提出问题：使用遗传算法和未使用遗传算法的差异性是否显著?

实验步骤：

1、提出检验假设Ho:u=u0, H1:u≠u0；

2、下载数据，从MongoDB数据库上下载ads集合和insights集合，并将两表的ad_id数据字段的进行集合的合并；

3、判断数据样本是否使用遗传算法，使用遗传算法采用is_ga标记为1，未使用遗传算法is_ga标记未0；

4、讲is_ga相同的合并为新的数据集合，即使用遗传算法的数据集合和未使用遗传算法的数据集合；

5、分别从使用遗传算法和未使用遗传算法的数据集合随机不放回各抽取100条数据样本，分别计算出100条数据样本的总共花费(spend)、安装次数(install)和支付次数(pay)，最后计算出一个平均cpi，即cpi=sum(spend)/sum(install),cpi=sum(spend)/sum(pay);

6、重复步骤5，直到满足自己设定的数量，本次实验步骤5重复做了100次，产生了100个cpi；

7、根据独立样本t检验，计算出cpi的样本均值、样本容量和样本方差；

8、进行t检验，计算出t值，再计算出p值；

9、计算出的p值小于0.01，使用遗传算法和未使用遗传算法差别有非常显著意义。

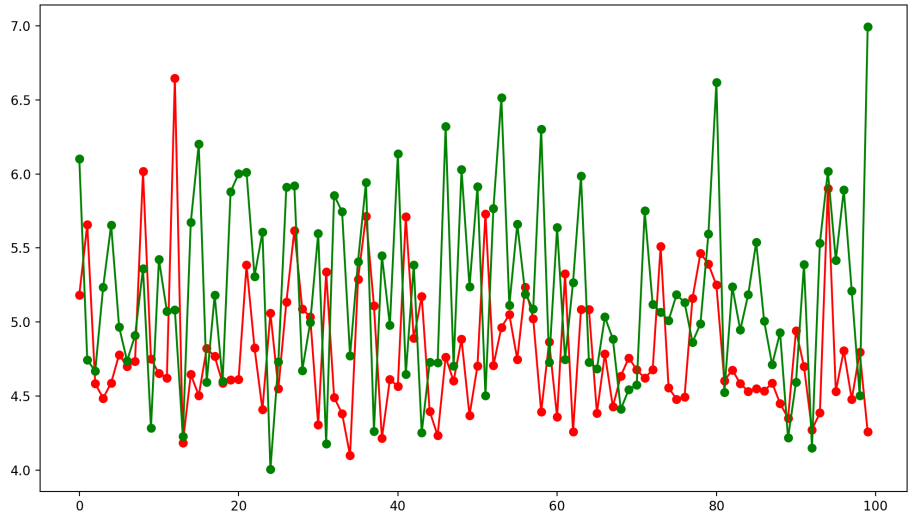
10、结论：使用遗传算法相比没有使用遗传算法更加平稳

p值参照表

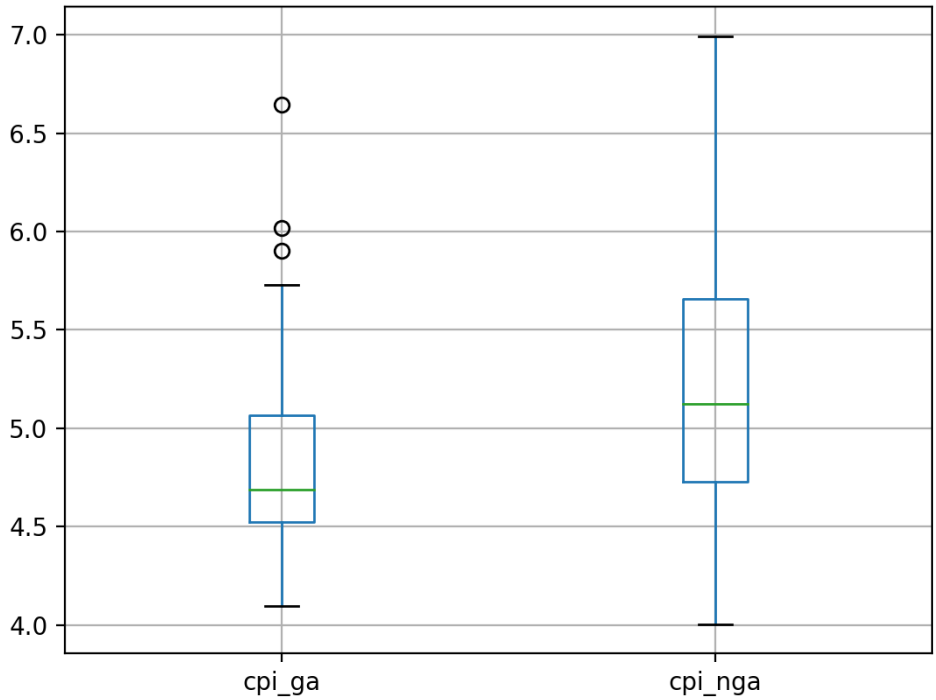
P值	碰巧概率	对无效假设	统计意义
P>0.05	碰巧出现的可能性大于5%	不能否定无效假设	两组差别无显著意义
P<0.05	碰巧出现的可能性小于5%	可以否定无效假设	两组差别有显著意义
P<0.0	碰巧出现的可能性小	可以否定无效	两组差别有非常显著

1	于1%	假设	意义

抽样数据产生的cpi：



利用产生的数据创建箱线图：



计算使用遗传算法和未使用遗传算法的cpi数据的均值和标准差：

使用遗传算法的均值和标准差：4.8106，0.4495

未使用遗传算法的均值和标准差：5.2068，0.6169

最终的p值：7.417e-07

p值含义参考

P值是用来判定假设检验结果的一个参数，也可以根据不同的分布使用分布的拒绝域进行比较。由R·A·Fisher首先提出。 P值（P value）就是当原假设为真时所得到的样本观察结果或更极端结果出现的概率。如果P值很小，说明原假设情况的发生的概率很小，而如果出现了，根据小概率原理，我们就有理由拒绝原假设，P值越小，我们拒绝原假设的理由越充分。总之，P值越小，表明结果越显著。但是检验的结果究竟是“显著的”、“中度显著的”还是“高度显著的”需要我们自己根据P值的大小和实际问题来解决。

附件：

if **name** == '**main**':
'''host=None,port=None,document_class=dict,tz_aware=None,connect=None, '''
client = MongoClient(host='34.218.190.195',port=27017) db = client.ai_explore
db.authenticate('app','DvNL8pEV8G5w2v6u@ai_explore') df = pd.DataFrame()

```
def load_ads(db):
    colle_ads = db.ads.find({},
{'_id':0,'ad_id':1,'pt.name':1},no_cursor_timeout=True)

    df_ads = pd.DataFrame()
    for item in colle_ads:
        dic = {}
        dic['ad_id'] = item['ad_id']
        dic['pt_name'] = item['pt']['name']
        df_ads = df_ads.append(pd.DataFrame(dic,index=[0]))

    df_ads.to_csv('ads.txt',index=False)

    return df_ads

df_ads = pd.DataFrame()
if os.path.exists('ads.txt'):
    df_ads = pd.read_csv('ads.txt')
```

```
else:
    df_ads = load_ads(db)
print(len(df_ads))  # 78746

def judge_res(cate,dt):
    if cate in dt:
        return 1
    else:
        return 0
df_ads['is_ga'] = df_ads.apply(lambda
x:judge_res('GA',x[1]),axis=1)
df_ads['is_ga1'] = df_ads.apply(lambda
x:judge_res('GA1',x[1]),axis=1)
df_ads['is_ga2'] = df_ads.apply(lambda
x:judge_res('GA2',x[1]),axis=1)

def load_insights(db):
    colle_insights = db.insights.find({},
{'_id':0,'ad_id':1,'spend':1,'install':1,'pay':1,'update':1})
    df_insights = pd.DataFrame()
    for item in colle_insights:
        df_insights = df_insights.append(pd.DataFrame(item,index=
[0]))
    df_insights.to_csv('insights.txt', index=False)
    return df_insights

df_insights = pd.DataFrame()
if os.path.exists('insights.txt'):
    df_insights = pd.read_csv('insights.txt')
else:
    df_insights = load_insights(db)
print(len(df_insights))  # 53390

df = pd.merge(df_ads,df_insights,how='inner',on="ad_id")
# print(len(set(df['ad_id'])))  #2215
'''  %Y-%m-%d %H:%M:%S  '''
def to_time(update):
    return datetime.datetime.strptime(update,'%Y-%m-%d %H:%M:%S')

# ['ad_id', 'pt_name', 'is_ga', 'is_ga1', 'is_ga2', 'spend',
'install','pay', 'update']
df['update'] = df.apply(lambda x:to_time(x['update']), axis=1)

# print(len(df))
# print(df.columns)
# print(df.head())
delta = datetime.timedelta(days=15)
```

```
df_15 = df[df['update']>=(max(df['update'])-delta)]
# df_ga = df[df['is_ga'] ==1][['ad_id','spend','install','pay']]
# df_ga = df_ga.groupby(['ad_id']).sum()
# df_nga = df[df['is_ga'] ==0][['ad_id','spend','install','pay']]
df_15 = df_15[['ad_id','is_ga','spend','install','pay']]
# df_15 = df_15.groupby(['ad_id','is_ga'],as_index=False).sum()

# print(df_15.head())
# print(df_15.shape)
df_15['spend_install'] = df_15['spend']/df_15['install']
df_15['spend_pay'] = df_15['spend']/df_15['pay']

df_15[df_15==np.inf] = 0

df_ga = df_15[df_15['is_ga']==1]
[['ad_id','spend','install','pay','spend_install','spend_pay','is_g
a']]
df_nga = df_15[df_15['is_ga']==0]
[['ad_id','spend','install','pay','spend_install','spend_pay','is_g
a']]

print('GA : ',df_ga.shape)  # GA :  (762, 7)
print('not GA: ',df_nga.shape)  # not GA:  (1453, 7)


def check_value(df_ga,df_nga):
    df_ga_stat = {}
    df_ga_stat['mean'] = np.mean(df_ga)
    df_ga_stat['median'] = np.median(df_ga)
    df_ga_stat['sum'] = np.sum(df_ga)
    df_ga_stat['std'] = np.std(df_ga)
    df_ga_stat['var'] = np.var(df_ga)

    df_nga_stat = {}
    df_nga_stat['mean'] = np.mean(df_nga)
    df_nga_stat['median'] = np.median(df_nga)
    df_nga_stat['sum'] = np.sum(df_nga)
    df_nga_stat['std'] = np.std(df_nga)
    df_nga_stat['var'] = np.var(df_nga)

    # print("GA:",df_ga_stat)
    # print("Not GA:",df_nga_stat)

    ttest_ind = stats.ttest_ind(df_ga,df_nga)
    # print(ttest_ind)

    print('\nT-tests: ')
```

```
    all_mean = (sum(df_ga)+sum(df_nga))/(len(df_ga)+len(df_nga))

    #  (x-u)/(std*sqrt(n))
    t_spend_install_ga = (df_ga_stat['mean']-
all_mean)/(df_ga_stat['std']/np.sqrt(len(df_ga)))
    pv_spend_install_ga = stats.t.sf(np.abs(t_spend_install_ga),
len(df_ga)-1)*2
    #  (x-u)/(std*sqrt(n))
    t_spend_install_nga = (df_nga_stat['mean']-
all_mean)/(df_nga_stat['std']/np.sqrt(len(df_nga)))
    pv_spend_install_nga = stats.t.sf(np.abs(t_spend_install_nga),
len(df_nga)-1)*2

    # print('GA的自由度为：',len(df_ga)-1)
    # print('t_spend_install_ga: ',t_spend_install_ga)
    # print('pv_spend_install_ga: ',pv_spend_install_ga)
    # print('not GA的自由度为：',len(df_nga)-1)
    # print('t_spend_install_nga: ',t_spend_install_nga)
    # print('pv_spend_install_nga: ',pv_spend_install_nga)


    print('\n独立样本t检验:')

    '''
         独立样本t检验
    Sw = ((m-1)S1^2+(n-1)S2^2)/(m+n-2)
    t = ((x-y)/(Sw*sqrt(1/m+1/n))

    x:第一个样本均值
    y:第二个样本均值
    m:第一个样本容量
    n:第二个样本容量
    S1^2:第一个样本方差
    S2^2第二个样本方差
    '''
    sw_si = ((len(df_ga)-1)*df_ga_stat['var']+
(len(df_nga)-1)*df_nga_stat['var'])/(len(df_ga)+len(df_nga)-2)
    t_spend_install = (df_ga_stat['mean']-
df_nga_stat['mean'])/np.sqrt(sw_si*
(1.0/len(df_ga)+1.0/len(df_nga)))
    pv_spend_install = stats.t.sf(np.abs(t_spend_install),
len(df_ga)+len(df_nga)-2)*2

    '''
    p值参照表
    p值             碰巧的概率              对原假设          统计意义
    P>0.05   碰巧出现的可能性不大于5%    不能否定原假设     两组差别无显著意义
    P<0.05   碰巧出现的可能性小于5%      可以否定原假设     两组差别有显著意义
```

```
    p<0.01  碰巧出现的可能性小于1%      可以否定原假设      两组差别有非常显著意义
    '''
    print('*'*100)
    print('t: ',t_spend_install)
    print('p-value: ',pv_spend_install)

cpi_install_ga = []
cpi_pay_ga = []

cpi_install_nga = []
cpi_pay_nga = []


for i in range(1,101):
    df_ga = df_ga.sample(frac=1)
    df_nga = df_nga.sample(frac=1)
    random_ga_index = random.sample(set(np.arange(len(df_ga))),100)
    random_nga_index =
random.sample(set(np.arange(len(df_nga))),100)
    df_ga_sample = df_ga.iloc[random_ga_index]
    df_nga_sample = df_nga.iloc[random_nga_index]

    cpi_install_ga.append(0 if df_ga_sample['install'].sum() == 0
else df_ga_sample['spend'].sum()/df_ga_sample['install'].sum())
    cpi_pay_ga.append(0 if df_ga_sample['pay'].sum() == 0 else
df_ga_sample['spend'].sum() / df_ga_sample['pay'].sum())

    cpi_install_nga.append(0 if df_nga_sample['install'].sum() == 0
else df_nga_sample['spend'].sum() / df_nga_sample['install'].sum())
    cpi_pay_nga.append(0 if df_nga_sample['pay'].sum() == 0 else
df_nga_sample['spend'].sum() / df_nga_sample['pay'].sum())

print(cpi_install_ga)
print(cpi_install_nga)

# # 随机不放回抽样产生的箱线图
def show_boxplot(cpi_ga,cpi_nga):
    box_ds = pd.DataFrame({'cpi_ga':cpi_ga,'cpi_nga':cpi_nga})
    box_ds.boxplot()
    plt.show()

show_boxplot(cpi_install_ga,cpi_install_nga)
show_boxplot(cpi_pay_ga,cpi_pay_nga)


print(df_ga['spend'].sum()/df_ga['install'].sum())
print(df_nga['spend'].sum()/df_nga['install'].sum())
```

```
print(df_ga['spend'].sum()/df_ga['pay'].sum())
print(df_nga['spend'].sum()/df_nga['pay'].sum())
print('-'*30)
print(np.mean(cpi_install_ga),np.std(cpi_install_ga))
print(np.mean(cpi_install_nga),np.std(cpi_install_nga))
print(np.mean(cpi_pay_ga),np.std(cpi_pay_ga))
print(np.mean(cpi_pay_nga),np.std(cpi_pay_nga))

check_value(cpi_install_ga, cpi_install_nga)

check_value(cpi_pay_ga, cpi_pay_nga)
```