(a)	Age	Marital	Default	Prediciton on best
2	44	0	1	1
3	28	1	0	0
3	28	1	0	0
1	42	0	0	0
5	52	1	1	1
4	26	2	1	0

Age ≤ 27

1   1,0,0,0,1

2 misclassified

Age ≤ 35

0,0,1   1,0,1

2 misclassified

Age ≤ 43

0,0,0,1   1,1

1 misclassified

Age ≤ 48

1,0,0,0,1   1

2 misclassified

~~Age~~ Marital = 0

1, 0   0,0,1,1

3 misclassified

Marital = 1

0,0,1   1,0,1

2 misclassified

Marital = 2

1   1,0,0,0,1

2 misclassified

Age≤43 is the best split

prediction on all instances :

1	0
2	1
3	0
4	0
5	1
6	0

0 – unmarried

1 – married

2 – separated

0 – High school

1 – college

(b) change

	Age	Education	Default	Predicition on education = 0
5	1	52	1	1
6	0	27	0	0
2	1	44	1	1
4	0	26	1	0
6	0	27	0	0
5	1	52	1	1

Age ≤ 26.5

1   1, 0, 1, 0, 1

2 misclassified

Age ≤ 35.5

0, 1, 0   1, 1, 1

1 misclassified

Age ≤ 48

0, 1, 1, 0   1, 1

2 misclassified

Education = 0

0, 1, 0   1, 1, 1

1 misclassified

Education = 0 or Age ≤ 35.5

I choose education = 0

predicition on all instances:

1	0
2	1
3	1
4	0
5	1
6	0

(b)2.

	Balance	Marital	Default	Predicitions on marital
5	10000	1	1	0
2	20000	0	1	1
4	120000	2	1	1
2	20000	0	1	1
3	30000	1	0	0
3	30000	1	0	0

Balance ≤ 15000
1   1,1,1,0,0
2 misclassified

Balance ≤ 25000
1,1,1   1,0,0
1 misclassified

Balance ≤ 80000
1,1,1,0,0   1
2 misclassified

Marital = 0
1, 1   1,1,0,0
2 misclassified

Marital = 1
1,0,0   1,1,1
1 misclassified

Marital = 2
1   1,1,1,0,0
2 misclassified

Balance ≤ 25000 or Marital = 1

I choose marital = 1

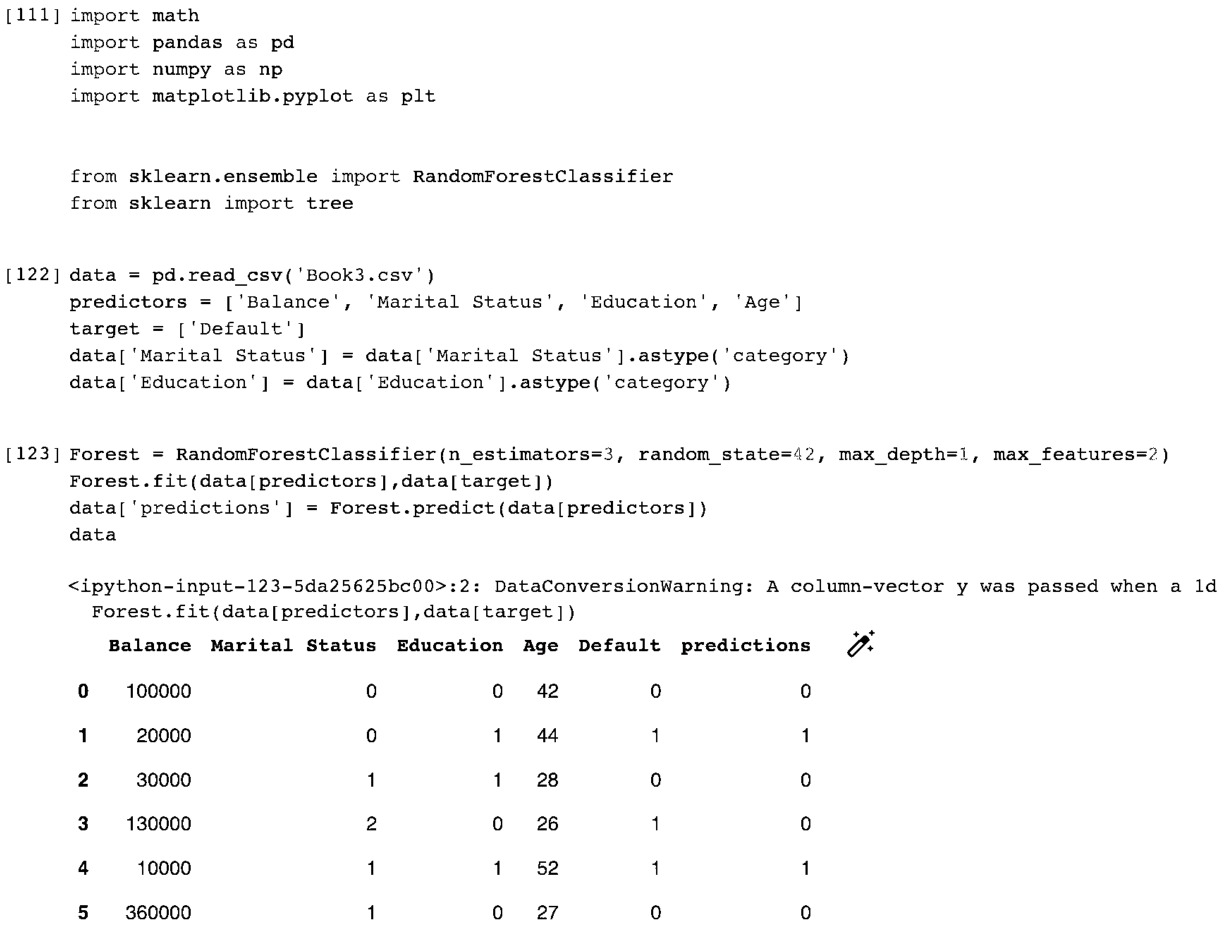
predicition on all instance

1	1
2	1
3	0
4	1
5	0
6	0

C.

	Tree 1	Tree 2	Tree 3	Prediction
1	0	0	1	False
2	1	1	1	True
3	0	1	0	False
4	0	0	1	False
5	1	1	0	True
6	0	0	0	False

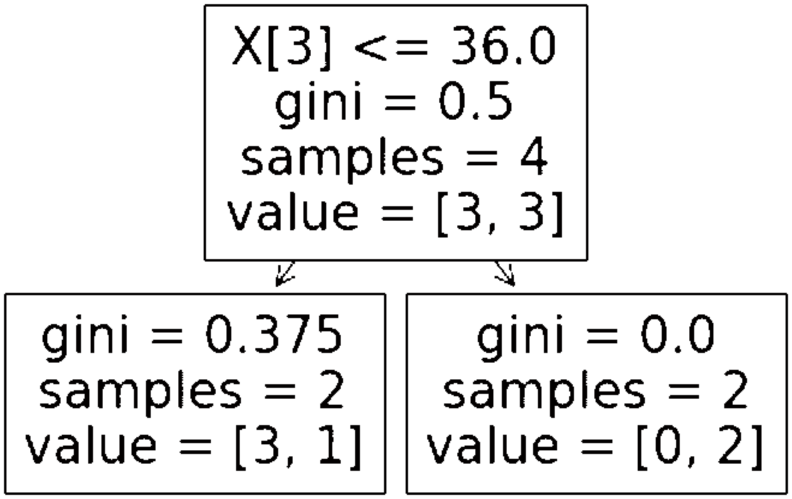
1d

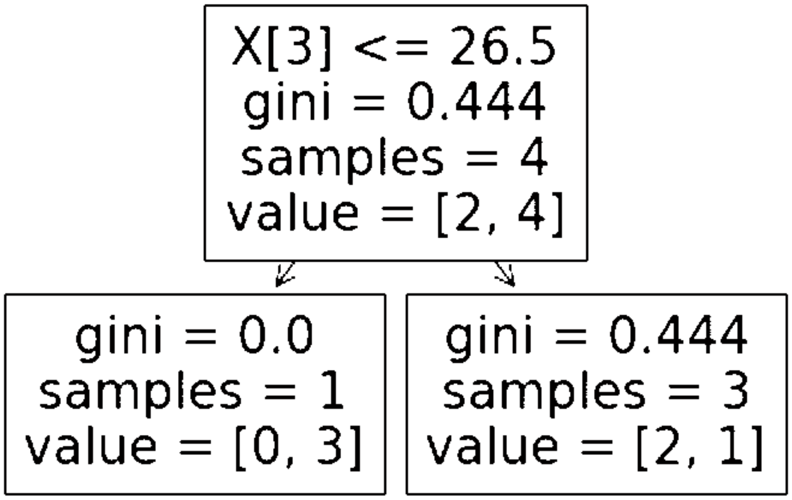


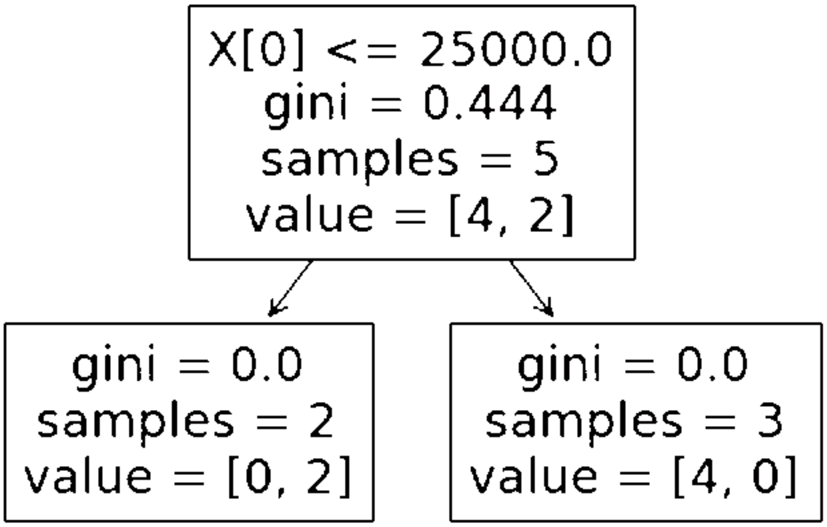
```
[111] import math
      import pandas as pd
      import numpy as np
      import matplotlib.pyplot as plt


      from sklearn.ensemble import RandomForestClassifier
      from sklearn import tree


[122] data = pd.read_csv('Book3.csv')
      predictors = ['Balance', 'Marital Status', 'Education', 'Age']
      target = ['Default']
      data['Marital Status'] = data['Marital Status'].astype('category')
      data['Education'] = data['Education'].astype('category')


[123] Forest = RandomForestClassifier(n_estimators=3, random_state=42, max_depth=1, max_features=2)
      Forest.fit(data[predictors],data[target])
      data['predictions'] = Forest.predict(data[predictors])
      data

      <ipython-input-123-5da25625bc00>:2: DataConversionWarning: A column-vector y was passed when a 1d
        Forest.fit(data[predictors],data[target])
```

```
   Balance  Marital Status  Education  Age  Default  predictions
0   100000               0          0   42        0            0
1    20000               0          1   44        1            1
2    30000               1          1   28        0            0
3   130000               2          0   26        1            0
4    10000               1          1   52        1            1
5   360000               1          0   27        0            0
```

```
[114] for i in Forest.estimators_:
        tree.plot_tree(i)
        plt.show()
```

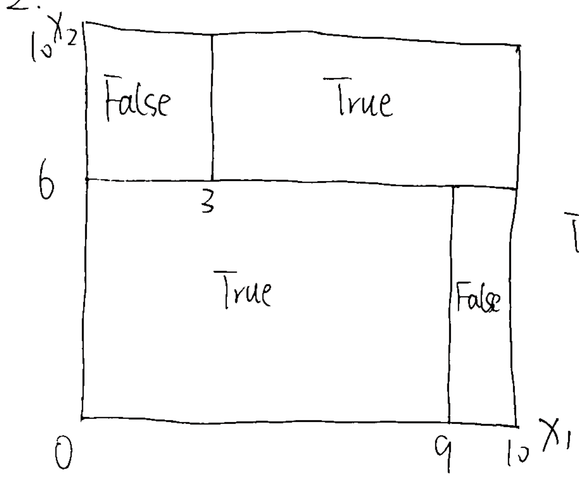




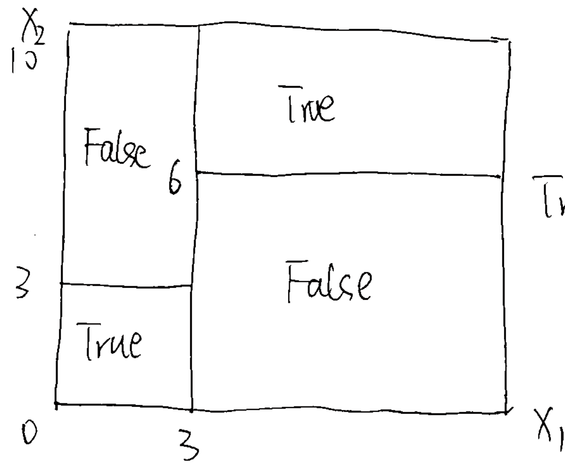

X[0] <= 25000.0
gini = 0.444
samples = 5
value = [4, 2]
gini = 0.0
samples = 2
value = [0, 2]
gini = 0.0
samples = 3
value = [4, 0]

2a

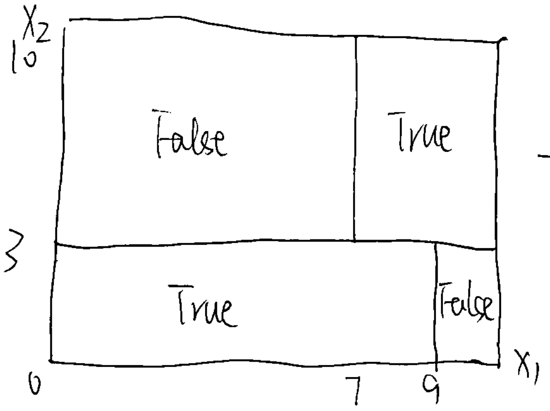
2.



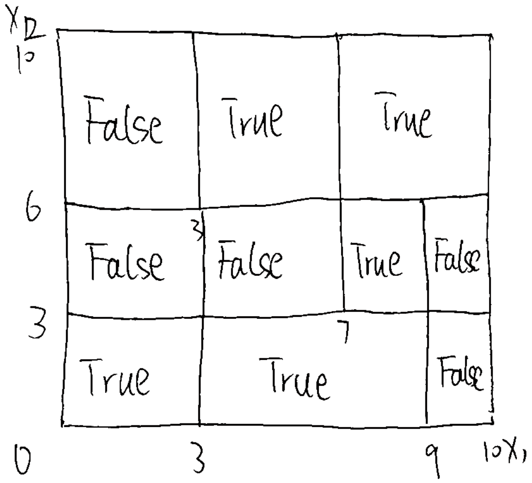
Tree 1



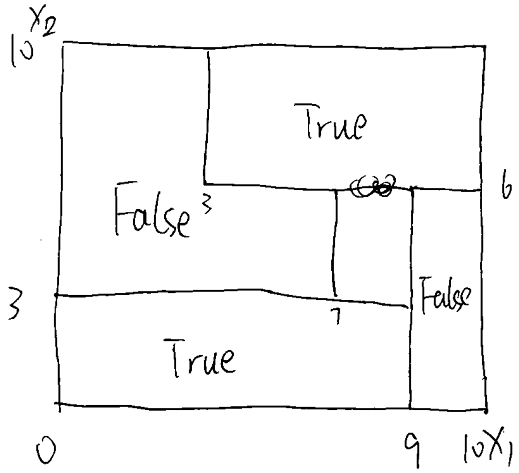
Tree 2



Tree3

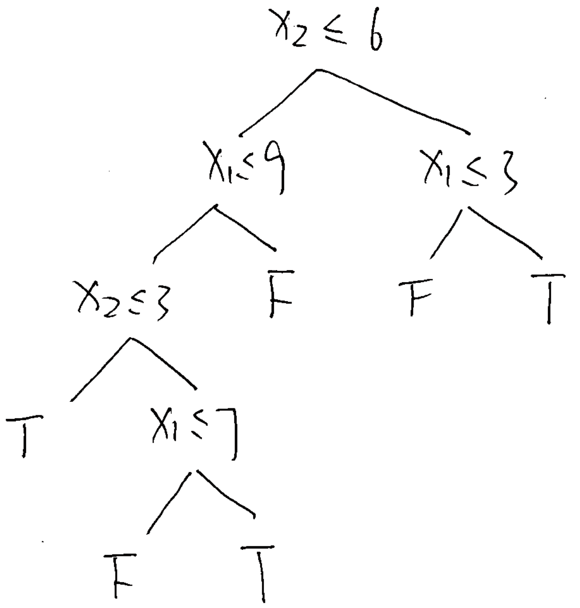


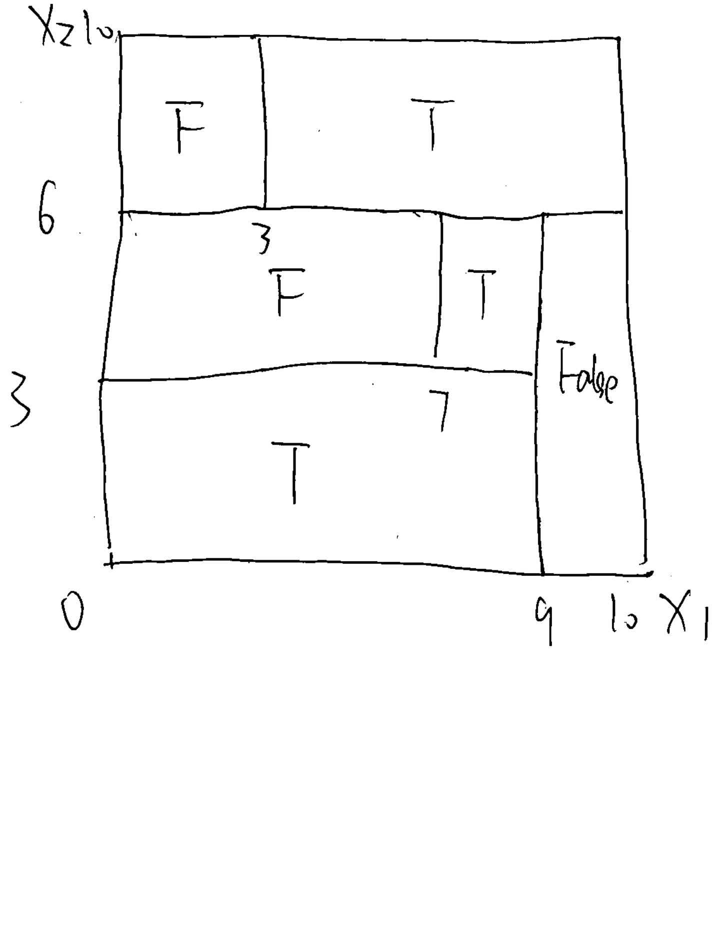
random forest



simplified randomforese partition

2.b.


$X_2 \le 6$
$X_1 \le 9$
$X_1 \le 3$
$X_2 \le 3$
F
F
T
T
$X_1 \le 7$
F
T


X2 10
6
3
F
T
3
F
T
False
7
T
0
9
10 X1

3. Increase

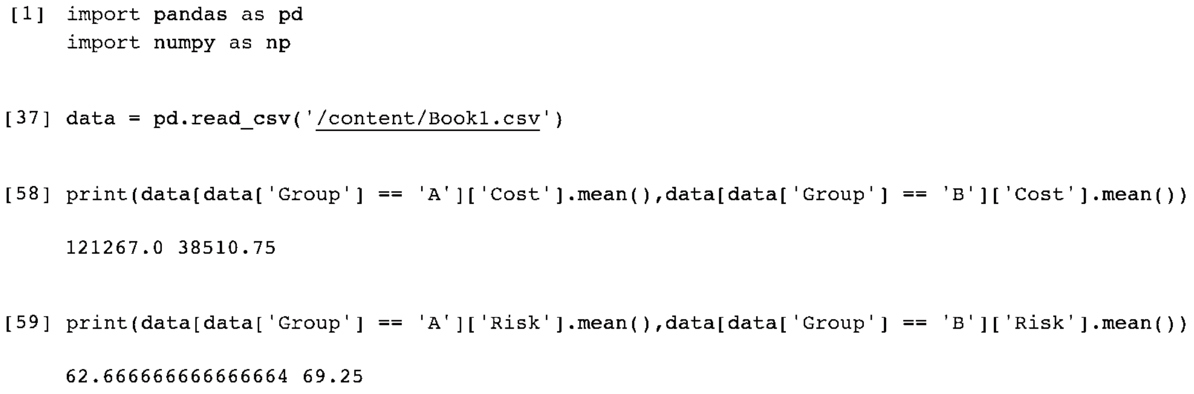
decrease

decrease

Increase

decrease

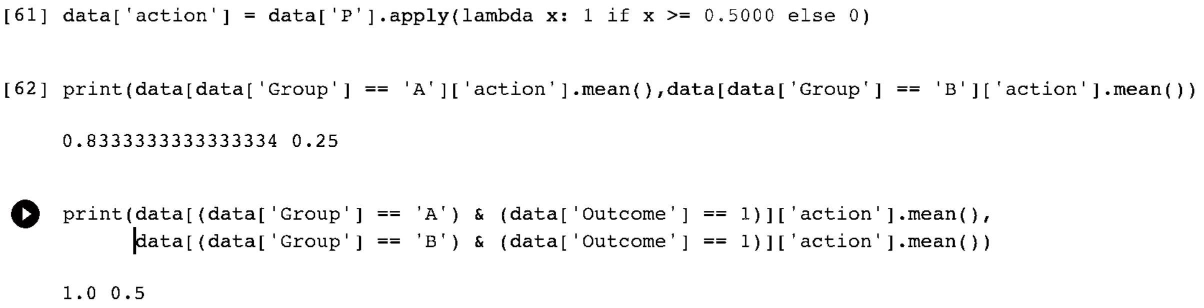
4a



```
[1]  import pandas as pd
     import numpy as np


[37] data = pd.read_csv('/content/Book1.csv')


[58] print(data[data['Group'] == 'A']['Cost'].mean(),data[data['Group'] == 'B']['Cost'].mean())

     121267.0 38510.75


[59] print(data[data['Group'] == 'A']['Risk'].mean(),data[data['Group'] == 'B']['Risk'].mean())

     62.666666666666664 69.25
```

Group A tends to have a higher health care cost, Group B tends to have a higher risk score, but the difference between risk score will be much smaller without outlier(A instance with only 46 risk score in group A)

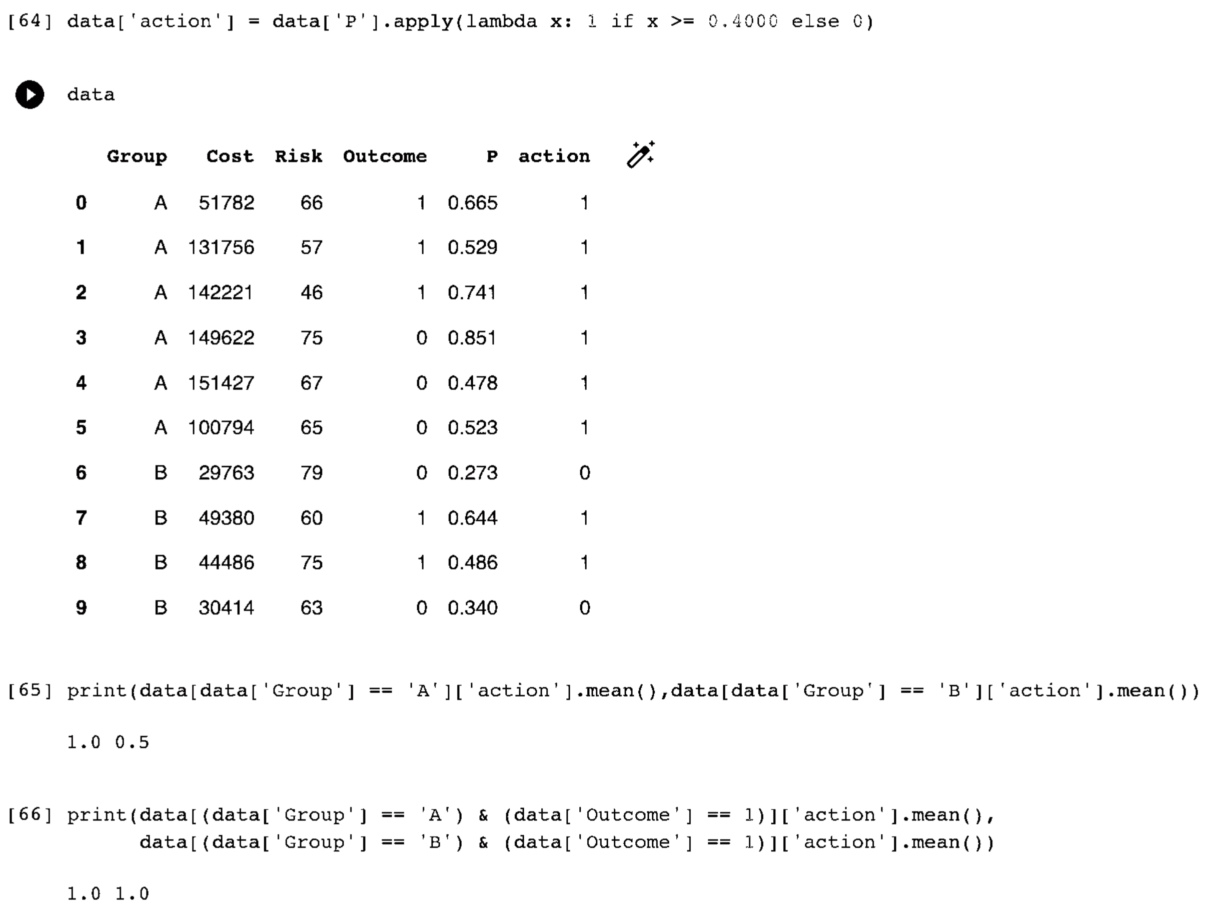
4b



```
[61] data['action'] = data['P'].apply(lambda x: 1 if x >= 0.5000 else 0)


[62] print(data[data['Group'] == 'A']['action'].mean(),data[data['Group'] == 'B']['action'].mean())

     0.8333333333333334 0.25


     print(data[(data['Group'] == 'A') & (data['Outcome'] == 1)]['action'].mean(),
           data[(data['Group'] == 'B') & (data['Outcome'] == 1)]['action'].mean())

     1.0 0.5
```

As we can see from the file that both true positive rate and positive rate is different for group A and B, therefore, demographic parity and equality of opportunity are both not satisfied.

4c



```
[64] data['action'] = data['P'].apply(lambda x: 1 if x >= 0.4000 else 0)
```

```
data
```

	Group	Cost	Risk	Outcome	P	action
0	A	51782	66	1	0.665	1
1	A	131756	57	1	0.529	1
2	A	142221	46	1	0.741	1
3	A	149622	75	0	0.851	1
4	A	151427	67	0	0.478	1
5	A	100794	65	0	0.523	1
6	B	29763	79	0	0.273	0
7	B	49380	60	1	0.644	1
8	B	44486	75	1	0.486	1
9	B	30414	63	0	0.340	0

```
[65] print(data[data['Group'] == 'A']['action'].mean(),data[data['Group'] == 'B']['action'].mean())

     1.0 0.5
```

```
[66] print(data[(data['Group'] == 'A') & (data['Outcome'] == 1)]['action'].mean(),
           data[(data['Group'] == 'B') & (data['Outcome'] == 1)]['action'].mean())

     1.0 1.0
```

Equal opportunity will be satisfied with same true positive rate if threshold is set to 0.40, but demographic parity will not be satisfied .

4d.

The sample size is too small, and the true distribution might be different in real world. Using this threshold might result in favor to a particular group. In addition, Imbalance and difference between group A and B in need of medical care will be perpetuated.