original-field-experiment2

March 5, 2021

[316]:
```
import pandas as pd
from datetime import datetime
import numpy as np
from scipy import stats
import matplotlib.pyplot as plt
from pymatch.Matcher import Matcher
from sklearn.linear_model import LogisticRegression
import seaborn as sns
```

[42]:
```
import pickle
```

[2]:
```
T_e2e_comp = pd.read_csv('order_e2e_pre.csv')  #Treatment
T_other_comp = pd.read_csv('order_control_pre.csv')  #Control
```

0.0.1 E2E Dataset

[156]:
```
idx = []
for i in range(len(T_e2e_comp)):
    if len(T_e2e_comp.iloc[i]['test_inv']) > 0:
        idx.append(i)

T_e2e_comp2 = T_e2e_comp.iloc[idx]

# calculate inventory metric
h = 1
b = 9
N_e2e = len(T_e2e_comp2)
e2e_holding_cost =  []
e2e_stockout_cost = []
e2e_total_cost = []
e2e_turnover = []
e2e_stockout_ratio = []
for n in range(N_e2e):
    inv = T_e2e_comp2.iloc[n].test_inv
    holding_cost = 0
    stockout_cost = 0
    stockout_day = 0
```

```
    T = len(inv)
    for t in range(T):
        cur_inv = inv[t]
        if(cur_inv>=0):
            holding_cost += h*cur_inv
        else:
            stockout_cost += -b*cur_inv
            stockout_day+=1
    turnover = np.maximum(0,T_e2e_comp2.iloc[n].ave_inv/T_e2e_comp2.iloc[n].
↪ave_demand)

    stockout_ratio = stockout_day/T

    e2e_holding_cost.append(holding_cost)
    e2e_stockout_cost.append(stockout_cost)
    e2e_total_cost.append(holding_cost+stockout_cost)
    e2e_turnover.append(turnover)
    e2e_stockout_ratio.append(stockout_ratio)

print('Algorithm A Average Holding cost: ', np.mean(e2e_holding_cost))
print('Algorithm A Average Stockout cost: ', np.mean(e2e_stockout_cost))
print('Algorithm A Average Total cost: ', np.mean(e2e_total_cost))
e2e_turnover1 = [max(0, item) for item in e2e_turnover]
print('Algorithm A Average Turnover rate: ', np.mean(e2e_turnover1))
print('Algorithm A Average Stockout rate: ', np.mean(e2e_stockout_ratio))
```

```
Algorithm A Average Holding cost:  688.134627046695
Algorithm A Average Stockout cost:  880.504548211037
Algorithm A Average Total cost:  1568.639175257732
Algorithm A Average Turnover rate:  16.406984656067063
Algorithm A Average Stockout rate:  0.2426318981200728
```

[184]:
```
idx = []
for i in range(len(T_other_comp)):
    if len(T_other_comp.iloc[i]['test_inv']) != 0:
        idx.append(i)

T_matched_comp = T_other_comp.iloc[idx]

T_matched_comp.test_inv
# calculate inventory metric
N_match = len(T_matched_comp)
T = 14
jd_holding_cost =  []
jd_stockout_cost = []
jd_total_cost = []
jd_turnover = []
```

```
jd_stockout_ratio = []
for n in range(N_match):
    inv = T_matched_comp.iloc[n].test_inv
    holding_cost = 0
    stockout_cost = 0
    stockout_day = 0
    T = len(T_matched_comp.iloc[n].test_inv)
    for t in range(T):
        cur_inv = inv[t]
        if(cur_inv>=0):
            holding_cost += h*cur_inv
        else:
            stockout_cost += -b*cur_inv
#             if(T_matched_comp.iloc[n].test_demand[t]<=0):
            stockout_day+=1
#     turnover = T_matched_comp.iloc[n].ave_inv/T_matched_comp.iloc[n].
 ↪ave_demand
    turnover = np.average(inv)/np.average(T_matched_comp.iloc[n].test_demand)
    stockout_ratio = stockout_day/T

    jd_holding_cost.append(holding_cost)
    jd_stockout_cost.append(stockout_cost)
    jd_total_cost.append(holding_cost+stockout_cost)
    jd_turnover.append(turnover)
    jd_stockout_ratio.append(stockout_ratio)
```

[185]:
```
print('Algorithm B Average Holding cost: ', np.mean(jd_holding_cost))
print('Algorithm B Average Holding Stockout cost: ', np.mean(jd_stockout_cost))
print('Algorithm B Average Total cost: ', np.mean(jd_total_cost))
jd_turnover1 = [max(0, item) for item in jd_turnover]
print('Algorithm B Average Turnover rate: ', np.mean(jd_turnover1))
print('Algorithm B Average Stockout rate: ', np.mean(jd_stockout_ratio))
```

```
Algorithm B Average Holding cost:  786.6887254901961
Algorithm B Average Holding Stockout cost:  887.6397058823529
Algorithm B Average Total cost:  1674.328431372549
Algorithm B Average Turnover rate:  16.399902555089984
Algorithm B Average Stockout rate:  0.22265626057186155
```

[32]:
```
x_demand = np.concatenate((T_e2e_comp.ave_demand, T_matched_comp.ave_demand),␣
 ↪axis = 0)
```

[33]:
```
vlt_e2e = T_e2e_comp.vlt.dt.days.values
# vlt_e2e = []
# for i in range(N_e2e):
#     if type(T_e2e_comp.vlt[i]) == int:
#         vlt_e2e.append(T_e2e_comp.vlt[i])
```

```
#     else:
#         vlt_e2e.append(T_e2e_comp.vlt[i].days)
```

[34]:
```
vlt_match = T_matched_comp.vlt.dt.days.values
```

[35]:
```
x_vlt = np.concatenate((vlt_e2e,vlt_match), axis = 0)

x_e2e = np.concatenate((np.ones(N_e2e), np.zeros(N_e2e)), axis = 0)
```

[45]:
```
# load post-exp results:
file = open("e2e_postexp.pkl",'rb')
post_exp_e2e = pickle.load(file )
```

[49]:
```
# load post-exp results:
file = open("jd_postexp.pkl",'rb')
post_exp_jd = pickle.load(file )
```

[306]:
```
# DID holding cost
post = np.array(post_exp_e2e[0])- np.array(post_exp_jd[0])
pre = np.array(e2e_holding_cost)- np.array(jd_holding_cost)
```

[307]:
```
a = post#[0:1649]
b = pre
t, p = stats.ttest_ind(a,b)
u,p2 = stats.mannwhitneyu(a,b)
print("t = " , str(t))
print("p = " , str(p))
print("u = " , str(u))
print("p2 = " , str(p2))
```

```
t =  -4.332632598460047
p =  1.491969356813226e-05
u =  4896646.5
p2 =  0.052863521503245046
```

[308]:
```
# DID stockout cost
post = np.array(post_exp_e2e[1])- np.array(post_exp_jd[1])
pre = np.array(e2e_stockout_cost)- np.array(jd_stockout_cost)
```

[309]:
```
a = post
b = pre
t, p = stats.ttest_ind(a,b)
u,p2 = stats.mannwhitneyu(a,b)
print("t = " , str(t))
print("p = " , str(p))
print("u = " , str(u))
```

```
print("p2 = " , str(p2))
```

```
t =  -9.427528884653327
p =  5.4385741263322774e-21
u =  4361277.5
p2 =  3.775058364149706e-17
```

[310]:
```
# DID total cost
post = np.array(post_exp_e2e[2])- np.array(post_exp_jd[2])
pre = np.array(e2e_stockout_cost)- np.array(jd_stockout_cost)
```

[311]:
```
a = post
b = pre
t, p = stats.ttest_ind(a,b)
u,p2 = stats.mannwhitneyu(a,b)
print("t = " , str(t))
print("p = " , str(p))
print("u = " , str(u))
print("p2 = " , str(p2))
```

```
t =  -14.409108592386088
p =  1.8035175535570724e-46
u =  3917706.5
p2 =  1.9618186301750062e-43
```

[312]:
```
# DID turnover rate
post = np.array(post_exp_e2e[3])- np.array(post_exp_jd[3])
pre = np.array(e2e_stockout_cost)- np.array(jd_stockout_cost)
```

[313]:
```
a = post
b = pre
t, p = stats.ttest_ind(a,b)
u,p2 = stats.mannwhitneyu(a,b)
print("t = " , str(t))
print("p = " , str(p))
print("u = " , str(u))
print("p2 = " , str(p2))
```

```
t =  2.0462102301771417
p =  0.0407693906503124
u =  4881790.0
p2 =  0.03576277085407192
```

[314]:
```
# DID stockout rate
post = np.array(post_exp_e2e[4])- np.array(post_exp_jd[4])
pre = np.array(e2e_stockout_cost)- np.array(jd_stockout_cost)
```

[315]:
```
a = post
b = pre
t, p = stats.ttest_ind(a,b)
u,p2 = stats.mannwhitneyu(a,b)
print("t = " , str(t))
print("p = " , str(p))
print("u = " , str(u))
print("p2 = " , str(p2))
```

```
t =  2.0175151731708993
p =  0.043676208736025846
u =  4984024.5
p2 =  0.29507512698908345
```

[ ]: