Karlsruhe Institute of Technology

SDQ Software Design and Quality

LAMA Presentation
Elo Merchant Category Recommendation

Team Members

Zeyu Li, Toprak Emrah | February 16, 2022

INSTITUT FÜR TECHNIK DER INFORMATIONSVERARBEITUNG

Outline

Zeyu Li, Toprak Emrah

1. Overview
   - Motivation and goals
   - Overview of data dictionary
   - Data Processing
2. Model Training
   - Methodology
   - Tools and Models
   - Further Improvement

Motivation and goals

Motivations

- Competition from Kaggle
- Recommendation System is one of the most typical commercial requests
- Many hands-on experience we can refer to
- Many challenges to overcome

Goals

- make a regression model to fit the loyalty of different card owner

Zeyu Li, Toprak Emrah

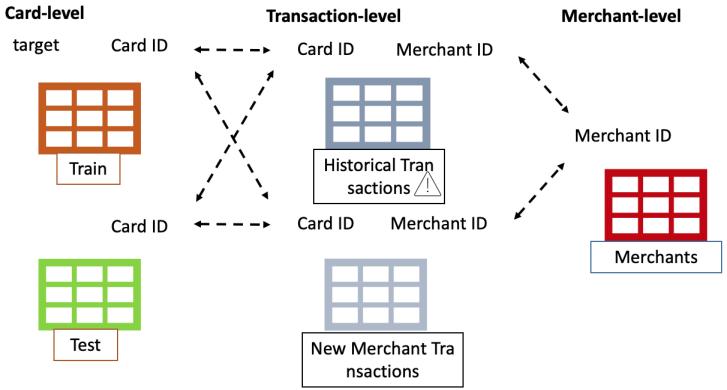
Overview of data dictionary

5 Datasets in all

- train, test datasets that contains features and target
- 3 additional datasets requires features engineering
- total size 3.1 GB
- one of the challenge in our project

Overview of data dictionary

Zeyu Li, Toprak Emrah



Solution to ease memory pressure

- read dataframe in chunk, process the collections of dataframes as stream pipeline

```
@experimental
class StreamerBuilder(t.Generic[T]):
    """      You, a month ago • fix standarize problem …

    def __init__(self, iterator: It[T]):
        if not isinstance(iterator, t.Iterable):
            raise Exception(
                "Streamer Builder must accept an instance from Iterable")
        self._iterator = iterator
        self._callbacks = []

    def __del__(self):
        # finalize stream
        if isinstance(self._iterator, IOBase) and not self._iterator.closed:
            self._iterator.close()
        del self._iterator

    @staticmethod
    def build(iterator: It[T]) -> StreamerBuilder:
        return StreamerBuilder(iterator)
```

Solution to leverage memory pressure

- python buildin mappings can't tackle with iterators with high dimensions in our case it is a two-dimensional streams

```
    def map(self, function: t.Callable[[T], U]) -> StreamerBuilder:
        """      You, a month ago • first commit …

        def _mapper(iterator: It[T]) -> It[U]:
            for data in iterator:
                yield function(data)

        self._register_callback(_mapper)
        return self

    def filter(self, f: t.Callable[[T], bool]) -> StreamerBuilder:

        def _filter(iterator: t.Iterable[T]):
            for data in iterator:
                if f(data):
                    yield data

        self._register_callback(_filter)
        return self
```

Example in Jupyter Notebook

- another merit is that we can postpone our dataset operations until the stream is consumed. which enables us to define behaviour before actual processing

```
lder = histories_builder() \
            .map(lambda df: reformat_dataframe(df, features, change_
            .map(lambda df: df.merge(df_merchant[cols], how='left',
            .map(convert_columns) \
            .map(add_datetime_index)


csv
lder.consume(lambda df: df.to_csv(transactions_path, mode='a', inc
```

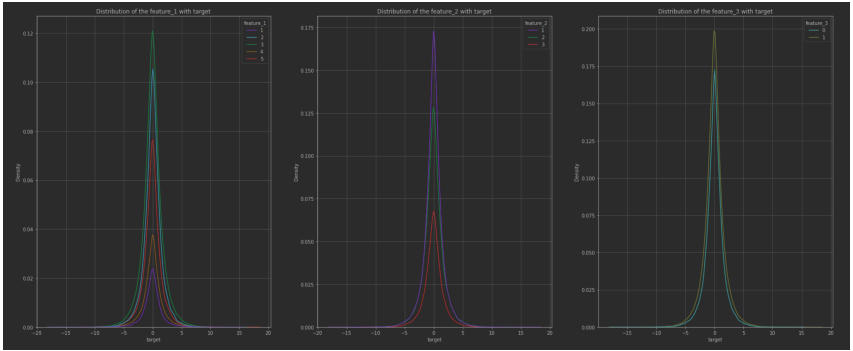
Zeyu Li, Toprak Emrah

General operations to all tables

- check nans and drop or fill the rows
- change category to numbers with one-hot-encoder
- analyze features co-variance and visualize data
- write the result back into new csv files

Data distributions

Zeyu Li, Toprak Emrah



Merging each tables

Merge merchants table with transcations

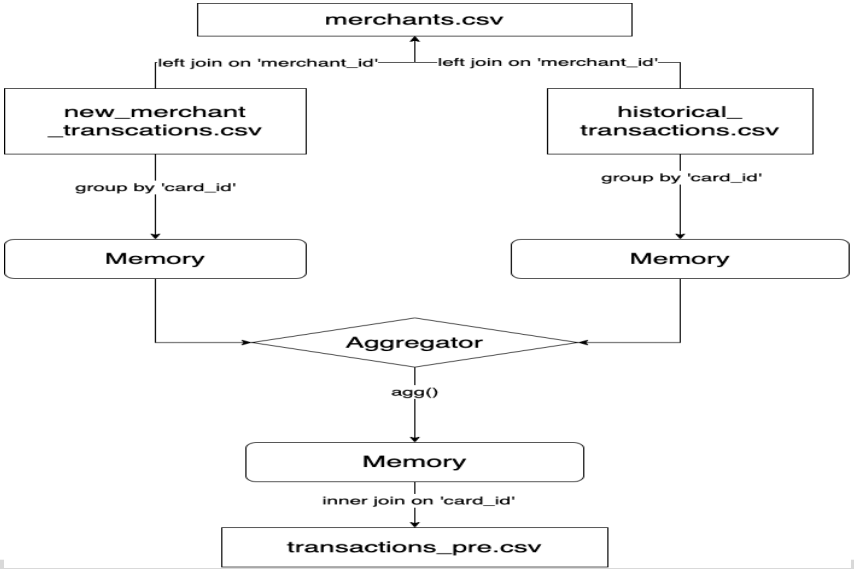
- Relationships on merge
- Aggregator

Merging each tables

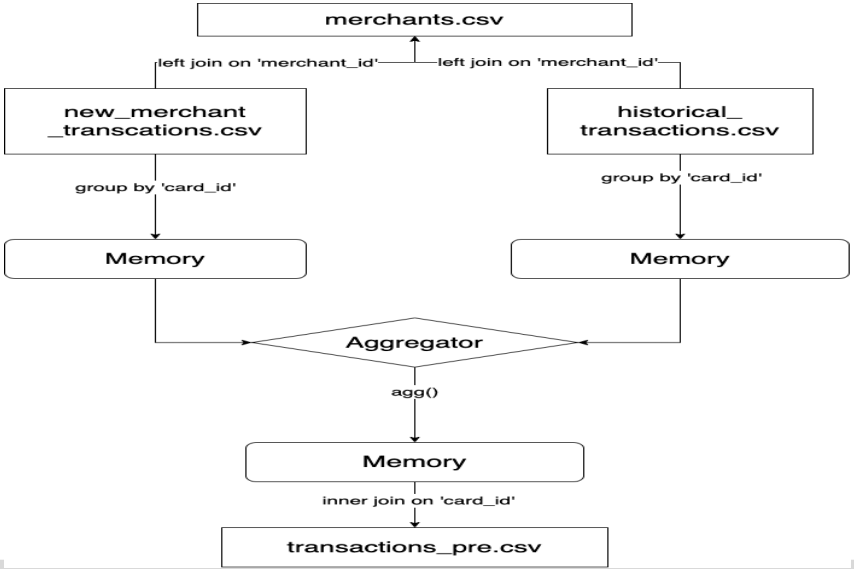
Merge merchants table with transcations

- Relationships on merge
- Aggregator

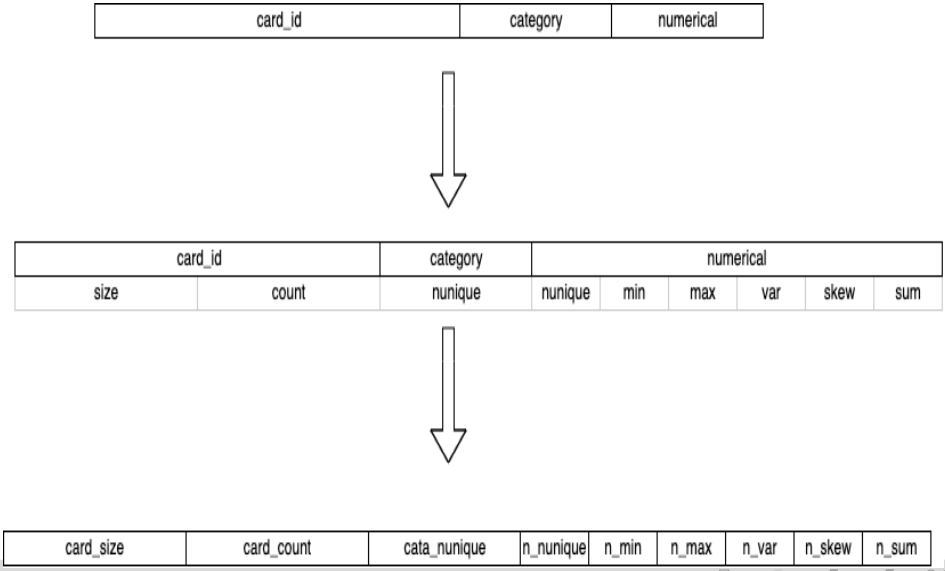
Merging each tables



Merging each tables



Aggregator


card_id
category
numerical
card_id
category
numerical
size
count
nunique
nunique
min
max
var
skew
sum
card_size
card_count
cata_nunique
n_nunique
n_min
n_max
n_var
n_skew
n_sum

Solution to challenge

- group rows with the same key into one
- Transactions table are too large to be read into memory
- Challenges here: How to split the chunk to ensure the rows with the same group by key are in the same table

```
>>> df = pd.DataFrame({'Animal': ['Falcon', 'Falcon',
...                               'Parrot', 'Parrot'],
...                    'Max Speed': [380., 370., 24., 26.]})
>>> df
   Animal  Max Speed
0  Falcon      380.0
1  Falcon      370.0
2  Parrot       24.0
3  Parrot       26.0
>>> df.groupby(['Animal']).mean()
        Max Speed
Animal
Falcon      375.0
Parrot       25.0
```

Stream Group by

- luckily key ids are sorted by name
- make fully usage of space locality, use greedy algorithms

```
authorized_flag,card_id,city_id,category_1,installments,category_3,merch
Y,C_ID_415bb3a509,107,N,1,B,307,M_ID_b0c793002c,1,-0.55757375,2018-03-11
Y,C_ID_415bb3a509,140,N,1,B,307,M_ID_88920c89e8,1,-0.56957993,2018-03-19
Y,C_ID_415bb3a509,330,N,1,B,507,M_ID_ad5237ef6b,2,-0.55103721,2018-04-26
Y,C_ID_415bb3a509,-1,Y,1,B,661,M_ID_9e84cda3b1,1,-0.67192550,2018-03-07
Y,C_ID_ef55cf8d4b,-1,Y,1,B,166,M_ID_3c86fa3831,1,-0.65990429,2018-03-22
Y,C_ID_ef55cf8d4b,231,N,1,B,367,M_ID_8874615e00,2,-0.63300684,2018-04-02
Y,C_ID_ef55cf8d4b,69,N,1,B,333,M_ID_6d061b5ddc,1,5.26369692,2018-03-28 1
Y,C_ID_ef55cf8d4b,231,N,1,B,307,M_ID_df1e022f41,2,-0.55378707,2018-04-05
Y,C_ID_ef55cf8d4b,69,N,1,B,278,M_ID_d15eae0468,2,-0.59664268,2018-04-07
Y,C_ID_ef55cf8d4b,69,N,1,B,437,M_ID_5f9bffd028,1,-0.60719129,2018-03-17
Y,C_ID_ef55cf8d4b,69,N,-1,,45,M_ID_3ffd43b4cd,1,4.45226529,2018-03-31 09
Y,C_ID_ef55cf8d4b,69,N,1,B,108,M_ID_e6f5213fbf,1,-0.60595911,2018-03-11
Y,C_ID_ef55cf8d4b,69,N,1,B,278,M_ID_aa97bc87f6,1,-0.63420896,2018-03-14
```

Stream Group by

- check last row with key of current stream
- yield all items to dataframe (we call it orphans) if key value equals last key values
- concate orphans to chunk at the beginning of each iteration

```python
def stream_groupby_csv(path: str, key: str, chunk_size: int = 10**6, dtype=None) ->
    """…

    with pd.read_csv(path, chunksize=chunk_size, dtype=dtype) as reader:
        orphans = pd.DataFrame()
        for chunk in reader:
            # Add the previous orphans to the chunk
            chunk = pd.concat([orphans, chunk])

            last_val = chunk[key].iloc[-1]
            is_orphan = chunk[key] == last_val

            orphans = chunk[is_orphan]
            yield chunk[~is_orphan]
        # yield orphans if not empty      You, seconds ago • Uncommitted changes
        if len(orphans):
            yield orphans
```

Zeyu Li, Toprak Emrah

Model Training

Tree Ensembled Methodology

- Bagging – RandomForestRegressor
- Gradient Boosting – LightGBM, XGBoost
- Stacking – We made our own

Model Training

Bagging

- Learn base learners in parallel, combine to reduce model variance
- Each base learner is trained on a bootstrap sample
- Combine learners by averaging the outputs (regression) or majority voting (classification)

Model Training

Boosting

- Learn weak learners sequentially, combine to reduce model bias
- At step t, repeat:
  - Evaluate the existing learners' errors
  - Train a weak learner f_t, focus on wrongly predicted examples
  - Train learner to predict errors
- Additively combining existing weak learners with f_t.

Stacking

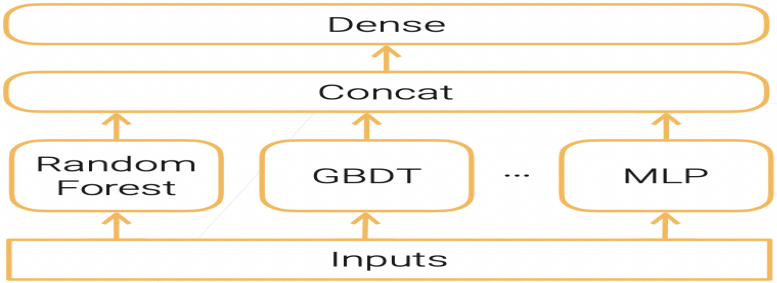
- Combine multiple base learners to reduce variance

Zeyu Li, Toprak Emrah

Stacking with one layer

Stacking

- bagging VS stacking
- Bagging: bootstrap samples to get diversity
- Stacking: different types of models extract different features



Tools and Models

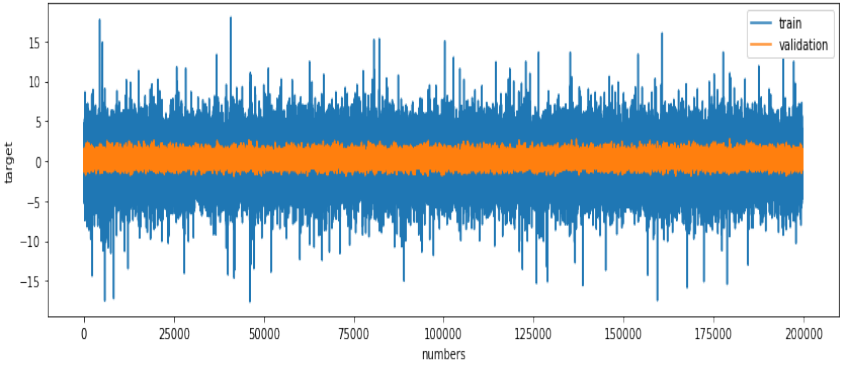
Zeyu Li, Toprak Emrah

Tools

- sklearn
- lightgbm
- xgboost
- All models are trained with kfold
- plots are plotted with validation and train set

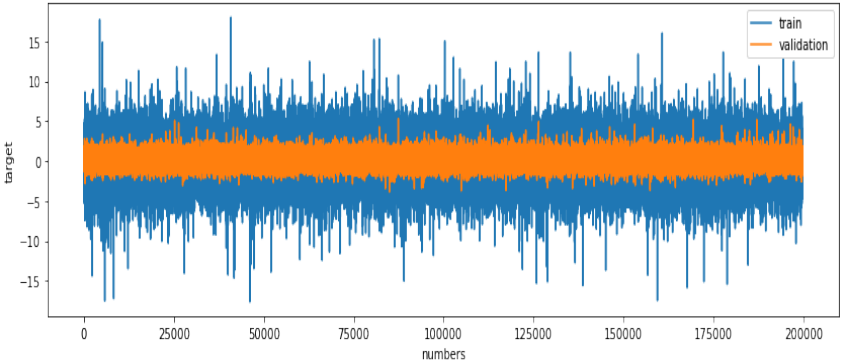
Result with validations random forest

Zeyu Li, Toprak Emrah



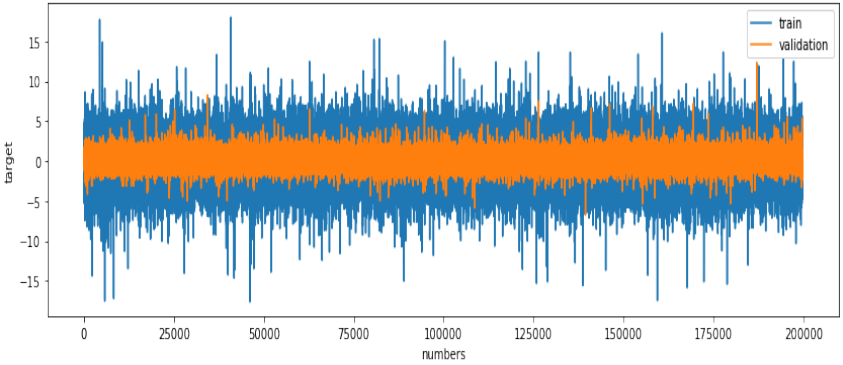
Result with validations lightGBM

Zeyu Li, Toprak Emrah



Result with validations xgboost

Zeyu Li, Toprak Emrah



Result with validations xgboost

Zeyu Li, Toprak Emrah

submission_stacking_kfold.csv a day ago by Zeyu Li666 add submission details	4.29987	4.39653
submission_stacking.csv a day ago by Zeyu Li666 add submission details	4.47945	4.57733
submission_stacking.csv a day ago by Zeyu Li666 add submission details	4.18149	4.28077
submission_light_gbm_best_params.csv 3 days ago by Zeyu Li666 add submission details	3.78958	3.90831
submission_light_gbm_kfold.csv 3 days ago by Zeyu Li666 add submission details	3.78916	3.90822
submission_random_forest_kfold.csv 5 days ago by Zeyu Li666 kfold	3.81648	3.93563

Zeyu Li, Toprak Emrah

Further Improvement

Features Engineering

- Features Engineering improvement
- more visualization with models
- not 100 percent finish