Simpson’s paradox

It refers to a trend appears in several groups of data but disappears or reverses when the groups are combined,the reason for this phenomenon is the existence of a third variable,which is called confounding variable.

	Treatment A	Treatment B
Small Stones	*Group 1* **93% (81/87)**	*Group 2* 87% (234/270)
Large Stones	*Group 3* **73% (192/263)**	*Group 4* 69% (55/80)
Both	78% (273/350)	**83% (289/350)**

For example treatment A is more effective for both small and large stone sizes,but when combined together,treatment B is more effective, which is caused by the confounding variable:the size of stones.

Introduction

In recommendation scenarios,the confounding variable is the deployed model (or system) from which the interaction data were collected.Specifically, in a closed loop feed back system,the exposure of items is affected by the system itself,which makes difficult to get the users’ true preferences.

literature review

In this paper,a new evaluation method was suggested,which stratify on the confounding variable and to investigate the potential outcome in each stratum,and finally a marginalisation over the confounding variable can be leveraged as a combined estimate.

$$\mathrm{E(outcome)} = \sum_{x} \mathrm{E(outcome|X = x)P(X = x)}$$

Here x is the confounding variable.For a deployed model,the confounding variable,namely the model’ s characteristics,is quantified as propensity score,

$P_{u,i}$ = the tendency of the deployed model to expose item i $\in$ I to user u $\in$ U

It was casted into a categorical variable by sorting and segmenting the propensity scores into a predefined number of strata.
To measure the effectiveness of this new evaluation, we measure how better it correlates with the open loop evaluation in comparison to the standard offline holdout evaluation.

Implement

1. Find the propensity scores

The paper assumed that the propensity scores are user-independent and utilized the items' frequency to estimate the propensity scores.

```
item_freq = defaultdict(int)
for u, i, r in self._data:
    item_freq[i] += 1
return item_freq
```

2. Build stratified datasets

```
# match the corresponding propensity score for each feedback
test_props = np.array([self.props[i] ,for u, i, r in test_data], dtype=np.float64)
# stratify
strata, bins = pd.cut(x=test_props, bins=self.n_strata, labels=['Q%d' % i for i in range(1, self.n_strata+1)],
retbins=True)
# sample the corresponding sub-population
for stratum in sorted(np.unique(strata)):
    qtest_data = []
    for (u, i, r), q in zip(test_data, strata):
        if q == stratum:
            qtest_data.append((u, i, r)
    self.stratified_sets[stratum] = qtest_set
```

3. Evaluate on different strata

```
for stratum, qtest_set in self.stratified_sets.items():
    qtest_result = self._eval(model=model,test_set=qtest_set,val_set=self.val_set,user_based=user_based)
```

Experiment

Evaluation Method		$MMMF^{100}$	MF^{50}	Number of Ratings	Number of Items
Holdout		**0.269***	0.263	121,460	3,962
IPS		**0.246**	0.245	121,460	3,962
Stratified		0.239	**0.241***	121,460	3,962
	Q1	0.236	**0.242***	**109,452 (90%)**	**3,944**
	Q2	**0.271***	0.229	12,008 (10%)	18

Stratum Q1 cover the 99% user-item interactions and model MF50 performed better than MMMF100,but when combined with the Stratum Q2 this trend reversed,which revealed the phenomenon of Simpson’s paradox