Problem Set 2 Solution Template

Xiaoli Sun (xs2338)

Sep.26 2020

Problem 1: Installment Loan Terms and Trade-offs

(a) See table below.
Python code:

```
#the value of m and r can change
m= 72
r = 2/100
P = 25000
M = ((1/m)+0.5*(r/12)+(1/8)*(m-2)*(r/12)*(r/12))*P
Interest_paid = M*m-P
DTI = M*12/50000
```

(i) Monthly payments $= M = P \times (\sum_{i=1}^{m} \frac{1}{(1+r_{loan}/12)^i})^{-1}$
$\approx\approx P \times [\frac{1}{m} + \frac{1}{2}(r_{loan}/12) + \frac{1}{8}(m-2)(r_{loan}/12)^2]$

(ii) Total interest paid over the life of the loan $= M \times m - P$

(iii) DTI (debt-to-income) change for your cousin assuming an income of \$50, 000 annually= $DTI = \frac{monthlydebt}{\frac{50,000}{12}}$

Loan Term	Lowest Rate	(i) Monthly Payment	(ii) Total Interest Paid	(iii) DTI change
36 months	3.00%	726.3585069444445	1148.90625	0.17432604166666668
60 months	4.50%	466.0904947916666	2965.4296874999927	0.11186171874999998
72 months	2.00%	368.6631944444444	1543.7499999999964	0.08847916666666666

(b) I recommend loan term 72 months with the lowest monthly payment and lowest DTI. It is more flexible. The total interest paid is acceptable. Only 400\$ higher than 36-month term, but has a much more low DTI and monthly payment.

(c) (i) ""

Loan Term	New/Lower Loan Rate	Monthly Payment Savings	Total Interest Savings
36 months	2.90%	1.0851996527778738	39.067187500004366
60 months	4.40%	1.153689236111063	69.22135416666379
72 months	1.90%	1.1009114583333144	79.26562499999864

(ii) Python code:

```
#M = Original Monthly Payment+25
m = 72
r = 2/100
M = 368.6631944444444+25
P = M*m/((1+r/12)**(m/2))
```

Loan Term	Loan Rate	Original Monthly Payment	New Monthly Payment	New Principal
36 months	3.00%	726.3585069444445	751.3585069444445	25860.138433500404
72 months	2.00%	368.6631944444444	393.6631944444444	26694.471707895496

(iii) Python code:

```
M=368.6631944444444
m=84
r=2/100
P = M*m/((1+r/12)**(m/2))
```

$P = 28875.78236902694$

Problem 2: Lending Club Loan Examples

(a) Python code:

```
import pandas as pd
#import data
df = pd.read_csv("/Users/liloli/Desktop/GR5293/hw2/LendingClub_2016Q1.csv")
# pick the first four loans with different grades
df_a = df[df['grade']=='A']
row_a = df_a.head(1)
df_b = df[df['grade']=='B']
row_b = df_b.head(1)
df_c = df[df['grade']=='C']
row_c = df_c.head(1)
df_d = df[df['grade']=='D']
row_d = df_d.head(1)
table = pd.concat([row_a, row_b, row_c, row_d])
#calculate total interest due
table['total_interest_due'] = table['installment']*36-table['loan_amnt']
#select columns needed
table = table[['loan_amnt', 'int_rate', 'installment','grade',  'sub_grade',
'total_interest_due']]
display(table.transpose())
```

(i) Table from python output.

	4	2	0	5
loan_amnt	6000	8000	12800	5625
int_rate	0.0739	0.1075	0.1199	0.1825
installment	186.34	260.97	425.09	204.07
grade	A	B	C	D
sub_grade	A4	B4	C1	D3
total_interest_due	708.24	1394.92	2503.24	1721.52

Figure 1: result

(b) Python code:

```
# Begin example
# You might be interested in loans with relatively low installments and low interest ra
# You could look at a distribution of interest rates and then look for loans with certa
# After filtering out loans outside of the desired range, you can examine

# Look at a distribution plot of 'int_rate'
```

```
import seaborn as sns
sns.distplot(df['int_rate']);
```

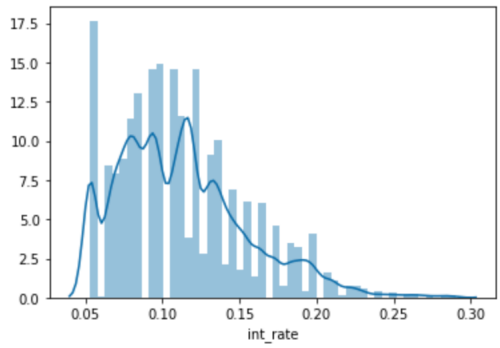

Figure 2: A distribution plot of interest rate

Select loans of grade A: Python code

```
mydata = df[df['grade']=='A']
mydata.head()
```

	loan_amnt	int_rate	installment	grade	sub_grade	emp_length	annual_inc	dti	delinq_2yrs	earliest_cr_line	revol_util	total_pymnt	total_rec_pr
4	6000.0	0.0739	186.34	A	A4	3.0	72000.0	10.68	0.0	11	0.236	6369.416519	600
9	4500.0	0.0697	138.89	A	A3	10.0	72000.0	12.57	1.0	21	0.351	4996.369224	450
13	22000.0	0.0649	674.18	A	A2	10.0	134000.0	26.33	0.0	47	0.670	24257.463888	2200
15	8000.0	0.0697	246.91	A	A3	2.0	75000.0	26.43	1.0	23	0.649	3696.320000	248
26	10000.0	0.0649	306.45	A	A2	1.0	40000.0	23.13	0.0	14	0.377	10810.322222	1000

Figure 3: grade A loans

Describe loans in grade A:
Python code:

```
mydata.describe()
```

Distribution of the installment:
Python code:

```
import seaborn as sns
sns.distplot(mydata['installment'])
```

	loan_amnt	int_rate	installment	emp_length	annual_inc	dti	delinq_2yrs	earliest_cr_line	revol_util	total_pymnt
count	24705.000000	24705.000000	24705.000000	23041.000000	2.470500e+04	24703.000000	24705.000000	24705.000000	24696.000000	24705.000000
mean	14880.756932	0.066266	456.982488	6.243739	9.349856e+04	16.086117	0.241044	22.292329	0.404177	15827.462835
std	8231.789034	0.010079	253.406535	3.713650	8.943261e+04	7.764258	0.722890	8.164563	0.225835	9047.096872
min	1000.000000	0.053200	30.120000	0.000000	0.000000e+00	0.000000	0.000000	6.000000	0.000000	0.000000
25%	8000.000000	0.053200	250.360000	3.000000	5.500000e+04	10.450000	0.000000	16.000000	0.233000	8722.485954
50%	13000.000000	0.069700	395.060000	7.000000	7.800000e+04	15.390000	0.000000	21.000000	0.381000	13480.812728
75%	20000.000000	0.074900	621.120000	10.000000	1.100000e+05	21.300000	0.000000	27.000000	0.557000	21768.340001
max	40000.000000	0.079100	1251.430000	10.000000	6.702150e+06	215.880000	20.000000	71.000000	1.720000	45165.280181

Figure 4: Get initial description of the grade A loans

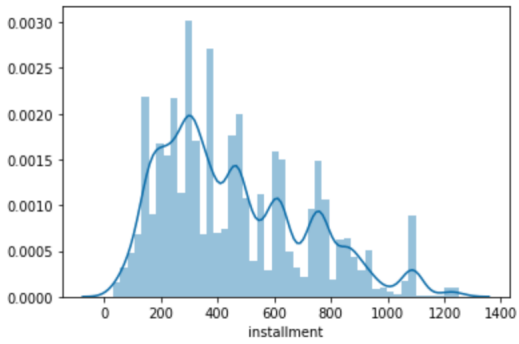

Figure 5: Get initial description of the installment

(c) **the result table is in the Figure 6**
total-count=96120
Python output table:

	pop%	dti	annual_inc	installment	loan_amnt	int_rate
A	25.7%	16.1	93,499	456.982	14,881	6.6%
B	34.3%	18.2	77,587	414.314	12,849	10.0%
C	25.3%	20.7	72,073	451.367	13,321	13.4%
D	9.9%	22.9	67,751	483.722	13,470	17.5%
E	3.5%	25.1	57,839	469.371	12,514	20.7%
F	1.0%	24.7	51,332	439.795	11,152	24.4%
G	0.3%	24.7	45,254	428.785	10,354	28.1%

Figure 6: Over view of Lending Club Grades

Problem 3: Probability of Default and Loss Given Default from LC Data

	avg int rate	default prev	avg return default	avg return no default	expected return	total return portfolio
A	6.627%	5.404%	-37.825%	8.712%	6.197%	6.362%
B	9.964%	12.536%	-37.617%	12.992%	6.648%	6.708%
C	13.381%	20.625%	-38.827%	17.160%	5.613%	5.545%
D	17.523%	28.428%	-40.171%	22.096%	4.394%	3.599%
E	20.667%	36.223%	-41.390%	26.375%	1.828%	0.082%
F	24.402%	39.610%	-42.540%	31.406%	2.115%	-1.406%
G	28.087%	45.455%	-46.250%	33.892%	-2.536%	-9.013%

Figure 7: summarize the results in a table

(a) most favorable portfolio return overall: **grade B 6.708%**
the least favorable portfolio return overall: **grade G -9.013%**
Which of the two quantities presents a higher variability as the loan grade changes?
Calculate the standard deviation of these two quantities:
Python code:

```
summary_stats[['default prev', 'avg return default']].std()
```

std(default prev) = 0.147358
std(avg return default) = 0.030608
default prev presents a higher variability as the loan grade changes.

(b) (i) If all loans had the same original loan balance/"loan-amnt", there will be **NO** difference.

$$ExpectedReturn = \frac{2n_d - n}{n} + \frac{1}{n} \times [\sum_{i \in NoDefault} \frac{TotalLoanPayment_i}{LoanAmount_i} - \sum_{j \in Default} \frac{TotalLoanPayment_j}{LoanAmount_j}]$$

Expected Return take "Prob of Default" into consideration.

$$PortfolioReturn = \frac{\sum_{i=1}^{n} TotalPayment_i}{\sum_{i=1}^{n} TotalLoanAmount_i} - 1$$

Portfolio Return does not take "Prob of Default" into consideration.

(ii) A-grade and B-grade: larger loans earn more, smaller loans earn less.
D-grade and E-grade: larger loans loss mode, smaller loans loss less.

(c) See table below:

	ret v ir ratio
A	1.314730
B	1.303946
C	1.282472
D	1.260957
E	1.276224
F	1.287011
G	1.206674

Figure 8: Ratio