Lending Club Case Study:

Introduction:

This case study is all about application of Exploratory Data Analysis (EDA) techniques to solve real-life business problems using data driven insights. The project will help in the development of fundamental understanding of the "risk analytics" in the banking and financial services domain ensuring data driven approach to minimize the risk of credit loss

Objective:

The objective of this case study is to identify factors/variables/patterns that would indicate if a customer would likely to "default" or not if a customer applies for a loan basis which business decisions to be taken such as loan denial, reduction in loan amount, lending loan at a higher interest rate etc. It is expected to study how each variable influence the tendency of "default". Identification of risky loan applicants will eventually help in cutting down the credit loss

Risk Analytics in banking/financial services sector:

Predicting whether a bank customer is likely to "default" is a complex yet important task that involves factoring in various parameters. Risk assessment models and algorithms are used to make these predictions. Few common factors considered in determining credit risk includes:

- Past credit history & credit score
- Debt-to-income ratio
- Income
- Stability of employment
- Loan amount
- Term of repayments
- Age
- Marital status
- Credit utilization ratio
- Payment history
- Economic outlook in the market

Data Analytics:

1. **Data Understanding from the Input dataset provided (for this case study)**:

- Relevant variables assumed initially:

variable	Comments
loan_amnt	This is the amount of loan that the customer wants to seek from the bank
Term	This represents the term duration for repayment of the loan with interest
emp_title	Organization where the customer is associated with
emp_length	Total years of experience of the customer
home_ownership	Whether RENT or MORTGAGE etc…
annual_inc	Total annual income of the customer
loan_status	**Whether customer had fully paid the loan or in the process of repayment or charged off by the bank**

- Data quality checks of the relevant variables:

variable	Comments	Data quality checks
loan_amnt	This is the amount of loan that the customer wants to seek from the bank	Looks good & no quality issues
Term	This represents the term duration for repayment of the loan with interest	"months" is separated out during analysis keeping only the numeric value but otherwise data looks good
emp_title	Organization where the customer is associated with	Too many missing values
emp_length	Total years of experience of the customer	Looks good
home_ownership	Whether RENT or MORTGAGE etc…	No missing values & all relevant values
annual_inc	Total annual income of the customer	Outliers present in the data
loan_status	**Whether customer had fully paid the loan or in the process of repayment or charged off by the bank**	**Looks good**

2. **Data cleansing & manipulation:**

- ***Missing value treatment*** – Done using Python

  So far from the set of relevant columns figured out for this analysis, only "emp_title" column has missing values and it makes sense to create a sub-dataset from the original input dataset where we are removing the rows in which "emp_title" is missing because "emp_title" represents the employee's organization and it's not fair to assume organization name for any employee unless we really get the details from some source. Moreover, in this Input dataset we would still end up with enough sample size to do analysis even if we discard those rows where "emp_title" is missing

```
#Reading the Input dataset using pandas as a dataframe
import pandas as pd;
file_path = '/content/loan.csv';
data = pd.read_csv(file_path);

#Capturing only relevant columns for analysis and creating a new dataframe
data_relevant_columns = data[
['loan_amnt','term','emp_title','emp_length','home_ownership','annual_inc'
] ];

#Discarding those rows in which "emp_title" is missing
data_relevant_columns1 =
data_relevant_columns.dropna(subset=['emp_title']);
```

- ***Outlier detection*** – When we did look at the "annual_inc" column in the Input dataset we had figured out values that are too high and hence analysis will produce better outcomes if we remove "outlier" values from the column as "annual_inc" is a driver of deciding whether a customer would "default". However, in this case study it made more sense to actually keep outliers and then use "Median" instead of "Mean" while judging the outcome of the "loan_status" basis the "annual_inc" values. All these analysis to be covered in the next section where we present the Bivariate analysis

3. **Data Analysis:**

- Univariate & Segmented Univariate analysis on the relevant columns:

relevant variables	**missing values**	**Outliers**

loan_amnt	No	No
Term	No	No
emp_title	Yes	NA
emp_length	No	No
home_ownership	No	No
annual_inc	No	Yes
loan_status	No	No

loan_amnt:

SUM	445602650
MAX	35000
MIN	500
AVG	11219.44381
# of blanks:	0

emp_length: Bucketed into < 1, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10+ & n/a

home_ownership: 3 distinct values: **MORTGAGE , OWN and RENT**

annual_inc:

SUM	2739238849
MAX	6000000
MIN	4000
AVERAGE	68968.92638
# of blanks	0

- Bivariate analysis on the relevant columns:

CASE-1: “annual_inc” on “loan_status”

```
#Reading the Input dataset using pandas as a dataframe
import pandas as pd;
file_path = '/content/loan.csv';
data = pd.read_csv(file_path);

#Capturing only relevant columns for analysis and creating a new dataframe
data_relevant_columns = data[
['loan_amnt','term','emp_title','emp_length','home_ownership','annual_inc'
,'loan_status'] ];
```

```
#Discarding those rows in which "emp_title" is missing
data_relevant_columns1 =
data_relevant_columns.dropna(subset=['emp_title']);

#Bivariate analysis between 'annual_inc' and 'loan_status':
data_relevant_columns2 = data_relevant_columns1[
['annual_inc','loan_status'] ];
#print(data_relevant_columns2);

#Grouping based on 'loan_status'
grouped_data_relevant_columns2 =
data_relevant_columns2.groupby('loan_status').agg({'annual_inc':'median'})
;
print(grouped_data_relevant_columns2);
```

loan_status	annual_inc
Charged Off	54000.0
Current	65000.0
Fully Paid	60000.0

Though ‘annual_inc’ creates some influence on the ‘loan_status’ however based on the data shared the same is not significant enough unless we add more variables. Let’s look at the impact of other variables on ‘loan_status’

CASE-2: “loan_amnt” on “loan_status”

```
#Grouping based on 'loan_status' for 'loan_amnt'
data_relevant_columns3 = data_relevant_columns1[
['loan_amnt','loan_status'] ];
grouped_data_relevant_columns3 =
data_relevant_columns3.groupby('loan_status').agg({'loan_amnt':'median'});
print(grouped_data_relevant_columns3);
```

loan_status	loan_amnt
Charged Off	10000.0
Current	16000.0
Fully Paid	9600.0

Though ‘loan_amnt’ creates some influence on the ‘loan_status’ however based on the data shared the same is not significant enough unless we add more variables. Let’s look at the impact of other variables on ‘loan_status’ or combination of variables

CASE-3: “home_ownership” on “loan_status”

```
#Take counts of each combinations of 'loan_status' and 'home_ownership'
data_relevant_columns4 = data_relevant_columns1[
['loan_status','home_ownership'] ];
#print(data_relevant_columns4);
counts_data_relevant_columns4 =
data_relevant_columns4.groupby(['loan_status','home_ownership']).size().re
set_index(name='Count');
print(counts_data_relevant_columns4);
```

```
    loan_status home_ownership  Count
0   Charged Off       MORTGAGE   2121
1   Charged Off          OTHER     18
2   Charged Off            OWN    366
3   Charged Off           RENT   2638
4       Current       MORTGAGE    597
5       Current            OWN     71
6       Current           RENT    399
7    Fully Paid       MORTGAGE  13856
8    Fully Paid           NONE      1
9    Fully Paid          OTHER     78
10   Fully Paid            OWN   2217
11   Fully Paid           RENT  14896
```

loan_status	**home_ownership**	**Count**
Charged Off	MORTGAGE	2121
Charged Off	OTHER	18
Charged Off	OWN	366
Charged Off	RENT	2638
Current	MORTGAGE	597
Current	OWN	71
Current	RENT	399
Fully Paid	MORTGAGE	13856
Fully Paid	NONE	1
Fully Paid	OTHER	78
Fully Paid	OWN	2217
Fully Paid	RENT	14896

So, from the above analysis it initially looked like that ‘home_ownership’ plays a big role in determining if the customer should be ‘default’ or not. From the highlighted data, it appeared to be that if the ‘home_ownership’ is ‘MORTGAGE’ or ‘RENT’ then maximum chances that the customer will be ‘default’ however upon closer analysis of data-driven metric we did figure out that percentage (% of home_ownership on ‘Charged Off’ w.r.t total home_ownership) will be better judge in this case from the data

loan_status	home_ownership	Count	Ratio
Charged Off	MORTGAGE	2121	13%
Charged Off	OTHER	18	19%
Charged Off	OWN	366	14%
Charged Off	RENT	2638	15%
Current	MORTGAGE	597	
Current	OWN	71	
Current	RENT	399	
Fully Paid	MORTGAGE	13856	
Fully Paid	NONE	1	
Fully Paid	OTHER	78	
Fully Paid	OWN	2217	
Fully Paid	RENT	14896	

Hence from the above percentages, it is evident that if the ‘home_ownership’ = ‘OTHER’ we have the highest possible chances of, ‘Charged Off’ compared to any other value in ‘home_ownership’

CASE-4: “emp_length” on “loan_status”

```
#Take counts of each combinations of 'loan_status' and 'emp_length'
data_relevant_columns5 = data_relevant_columns1[
['loan_status','emp_length'] ];
counts_data_relevant_columns5 =
data_relevant_columns5.groupby(['loan_status','emp_length']).size().reset_
index(name='Count');
print(counts_data_relevant_columns5);
```

```
   loan_status emp_length  Count
0   Charged Off     1 year    435
1   Charged Off  10+ years   1270
2   Charged Off    2 years    547
3   Charged Off    3 years    534
4   Charged Off    4 years    447
5   Charged Off    5 years    436
6   Charged Off    6 years    296
7   Charged Off    7 years    254
```

```
8   Charged Off     8 years    198
9   Charged Off     9 years    152
10  Charged Off   < 1 year     562
11      Current     1 year      67
12      Current   10+ years    379
13      Current     2 years     95
14      Current     3 years     81
15      Current     4 years     91
16      Current     5 years     86
17      Current     6 years     59
18      Current     7 years     59
19      Current     8 years     44
20      Current     9 years     31
21      Current   < 1 year      74
22   Fully Paid     1 year    2632
23   Fully Paid   10+ years   6886
24   Fully Paid     2 years   3631
25   Fully Paid     3 years   3367
26   Fully Paid     4 years   2819
27   Fully Paid     5 years   2655
28   Fully Paid     6 years   1803
29   Fully Paid     7 years   1409
30   Fully Paid     8 years   1193
31   Fully Paid     9 years   1045
32   Fully Paid   < 1 year    3565
```

loan_status	**emp_length**	**Count**	**Total Count(emp_length)**	**Percentage**
Charged Off	1	435	3134	13.88002553
Charged Off	10+	1270	8535	14.87990627
Charged Off	2	547	4273	12.80131055
Charged Off	3	534	3982	13.41034656
Charged Off	4	447	3357	13.31546023
Charged Off	5	436	3177	13.72363865
Charged Off	6	296	2158	13.71640408
Charged Off	7	254	1722	14.75029036
Charged Off	8	198	1435	13.79790941
Charged Off	9	152	1228	12.37785016
Charged Off	< 1	562	4201	13.3777672

From the above percentages, it is evident that if the ‘emp_length’ is not a strong indicator of ‘loan_status’

CASE-5: Multiplication of ‘loan_amnt’ & ‘term’ to derive a new column called ‘flag’ is used to check if ‘flag’ value influences the ‘loan_status’

```
#Reading the Input dataset using pandas as a dataframe
import pandas as pd;
```

```
file_path = '/content/loan.csv';
data = pd.read_csv(file_path);
data_req_columns = data[ ['loan_amnt','term','loan_status'] ];
data_req_columns['flag'] =
data_req_columns['loan_amnt'].mul(data_req_columns['term']);
#print(data_req_columns);

#Group By 'loan_status' & aggregation of 'flag'
grouped_df = data_req_columns.groupby(['loan_status']).mean('flag');
print(grouped_df);
```

```
loan_status  flag
Charged Off  5.970401e+05
Current      1.023239e+06
Fully Paid   4.707537e+05
```

From the above analysis it is evident that 'loan_amnt' & 'term' together influences the 'loan_status' and based on that we can judge if a customer is 'default' or not

CASE-6: "grade" on "loan_status"

```
#Checking 'grade' on 'loan_status':
grouped_df =
data_req_columns.groupby(['loan_status','grade']).size().reset_index(name=
'Count');
print(grouped_df);
```

```
    loan_status grade  Count
0   Charged Off     A    602
1   Charged Off     B   1425
2   Charged Off     C   1347
3   Charged Off     D   1118
4   Charged Off     E    715
5   Charged Off     F    319
6   Charged Off     G    101
7       Current     A     40
8       Current     B    345
9       Current     C    264
10      Current     D    222
11      Current     E    179
12      Current     F     73
13      Current     G     17
14   Fully Paid     A   9443
15   Fully Paid     B  10250
16   Fully Paid     C   6487
17   Fully Paid     D   3967
18   Fully Paid     E   1948
```

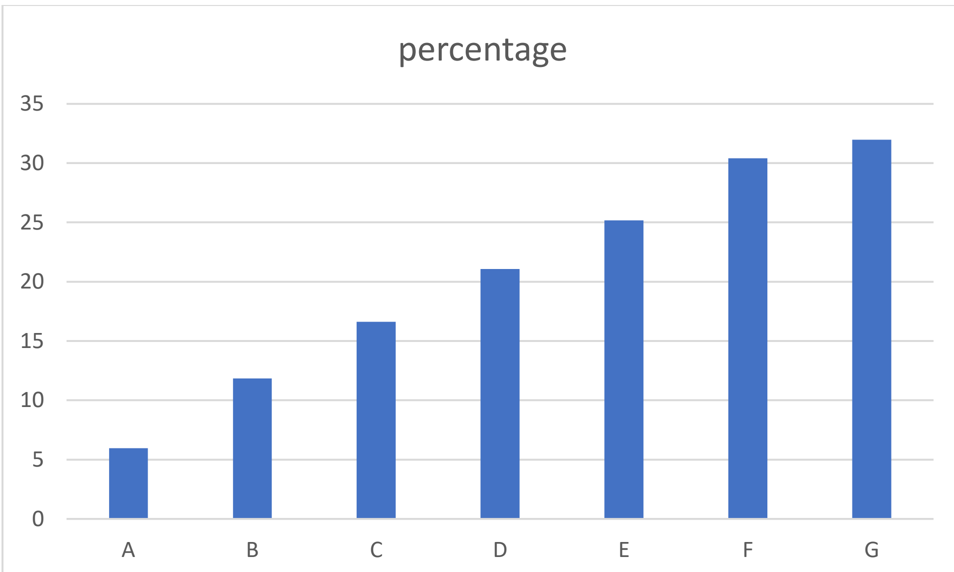
```
19    Fully Paid      F     657
20    Fully Paid      G     198
```

loan_status	Grade	Count	Total Count(grades)	Percentages
Charged Off	A	602	10085	5.969261279
Charged Off	B	1425	12020	11.85524126
Charged Off	C	1347	8098	16.63373673
Charged Off	D	1118	5307	21.06651592
Charged Off	E	715	2842	25.1583392
Charged Off	F	319	1049	30.4099142
Charged Off	G	101	316	31.96202532
Current	A	40		
Current	B	345		
Current	C	264		
Current	D	222		
Current	E	179		
Current	F	73		
Current	G	17		
Fully Paid	A	9443		
Fully Paid	B	10250		
Fully Paid	C	6487		
Fully Paid	D	3967		
Fully Paid	E	1948		
Fully Paid	F	657		
Fully Paid	G	198		

INFERENCE drawn from the data:

Looking at the percentages above, it is clearly evident that ‘grade’ plays a big factor in determining if a customer is likely to default or not. Higher grades have less chances of a ‘default’ & vice-versa.

Visualization Plot:



So far, we have figured out that 'home_ownership' and 'grade' are strong indicator variables to determine if a customer is likely to 'default' or not. We will study the pattern of more variables in the Input dataset to figure out variables that are likely to influence 'loan_status'

CASE-7: "purpose" on "loan_status"

```
#Reading the Input dataset using pandas as a dataframe
import pandas as pd;
file_path = '/content/loan.csv';
data = pd.read_csv(file_path);

#Checking 'grade' on 'loan_status':
grouped_df =
data.groupby(['loan_status','purpose']).size().reset_index(name='Count');
print(grouped_df);
```

loan_status	purpose	Count

```
0   Charged Off                 car    160
1   Charged Off         credit_card    542
2   Charged Off  debt_consolidation   2767
3   Charged Off         educational     56
4   Charged Off    home_improvement    347
5   Charged Off               house     59
6   Charged Off      major_purchase    222
7   Charged Off             medical    106
8   Charged Off              moving     92
9   Charged Off               other    633
10  Charged Off    renewable_energy     19
11  Charged Off      small_business    475
12  Charged Off            vacation     53
13  Charged Off             wedding     96
14      Current                 car     50
15      Current         credit_card    103
16      Current  debt_consolidation    586
17      Current    home_improvement    101
18      Current               house     14
19      Current      major_purchase     37
20      Current             medical     12
21      Current              moving      7
22      Current               other    128
23      Current    renewable_energy      1
24      Current      small_business     74
25      Current            vacation      6
26      Current             wedding     21
27   Fully Paid                 car   1339
28   Fully Paid         credit_card   4485
29   Fully Paid  debt_consolidation  15288
30   Fully Paid         educational    269
31   Fully Paid    home_improvement   2528
32   Fully Paid               house    308
33   Fully Paid      major_purchase   1928
34   Fully Paid             medical    575
35   Fully Paid              moving    484
36   Fully Paid               other   3232
37   Fully Paid    renewable_energy     83
38   Fully Paid      small_business   1279
39   Fully Paid            vacation    322
40   Fully Paid             wedding    830
```

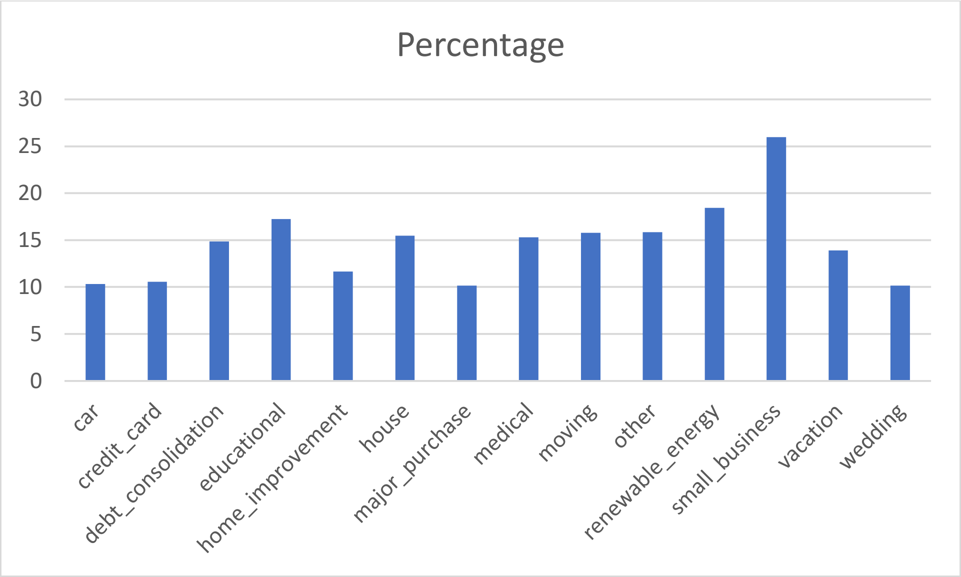
loan_status	purpose	Count	Total Count(purpose)	Percentage
Charged Off	Car	160	1549	10.32924467
Charged Off	credit_card	542	5130	10.56530214
Charged Off	debt_consolidation	2767	18641	14.84362427
Charged Off	Educational	56	325	17.23076923
Charged Off	home_improvement	347	2976	11.65994624
Charged Off	House	59	381	15.4855643

Charged Off	major_purchase	222	2187	10.15089163
Charged Off	Medical	106	693	15.2958153
Charged Off	Moving	92	583	15.78044597
Charged Off	Other	633	3993	15.8527423
Charged Off	renewable_energy	19	103	18.44660194
Charged Off	small_business	475	1828	25.98468271
Charged Off	Vacation	53	381	13.91076115
Charged Off	Wedding	96	947	10.13727561
Current	Car	50		
Current	credit_card	103		
Current	debt_consolidation	586		
Current	home_improvement	101		
Current	House	14		
Current	major_purchase	37		
Current	Medical	12		
Current	Moving	7		
Current	Other	128		
Current	renewable_energy	1		
Current	small_business	74		
Current	Vacation	6		
Current	Wedding	21		
Fully Paid	Car	1339		
Fully Paid	credit_card	4485		
Fully Paid	debt_consolidation	15288		
Fully Paid	Educational	269		
Fully Paid	home_improvement	2528		
Fully Paid	House	308		
Fully Paid	major_purchase	1928		
Fully Paid	Medical	575		
Fully Paid	Moving	484		
Fully Paid	Other	3232		
Fully Paid	renewable_energy	83		
Fully Paid	small_business	1279		
Fully Paid	Vacation	322		
Fully Paid	Wedding	830		

INFERENCE drawn from the data:

'purpose' has some significance when it comes to influencing 'loan_status' as we can see that percentage is skewed towards 'default' when purpose is 'small_business' hence 'purpose' is a fair indicator of the 'default' customer probability

Visualization Plot:



CASE-8: “addr_state” on “loan_status”:

```
#Reading the Input dataset using pandas as a dataframe
import pandas as pd;
file_path = '/content/loan.csv';
data = pd.read_csv(file_path);

#Impact of 'addr_state' on 'loan_status':
grouped_df =
data.groupby(['loan_status','addr_state']).size().reset_index(name='Count'
);
print(grouped_df);

# Export the DataFrame to an Excel file with a specific sheet name
output_file = '/content/output.xlsx'
sheet_name  = 'MySheet'
grouped_df.to_excel(output_file, index=False, sheet_name=sheet_name);
```

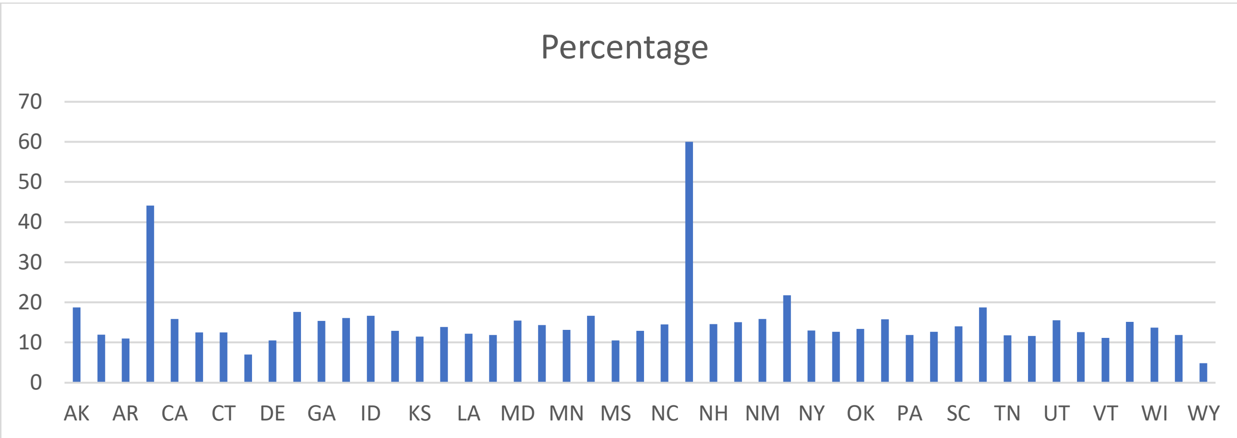
loan_status	addr_state	Count	Total Count(addr_state)	Percentage	Comments
Charged Off	AK	15	80	18.75	
Charged Off	AL	54	452	11.94690265	
Charged Off	AR	27	245	11.02040816	
Charged Off	AZ	123	279	44.08602151	
Charged Off	CA	1125	7099	15.84730244	

Charged Off	CO	98	782	12.53196931	
Charged Off	CT	94	751	12.51664447	
Charged Off	DC	15	214	7.009345794	
Charged Off	DE	12	114	10.52631579	
Charged Off	FL	504	2866	17.585485	
Charged Off	GA	215	1398	15.37911302	
Charged Off	HI	28	174	16.09195402	
Charged Off	ID	1	6	16.66666667	
Charged Off	IL	197	1525	12.91803279	
Charged Off	KS	31	271	11.43911439	
Charged Off	KY	45	325	13.84615385	
Charged Off	LA	53	436	12.1559633	
Charged Off	MA	159	1340	11.86567164	
Charged Off	MD	162	1049	15.44327931	
Charged Off	MI	103	720	14.30555556	
Charged Off	MN	81	615	13.17073171	
Charged Off	MO	114	686	16.6180758	
Charged Off	MS	2	19	10.52631579	
Charged Off	MT	11	85	12.94117647	
Charged Off	NC	114	788	14.46700508	
Charged Off	NE	3	5	60	low sample size
Charged Off	NH	25	171	14.61988304	
Charged Off	NJ	278	1850	15.02702703	
Charged Off	NM	30	189	15.87301587	
Charged Off	NV	108	497	21.73038229	
Charged Off	NY	495	3812	12.98530955	
Charged Off	OH	155	1223	12.67375307	
Charged Off	OK	40	299	13.37792642	
Charged Off	OR	71	451	15.74279379	
Charged Off	PA	180	1517	11.86552406	
Charged Off	RI	25	198	12.62626263	
Charged Off	SC	66	472	13.98305085	
Charged Off	SD	12	64	18.75	
Charged Off	TN	2	17	11.76470588	
Charged Off	TX	316	2727	11.58782545	
Charged Off	UT	40	258	15.50387597	
Charged Off	VA	177	1407	12.57995736	
Charged Off	VT	6	54	11.11111111	
Charged Off	WA	127	840	15.11904762	
Charged Off	WI	63	460	13.69565217	
Charged Off	WV	21	177	11.86440678	
Charged Off	WY	4	83	4.819277108	

<u>INFERENCE drawn from the data</u>:

"addr_state" plays a role in determining if a customer can be 'default' or not as we can see here, percentages are skewed towards 'AZ' followed by 'NV' & 'SD'. The below visualization plot shows us the regions where have the spikes

<u>Visualization Plot</u>:



Conclusion:

From the above data exploration/analysis using multiple data analysis techniques it is safe to assume that the below variables are the primary drivers in determining if a customer is likely to 'default' or not

home_ownership
Combination of 'loan_amnt' & 'term'
grade
purpose
addr_state

****