Team BruteForce

Coupon Purchase Prediction

Blended Cosine Similarity with GBT

This notebook takes the high-performing cosine similarity submission and merges it with the great predictions from TF Gradient Boosted Tree submission. The Cosine Similarity submission contains the top 10 coupons a user is likely to purchase based on actual purchase history. This means there are some users who did not get assigned any coupons. For those users who did not get assigned any coupons, we simply take the predictions from Gradient Boosted Trees (which is trained on **browsing data** and backfill those user predictions with this algorithm. In a way, this is like an ensemble method, though implemented manually for ease of submission to Kaggle.

In [14]:
```
import pandas as pd
import numpy as np
```

In [2]:
```
# get the data

# Cosine Similarity
!gdown --id 1UNHoz8KKd7uO8osyb7vSlc86bJ-GurFG
!gdown --id 1W8VTT72qdL_rtMgqOstubaJwouiAaxVN
```

```
Downloading...
From: https://drive.google.com/uc?id=1UNHoz8KKd7uO8osyb7vSlc86bJ-GurFG
To: /content/submission_cosine.csv
100% 8.27M/8.27M [00:00<00:00, 72.8MB/s]
Downloading...
From: https://drive.google.com/uc?id=1W8VTT72qdL_rtMgqOstubaJwouiAaxVN
To: /content/submission_gradient_boosted_hp.csv
100% 8.30M/8.30M [00:00<00:00, 73.1MB/s]
```

In [73]:
```
df_cosine = pd.read_csv('submission_cosine.csv')
df_gbt    = pd.read_csv('submission_gradient_boosted_hp.csv')

# Gradient Boosted is based on browsing data, so it is the superset here.
# We augment the cosine similarity set.
df_final = df_cosine

print(f'df_cosine length: {len(df_cosine)}')
print(f'df_gbt length: {len(df_gbt)}')

# empty calculations
empty_cosine = len(df_cosine[df_cosine['PURCHASED_COUPONS'].isna()])
empty_gbt    = len(df_gbt[df_gbt['PURCHASED_COUPONS'].isna()])
print(f'Empty values: COSINE {empty_cosine}; GBT {empty_gbt}')
```

```
df_cosine length: 22873
df_gbt length: 22873
Empty values: COSINE 91; GBT 0
```

In [74]:
```
# As you can see, we can augment this cosine result with more users
```

In [75]:
```
augmented = 0
df_final_empty = df_final[df_final['PURCHASED_COUPONS'].isna()]
print(f'adjusting {len(df_final_empty)} rows')

# loop only through the couponless-users
for i, r in df_final_empty.iterrows():
  user_id = r['USER_ID_hash']
  # get the gbt row's coupons
  coupons = df_gbt.loc[df_gbt['USER_ID_hash'] == user_id].PURCHASED_COUPONS

  # assign to final dataset
  df_final.loc[df_final['USER_ID_hash'] == user_id, 'PURCHASED_COUPONS'] = coupo
  augmented += 1
```

```
adjusting 91 rows
```

In [77]:
```
# see how many rows we gained
print(f'added {augmented} sets of coupons')

#  null length
nulls = df_final[df_final['PURCHASED_COUPONS'].isna()]
print(f'null count: {len(nulls)}')
```

```
added 91 sets of coupons
null count: 0
```

In [79]:
```
# re-check length
print(f'final length: {len(df_final)}')
assert len(df_final) == len(df_cosine), 'Length does not match'
```

```
final length: 22873
```

In [80]:
```
df_final.to_csv('submission_cosine_blended_gbt.csv', header=True, index=False)
```