西南财经大学
SOUTHWESTERN UNIVERSITY OF FINANCE AND ECONOMICS

数据科学
期末课程作业

作业名称：房贷美抵押贷款数据分析

学生姓名：李楷

所在学院：经济信息工程学院

专　　业：金融学

学　　号：219020204210

成　　绩：

2020 年　1　月

目录

一、 研究背景及目的

在 2008 年金融危机前，许多人购买房屋是为了投资目的，即他们期望房产价值迅速上升，以便从转售中获利，尽管他们从收入来说并不都能负担得起。2008 年金融危机发生后，贷款人在审查贷款申请时加大了收入核实或首期付款的强度。例如，对 FICO 评分的要求变得更高，需要进行收入核实。

因此本文主要研究目的是探寻 2008 年金融危机后来自于房贷美的抵押贷款数据的相关变化，并提出以下几个问题并试图验证：

1. 相较以金融危机前，贷款所需的信用评分变化。
2. 借款渠道的不同在其他数据上的体现。
3. 借款人群和借款行为的特征。
4. 数据之间的相关关系是怎样的。

以上问题的结论主要在第三和第四部分得到了解答。另外本文还就该数据分别使用了神经网络和逻辑回归进行了分类预测，使用了 OLS 探究变量之间的关系，使用了梯度提升树回归进行了回归，并得出了结论：数据内在的相关性强弱是影响机器学习任务效果好坏的根本因素。

二、 数据处理

（一）数据获取

数据源来自于房贷美官网在 2018-2019 的月度单个家庭的抵押贷款情况按季度下载，数据名依次为 historical_data1_Q12008.txt 至 istorical_data1_Q42009 .txt。

```
771|200803|N|203802||000|1|P|95|61|272000|80|5.875|R|N|FRM|CO|SF|81200
|F108Q1000001|N|360|01|Other sellers|USBANKNA|
729|200805|N|203804|17140|000|1|P|73|20|87000|73|6.5|R|N|FRM|OH|SF|45200
|F108Q1000002|C|360|01|Other sellers|Other servicers|
769|200803|N|203802||000|1|P|59|17|59000|59|6.375|R|N|FRM|KY|SF|40300
|F108Q1000003|C|360|01|Other sellers|Other servicers|
755|200803|Y|203802||35|1|P|100|28|81000|100|5.875|R||FRM|PA|SF|17900
|F108Q1000004|P|360|01|Other sellers|Other servicers|
```

由于单个数据集过大，用以下代码作了 18-19 年的数据合并并且进行了筛选，只选取信用评分在 550 至 850 之间，并且抵押资产为 ['PROPERTY TYPE '] == 'SF'，意为对应到单个家庭的 Single-Family 数据记录。

```
Q108 = pd.read_csv("/Users/scottlai/Desktop/work/FreddieMacProjects/cod
e/historical_data1_Q12008/Q12008.csv",
                            thousands=',')
x18 = Q108.loc[(Q108['PROPERTY TYPE '] == 'SF')& (Q108['Credit Score'] >
=550) & (Q108['Credit Score'] <=850)]
Q109 = pd.read_csv("/Users/scottlai/Desktop/work/FreddieMacProjects/cod
e/historical_data1_Q12009/Q109.csv",
                        thousands = ',')
x19 = Q109.loc[(Q109['PROPERTY TYPE '] == 'SF')& (Q109['Credit Score'] >
=550) & (Q109['Credit Score'] <=850)]
```

……省略以上过程

```
data.head()
data.to_csv('/Users/scottlai/Desktop/work/FreddieMacProjects/code/data.c sv') #存储表
格
```

因此在代码文件中，直接读取数据源的是生成后的单个 data.csv 文件，方便直接运行。

（二）数据变量说明

原始数据包含 2008-2009 年房贷美 Freddie Mac 对于房屋贷款购置的相关信息，其中包含是否第一次购置房屋，现行贷款违约状态，抵押贷款百分比等贷款数据及房屋购置数据，其中本研究用到的主要变量如下：

[1] ORIGINAL COMBINED LOAN-TO-VALUE (CLTV): 原始组合贷款价值

在购买抵押贷款的情况下，该比率是通过将票据日期的原始抵押贷款金额加上卖方披露的任何二级抵押贷款金额除以抵押财产在票据日或其购买价格上的评估价值。在再融资抵押贷款的情况下，该比率是通过将票据日期的原始抵押贷款金额加上卖方披露的任何二级抵押贷款金额除以抵押日期的抵押房地产的评估价值获得的。如果卖方披露的二次融资金额包括房屋净值信贷额度，那么 CLTV 计算反映了第一次留置权抵押贷款结束时的已支付金额，而不是房屋净值信贷额

度下可用的最高贷款金额。对于经验丰富的抵押贷款，如果卖方不能保证抵押财产的价值自注释日期以来没有下降，则 Freddie Mac 要求卖方必须提供新的评估值，用于 CLTV 计算。在某些情况下，如果卖方向 Freddie Mac 提供贷款，并附有指示额外二级抵押贷款金额的特殊代码，则这些金额可能已包含在 CLTV 计算中。

公式如下：

CLTV Formula and Calculation

$$\text{CLTV} = \frac{\text{VL1} + \text{VL2} + \ldots + \text{VLn}}{\text{Total Value of the Property}}$$

where:

$$\text{VL} = \text{Value of loan}$$

例子如下：

Calculating CLTV

First mortgage balance	$90,000
Second mortgage balance	$70,000
Drawn balance on HELOC	$5,000
Sum of all loans and lines of credit	$165,000
Lesser of home sales price or appraised value	$200,000
Sum of loans divided by value = CLTV	82.5%

[2] ORIGINAL DEBT-TO-INCOME (DTI)：原始债务收入比率

债务与收入比率的披露基于（1）借款人每月债务支付的总和，包括纳入借款人在当时支付的抵押支付的月度住房支出。 将抵押贷款交付给房地美，除以（2）用于在该贷款发起之日承保贷款的每月总收入。

[3] ORIGINAL LOAN-TO-VALUE (LTV): 原始贷款价值

在购买抵押贷款的情况下，通过将票据日期的原始抵押贷款金额除以抵押日期或其购买价格中抵押房产的评估价值中的较小者而获得的比率。在再融资抵押贷款的情况下，通过将票据日期的原始抵押贷款金额与抵押财产在评估日的评估价值除以获得的比率。对于经验丰富的抵押贷款，如果卖方不能保证抵押财产的价值自注明日期以来没有下降，则 Freddie Mac 要求卖方必须提供新的评估值，

用于 LTV 计算。

组合贷款与价值（CLTV）比率是一个财产上的所有担保贷款与一个财产价值的比率。贷款人使用 CLTV 比率来确定潜在的购房者在使用多笔贷款时的违约风险。一般来说，贷款人愿意以 80%及以上的 CLTV 比率向信用评级较高的借款人贷款。CLTV 不同于简单贷款与价值（LTV）比率，因为 LTV 在计算中只包括第一或主要抵押贷款。对应公式如下：

$$LTV\,ratio = \frac{MA}{APV}$$

where:

$$MA = \text{Mortgage Amount}$$

$$APV = \text{Appraised Property Value}$$

以下为例子：

Calculating LTV

Outstanding mortgage balance	$160,000
Sales price or appraised value of your home	$200,000
Loan balance divided by value = LTV	80%

[4] Credit Score 个人信用

信用评级得分比较低。美国的信用评级公司 (FICO)将个人信用评级分为五等：优(750～850 分)，良(660～749 分)，一般(620～659 分)，差(350～619 分)，不确定(350 分以下)。次级贷款的借款人信用评分多在 620 分以下，除非个人可支付高比例的首付款，否则根本不符合常规抵押贷款的借贷条件。

[5]CHANNEL 贷款渠道

[6]Monthly report period 数据记录所在的日期

使用 dropna 去除空缺值。最终打印 data.head()如下。后文还会有对数据作变换以作特定用途，会在具体地方表示再说明。

```
In [5]: data.head()
```

Out[5]:

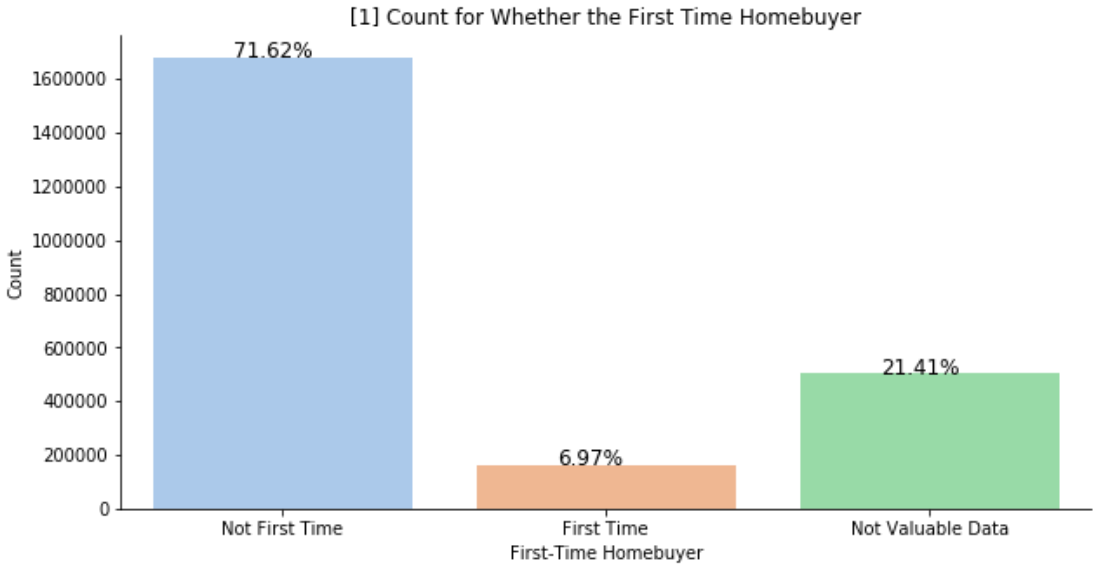
	Unnamed: 0	Unnamed: 0.1	Credit Score	Monthly report period	FIRST TIME HOMEBUYER FLAG	CURRENT LOAN DELINQUENCY STATUS	X5	MORTGAGE INSURANCE PERCENTAGE (MI %)	NUMBER OF UNITS	OCCUPANCY STATUS	...	PROPERTY TYPE	X19	Loa Sequenc Numb
0	0	0	771	200803	N	203802	NaN	0	1	P	...	SF	81200.0	F108Q10000(
1	1	1	729	200805	N	203804	17140.0	0	1	P	...	SF	45200.0	F108Q10000(
2	2	2	769	200803	N	203802	NaN	0	1	P	...	SF	40300.0	F108Q10000(
3	3	3	755	200803	Y	203802	NaN	35	1	P	...	SF	17900.0	F108Q10000(
4	4	4	760	200804	N	203803	48300.0	0	1	P	...	SF	98800.0	F108Q10000(

5 rows × 29 columns

三、 数据描述性分析

（一）是否为第一次购房

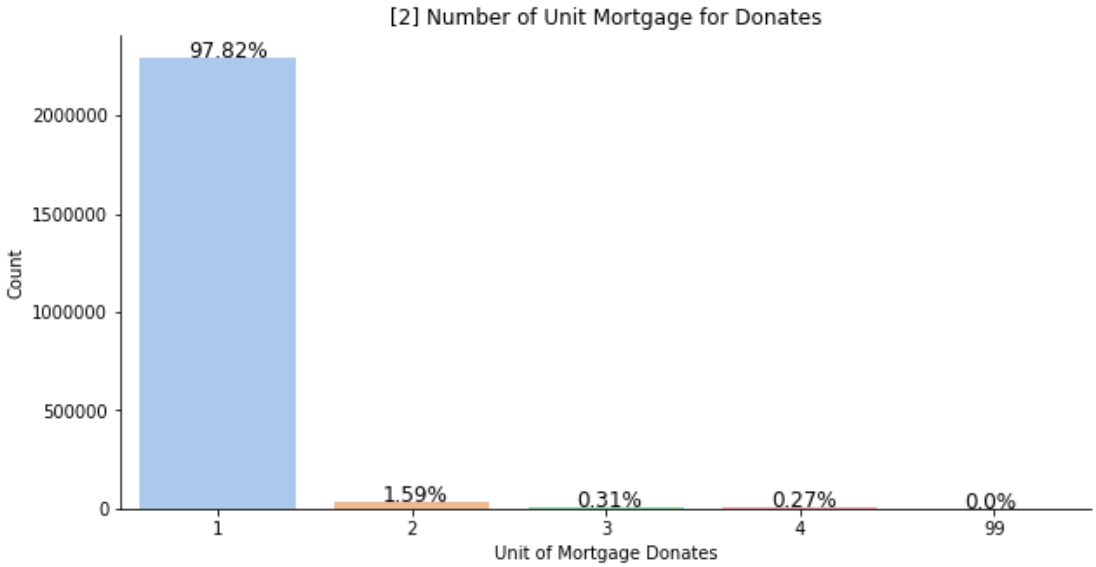
```
# 1）是否为第一次购房

data['FIRST TIME HOMEBUYER FLAG '].replace('N', 'Not First Time',inplace = True) #将数据中的 N替换成Not First Time
data['FIRST TIME HOMEBUYER FLAG '].replace('Y', 'First Time',inplace = True)
data['FIRST TIME HOMEBUYER FLAG '].replace('9', 'Not Valuable Data',inplace = True)

#做图
fig, firsthouse = plt.subplots(figsize=(10,5)) #图大小设置


sns.set_palette('pastel') #设置图的格式及颜色为Pastel
sns.countplot(x=data['FIRST TIME HOMEBUYER FLAG '], data=data) #做频率图
data

firsthouse.set_title('[1] Count for Whether the First Time Homebuyer') #改表头
firsthouse.set_ylabel('Count') #改纵坐标
firsthouse.set_xlabel('First-Time Homebuyer') #改横坐标

firsthouse.spines['right'].set_visible(False) # 去除右边的边框
firsthouse.spines['top'].set_visible(False)

#设置百分比
totals = []

for i in firsthouse.patches:
    totals.append(i.get_height())


total = sum(totals)

for i in firsthouse.patches:
    firsthouse.text(i.get_x() +.25, i.get_height()+20,
            str(round((i.get_height()/total)*100,2))+"%",fontsize = 12,
            color = "black")
plt.show()
```



上图显示抵押贷款的家庭单位是否为第一次购买房屋的分布数据，根据图像显示，有近 71.62%的借贷者不是第一次购买房屋，而只有 6.97% 的借贷者为第一次购买房屋。而数据有接近 21.41%的无效数据。

（二）用多少房子做抵押

```
fig, firsthouse = plt.subplots(figsize=(10,5)) #图大小设置


sns.set_palette('pastel')#设置图的格式及颜色为Pastel
sns.countplot(x=data['NUMBER OF UNITS '], data=data)#做频率图


firsthouse.set_title('[2] Number of Unit Mortgage for Donates') #设置标题
firsthouse.set_ylabel('Count') #Y坐标名
firsthouse.set_xlabel('Unit of Mortgage Donates') #x坐标名


firsthouse.spines['right'].set_visible(False) # 去除右边框
firsthouse.spines['top'].set_visible(False) #去除上边框

#设置每一列bar的百分比函数
totals = []

for i in firsthouse.patches:
    totals.append(i.get_height())


total = sum(totals)

#设置位置百分比位置
for i in firsthouse.patches:
    firsthouse.text(i.get_x() +.25, i.get_height()+20, \
            str(round((i.get_height()/total)*100,2))+"%",fontsize = 12,
            color = "black")
plt.show()
```



从图【2】中可看出 97.82%的借贷着只有一座房子作为贷款抵押。

（三）信用分数基本分布

```
fig, FICO = plt.subplots(figsize=(10,5)) #图小大小设置
sns.set_palette('dark') #设置图像颜色格式为dark
sns.boxplot(x = data['Credit Score'],color = 'pink') # 做箱型图

x = data['Credit Score']
FICO.set_title("[3] Distribution of Borrower's Credit Score") #设置标题

FICO.set_xlabel('Credit Score') #设置想坐标名称

#去除边框（左右上）
FICO.spines['right'].set_visible(False)
FICO.spines['top'].set_visible(False)
FICO.spines['left'].set_visible(False)

print('平均数：',x.median())
print('')
print('下四分位数：',x.quantile(0.25))
print('')
print('中位数:',x.quantile(0.5))
print('')
print('上四分位数：',x.quantile(0.75))
print('')
print('最小值：',x.min())
print('')
print('最大值：',x.max())
```

平均数： 764.0

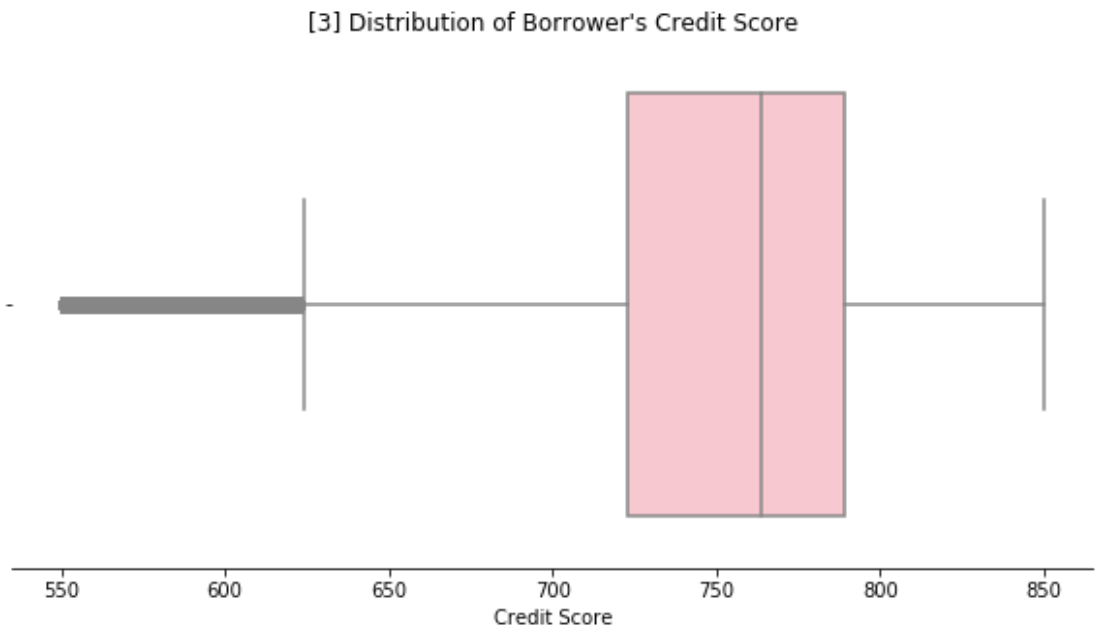
下四分位数： 723.0

中位数: 764.0

上四分位数： 789.0

最小值： 550

最大值： 850



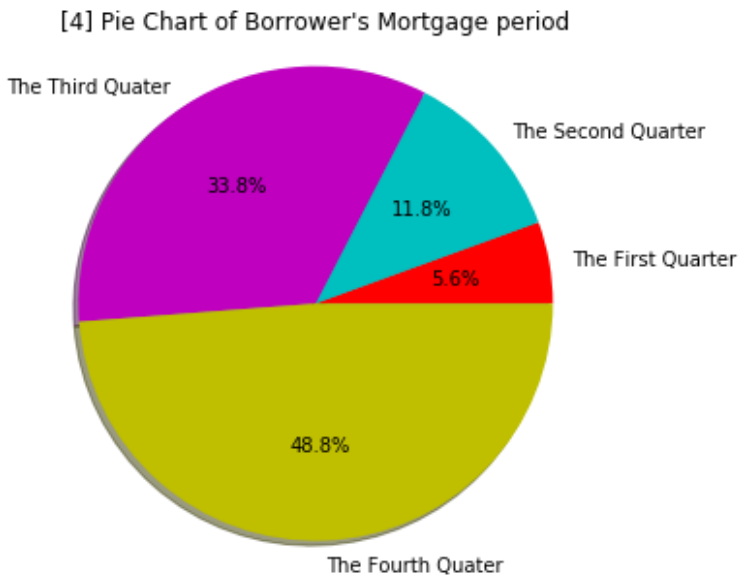
关于借贷人信用分数的分布数据图，结合箱形图及数据概括，我们可以看出，借贷人的平均信用分数为 752 分左右，大部分人（25%-75%）的信用分数集中在 723 到 789 分之间，贷款人信用分最低为 550 分，最高接近 850.

（四）信用分数基本分布

先转换数据库中的贷款时间为对应一年中的 4 个季度，统计其综合，画出饼图。

```
#data['Monthly report period'].replace({200803:3, 200903:3, 200804:4, 200904:4,
       #                                200801:1, 200901:1, 200802:2, 200902:2}, inplace = True)
#(data['Monthly report period']== 1).sum() #31490
#(data['Monthly report period']== 2).sum() #67074
#(data['Monthly report period']== 3).sum() #191467
#(data['Monthly report period']== 4).sum() #276164

#计算各个季度的占比。
```

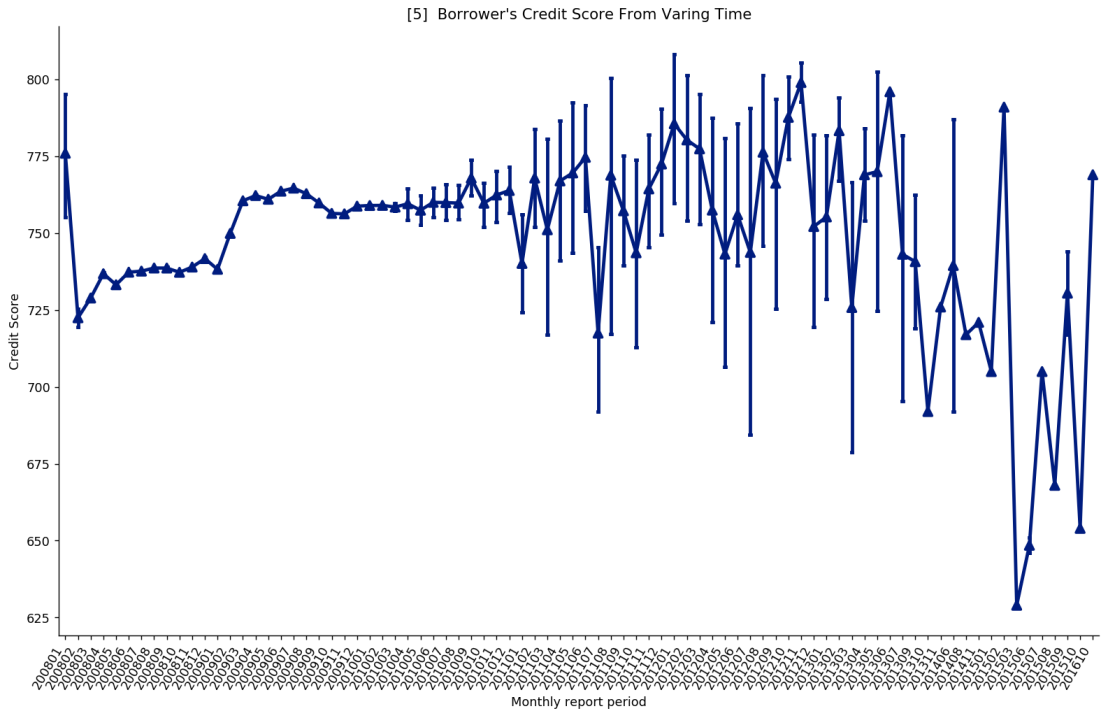
```
sizes = [31490,67074,191467,276164]#各个季度占比

labels = ['The First Quarter', 'The Second Quarter','The Third Quater','The Fourth Quater'] #设置季度名称

fig, ax = plt.subplots(figsize=(10,5)) #设置图像大小

sns.set_palette('pastel') #设置图格式为pastel
ax.pie(sizes,labels = labels,autopct = '%1.1f%%',shadow = True, colors=('r', 'c', 'm', 'y', 'k', 'w')) #做饼图
ax.axis('equal')


ax.set_title("[4] Pie Chart of Borrower's Mortgage period")

plt.show()
```



从饼图中可以看出有 48.8%的借贷者都在第四季度（9-12 月）间进行借贷。第一季度借贷人数最少，只有 5.6%

（五）信用分数随时间变化的分布

```
fig, FICO = plt.subplots(dpi=128,figsize=(15,10)) #图小大小设置
sns.set_palette('dark') #设置图像颜色格式为dark

FICO.set_title("[5]  Borrower's Credit Score From Varing Time") #设置标题

FICO.set_xlabel('Credit Score') #设置想坐标名称
fig.autofmt_xdate(rotation=60)

#scale_ls = range(5)
#index_ls = ['2008-1', '2008-7','2009-1', '2009-7', '2009-12']
#plt.bar(scale_ls, index_ls)
#plt.xticks(scale_ls,index_ls)
#_ = plt.xticks(scale_ls,index_ls) ## 可以设置坐标字
#FICO.set_xticks(['2008-1', '2008-7','2009-1', '2009-7', '2009-12'])
#去除边框（左右上）



FICO.spines['right'].set_visible(False)
FICO.spines['top'].set_visible(False)
FICO.spines['left'].set_visible(True)
#
# for label in FICO.get_xticklabels():
#      label.set_visible(False)

i=0
for label in FICO.get_xticklabels():  ##防止标签堆叠
    i=i+1
    if i%2==0:
        label.set_visible(True)



sns.pointplot(x=data['Monthly report period'], y=data['Credit Score'],
markers=["^", "o"], linestyles=["-", "--"],capsize=0.1)
```



可以从上图看出借款人对应的信用分数在 2008 年金融危机正盛的时候要求

很严格，2008 年信用分数稳定在 720-800 区间，而在经济恢复后，只要大于 600 就有机会获得贷款。这是先前探讨问题的重大结论，也就是说金融危机后，贷款的特征已经发生了比较大的变化。

四、 相关性分析

在这部分内容中采取元数据采样 30000 个记录的做法，以方便可视化同时不失一般性。

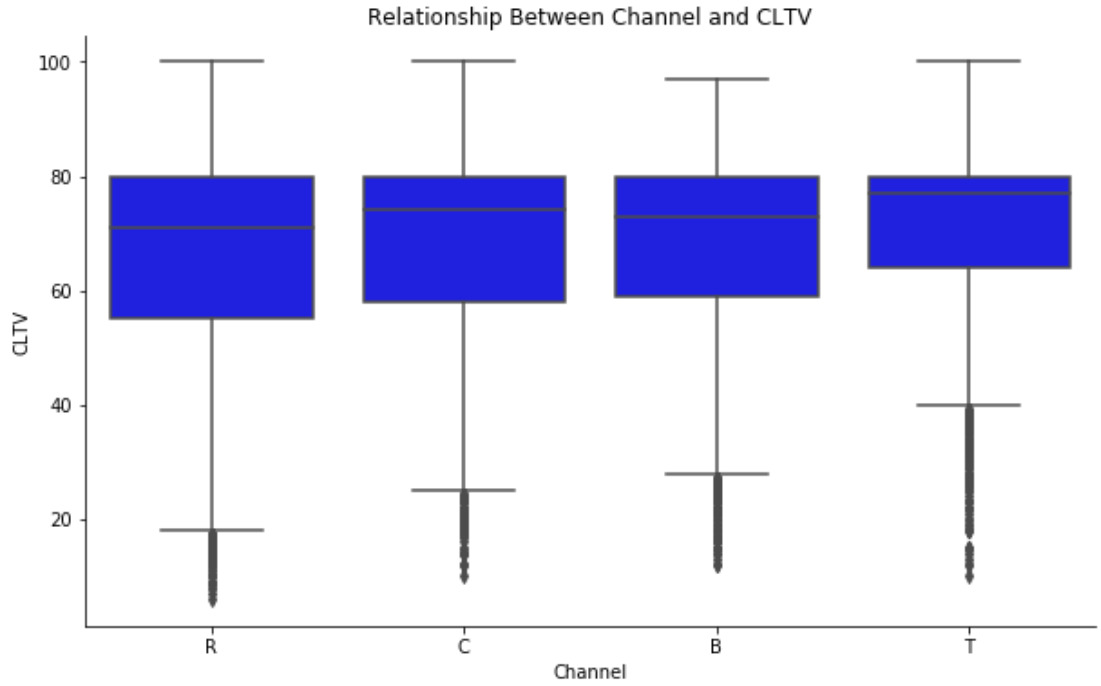
```
dataq = data.sample(n = 30000) #从data中提取30000个随机数据
data.rename(columns={'ORIGINAL DEBT-TO-INCOME (DTI) RATIO ':'DTI','ORIGINAL LOAN-TO-VALUE (LTV) ':'LTV'}, inplace=True)
data.rename(columns={'ORIGINAL COMBINED LOAN-TO-VALUE (CLTV) ':'CLTV','Monthly report period':'Time','Credit Score':'CS','CHANNEL ':'Channel'},

pd.options.display.max_columns = None
dataq.head(100)
```

```
dataq['CLTV']=dataq[['CLTV']][dataq['CLTV']<=100]
```

```
print(dataq['Channel'], dataq['CLTV'])
max(dataq['CLTV'])
dataq['DTI']=dataq[['DTI']][dataq['DTI']<=100]
dataq['LTV']=dataq[['LTV']][dataq['LTV']<=100]
```

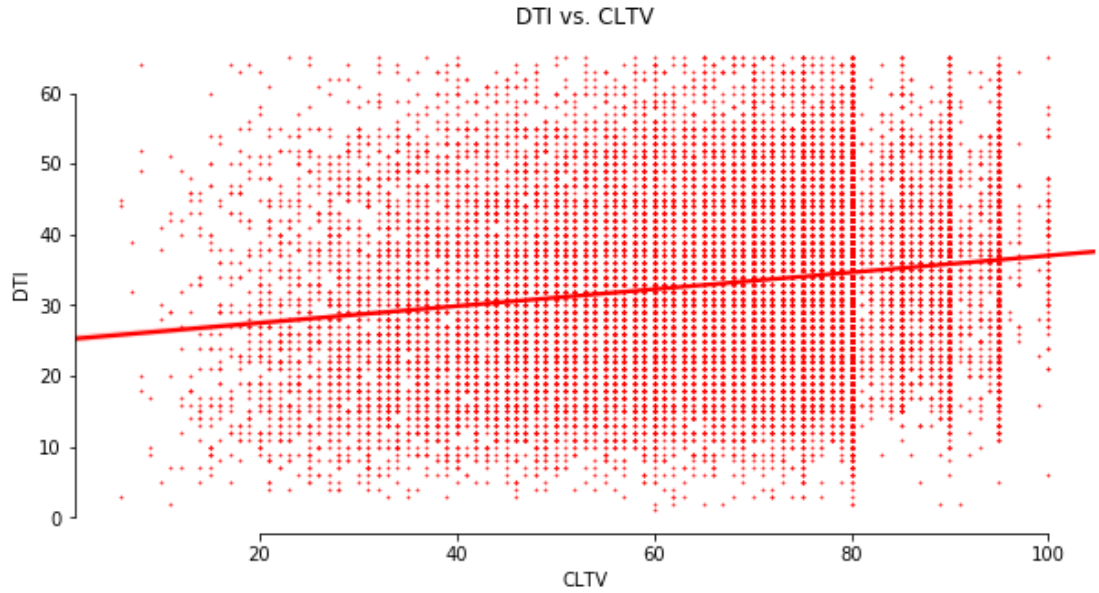
由于有一个异常点记录 CLTV 是在 1000 的水平，故去掉 CLTV 大于 100 的异常值，因为绝大多数 CLTV 都是在大于 0 到 100 的区间。

```
fig, ax = plt.subplots(figsize=(10,6))

#sns.despine(fig,left = True, bottom = True)

#clarity_ranking = ['R','B','C','T']#设置X坐标值
sns.boxplot(dataq['Channel'], dataq['CLTV'], color='blue')  # 做箱型图

ax.set_title('Relationship Between Channel and CLTV')
ax.set_ylabel('CLTV')
ax.set_xlabel('Channel')

ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)

plt.show()
```



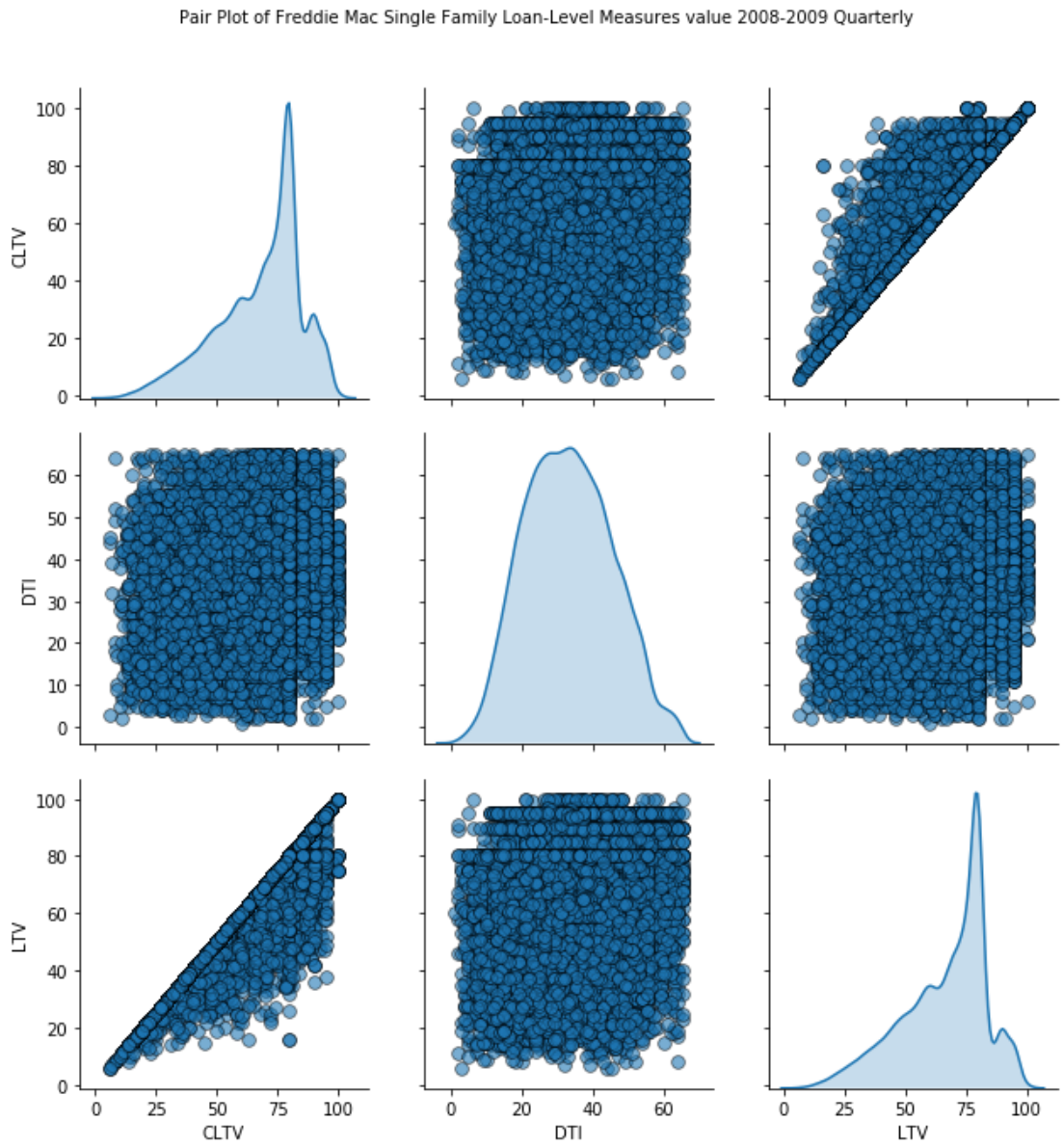
最终结果如上，横轴的变量表示为 R = Retail 零销商 B = Broker 表示经纪人，C = Correspondent 表示代理人，T = TPO Not Specified 表示未知但是明确是经过第三方的贷款。在这些渠道中，未指明 T 来源的贷款的 CTLV 最高，其次是代理人，最低的是来自零销商渠道的。

```
fig, ax = plt.subplots(figsize=(10,5))

#seaborn regplot命令。不会在轴对象上调用它（例如在matplotlib中），
#但是我们可以将其传递给轴，这样我们就可以进行类似matplotlib的调整。

sns.regplot(x=dataq['CLTV'], y=dataq['DTI'],  # 数据
            ax = ax,                                          # ax值
            color = 'r',                                    # 颜色
            ci = 95,
           marker='*',scatter_kws={"s": 1}
    )                                          # 置信区间confidence interval: in（%）

sns.despine(ax = ax, trim=True)    #去掉上边和右边的边框


# Our usual labeling
ax.set_title('DTI vs. CLTV')
ax.set_ylabel('DTI')
ax.set_xlabel('CLTV')
# plt.ylim(ymax = 5)
# plt.ylim(ymin= -5)
plt.show()
```



利用 seaborn 的 regplot 方法在置信水平为 95 的下绘制 DTI 与 CLTV 的关系，可以看出 DTI 与 CLTV 之间呈现正向关系。其实直觉上也很好理解，债务对应贷款水平提高了，那么债务对收入的占比也就提高了。但是这个结果从图像来看并不是非常直接。尤其是在以下的矩阵图中可以发现。

```
df = dataq.loc[:,['CLTV','DTI',"LTV"]]#palette=sns.hls_palette(1,l=.7,s=.9)
sns.pairplot(df,diag_kind= 'kde',plot_kws = {'alpha':0.6,'s':60,'edgecolor':'k'},height = 3) #做匹配图
plt.suptitle('Pair Plot of Freddie Mac Single Family Loan-Level Measures value 2008-2009 Quarterly',size = 10,y = 1.05)#设置标题
plt.show()
```



上图所展示了 CLTV，DTI，及 LTV 三者的矩阵图，可见 CLTV & DTI 和 DTI & LTV 的相关关系没那么明显，而 CLTV & LTV 关系明显为正向相关。同时可以看出三个数值的均值分布。CLTV 大约为 75， DTI 大约为 30，LTV 则接近 80,CLTV 和 LTV 均为左偏，峰度较高，而 DTI 的峰度更低，偏度较小。在后文的分析中我们会发现，数据相关性较高的机器学习任务效果会很好，相反，无强烈相关性的机器学习趋于失效。

五、 回归分析

在这部分内容中，主要是使用 OLS 方法对变量之间的数量关系再深入探究。

我们首先考虑基本线性模型

```
y ~ x1 + x2
```

概括方程式:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \epsilon.$$

在做回归分析前，需要对数据标准化，消除量纲大小不一样带来的影响。

```
data = pd.read_csv('data.csv',low_memory=False) # 导入数据
data['FIRST TIME HOMEBUYER FLAG '].replace('N', 'Not First Time',inplace = True) #将数据中的 N替换成Not First Time
data['FIRST TIME HOMEBUYER FLAG '].replace('Y', 'First Time',inplace = True)
data['FIRST TIME HOMEBUYER FLAG '].replace('9', 'Not Valuable Data',inplace = True)
```

```
#简化columns的名称
data.rename(columns={'ORIGINAL DEBT-TO-INCOME (DTI) RATIO ':'DTI','ORIGINAL LOAN-TO-VALUE (LTV) ':'LTV'}, inplace=True)
data.rename(columns={'ORIGINAL COMBINED LOAN-TO-VALUE (CLTV) ':'CLTV','Monthly report period':'Time','Credit Score':'CS','CHANNEL ':'Channel'},
```

```
data = data[data['CLTV'] != 999]#提取CLTV中不是999（无限）的变量
data = data[data['DTI'] != 999]
data = data[data['LTV'] != 999]
dataq['CLTV']=dataq[['CLTV']][dataq['CLTV']<=100]
dataq['DTI']=dataq[['DTI']][dataq['DTI']<=100]
dataq['LTV']=dataq[['LTV']][dataq['LTV']<=100]
adddata=data

stand_data=data.loc[:,['CLTV','DTI','LTV','CS','Time']]
stand_data
```

数据标准化

```
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import MinMaxScaler
ss = StandardScaler()
stand_data2 = ss.fit_transform(stand_data)
origin_data = ss.inverse_transform(stand_data2)
print('data is ',stand_data)
print('after standard ',stand_data2)
print('after inverse ',origin_data)
print('after standard mean and std is ',np.mean(stand_data), np.std(stand_data))
stand_data2=pd.DataFrame(stand_data2)

stand_data2.columns = ['CLTV','DTI','LTV','CS','Time']
stand_data2
```

```
data=stand_data2
data['Channel']=adddata['Channel']
#data['Time']=adddata['Time']
#data['FLAG']=adddata.iloc[:,6]#FIRST TIME HOMEBUYER FLAG

data['FLAG'] = adddata.iloc[:,4].loc[adddata.iloc[:,4]!='Not Valuable Data'] #去除无效数据
data['FLAG'] = np.where(data['FLAG'] == 'First Time',1,-1)
dataq=data
```

```
#data['Channel'][data['Channel']=='B']
```

```
dataq  ## 对CLTV-CS标准化后加入类别的数据
```

数据处理后的结果如下：

```
dataq  ## 对CLTV-CS标准化后加入类别的数据
```

	CLTV	DTI	LTV	CS	Time	Channel	FLAG
0	1.553469	2.259483	0.767012	0.399944	-1.359789	R	-1
1	0.290826	-1.079900	0.366215	-0.504194	-1.324984	R	-1
2	-0.512674	-1.324245	-0.435378	0.356890	-1.359789	R	-1
3	1.840433	-0.428313	1.912146	0.055510	-1.359789	R	1
4	0.348219	2.015138	0.423472	0.163146	-1.342386	R	-1
...	...	...	...	...	...	...	...
2321629	0.635183	-0.265416	0.709755	0.055510	2.155516	R	-1
2321630	1.553469	-1.161348	1.625862	-0.267396	2.155516	R	-1
2321631	1.668255	0.711964	1.740376	-0.956264	2.120711	R	-1
2321632	0.692576	-0.591210	0.767012	-1.451387	2.259931	R	-1
2321633	-0.053531	1.037757	0.022675	0.442998	2.155516	C	-1

2321634 rows × 7 columns

在这部分内容中需要注意的是除了(4),方程的 R 方系数都比较低，但自变量对于解释因变量大多都十分显著，主要从系数中获得变量之间关系的意义。

以下研究内容分为五个部分。

（一）关于 CTLV 的线性回归

（1）如果假设因变量(y)为CLTV，自变量则为时间(x_1)，信用分数(x_2)及渠道(x_3)。

则有：

$$CLTV = \beta_0 + \beta_1 Time + \beta_2 CreditScore + \beta_3 Channel + \epsilon.$$

```
y, X = patsy.dmatrices('CLTV~ Time + CS + Channel', data) #传递模型公式和相关数据以创建设计矩阵
cltv = sm.OLS(y, X) #将设计矩阵传递给OLS以指定普通的最小二乘模型
res = cltv.fit()#估计模型并将结果存储在res中
print(res.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                   CLTV   R-squared:                       0.029
Model:                            OLS   Adj. R-squared:                  0.029
Method:                 Least Squares   F-statistic:                 1.385e+04
Date:                Sun, 12 Jan 2020   Prob (F-statistic):               0.00
Time:                        20:12:10   Log-Likelihood:            -3.2263e+06
No. Observations:             2297973   AIC:                         6.453e+06
Df Residuals:                 2297967   BIC:                         6.453e+06
Df Model:                           5
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.0271      0.002     15.648      0.000       0.024       0.031
Channel[T.C]  -0.0308      0.002    -13.961      0.000      -0.035      -0.026
Channel[T.R]  -0.0483      0.002    -24.843      0.000      -0.052      -0.045
Channel[T.T]   0.0512      0.003     17.585      0.000       0.046       0.057
Time          -0.0648      0.001    -87.839      0.000      -0.066      -0.063
CS            -0.1353      0.001   -202.995      0.000      -0.137      -0.134
==============================================================================
Omnibus:                   184004.100   Durbin-Watson:                   1.752
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           232344.291
Skew:                          -0.778   Prob(JB):                         0.00
Kurtosis:                       3.090   Cond. No.                         6.93
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

从上表线性回归测试我们可以看出原始组合贷款价值（CLTV）与渠道的变化关系较为明显，其中通过零售商(T.R)和代理人（T.C）渠道进行借贷与 CLTV 数值成反向关系。反之，通过通信人（T.T)或未表明具体渠道了解借贷渠道则会使 CLTV 数值升高。同理，CLTV 与时间及信用分数成反比。

（二）关于 DTI 的线性回归

(2) 如果假设因变量(y) 为DTI，自变量则为时间(x_1)，信用分数(x_2)及渠道(x_3)。

则有:

$$DTI = \beta_0 + \beta_1 Time + \beta_2 CreditScore + \beta_3 Channel + \epsilon.$$

```
y, X = patsy.dmatrices('DTI~ Time + CS + Channel', data)#传递模型公式和相关数据以创建设计矩阵
DTI = sm.OLS(y, X)#将设计矩阵传递给OLS以指定普通的最小二乘模型
res1 = DTI.fit()#估计模型并将结果存储在res1中
print(res1.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    DTI   R-squared:                       0.070
Model:                            OLS   Adj. R-squared:                  0.070
Method:                 Least Squares   F-statistic:                 3.455e+04
Date:                Sun, 12 Jan 2020   Prob (F-statistic):               0.00
Time:                        20:19:47   Log-Likelihood:            -3.1761e+06
No. Observations:             2297973   AIC:                         6.352e+06
Df Residuals:                 2297967   BIC:                         6.352e+06
Df Model:                           5
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.0168      0.002      9.903      0.000       0.013       0.020
Channel[T.C]  -0.0148      0.002     -6.870      0.000      -0.019      -0.011
Channel[T.R]  -0.0312      0.002    -16.396      0.000      -0.035      -0.027
Channel[T.T]   0.0284      0.003      9.947      0.000       0.023       0.034
Time          -0.1011      0.001   -140.053      0.000      -0.102      -0.100
CS            -0.2194      0.001   -336.473      0.000      -0.221      -0.218
==============================================================================
Omnibus:                    49035.514   Durbin-Watson:                   1.876
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            30458.430
Skew:                           0.140   Prob(JB):                         0.00
Kurtosis:                       2.510   Cond. No.                         6.93
==============================================================================
```

从上表线性回归测试我们可以看出原始债务收入（DTI）比率与渠道的变化关系较为明显，其中通过零售商(T.R)渠道及通过通信人（T.C）渠道进行借贷了解与 CLTV 数值成反比。反之，未表明具体渠道（T.T）了解借贷渠道则会使 CLTV 数值升高。同理，CLTV 与时间及信用分数成反比。

（三）关于 LTV 的线性回归

（3）如果假设因变量(y)为LTV，自变量则为时间(x_1)，信用分数(x_2)及渠道(x_3)。

则有:

$$LTV = \beta_0 + \beta_1 Time + \beta_2 CreditScore + \beta_3 Channel + \epsilon.$$

```
y, X = patsy.dmatrices('LTV~ Time + CS + Channel', data)#传递模型公式和相关数据以
ltv = sm.OLS(y, X)#将设计矩阵传递给OLS以指定普通的最小二乘模型
res2 = ltv.fit()#估计模型并将结果存储在res2中
print(res2.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    LTV   R-squared:                       0.031
Model:                            OLS   Adj. R-squared:                  0.031
Method:                 Least Squares   F-statistic:                 1.460e+04
Date:                Sun, 12 Jan 2020   Prob (F-statistic):               0.00
Time:                        20:20:00   Log-Likelihood:            -3.2245e+06
No. Observations:             2297973   AIC:                         6.449e+06
Df Residuals:                 2297967   BIC:                         6.449e+06
Df Model:                           5
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.0294      0.002     16.972      0.000       0.026       0.033
Channel[T.C]  -0.0292      0.002    -13.286      0.000      -0.034      -0.025
Channel[T.R]  -0.0507      0.002    -26.093      0.000      -0.055      -0.047
Channel[T.T]   0.0381      0.003     13.101      0.000       0.032       0.044
Time          -0.0670      0.001    -90.853      0.000      -0.068      -0.066
CS            -0.1405      0.001   -210.899      0.000      -0.142      -0.139
==============================================================================
Omnibus:                   170124.889   Durbin-Watson:                   1.745
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           211428.113
Skew:                          -0.743   Prob(JB):                         0.00
Kurtosis:                       2.997   Cond. No.                         6.93
==============================================================================
```

从上表线性回归测试我们可以看出原始贷款价值（LTV）与渠道的变化关系较为明显，其中通过零售商(T.R)渠道渠道进行借贷了解与 CLTV 数值成反比，及通过零售进行借贷了解会造成 CLTV 数值降低（-2.0986）。反之，及通过通信人（T.C)及未表明具体渠道了解借贷渠道则会使 CLTV 数值升高。同理，CLTV 与时间及信用分数成反比。

（四）关于 LTV 的线性回归

（4）如果假设因变量(y)为LTV，自变量则为CLTV(x_1)，DTI(x_2)。

则有：

$$LTV = \beta_0 + \beta_1 CLTV + \beta_2 DTI + \epsilon.$$

```
y, X = patsy.dmatrices('LTV~ CLTV + DTI', data)#传递模型公式和相关数据以创建设计矩阵
ae = sm.OLS(y, X)#将设计矩阵传递给OLS以指定普通的最小二乘模型
res3 = ae.fit()#估计模型并将结果存储在res3中
print(res3.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    LTV   R-squared:                       0.922
Model:                            OLS   Adj. R-squared:                  0.922
Method:                 Least Squares   F-statistic:                 1.372e+07
Date:                Sun, 12 Jan 2020   Prob (F-statistic):               0.00
Time:                        20:20:02   Log-Likelihood:            -3.3305e+05
No. Observations:             2321634   AIC:                         6.661e+05
Df Residuals:                 2321631   BIC:                         6.661e+05
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept  -6.516e-15      0.000  -3.55e-11      1.000      -0.000       0.000
CLTV          0.9593      0.000   5162.058      0.000       0.959       0.960
DTI           0.0057      0.000     30.485      0.000       0.005       0.006
==============================================================================
Omnibus:                  2202937.612   Durbin-Watson:                   1.908
Prob(Omnibus):                  0.000   Jarque-Bera (JB):         84572632.133
Skew:                          -4.750   Prob(JB):                         0.00
Kurtosis:                      31.000   Cond. No.                         1.18
==============================================================================
```

从上表线性回归测试我们可以看出原始贷款价值（LTV）与原始债务收入（DTI）比率及原始组合贷款价值（CLTV）的关系都成正比。LTV 与 CLTV 正相关很强，我们 从公式中可以直观得到这一结论。在这部分实验中，R 平方系数是最高的，达到了 0.92，说明变量之间的确有着比较强的映射关系。

（五）关于 SCORE 的线性回归

（5）如果假设因变量(y) 为SCORE，自变量则为：CLTV，DTI,Channel,Time, FLAG。

则有：

$$SCORE = \beta 0 + \beta 1 CLTV + \beta 2 DTI + \beta 3 \text{Channel} + \beta 4 Time + \beta 5 \text{FLAG} + \epsilon.$$

```
y, X = patsy.dmatrices('CS~ CLTV + DTI+Channel+Time+ FLAG ', data)#传递模型公式和相关数据
ae = sm.OLS(y, X)#将设计矩阵传递给OLS以指定普通的最小二乘模型
res4 = ae.fit()#估计模型并将结果存储在res3中
print(res4.summary())
```

OLS Regression Results

Dep. Variable:	CS	R-squared:	0.079
Model:	OLS	Adj. R-squared:	0.078
Method:	Least Squares	F-statistic:	2.797e+04
Date:	Sun, 12 Jan 2020	Prob (F-statistic):	0.00
Time:	17:38:50	Log-Likelihood:	-3.1651e+06
No. Observations:	2297971	AIC:	6.330e+06
Df Residuals:	2297963	BIC:	6.330e+06
Df Model:	7		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	0.0602	0.002	30.327	0.000	0.056	0.064
Channel[T.C]	0.0140	0.002	6.509	0.000	0.010	0.018
Channel[T.R]	-0.0406	0.002	-21.409	0.000	-0.044	-0.037
Channel[T.T]	-0.2000	0.003	-70.291	0.000	-0.206	-0.194
CLTV	-0.1126	0.001	-175.417	0.000	-0.114	-0.111
DTI	-0.2193	0.001	-341.262	0.000	-0.221	-0.218
Time	0.0364	0.001	51.250	0.000	0.035	0.038
FLAG	0.0245	0.001	19.735	0.000	0.022	0.027

Omnibus:	262858.438	Durbin-Watson:	1.827
Prob(Omnibus):	0.000	Jarque-Bera (JB):	363719.911
Skew:	-0.924	Prob(JB):	0.00
Kurtosis:	3.617	Cond. No.	8.79

从上表线性回归测试我们可以看出信用分数（CREDIT SCORE）与零售商(T.R)和未知渠道（T.T）成反比和代理人（T.C）借贷渠道成正比，信用分数与原始债务收入（DTI）比率及原始组合贷款价值（CLTV）的关系都成反比，与事件（Time）和所购房子是否是第一套（FLAG）成正比。

六、 分类预测分析

（一）Logistic 预测

这部分内容其实从第四大部分，相关性分析部分可知，其实除了 CLTV 和 LTV 本质上的相关性来说，变量之间并没有很强的相关属性。尽管可想而知，效果不会理想，也没有实际用处。因为数据内生的相关性并不出存在，但是为了尝试一些机器学习方法，本文依旧对一些离散型数据实用其他变量对其进行分类预测。因此将结果和代码放在下面。

```
dataq = dataq.loc[dataq['Channel'] != 'Not Valuable Data'] #去除无效数据


dataq['Channel'] .replace('T', '0',inplace = True) #将数据中的 N替换成Not First Time
dataq['Channel'] .replace('R', '1',inplace = True)
dataq['Channel'] .replace('B', '2',inplace = True)
dataq['Channel'] .replace('C', '3',inplace = True)
```

```
dataq.dropna(axis=0,how='any', inplace=True)
```

```
X = dataq[['LTV','CLTV','DTI','CS']]
y = dataq.FLAG
X_train, X_test, y_train, y_test = train_test_split(X, y)

# 导入LogisticRegression数据包。
from sklearn.linear_model import LogisticRegression
# 实例化模型（使用默认参数）
logreg = LogisticRegression()
# 用数据拟合模型
logreg.fit(X_train,y_train)
y_pred=logreg.predict(X_test)
# 导入矩阵class
from sklearn import metrics
cnf_matrix = metrics.confusion_matrix(y_test, y_pred)
cnf_matrix
```

```
from sklearn.metrics import classification_report,confusion_matrix
print(classification_report(y_test,y_pred))
```

```
C:\lklklklklk\sofeware\anacoda\lib\site-packages\sklearn\metrics\_cl
ill-defined and being set to 0.0 in labels with no predicted samples
  _warn_prf(average, modifier, msg_start, len(result))
```

	precision	recall	f1-score	support
-1	0.93	1.00	0.96	533841
1	0.00	0.00	0.00	40652
accuracy			0.93	574493
macro avg	0.46	0.50	0.48	574493
weighted avg	0.86	0.93	0.90	574493

以上是对借款人是否是第一次购买房屋的分类预测，可以

```
X = dataq[['LTV','CLTV','DTI','CS']]
y = dataq.Channel
X_train, X_test, y_train, y_test = train_test_split(X, y)

# 导入LogisticRegression数据包。
from sklearn.linear_model import LogisticRegression
# 实例化模型（使用默认参数）
logreg = LogisticRegression()
# 用数据拟合模型
logreg.fit(X_train,y_train)
y_pred=logreg.predict(X_test)
# 导入矩阵class
from sklearn import metrics
cnf_matrix = metrics.confusion_matrix(y_test, y_pred)
cnf_matrix
```

```
array([[     0,   57271,       0,       0],
       [     0,  306886,       0,       0],
       [     0,   81538,       0,       0],
       [     0,  128798,       0,       0]], dtype=int64)
```

```
from sklearn.metrics import classification_report,confusion_matrix
print(classification_report(y_test,y_pred))
```

```
C:\lklklklklk\sofeware\anacoda\lib\site-packages\sklearn\metrics\_cl
ill-defined and being set to 0.0 in labels with no predicted sample
  _warn_prf(average, modifier, msg_start, len(result))
```

	precision	recall	f1-score	support
0	0.00	0.00	0.00	57271
1	0.53	1.00	0.70	306886
2	0.00	0.00	0.00	81538
3	0.00	0.00	0.00	128798
accuracy			0.53	574493
macro avg	0.13	0.25	0.17	574493
weighted avg	0.29	0.53	0.37	574493

（二）神经网络测试

```
X = dataq[['LTV','CLTV','DTI','CS']]
y = dataq.FLAG
from sklearn.preprocessing import StandardScaler
#scaler = StandardScaler()
# 适配训练集。
#scaler.fit(X_train)
# 现在将转换应用于数据:
X_train, X_test, y_train, y_test = train_test_split(X, y)#X_train = scaler.transform(X_train)
#X_test = scaler.transform(X_test)

from sklearn.neural_network import MLPClassifier #使用导入神经网络包
mlp = MLPClassifier(hidden_layer_sizes=(30,30,30)) #简单起见,
                                                   #我们将选择3个具有相同神经元数的层，这与我们的数据集中的特征相同
mlp.fit(X_train,y_train)

predictions = mlp.predict(X_test)
```

```
print(classification_report(y_test,predictions))
```

```
C:\lklklklklk\sofeware\anacoda\lib\site-packages\sklearn\
ill-defined and being set to 0.0 in labels with no predic
  _warn_prf(average, modifier, msg_start, len(result))
```

	precision	recall	f1-score	support
-1	0.93	1.00	0.96	534083
1	0.00	0.00	0.00	40374
accuracy			0.93	574457
macro avg	0.46	0.50	0.48	574457
weighted avg	0.86	0.93	0.90	574457

神经网络的分类结果同逻辑回归的分类效果没有差异。

```
dataq[dataq['FLAG']=='-1']
```

```
C:\lklklklklk\sofeware\anacoda\lib\site-packages\panda
calar instead, but in the future will perform elementw
  result = method(y)
```

CLTV	DTI	LTV	CS	Time	Channel	FLAG

```
dataq[dataq['FLAG']==1].count()/dataq.count()
```

```
CLTV        0.070284
DTI         0.070284
LTV         0.070284
CS          0.070284
Time        0.070284
Channel     0.070284
FLAG        0.070284
dtype: float64
```

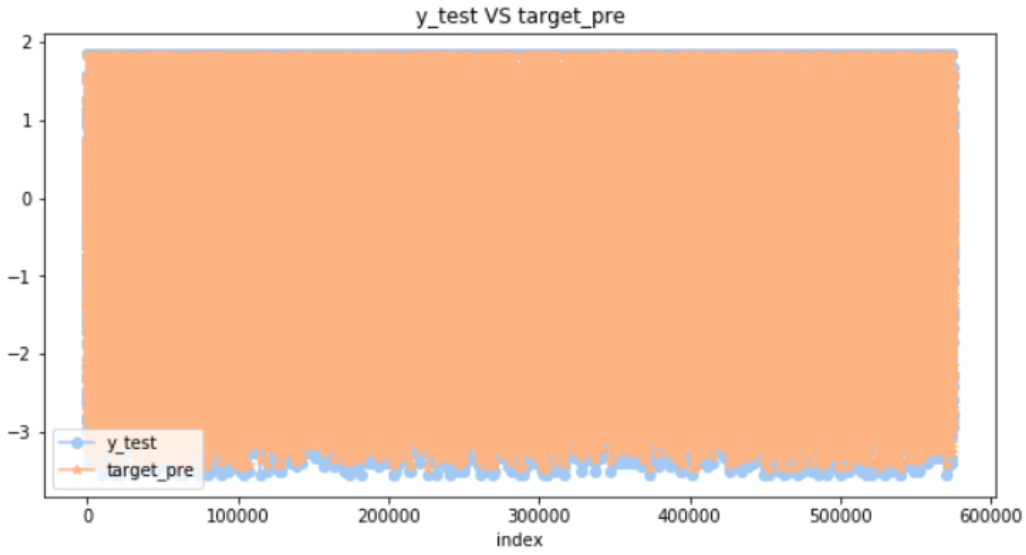
根据以上代码和结果可知，模型都将所有借款者是否购买第一套房子的情况预测成了-1，所以由于数据之间的非相关性，将会导致分类任务无法成果。

（三）测试梯度提升回归树模型 对 CLTV 的拟合效果

```
import pandas as pd
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.model_selection import train_test_split

X = dataq[['LTV','CS','DTI']]
y = dataq['CLTV']
X_train, X_test, y_train, y_test = train_test_split(X, y)


GBR_house=GradientBoostingRegressor().fit(X_train,y_train)
print(GBR_house)

#评价
target_pre=GBR_house.predict(X_test)
from sklearn.metrics import explained_variance_score,\
    mean_absolute_error,mean_squared_error,\
    median_absolute_error,r2_score
print('梯度提升回归树模型的平均绝对误差为：',mean_absolute_error(y_test,target_pre))
print('梯度提升回归树模型的均方误差为：',mean_squared_error(y_test,target_pre))
print('梯度提升回归树模型的中值绝对误差为：',median_absolute_error(y_test,target_pre))
print('梯度提升回归树模型的可解释方差值为：',explained_variance_score(y_test,target_pre))
print('梯度提升回归树模型的R^2值为：',r2_score(y_test,target_pre))
```

```
GradientBoostingRegressor(alpha=0.9, ccp_alpha=0.0, criterion='friedman_mse',
                          init=None, learning_rate=0.1, loss='ls', max_depth=3,
                          max_features=None, max_leaf_nodes=None,
                          min_impurity_decrease=0.0, min_impurity_split=None,
                          min_samples_leaf=1, min_samples_split=2,
                          min_weight_fraction_leaf=0.0, n_estimators=100,
                          n_iter_no_change=None, presort='deprecated',
                          random_state=None, subsample=1.0, tol=0.0001,
                          validation_fraction=0.1, verbose=0, warm_start=False)
梯度提升回归树模型的平均绝对误差为： 0.13361973612984251
梯度提升回归树模型的均方误差为： 0.07744759037913633
梯度提升回归树模型的中值绝对误差为： 0.06532583154450278
梯度提升回归树模型的可解释方差值为： 0.9226848799605224
梯度提升回归树模型的R^2值为： 0.9226847804108385
```

```
fig, ax = plt.subplots(figsize=(10,5))
plt.plot(range(len(y_test)),y_test, marker='o', label='y_test')
plt.plot(range(len(y_test)),target_pre,marker='*', label='target_pre')
plt.legend() # 让图例生效


plt.xlabel("index") #X轴标签
plt.ylabel("") #Y轴标签
plt.title("y_test VS target_pre") #标题

plt.show()
```



同样对应的，由于 CLTV 和 LTV 数据之间的强相关性，使用 LTV 等数据来预测 CLTV，达到的效果比较好，并且 R 方的系数同 OLS 回归几乎都在 0.922 的水平，因此我们得出一个重要的数据分析任务的结论就是，捕获数据之间的内在关系相对于模型的选择是更为重要的。