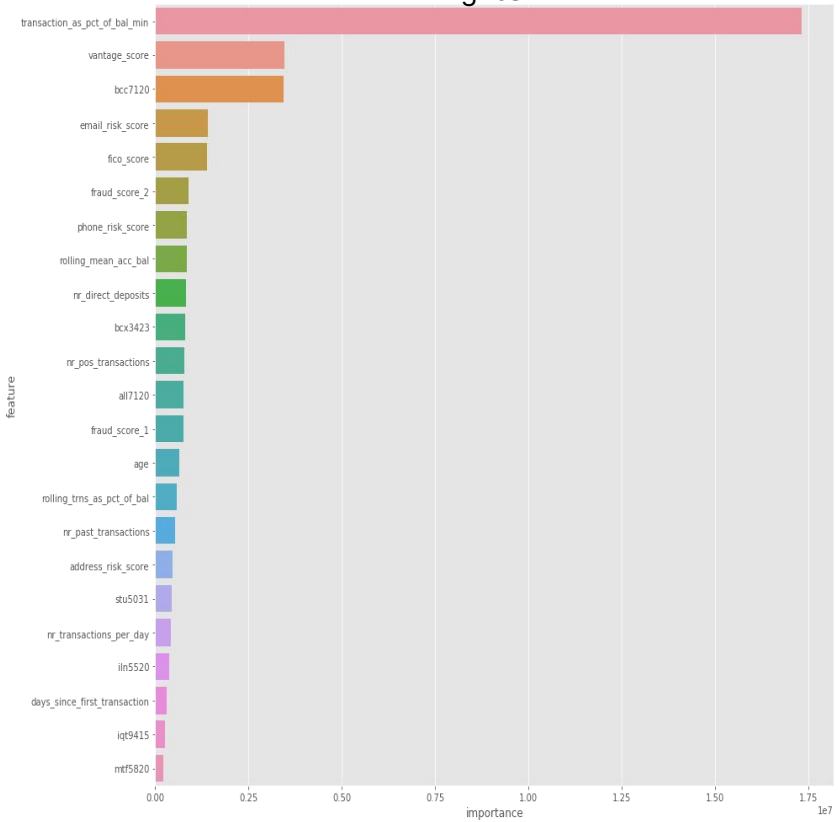
	Percentile	Threshold	Fraud Capture Rate (%)	False Positive Rate (%)	#Above Threshold	#Fraudulent Above Threshold	#Good Above Threshold	#Fraudulent Below Threshold
0	95.0	0.802368	88.490988	9.095917	32674	29702	2972	3863
1	95.5	0.915657	83.819455	4.332155	29408	28134	1274	5431
2	96.0	0.958694	75.957098	2.474945	26142	25495	647	8070
3	96.5	0.980042	67.576344	0.856718	22878	22682	196	10883
4	97.0	0.990019	58.143900	0.448888	19604	19516	88	14049
5	97.5	0.995419	48.657828	0.103982	16349	16332	17	17233
6	98.0	0.998063	38.885744	0.130079	13069	13052	17	20513
7	98.5	0.999174	29.259645	0.000000	9821	9821	0	23744
8	99.0	0.999748	19.478624	0.000000	6538	6538	0	27027
9	99.5	0.999957	9.736332	0.000000	3268	3268	0	30297

  - TCN + LGBM:

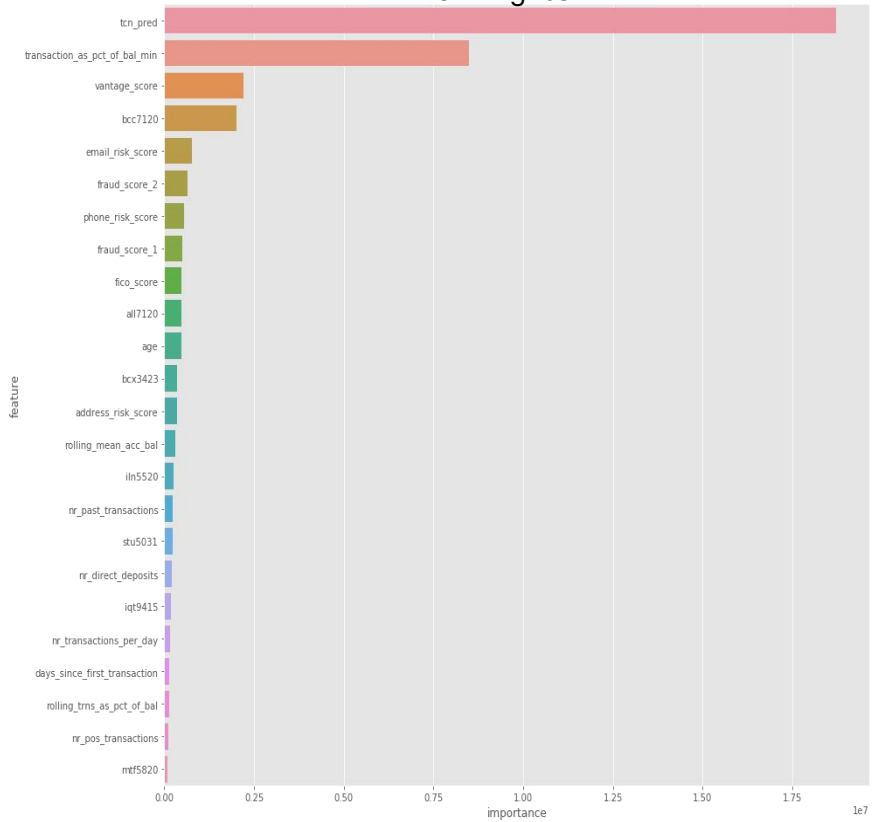
	Percentile	Threshold	Fraud Capture Rate (%)	False Positive Rate (%)	#Above Threshold	#Fraudulent Above Threshold	#Good Above Threshold	#Fraudulent Below Threshold
0	95.0	0.820746	92.170416	5.345123	32684	30937	1747	2628
1	95.5	0.958805	86.211828	1.591566	29405	28937	468	4628
2	96.0	0.986960	77.294801	0.795350	26152	25944	208	7621
3	96.5	0.994939	67.772978	0.559538	22876	22748	128	10817
4	97.0	0.997830	58.146879	0.443787	19604	19517	87	14048
5	97.5	0.998946	48.559511	0.232601	16337	16299	38	17266
6	98.0	0.999462	38.933413	0.129920	13085	13068	17	20497
7	98.5	0.999776	29.170267	0.173328	9808	9791	17	23774
8	99.0	0.999923	19.472665	0.000000	6536	6536	0	27029
9	99.5	0.999980	9.736332	0.000000	3268	3268	0	30297

Feature Importance

LightGBM



TCN+LightGBM



Thanks!

- Any questions, comments, and suggestions?
- Currently building an end-to-end TCN model that takes in the sequential features and tabular features at the same time.

Appendix!

- Information on TCN:
  - Introduction of the TCN Architecture: https://medium.com/the-artificial-impostor/notes-understanding-tensorflow-part-3-7f6633fcc7c7
  - Original Paper: https://arxiv.org/pdf/1803.01271.pdf
  - Informative documentations: https://github.com/philipperemy/keras-tcn
- Current TCN API:
  - https://gitlab.com/sofiinc/data-science-risk/sequential-models
- Detailed Model Validation:
  - https://gitlab.com/sofiinc/data-science-risk/sequential-models/blob/include_stationary_features/TCN_v1/customer_risk_model_validation.ipynb

How does our TCN API work?

	borrower_id	transaction_datetime	transaction_type	nr_past_transactions	hr_of_transaction	days_since_first_transaction	transaction_as_pct_of_balance	target
0	3411668	2017-12-20 00:00:00	P2P	1.0	0.0	0.0	100.000000	True
1	3411668	2017-12-20 00:00:00	P2P	0.0	0.0	0.0	-0.250000	True
2	3411668	2017-12-29 00:00:00	P2P	3.0	0.0	9.0	-0.006042	True
3	3411668	2017-12-29 00:00:00	P2P	2.0	0.0	9.0	-0.006483	True
4	3411668	2017-12-29 12:19:00	ACH	4.0	12.0	9.0	24.242424	True
5	3411668	2018-01-11 00:00:00	P2P	5.0	0.0	22.0	0.004862	True
6	3411668	2018-01-16 17:04:34	ACH	6.0	17.0	27.0	0.483887	True
7	3411668	2018-01-17 15:20:24	POS	7.0	15.0	28.0	-0.019500	True
8	3411668	2018-01-17 17:46:37	POS	8.0	17.0	28.0	-0.037325	True
9	3411668	2018-01-17 18:12:03	POS	9.0	18.0	28.0	-0.043011	True
10	3411668	2018-01-17 18:15:24	POS	10.0	18.0	28.0	-0.019855	True
11	3411668	2018-01-18 09:40:33	POS	11.0	9.0	29.0	-0.064784	True
12	3411668	2018-01-18 12:35:11	ATM	12.0	12.0	29.0	-0.026382	True
13	3411668	2018-01-18 12:35:11	ATM	13.0	12.0	29.0	0.007742	True
14	3411668	2018-01-18 12:38:34	POS	14.0	12.0	29.0	-0.026827	True