+ Code + Text

```
import pandas as pd
```

```
data=pd.read_csv(r'/content/logistic_regression.csv')
```

```
---------------------------------------------------------------------------
ParserError                               Traceback (most recent call last)
<ipython-input-122-4356c5b689ec> in <cell line: 1>()
----> 1 data=pd.read_csv(r'/content/logistic_regression.csv')

9 frames

/usr/local/lib/python3.10/dist-packages/pandas/_libs/parsers.pyx in pandas._libs.parsers.raise_parser_error()

ParserError: Error tokenizing data. C error: EOF inside string starting at row 128273
```

```
data.columns
data['loan_status']

```

```
0        Fully Paid
1        Fully Paid
2        Fully Paid
3        Fully Paid
4       Charged Off
           ...
2017     Fully Paid
2018    Charged Off
2019     Fully Paid
2020     Fully Paid
2021     Fully Paid
Name: loan_status, Length: 2000, dtype: object
```

```
data
```

	loan_amnt	term	int_rate	installment	grade	sub_grade	emp_title	emp_length	home_ownership	annual_inc	...	open_acc
0	10000.0	36 months	11.44	329.48	B	B4	Marketing	10+ years	RENT	117000.0	...	16.0
1	8000.0	36 months	11.99	265.68	B	B5	Credit analyst	4 years	MORTGAGE	65000.0	...	17.0
2	15600.0	36 months	10.49	506.97	B	B3	Statistician	< 1 year	RENT	43057.0	...	13.0
3	7200.0	36 months	6.49	220.65	A	A2	Client Advocate	6 years	RENT	54000.0	...	6.0
4	24375.0	60 months	17.27	609.33	C	C5	Destiny Management Inc.	9 years	MORTGAGE	55000.0	...	13.0
...	...	...	...	...	...	...	...	...	...	...	...	..
2017	10625.0	36 months	14.33	364.85	C	C1	white wave food	3 years	RENT	31000.0	...	14.0
2018	6600.0	36 months	14.46	227.05	C	C4	SUPERVISOR	10+ years	OWN	58240.0	...	6.0
2019	16800.0	60 months	14.31	393.62	C	C4	Sr. Budget Manager	10+ years	RENT	145000.0	...	21.0
2020	17050.0	36 months	15.31	593.64	C	C2	Safeway	10+ years	MORTGAGE	35000.0	...	13.0
2021	35000.0	60 months	17.57	880.61	D	D4	Financial Advisor	7 years	RENT	200000.0	...	6.0

2000 rows × 27 columns

```
# data preprocessing
data.isnull().sum()
```

```
loan_amnt               0
term                    0
int_rate                0
```

```
installment             0
grade                   0
sub_grade               0
emp_title               0
emp_length              0
home_ownership          0
annual_inc              0
verification_status     0
issue_d                 0
loan_status             0
purpose                 0
title                   0
dti                     0
earliest_cr_line        0
open_acc                0
pub_rec                 0
revol_bal               0
revol_util              0
total_acc               0
initial_list_status     0
application_type        0
mort_acc                0
pub_rec_bankruptcies    0
address                 0
dtype: int64
```

```
Start coding or generate with AI.
```

```

# columns with   null values will be replaced with 0
data


```

	loan_amnt	term	int_rate	installment	grade	sub_grade	emp_title	emp_l
0	10000.0	36 months	11.44	329.48	B	B4	Marketing	10+
1	8000.0	36 months	11.99	265.68	B	B5	Credit analyst	4
2	15600.0	36 months	10.49	506.97	B	B3	Statistician	<
3	7200.0	36 months	6.49	220.65	A	A2	Client Advocate	6
4	24375.0	60 months	17.27	609.33	C	C5	Destiny Management Inc.	9
...	...	...	...	...	...	...	...	
2017	10625.0	36 months	14.33	364.85	C	C1	white wave food	3
2018	6600.0	36 months	14.46	227.05	C	C4	SUPERVISOR	10+
2019	16800.0	60 months	14.31	393.62	C	C4	Sr. Budget Manager	10+
2020	17050.0	36 months	15.31	593.64	C	C2	Safeway	10+
2021	35000.0	60 months	17.57	880.61	D	D4	Financial Advisor	7

2000 rows × 27 columns

```
data=data.fillna(0)
```

```

data.isnull().sum()
```

```
loan_amnt               0
term                    0
int_rate                0
```

```
installment             0
grade                   0
sub_grade               0
emp_title               0
emp_length              0
home_ownership          0
annual_inc              0
verification_status     0
issue_d                 0
loan_status             0
purpose                 0
title                   0
dti                     0
earliest_cr_line        0
open_acc                0
pub_rec                 0
revol_bal               0
revol_util              0
total_acc               0
initial_list_status     0
application_type        0
mort_acc                0
pub_rec_bankruptcies    0
address                 0
dtype: int64
```

```
1 data.duplicated().any()
```

```
False
```

```
1 data
```

	loan_amnt	term	int_rate	installment	grade	sub_grade	emp_title	emp_l
0	10000.0	36 months	11.44	329.48	B	B4	Marketing	10+
1	8000.0	36 months	11.99	265.68	B	B5	Credit analyst	4
2	15600.0	36 months	10.49	506.97	B	B3	Statistician	<
3	7200.0	36 months	6.49	220.65	A	A2	Client Advocate	6
4	24375.0	60 months	17.27	609.33	C	C5	Destiny Management Inc.	9
...	...	...	...	...	...	...	...	
2017	10625.0	36 months	14.33	364.85	C	C1	white wave food	3
2018	6600.0	36 months	14.46	227.05	C	C4	SUPERVISOR	10+
2019	16800.0	60 months	14.31	393.62	C	C4	Sr. Budget Manager	10+
2020	17050.0	36 months	15.31	593.64	C	C2	Safeway	10+
2021	35000.0	60 months	17.57	880.61	D	D4	Financial Advisor	7

2000 rows × 27 columns

```
1 data
```

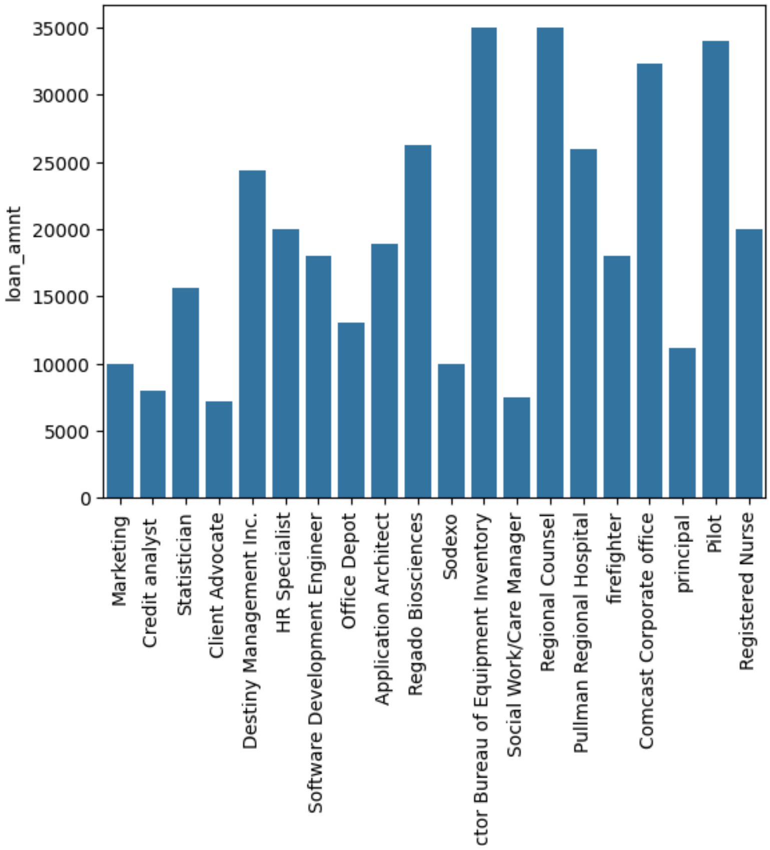
	loan_amnt	term	int_rate	installment	grade	sub_grade	emp_title	emp_l
0	10000.0	36 months	11.44	329.48	B	B4	Marketing	10+
1	8000.0	36 months	11.99	265.68	B	B5	Credit analyst	4
2	15600.0	36 months	10.49	506.97	B	B3	Statistician	<
3	7200.0	36 months	6.49	220.65	A	A2	Client Advocate	6
4	24375.0	60 months	17.27	609.33	C	C5	Destiny Management Inc.	9
...	...	...	...	...	...	...	...	
2017	10625.0	36 months	14.33	364.85	C	C1	white wave food	3
2018	6600.0	36 months	14.46	227.05	C	C4	SUPERVISOR	10+
2019	16800.0	60 months	14.31	393.62	C	C4	Sr. Budget Manager	10+
2020	17050.0	36 months	15.31	593.64	C	C2	Safeway	10+
2021	35000.0	60 months	17.57	880.61	D	D4	Financial Advisor	7

2000 rows × 27 columns

```
# data visualization
import seaborn as sns
p=data.head(20)

```

```
import matplotlib.pyplot as plt
sns.barplot(x='emp_title',y='loan_amnt',data=p)
plt.xticks(rotation=90)
```

```
([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19],
 [Text(0, 0, 'Marketing'),
  Text(1, 0, 'Credit analyst '),
  Text(2, 0, 'Statistician'),
  Text(3, 0, 'Client Advocate'),
  Text(4, 0, 'Destiny Management Inc.'),
  Text(5, 0, 'HR Specialist'),
  Text(6, 0, 'Software Development Engineer'),
  Text(7, 0, 'Office Depot'),
  Text(8, 0, 'Application Architect'),
  Text(9, 0, 'Regado Biosciences'),
  Text(10, 0, 'Sodexo'),
  Text(11, 0, 'Director Bureau of Equipment Inventory'),
  Text(12, 0, 'Social Work/Care Manager'),
  Text(13, 0, 'Regional Counsel'),
  Text(14, 0, 'Pullman Regional Hospital'),
  Text(15, 0, 'firefighter'),
  Text(16, 0, 'Comcast Corporate office'),
  Text(17, 0, 'principal'),
  Text(18, 0, 'Pilot'),
  Text(19, 0, 'Registered Nurse')])
```



New section

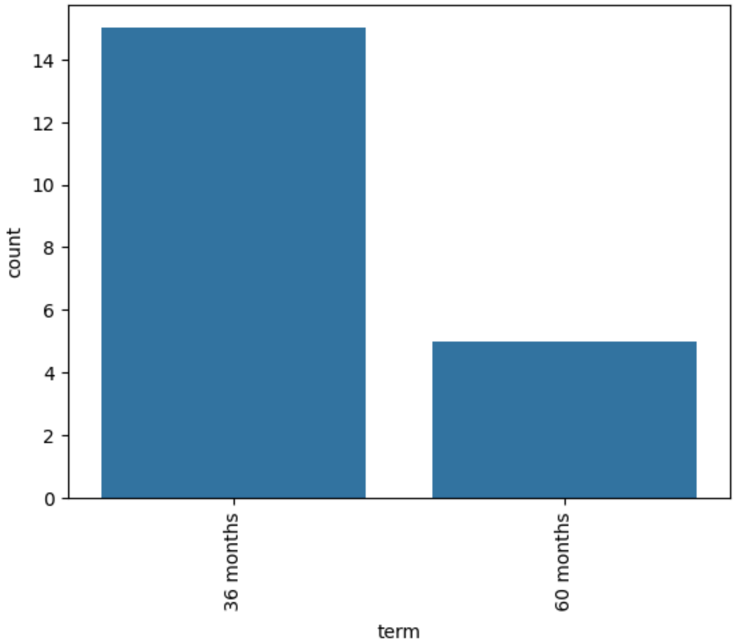
```
data.head(5)
```

	loan_amnt	term	int_rate	installment	grade	sub_grade	emp_title	emp_lengt
0	10000.0	36 months	11.44	329.48	B	B4	Marketing	10+ years
1	8000.0	36 months	11.99	265.68	B	B5	Credit analyst	4 years
2	15600.0	36 months	10.49	506.97	B	B3	Statistician	< 1 yea
3	7200.0	36 months	6.49	220.65	A	A2	Client Advocate	6 years
4	24375.0	60 months	17.27	609.33	C	C5	Destiny Management Inc.	9 years

5 rows × 27 columns

```
sns.countplot(x='term',data=p)
plt.xticks(rotation=90)
# in this visualizing how many people opted 36 months plan and 60 months plan
# here more number of people are opted for 36 months plan
```

```
([0, 1], [Text(0, 0, ' 36 months'), Text(1, 0, ' 60 months')])
```



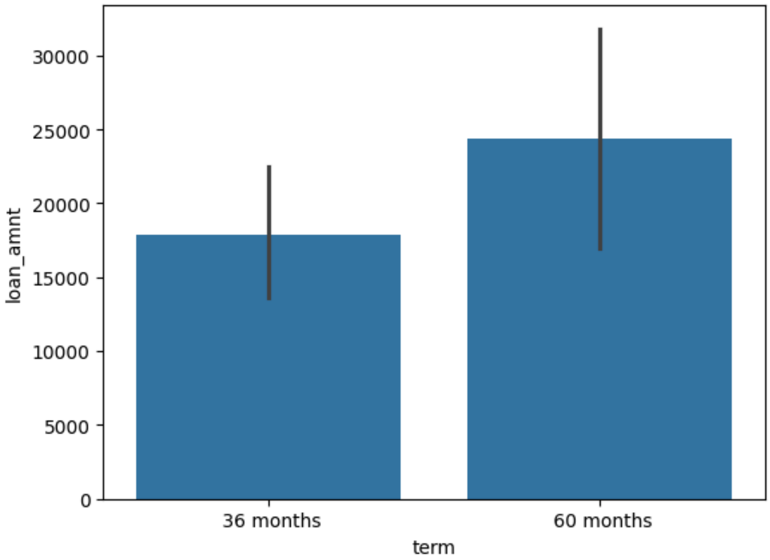
```
data.head(3)
```

	loan_amnt	term	int_rate	installment	grade	sub_grade	emp_title	emp_length
0	10000.0	36 months	11.44	329.48	B	B4	Marketing	10+ years
1	8000.0	36 months	11.99	265.68	B	B5	Credit analyst	4 years
2	15600.0	36 months	10.49	506.97	B	B3	Statistician	< 1 year

3 rows × 27 columns

```
sns.barplot(x='term',y='loan_amnt',data=p)
# for 36 months the highest loan_amnt 18000
# for 60 months the highest loan_amnt is 24000
# more months for more loan amount in order to clear thats why term will be more for that
```

```
<Axes: xlabel='term', ylabel='loan_amnt'>
```



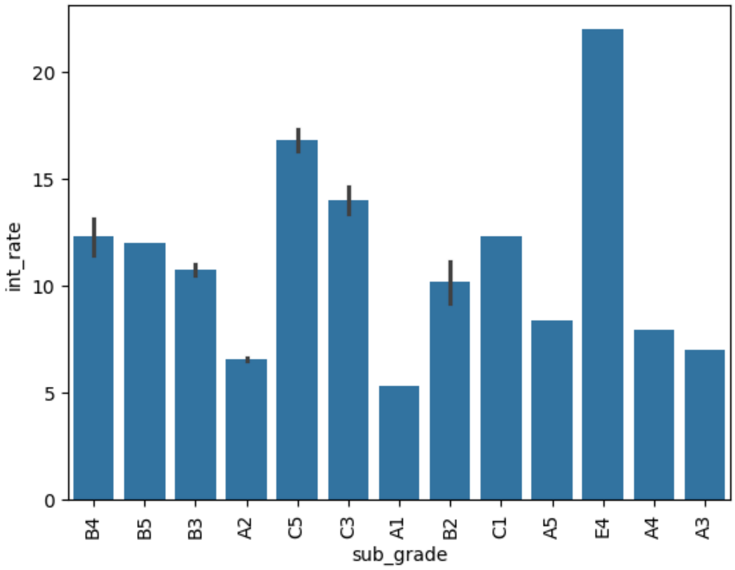
```
data
```

	loan_amnt	term	int_rate	installment	grade	sub_grade	emp_title	emp_l
0	10000.0	36 months	11.44	329.48	B	B4	Marketing	10+
1	8000.0	36 months	11.99	265.68	B	B5	Credit analyst	4
2	15600.0	36 months	10.49	506.97	B	B3	Statistician	<
3	7200.0	36 months	6.49	220.65	A	A2	Client Advocate	6
4	24375.0	60 months	17.27	609.33	C	C5	Destiny Management Inc.	9
...	...	...	...	...	...	...	...	
2017	10625.0	36 months	14.33	364.85	C	C1	white wave food	3
2018	6600.0	36 months	14.46	227.05	C	C4	SUPERVISOR	10+
2019	16800.0	60 months	14.31	393.62	C	C4	Sr. Budget Manager	10+
2020	17050.0	36 months	15.31	593.64	C	C2	Safeway	10+
2021	35000.0	60 months	17.57	880.61	D	D4	Financial Advisor	7

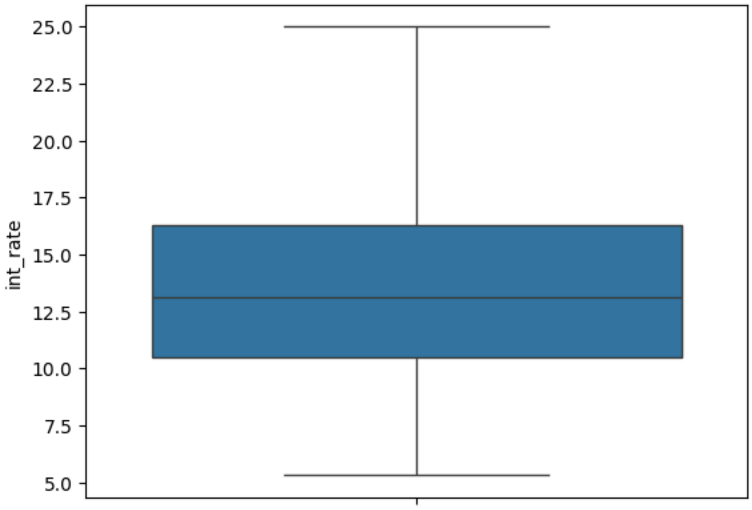
2000 rows × 27 columns

```
sns.barplot(x='sub_grade',y='int_rate',data=p)
plt.xticks(rotation=90)
# here subgrade implies based on credit score,history,
#employment status of a person,annual income the borrowers are classified in to subgrades
# A1,A2 are safest or risk free borrowers  e1,e2 are not safest  they are risky borrowers
# for banks also providing more intrest rate for risky borrowers
# less intrest rate for non risky borrowers

```

```
([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12],
 [Text(0, 0, 'B4'),
  Text(1, 0, 'B5'),
  Text(2, 0, 'B3'),
  Text(3, 0, 'A2'),
  Text(4, 0, 'C5'),
  Text(5, 0, 'C3'),
  Text(6, 0, 'A1'),
  Text(7, 0, 'B2'),
  Text(8, 0, 'C1'),
  Text(9, 0, 'A5'),
  Text(10, 0, 'E4'),
  Text(11, 0, 'A4'),
  Text(12, 0, 'A3')])
```



```
# checking for outliers
data.head(5)
iqr=data['int_rate'].quantile(0.75)-data['int_rate'].quantile(0.25)
iqr
sns.boxplot(data['int_rate'])
iqr
upper_limit=data['int_rate'].quantile(0.75)+1.5*(iqr)
lower_limit=data['int_rate'].quantile(0.25)-1.5*(iqr)


```



```
outliers=data[data['int_rate']>upper_limit]
```

```
outliers
```

loan_amnt	term	int_rate	installment	grade	sub_grade	emp_title	emp_length	hoı

0 rows × 27 columns

```
data=data[data['int_rate']<upper_limit]
# here the values greater than upper limit are considered as an outliers less than that is normal data
# 742 rows are dropped because of outliers
```

```
Start coding or generate with AI.
```

```
data
```

	loan_amnt	term	int_rate	installment	grade	sub_grade	emp_title	emp_l
0	10000.0	36 months	11.44	329.48	B	B4	Marketing	10+
1	8000.0	36 months	11.99	265.68	B	B5	Credit analyst	4
2	15600.0	36 months	10.49	506.97	B	B3	Statistician	<
3	7200.0	36 months	6.49	220.65	A	A2	Client Advocate	6
4	24375.0	60 months	17.27	609.33	C	C5	Destiny Management Inc.	9
...	...	...	...	...	...	...	...	
2017	10625.0	36 months	14.33	364.85	C	C1	white wave food	3
2018	6600.0	36 months	14.46	227.05	C	C4	SUPERVISOR	10+
2019	16800.0	60 months	14.31	393.62	C	C4	Sr. Budget Manager	10+
2020	17050.0	36 months	15.31	593.64	C	C2	Safeway	10+
2021	35000.0	60 months	17.57	880.61	D	D4	Financial Advisor	7

1995 rows × 27 columns

```

data
```

	loan_amnt	term	int_rate	installment	grade	sub_grade	emp_title	emp_l
0	10000.0	36 months	11.44	329.48	B	B4	Marketing	10+
1	8000.0	36 months	11.99	265.68	B	B5	Credit analyst	4
2	15600.0	36 months	10.49	506.97	B	B3	Statistician	<
3	7200.0	36 months	6.49	220.65	A	A2	Client Advocate	6
4	24375.0	60 months	17.27	609.33	C	C5	Destiny Management Inc.	9
...	...	...	...	...	...	...	...	
2017	10625.0	36 months	14.33	364.85	C	C1	white wave food	3
2018	6600.0	36 months	14.46	227.05	C	C4	SUPERVISOR	10+
2019	16800.0	60 months	14.31	393.62	C	C4	Sr. Budget Manager	10+
2020	17050.0	36 months	15.31	593.64	C	C2	Safeway	10+
2021	35000.0	60 months	17.57	880.61	D	D4	Financial Advisor	7

1995 rows × 27 columns

```
# finding relationship between different features  how much correlation is there among different independent features
data[data['loan_status']!=0]

```

	loan_amnt	term	int_rate	installment	grade	sub_grade	emp_title	emp_l
0	10000.0	36 months	11.44	329.48	B	B4	Marketing	10+
1	8000.0	36 months	11.99	265.68	B	B5	Credit analyst	4
2	15600.0	36 months	10.49	506.97	B	B3	Statistician	<
3	7200.0	36 months	6.49	220.65	A	A2	Client Advocate	6
4	24375.0	60 months	17.27	609.33	C	C5	Destiny Management Inc.	9
...	...	...	...	...	...	...	...	
2017	10625.0	36 months	14.33	364.85	C	C1	white wave food	3
2018	6600.0	36 months	14.46	227.05	C	C4	SUPERVISOR	10+
2019	16800.0	60 months	14.31	393.62	C	C4	Sr. Budget Manager	10+
2020	17050.0	36 months	15.31	593.64	C	C2	Safeway	10+
2021	35000.0	60 months	17.57	880.61	D	D4	Financial Advisor	7

1995 rows × 27 columns

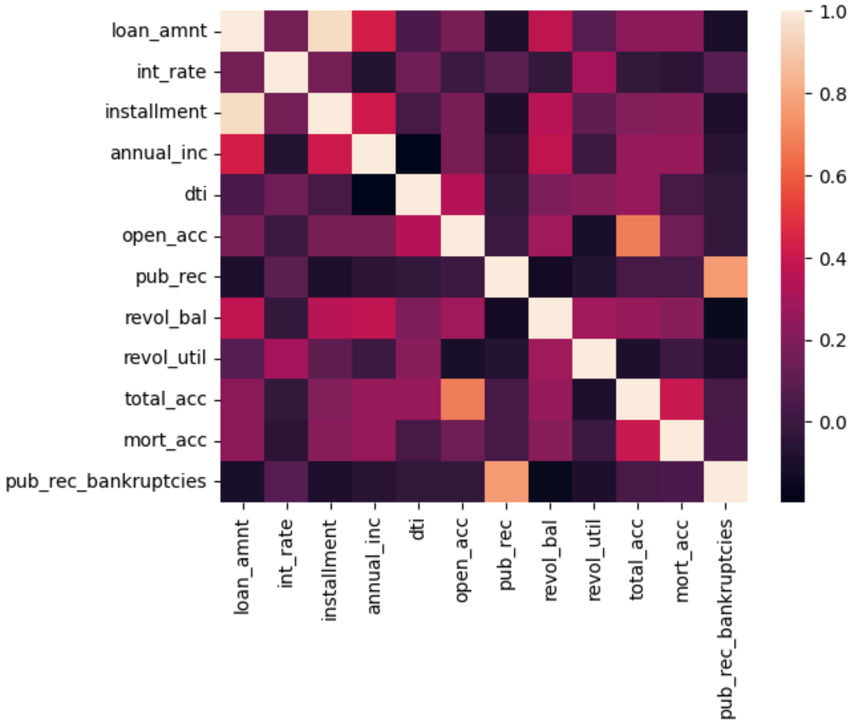
```
# in order to visualize it we are using heatmaps
import seaborn as sns
data.corr()
# it gives info of relation ship if the corelation coefficient value is approximately  equal to 1 then  there
# is a strong corelation is there  and positive corelation is there  among the features
# if it is equal to -1  then also strong corelation will be there but negative realion ship will be there among the features
```

```
<ipython-input-147-a8a95c6733ea>:3: FutureWarning: The default value of numeric_only
  data.corr()
```

	loan_amnt	int_rate	installment	annual_inc	dti	open_ac
loan_amnt	1.000000	0.162233	0.949505	0.436051	0.044343	0.18258
int_rate	0.162233	1.000000	0.162719	-0.071475	0.145377	0.00109
installment	0.949505	0.162719	1.000000	0.412926	0.038269	0.17389
annual_inc	0.436051	-0.071475	0.412926	1.000000	-0.195032	0.17916
dti	0.044343	0.145377	0.038269	-0.195032	1.000000	0.34534
open_acc	0.182580	0.001098	0.173896	0.179162	0.345349	1.00000
pub_rec	-0.093920	0.091129	-0.087728	-0.053697	-0.027934	-0.00067
revol_bal	0.370209	-0.028460	0.357469	0.374089	0.187107	0.28645
revol_util	0.084119	0.300191	0.109794	0.005211	0.211925	-0.11300
total_acc	0.228821	-0.030797	0.205956	0.260376	0.255460	0.68570
mort_acc	0.228476	-0.052571	0.212612	0.262846	0.032899	0.14210
pub_rec_bankruptcies	-0.108777	0.083819	-0.097273	-0.066238	-0.025011	-0.02728

```
sns.heatmap(data.corr())
```

```
<ipython-input-148-8b96879b4d02>:1: FutureWarning: The default value of numeric_only
  sns.heatmap(data.corr())
<Axes: >
```



```
# label encoding is required because features contains text data that data we cant use it in the mathematical model
# thats why we are going to convert text data into numerical values
data=data.head(2000)
```

```
y=data['loan_status']
x=data.drop(columns=['loan_status'],axis=1)

```

```
x.columns
```

```
Index(['loan_amnt', 'term', 'int_rate', 'installment', 'grade', 'sub_grade',
       'emp_title', 'emp_length', 'home_ownership', 'annual_inc',
       'verification_status', 'issue_d', 'purpose', 'title', 'dti',
       'earliest_cr_line', 'open_acc', 'pub_rec', 'revol_bal', 'revol_util',
       'total_acc', 'initial_list_status', 'application_type', 'mort_acc',
       'pub_rec_bankruptcies', 'address'],
      dtype='object')
```

```
y=y.to_frame()

```

```
y=y['loan_status'].map({'Fully Paid':0,'Charged Off':1})
```

```
Start coding or generate with AI.
```

```
from sklearn.model_selection import train_test_split
```

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=78)
```

```
x_train=x_train[x_train['emp_title']!=0]
```

```
y_train
```

```
617     0
863     0
111     0
1054    0
1835    1
       ..
108     0
40      0
1969    0
```

```
484     0
724     0
Name: loan_status, Length: 1396, dtype: int64
```

```
from sklearn.preprocessing import LabelEncoder
```

```


lr=LabelEncoder()
x_train['emp_title']
```

```
617        Teaching Assistant
863                 secretary
111     Systems Administrator
1054     saint francis church
1835                   Driver
                ...
555           Cocktail server
711                Supervisor
108               Maintenance
1969                Machinist
724         Financial Advisor
Name: emp_title, Length: 1328, dtype: object
```

```
x_train['emp_title'],x_train['home_ownership'],=lr.fit_transform(x_train['emp_title']),lr.fit_transform(x_train['emp_title'])
```

```
<ipython-input-170-feff6f5f2746>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus
  x_train['emp_title'],x_train['home_ownership'],=lr.fit_transform(x_train['emp_title']),lr.fit_transform(x_train['emp_title'])
```

```
x_train['initial_list_status']=lr.fit_transform(x_train['initial_list_status'])
x_train
```

```
<ipython-input-173-4ddcf83ed003>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/us
  x_train['initial_list_status']=lr.fit_transform(x_train['initial_list_status'])
```

	loan_amnt	term	int_rate	installment	grade	sub_grade	emp_title	emp_leng
617	5000.0	36 months	11.47	164.81	B	B5	864	2 yea
863	12000.0	36 months	11.67	396.69	B	B4	1118	10+ yea
111	5000.0	36 months	7.62	155.81	A	A3	857	10+ yea
1054	5000.0	36 months	11.48	164.85	B	B2	1111	4 yea
1835	7500.0	36 months	11.99	249.08	C	C1	282	5 yea
...	...	...	...	...	...	...	...	
555	8400.0	36 months	15.77	294.37	D	D1	184	< 1 ye
711	20000.0	60 months	11.53	440.16	B	B5	843	10+ yea
108	12000.0	36 months	7.69	374.33	A	A4	488	10+ yea
1969	12000.0	36 months	5.32	361.38	A	A1	486	9 yea
724	15000.0	36 months	7.12	463.98	A	A3	329	10+ yea

1328 rows × 26 columns

```
x_train['application_type']=lr.fit_transform(x_train['application_type'])
x_train
```

```
<ipython-input-176-9520e72b892b>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/us
  x_train['application_type']=lr.fit_transform(x_train['application_type'])
```

	loan_amnt	term	int_rate	installment	grade	sub_grade	emp_title	emp_leng
617	5000.0	36 months	11.47	164.81	B	B5	864	2 yea
863	12000.0	36 months	11.67	396.69	B	B4	1118	10+ yea
111	5000.0	36 months	7.62	155.81	A	A3	857	10+ yea
1054	5000.0	36 months	11.48	164.85	B	B2	1111	4 yea
1835	7500.0	36 months	11.99	249.08	C	C1	282	5 yea
...	...	...	...	...	...	...	...	
555	8400.0	36 months	15.77	294.37	D	D1	184	< 1 ye
711	20000.0	60 months	11.53	440.16	B	B5	843	10+ yea
108	12000.0	36 months	7.69	374.33	A	A4	488	10+ yea
1969	12000.0	36 months	5.32	361.38	A	A1	486	9 yea
724	15000.0	36 months	7.12	463.98	A	A3	329	10+ yea

1328 rows × 26 columns

```
x_train['grade']=lr.fit_transform(x_train['grade'])
```

```
<ipython-input-177-bf4657862768>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus
  x_train['grade']=lr.fit_transform(x_train['grade'])
```

```
x_train['sub_grade']=lr.fit_transform(x_train['sub_grade'])
```

```
<ipython-input-178-ce61d13d6af1>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus
  x_train['sub_grade']=lr.fit_transform(x_train['sub_grade'])
```

```
x_train['sub_grade']=lr.fit_transform(x_train['sub_grade'])
```

```
<ipython-input-180-ce61d13d6af1>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus
  x_train['sub_grade']=lr.fit_transform(x_train['sub_grade'])
```

```
x_train['address']=lr.fit_transform(x_train['address'])
```

```
<ipython-input-182-cb03381d5423>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus
  x_train['address']=lr.fit_transform(x_train['address'])
```

```
x_train
```

	loan_amnt	term	int_rate	installment	grade	sub_grade	emp_title	emp_leng
617	5000.0	36 months	11.47	164.81	1	9	864	2 yea
863	12000.0	36 months	11.67	396.69	1	8	1118	10+ yea
111	5000.0	36 months	7.62	155.81	0	2	857	10+ yea
1054	5000.0	36 months	11.48	164.85	1	6	1111	4 yea
1835	7500.0	36 months	11.99	249.08	2	10	282	5 yea
...	...	...	...	...	...	...	...	
555	8400.0	36 months	15.77	294.37	3	15	184	< 1 ye
711	20000.0	60 months	11.53	440.16	1	9	843	10+ yea
108	12000.0	36 months	7.69	374.33	0	3	488	10+ yea
1969	12000.0	36 months	5.32	361.38	0	0	486	9 yea
724	15000.0	36 months	7.12	463.98	0	2	329	10+ yea

1328 rows × 26 columns

```
def extract_numeric(value):

  return int(''.join(filter(str.isdigit, value)))

# Apply the function to the 'experience' column

x_train['emp_length'] = x_train['emp_length'].apply(extract_numeric)


#This is the code you are required to use
```

```
<ipython-input-186-1e23280ea934>:7: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus
  x_train['emp_length'] = x_train['emp_length'].apply(extract_numeric)
```

```
x_train
```

	loan_amnt	term	int_rate	installment	grade	sub_grade	emp_title	emp_leng
617	5000.0	36 months	11.47	164.81	1	9	864	
863	12000.0	36 months	11.67	396.69	1	8	1118	
111	5000.0	36 months	7.62	155.81	0	2	857	

```
def xtract_numeric(value1):
  return int(" ".join(filter(str.isdigit,value1)))
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-196-5be77f5f5b4e> in <cell line: 5>()
      3 
      4 
----> 5 x_train['term']=x_train['term'].apply(xtract_numeric)
      6 
      7 

                          4 frames

<ipython-input-196-5be77f5f5b4e> in xtract_numeric(value1)
      1 def xtract_numeric(value1):
----> 2   return int(" ".join(filter(str.isdigit,value1)))
      3 
      4 
      5 x_train['term']=x_train['term'].apply(xtract_numeric)

ValueError: invalid literal for int() with base 10: '3 6'
```

```
Start coding or generate with AI.
```

```
x_train
```

	loan_amnt	term	int_rate	installment	grade	sub_grade	emp_title	emp_leng
617	5000.0	36 months	11.47	164.81	1	9	864	
863	12000.0	36 months	11.67	396.69	1	8	1118	