C. Anirudh Bhardwaj

EY Project

Simulation of Bank Transactions

```
import numpy as np
import pandas as pd
import math
import matplotlib
import matplotlib.pyplot as plt

plt.show()
branch_a_cust_data = {'Branch' :'A', 'Balance': 100000*abs((np.random.randn(1000)))};
df = pd.DataFrame(data= branch_a_cust_data);
Branch_a = df;
branch_b_cust_data = {'Branch' :'B', 'Balance': 100000*abs((np.random.randn(3000)))};
df = pd.DataFrame(data= branch_b_cust_data);
Branch_b = df;
branch_c_cust_data = {'Branch' :'C', 'Balance': 100000*abs((np.random.randn(2000)))};
df = pd.DataFrame(data= branch_c_cust_data);
Branch_c =df;

Branch_a_Record = Branch_a.copy()
Branch_b_Record = Branch_b.copy()
Branch_c_Record = Branch_c.copy()

Ba_resrv = 0.1*(Branch_a.Balance.sum());
Bb_resrv = 0.1*(Branch_b.Balance.sum());
Bc_resrv = 0.1*(Branch_c.Balance.sum());

print Ba_resrv,Bb_resrv,Bc_resrv

Paa = (1)/((1/10.0000)+(1/5.0000000)+1)
Pab = (1/10.000)/((1/10.000000000000)+(1/5.0000000)+1)
Pac = (1/5.000)/((1/10.000000000000)+(1/5.0000000)+1)

Pba = (1/10.000)/((1/10.0000)+(1/10.0000000)+1)
```

```
Pbb = (1)/((1/10.0000)+(1/10.0000000)+1)
Pbc = (1/10.000)/((1/10.0000)+(1/10.0000000)+1)

Pca = (1/5.000)/((1/10.0000)+(1/5.0000000)+1)
Pcb = (1/10.000)/((1/10.0000)+(1/5.0000000)+1)
Pcc = (1)/((1/10.0000)+(1/5.0000000)+1)

print Paa,Pab,Pac,(Paa+Pab+Pac)

print Pba,Pbb,Pbc,(Pba+Pbb+Pbc)

print Pca,Pcb,Pcc,(Pca+Pcb+Pcc)

Branch_a_Record = Branch_a.Balance.copy()
Branch_b_Record = Branch_b.Balance.copy()
Branch_c_Record = Branch_c.Balance.copy()
c=1
s_ta = {c};
Old_bal_ser_A = pd.Series();
Old_bal_ser_B = pd.Series();
Old_bal_ser_C = pd.Series();

New_bal_ser_A = pd.Series();
New_bal_ser_B = pd.Series();
New_bal_ser_C = pd.Series();

Total_IO_ser = pd.Series();
#dftttta  =  pd.DataFrame({'Old_A': [c] ,'New_A': [c] ,'Old_B': [c],'New_B': [c],'Old_C': [c],'New_C': [c],'Change_Val': [c]},index=[0]);

for j in range(1,91,1):
    for i in range(0,3,1):
        a= math.floor(np.random.uniform(low=0, high=1000, size=None))
        print a
        c=a;
        Cb= Branch_a.Balance[a]
        Sdb= Cb/3.0000
        MV = Sdb * (np.random.randn(1))
        New_bal = Branch_a.Balance[a] + MV

        if(New_bal<0):
            New_bal = Branch_a.Balance
            Branch_a.Balance = Branch_a.Balance.replace(Branch_a.Balance[i],New_bal)

        x= np.random.uniform(0,1)

        if(x<Paa):
            print 'Branch A';
            if((Ba_resrv+MV)>0 and ((Branch_a.Balance[a] + MV)>0)):
                Ba_resrv = Ba_resrv + MV;
                Branch_a.set_value(a,'Balance',New_bal, takeable=False)
        elif(Paa<x<(Paa+Pab)):
            print 'Branch B';
            if((Bb_resrv+MV)>0 and ((Branch_a.Balance[a] + MV)>0)):
                Bb_resrv = Bb_resrv + MV;
                Branch_a.set_value(a,'Balance',New_bal, takeable=False)
        else:
            print 'Branch C';
            if((Bc_resrv+MV)>0 and ((Branch_a.Balance[a] + MV)>0)):
                Bc_resrv = Bc_resrv + MV;
                Branch_a.set_value(a,'Balance',New_bal, takeable=False)




    for i in range(0,10,1):
        a= math.floor(np.random.uniform(low=0, high=3000, size=None))
        print a
        c=a;
        Cb= Branch_b.Balance[a]
        Sdb= Cb/3.0000
        MV = Sdb * (np.random.randn(1))
        New_bal = Branch_b.Balance[a] + MV

        if(New_bal<0):
            New_bal = Branch_b.Balance
            Branch_b.Balance = Branch_b.Balance.replace(Branch_b.Balance[i],New_bal)

        x= np.random.uniform(0,1)

        if(x<Pba):
            print 'Branch A';
            if((Ba_resrv+MV)>0 and ((Branch_b.Balance[a] + MV)>0)):
                Ba_resrv = Ba_resrv + MV;
                Branch_b.set_value(a,'Balance',New_bal, takeable=False)
        elif(Pba<x<(Pba+Pbb)):
            print 'Branch B';
            if((Bb_resrv+MV)>0 and ((Branch_b.Balance[a] + MV)>0)):
                Bb_resrv = Bb_resrv + MV;
                Branch_b.set_value(a,'Balance',New_bal, takeable=False)
        else:
            print 'Branch C';
            if((Bc_resrv+MV)>0 and ((Branch_b.Balance[a] + MV)>0)):
                Bc_resrv = Bc_resrv + MV;
                Branch_b.set_value(a,'Balance',New_bal, takeable=False)



    for i in range(0,6,1):
        a= math.floor(np.random.uniform(low=0, high=1000, size=None))
        print a
        c=a;
        Cb= Branch_c.Balance[a]
        Sdb= Cb/3.0000
        MV = Sdb * (np.random.randn(1))
        New_bal = Branch_c.Balance[a] + MV

        if(New_bal<0):
            New_bal = Branch_c.Balance
            Branch_c.Balance = Branch_c.Balance.replace(Branch_c.Balance[i],New_bal)

        x= np.random.uniform(0,1)

        if(x<Pca):
            print 'Branch A';
            if((Ba_resrv+MV)>0 and ((Branch_c.Balance[a] + MV)>0)):
                Ba_resrv = Ba_resrv + MV;
                Branch_c.set_value(a,'Balance',New_bal, takeable=False)
        elif(Pca<x<(Pca+Pcb)):
            print 'Branch B';
            if((Bb_resrv+MV)>0 and ((Branch_c.Balance[a] + MV)>0)):
                Bb_resrv = Bb_resrv + MV;
                Branch_c.set_value(a,'Balance',New_bal, takeable=False)
        else:
            print 'Branch C';
            if((Bc_resrv+MV)>0 and ((Branch_c.Balance[a] + MV)>0)):
                Bc_resrv = Bc_resrv + MV;
                Branch_c.set_value(a,'Balance',New_bal, takeable=False)

    Branch_a_Record = pd.concat([Branch_a_Record, Branch_a.Balance], axis =1);
    Branch_b_Record = pd.concat([Branch_b_Record, Branch_b.Balance], axis =1);
    Branch_c_Record = pd.concat([Branch_c_Record, Branch_c.Balance], axis =1);


    temp_a = Ba_resrv;
    temp_b = Bb_resrv;
    temp_c = Bc_resrv;


    Ba_resrv = 0.1*(Branch_a.Balance.sum());
    Bb_resrv = 0.1*(Branch_b.Balance.sum());
    Bc_resrv = 0.1*(Branch_c.Balance.sum());


    t_val = ( temp_a[0] - Ba_resrv) + ( temp_b[0] - Bb_resrv) + ( temp_c[0] - Bc_resrv);


    Old_bal_ser_A = Old_bal_ser_A.set_value(j,temp_a[0]);
    Old_bal_ser_B = Old_bal_ser_B.set_value(j,temp_b[0]);
    Old_bal_ser_C = Old_bal_ser_C.set_value(j,temp_c[0]);


    New_bal_ser_A = New_bal_ser_A.set_value(j,Ba_resrv);
    New_bal_ser_B = New_bal_ser_B.set_value(j,Bb_resrv);
    New_bal_ser_C = New_bal_ser_C.set_value(j,Bc_resrv);


    Total_IO_ser = Total_IO_ser.set_value(j,t_val);
```

```
for i in range(0,4,1):
    a= math.floor(np.random.uniform(low=0, high=1000, size=None))
    print a
    c=a;
    Cb= Branch_a.Balance[a]
    Sdb= Cb/3.0000
    MV = Sdb * (np.random.randn(1))
    New_bal = Branch_a.Balance[a] + MV

    if(New_bal<0):
        New_bal = Branch_a.Balance
        Branch_a.Balance = Branch_a.Balance.replace(Branch_a.Balance[i],New_bal)

    x= np.random.uniform(0,1)

    if(x<Paa):
        print 'Branch A';
        if((Ba_resrv+MV)>0 and ((Branch_a.Balance[a] + MV)>0)):
            Ba_resrv = Ba_resrv + MV;
            Branch_a.set_value(a,'Balance',New_bal, takeable=False)
    elif(Paa<x<(Paa+Pab)):
        print 'Branch B';
        if((Bb_resrv+MV)>0 and ((Branch_a.Balance[a] + MV)>0)):
            Bb_resrv = Bb_resrv + MV;
            Branch_a.set_value(a,'Balance',New_bal, takeable=False)
    else:
        print 'Branch C';
        if((Bc_resrv+MV)>0 and ((Branch_a.Balance[a] + MV)>0)):
            Bc_resrv = Bc_resrv + MV;
            Branch_a.set_value(a,'Balance',New_bal, takeable=False)




for i in range(0,10,1):
    a= math.floor(np.random.uniform(low=0, high=3000, size=None))
    print a
    c=a;
    Cb= Branch_b.Balance[a]
    Sdb= Cb/3.0000
    MV = Sdb * (np.random.randn(1))
    New_bal = Branch_b.Balance[a] + MV

    if(New_bal<0):
        New_bal = Branch_b.Balance
        Branch_b.Balance = Branch_b.Balance.replace(Branch_b.Balance[i],New_bal)

    x= np.random.uniform(0,1)

    if(x<Pba):
        print 'Branch A';
        if((Ba_resrv+MV)>0 and ((Branch_b.Balance[a] + MV)>0)):
            Ba_resrv = Ba_resrv + MV;
            Branch_b.set_value(a,'Balance',New_bal, takeable=False)
    elif(Pba<x<(Pba+Pbb)):
        print 'Branch B';
        if((Bb_resrv+MV)>0 and ((Branch_b.Balance[a] + MV)>0)):
            Bb_resrv = Bb_resrv + MV;
            Branch_b.set_value(a,'Balance',New_bal, takeable=False)
    else:
        print 'Branch C';
        if((Bc_resrv+MV)>0 and ((Branch_b.Balance[a] + MV)>0)):
            Bc_resrv = Bc_resrv + MV;
            Branch_b.set_value(a,'Balance',New_bal, takeable=False)



for i in range(0,7,1):
    a= math.floor(np.random.uniform(low=0, high=1000, size=None))
    print a
    c=a;
    Cb= Branch_c.Balance[a]
    Sdb= Cb/3.0000
    MV = Sdb * (np.random.randn(1))
    New_bal = Branch_c.Balance[a] + MV

    if(New_bal<0):
        New_bal = Branch_c.Balance
        Branch_c.Balance = Branch_c.Balance.replace(Branch_c.Balance[i],New_bal)

    x= np.random.uniform(0,1)

    if(x<Pca):
        print 'Branch A';
        if((Ba_resrv+MV)>0 and ((Branch_c.Balance[a] + MV)>0)):
            Ba_resrv = Ba_resrv + MV;
            Branch_c.set_value(a,'Balance',New_bal, takeable=False)
    elif(Pca<x<(Pca+Pcb)):
        print 'Branch B';
        if((Bb_resrv+MV)>0 and ((Branch_c.Balance[a] + MV)>0)):
            Bb_resrv = Bb_resrv + MV;
            Branch_c.set_value(a,'Balance',New_bal, takeable=False)
    else:
        print 'Branch C';
        if((Bc_resrv+MV)>0 and ((Branch_c.Balance[a] + MV)>0)):
            Bc_resrv = Bc_resrv + MV;
            Branch_c.set_value(a,'Balance',New_bal, takeable=False)

Branch_a_Record = pd.concat([Branch_a_Record, Branch_a.Balance], axis =1);
Branch_b_Record = pd.concat([Branch_b_Record, Branch_b.Balance], axis =1);
Branch_c_Record = pd.concat([Branch_c_Record, Branch_c.Balance], axis =1);

temp_a = Ba_resrv;
temp_b = Bb_resrv;
temp_c = Bc_resrv;

Ba_resrv = 0.1*(Branch_a.Balance.sum());
Bb_resrv = 0.1*(Branch_b.Balance.sum());
Bc_resrv = 0.1*(Branch_c.Balance.sum());

t_val = ( temp_a[0] - Ba_resrv) + ( temp_b[0] - Bb_resrv) + ( temp_c[0] - Bc_resrv);
```

```
Old_bal_ser_A = Old_bal_ser_A.set_value(j,temp_a[0]);
Old_bal_ser_B = Old_bal_ser_B.set_value(j,temp_b[0]);
Old_bal_ser_C = Old_bal_ser_C.set_value(j,temp_c[0]);

New_bal_ser_A = New_bal_ser_A.set_value(j,Ba_resrv);
New_bal_ser_B = New_bal_ser_B.set_value(j,Bb_resrv);
New_bal_ser_C = New_bal_ser_C.set_value(j,Bc_resrv);

Total_IO_ser = Total_IO_ser.set_value(j,t_val);

for i in range(0,3,1):
    a= math.floor(np.random.uniform(low=0, high=1000, size=None))
    print a
    c=a;
    Cb= Branch_a.Balance[a]
    Sdb= Cb/3.0000
    MV = Sdb * (np.random.randn(1))
    New_bal = Branch_a.Balance[a] + MV

    if(New_bal<0):
        New_bal = Branch_a.Balance
        Branch_a.Balance = Branch_a.Balance.replace(Branch_a.Balance[i],New_bal)

    x= np.random.uniform(0,1)

    if(x<Paa):
        print 'Branch A';
        if((Ba_resrv+MV)>0 and ((Branch_a.Balance[a] + MV)>0)):
            Ba_resrv = Ba_resrv + MV;
            Branch_a.set_value(a,'Balance',New_bal, takeable=False)
    elif(Paa<x<(Paa+Pab)):
        print 'Branch B';
        if((Bb_resrv+MV)>0 and ((Branch_a.Balance[a] + MV)>0)):
            Bb_resrv = Bb_resrv + MV;
            Branch_a.set_value(a,'Balance',New_bal, takeable=False)
    else:
        print 'Branch C';
        if((Bc_resrv+MV)>0 and ((Branch_a.Balance[a] + MV)>0)):
            Bc_resrv = Bc_resrv + MV;
            Branch_a.set_value(a,'Balance',New_bal, takeable=False)




for i in range(0,10,1):
    a= math.floor(np.random.uniform(low=0, high=3000, size=None))
    print a
    c=a;
    Cb= Branch_b.Balance[a]
    Sdb= Cb/3.0000
    MV = Sdb * (np.random.randn(1))
    New_bal = Branch_b.Balance[a] + MV

    if(New_bal<0):
        New_bal = Branch_b.Balance
        Branch_b.Balance = Branch_b.Balance.replace(Branch_b.Balance[i],New_bal)

    x= np.random.uniform(0,1)

    if(x<Pba):
        print 'Branch A';
        if((Ba_resrv+MV)>0 and ((Branch_b.Balance[a] + MV)>0)):
            Ba_resrv = Ba_resrv + MV;
            Branch_b.set_value(a,'Balance',New_bal, takeable=False)
    elif(Pba<x<(Pba+Pbb)):
        print 'Branch B';
        if((Bb_resrv+MV)>0 and ((Branch_b.Balance[a] + MV)>0)):
            Bb_resrv = Bb_resrv + MV;
            Branch_b.set_value(a,'Balance',New_bal, takeable=False)
    else:
        print 'Branch C';
        if((Bc_resrv+MV)>0 and ((Branch_b.Balance[a] + MV)>0)):
            Bc_resrv = Bc_resrv + MV;
            Branch_b.set_value(a,'Balance',New_bal, takeable=False)
```

```
for i in range(0,6,1):
    a= math.floor(np.random.uniform(low=0, high=1000, size=None))
    print a
    c=a;
    Cb= Branch_c.Balance[a]
    Sdb= Cb/3.0000
    MV = Sdb * (np.random.randn(1))
    New_bal = Branch_c.Balance[a] + MV

    if(New_bal<0):
        New_bal = Branch_c.Balance
        Branch_c.Balance = Branch_c.Balance.replace(Branch_c.Balance[i],New_bal)

    x= np.random.uniform(0,1)

    if(x<Pca):
        print 'Branch A';
        if((Ba_resrv+MV)>0 and ((Branch_c.Balance[a] + MV)>0)):
            Ba_resrv = Ba_resrv + MV;
            Branch_c.set_value(a,'Balance',New_bal, takeable=False)
    elif(Pca<x<(Pca+Pcb)):
        print 'Branch B';
        if((Bb_resrv+MV)>0 and ((Branch_c.Balance[a] + MV)>0)):
            Bb_resrv = Bb_resrv + MV;
            Branch_c.set_value(a,'Balance',New_bal, takeable=False)
    else:
        print 'Branch C';
        if((Bc_resrv+MV)>0 and ((Branch_c.Balance[a] + MV)>0)):
            Bc_resrv = Bc_resrv + MV;
            Branch_c.set_value(a,'Balance',New_bal, takeable=False)

Branch_a_Record = pd.concat([Branch_a_Record, Branch_a.Balance], axis =1);
Branch_b_Record = pd.concat([Branch_b_Record, Branch_b.Balance], axis =1);
Branch_c_Record = pd.concat([Branch_c_Record, Branch_c.Balance], axis =1);

temp_a = Ba_resrv;
temp_b = Bb_resrv;
```

```
    temp_c = Bc_resrv;

    Ba_resrv = 0.1*(Branch_a.Balance.sum());
    Bb_resrv = 0.1*(Branch_b.Balance.sum());
    Bc_resrv = 0.1*(Branch_c.Balance.sum());

    t_val = ( temp_a[0] - Ba_resrv) + ( temp_b[0] - Bb_resrv) + ( temp_c[0] - Bc_resrv);

    Old_bal_ser_A = Old_bal_ser_A.set_value(j,temp_a[0]);
    Old_bal_ser_B = Old_bal_ser_B.set_value(j,temp_b[0]);
    Old_bal_ser_C = Old_bal_ser_C.set_value(j,temp_c[0]);

    New_bal_ser_A = New_bal_ser_A.set_value(j,Ba_resrv);
    New_bal_ser_B = New_bal_ser_B.set_value(j,Bb_resrv);
    New_bal_ser_C = New_bal_ser_C.set_value(j,Bc_resrv);

    Total_IO_ser = Total_IO_ser.set_value(j,t_val);



Net_inventory_change_A = pd.DataFrame(Old_bal_ser_A)
New_inventory_change_A = pd.DataFrame(New_bal_ser_A)
result_A = pd.concat([Net_inventory_change_A,New_inventory_change_A,(New_inventory_change_A -
Net_inventory_change_A)], axis=1);

Net_inventory_change_B = pd.DataFrame(Old_bal_ser_B)
New_inventory_change_B = pd.DataFrame(New_bal_ser_B)
result_B = pd.concat([Net_inventory_change_B,New_inventory_change_B,(New_inventory_change_B -
Net_inventory_change_B)], axis=1);

Net_inventory_change_C = pd.DataFrame(Old_bal_ser_C)
New_inventory_change_C = pd.DataFrame(New_bal_ser_C)
result_C = pd.concat([Net_inventory_change_C,New_inventory_change_C,(New_inventory_change_C -
Net_inventory_change_C)], axis=1);


t_val = result_A.iloc[:, 2] + result_B.iloc[: , 2] + result_C.iloc[: , 2]
```

```
Total_Change = pd.concat([result_A.iloc[:, 2],result_B.iloc[: , 2],result_C.iloc[: , 2],t_val], axis=1);
Total_Change.columns = ['A', 'B','C','Total']


Total_Change.to_csv('Total_Change_V2.csv')
Branch_a_Record.to_csv('Branch_A_V2.csv')
Branch_b_Record.to_csv('Branch_B_V2.csv')
Branch_c_Record.to_csv('Branch_C_V2.csv')
result_A.to_csv('Transaction_log_A_V2.csv')
result_B.to_csv('Transaction_log_B_V2.csv')
result_C.to_csv('Transaction_log_C_V2.csv')
```