

# CREDIT RISK PREDICTION

Bagas Shalahuddin Wahid | bagas.shalahuddin@ui.ac.id



Official  
Internship Partner



Kementerian Komunikasi  
dan Informatika RI

# Bussiness Understanding

01

“Business Understanding”. Dalam project ini, dapat dipahami bahwa dalam proses pengajuan kredit pada lembaga peminjaman dibutuhkan sistem yang secara cepat dan tepat sebuah peminjam beresiko atau tidak. Untuk itu dibutuhkan sebuah model yang dapat melakukan penilaian terhadap ajuan kredit dan memprediksi credit-risk dari ajuan kredit tersebut.

Tugasnya adalah membangun model yang dapat memprediksi credit risk menggunakan dataset yang disediakan oleh company yang terdiri dari data pinjaman yang diterima dan yang ditolak

# 02

## Analytic Approach

Setelah melakukan pemahaman dari segi bisnis dari project ini, diperlukan “Analytic Approach” untuk tahap kedua ini dilakukan pendefinisian masalah dan pemecahannya. Pada kasus ajuan kredit ini, permasalahan yang dihadapi adalah bagaimana caranya untuk dapat menentukan credit-risk dalam suatu pengajuan dengan cepat dan tepat.

# 03

## Data Requirements & Data Collections

Data Requirements Dalam project ini, terdapat dua data diberikan, yang merupakan file csv bernama “loan\_data\_2007\_2014” dan “LCDatadictionary”. “loan\_data\_2007\_2014” merupakan tabel yang terdiri dari 78 kolom berisikan data personal dari pengaju kredit yang diwakili oleh member\_id, tabel ini meliputi pendapatan, pekerjaan, tempat tinggal, loan status, deskripsi pinjaman, tujuan dari meminjam dan lain-lain. Sedangkan “LCDatadictionary” merupakan tabel 3 kolom yang berisikan penjelasan setiap kolom dari “loan\_data\_2007\_2014”,

# 04

## Data Understanding

Unnamed: 0		id	member_id	loan_amnt	funded_amnt	funded_amnt_inv	term	int_rate	installment	grade	...	total_bal_il	il_util	open_rv_1
0	1077501	1296599	5000	5000	4975.0	36 months	10.65	162.87	B ...	NaN	NaN	N		
1	1077430	1314167	2500	2500	2500.0	60 months	15.27	59.83	C ...	NaN	NaN	N		
2	1077175	1313524	2400	2400	2400.0	36 months	15.96	84.33	C ...	NaN	NaN	N		
3	1076863	1277178	10000	10000	10000.0	36 months	13.49	339.31	C ...	NaN	NaN	N		
4	1075358	1311748	3000	3000	3000.0	60 months	12.69	67.79	B ...	NaN	NaN	N		
5	1075269	1311441	5000	5000	5000.0	36 months	7.90	156.46	A ...	NaN	NaN	N		
6	1069639	1304742	7000	7000	7000.0	60 months	15.96	170.08	C ...	NaN	NaN	N		

6 rows x 75 columns

Data understanding dibutuhkan untuk mengolah data,  
Dapat dilihat terdapat beberapa type data: int, float, object, dan dengan sekilas terlihat beberapa missing value.

0	Unnamed: 0	466285	non-null	int64	35	total_acc	466256	non-null	float64
1	id	466285	non-null	int64	36	initial_list_status	466285	non-null	object
2	member_id	466285	non-null	int64	37	out_prncp	466285	non-null	float64
3	loan_amnt	466285	non-null	int64	38	out_prncp_inv	466285	non-null	float64
4	funded_amnt	466285	non-null	int64	39	total_pymnt	466285	non-null	float64
5	funded_amnt_inv	466285	non-null	float64	40	total_pymnt_inv	466285	non-null	float64
6	term	466285	non-null	object	41	total_rec_prncp	466285	non-null	float64
7	int_rate	466285	non-null	float64	42	total_rec_int	466285	non-null	float64
8	installment	466285	non-null	float64	43	total_rec_late_fee	466285	non-null	float64
9	grade	466285	non-null	object	44	recoveries	466285	non-null	float64
10	sub_grade	466285	non-null	object	45	collection_recovery_fee	466285	non-null	float64
11	emp_title	438697	non-null	object	46	last_pymnt_d	465909	non-null	object
12	emp_length	445277	non-null	object	47	last_pymnt_amnt	466285	non-null	float64
13	home_ownership	466285	non-null	object	48	next_pymnt_d	239071	non-null	object
14	annual_inc	466281	non-null	float64	49	last_credit_pull_d	466243	non-null	object
15	verification_status	466285	non-null	object	50	collections_12_mths_ex_med	466140	non-null	float64
16	issue_d	466285	non-null	object	51	mths_since_last_major_derog	98974	non-null	float64
17	loan_status	466285	non-null	object	52	policy_code	466285	non-null	int64
18	pymnt_plan	466285	non-null	object	53	application_type	466285	non-null	object
19	url	466285	non-null	object	54	annual_inc_joint	0	non-null	float64
20	desc	125983	non-null	object	55	dti_joint	0	non-null	float64
21	purpose	466285	non-null	object	56	verification_status_joint	0	non-null	float64
22	title	466265	non-null	object	57	acc_now_delinq	466256	non-null	float64
23	zip_code	466285	non-null	object	58	tot_coll_amt	396009	non-null	float64
24	addr_state	466285	non-null	object	59	tot_cur_bal	396009	non-null	float64
25	dti	466285	non-null	float64	60	open_acc_6m	0	non-null	float64
26	delinq_2yrs	466256	non-null	float64	61	open_il_6m	0	non-null	float64
27	earliest_cr_line	466256	non-null	object	62	open_il_12m	0	non-null	float64
28	inq_last_6mths	466256	non-null	float64	63	open_il_24m	0	non-null	float64
29	mths_since_last_delinq	215934	non-null	float64	64	mths_since_rcnt_il	0	non-null	float64
30	mths_since_last_record	62638	non-null	float64	65	total_bal_il	0	non-null	float64
31	open_acc	466256	non-null	float64	66	il_util	0	non-null	float64
32	pub_rec	466256	non-null	float64	67	open_rv_12m	0	non-null	float64
33	revol_bal	466285	non-null	int64	68	open_rv_24m	0	non-null	float64
34	revol_util	465945	non-null	float64	69	max_bal_bc	0	non-null	float64
35	total_acc	466256	non-null	float64	70	all_util	0	non-null	float64

dtypes: float64(46), int64(7), object(22)  
memory usage: 266.8+ MB



# 05

## Model Definition

### Defining Variable

```
In [20]: df.loan_status.unique()
Out[20]: array(['Fully Paid', 'Charged Off', 'Current', 'Default',
       'Late (31-120 days)', 'In Grace Period', 'Late (16-30 days)',
       'Does not meet the credit policy. Status:Fully Paid',
       'Does not meet the credit policy. Status:Charged Off'],
       dtype=object)

In [21]: #define values
ambiguous = ['Current', 'In Grace Period']
good = ['Fully Paid', 'Does not meet the credit policy. Status:Fully Paid']

#drop rows that contain ambiguous ending
df3 = df3[df3.loan_status.isin(ambiguous) == False]

#create new column to classify ending
df3['loan_endng'] = np.where(df3['loan_status'].isin(good), 'good', 'bad')

In [22]: # check the balance
plt.title('good vs risky loans balance')
sns.barplot(x=df3.loan_endng.value_counts().index,y=df3.loan_endng.value_counts().values)
Out[22]: <AxesSubplot:title={'center':'good vs risky loans balance'}>
```

Untuk memprediksi suatu pinjaman berisiko atau tidak, jadi, perlu mengetahui akhir dari setiap pinjaman secara historis, apakah itu defaulted / charged off, or fully paid. Seperti yang bisa kita lihat, ada nilai-nilai seperti "Saat ini(Current)", "Dalam Masa Tenggang" ("In Grace Period") yang ambigu. Pengakhiran pinjaman tersebut dapat dilunasi atau dilunasi, jadi kami tidak dapat menggunakan status tersebut. Terlambat("Late") juga agak ambigu, tetapi saya pribadi tidak ingin berinvestasi dalam pinjaman yang terlambat, jadi saya akan mengklasifikasikannya sebagai pinjaman macet.

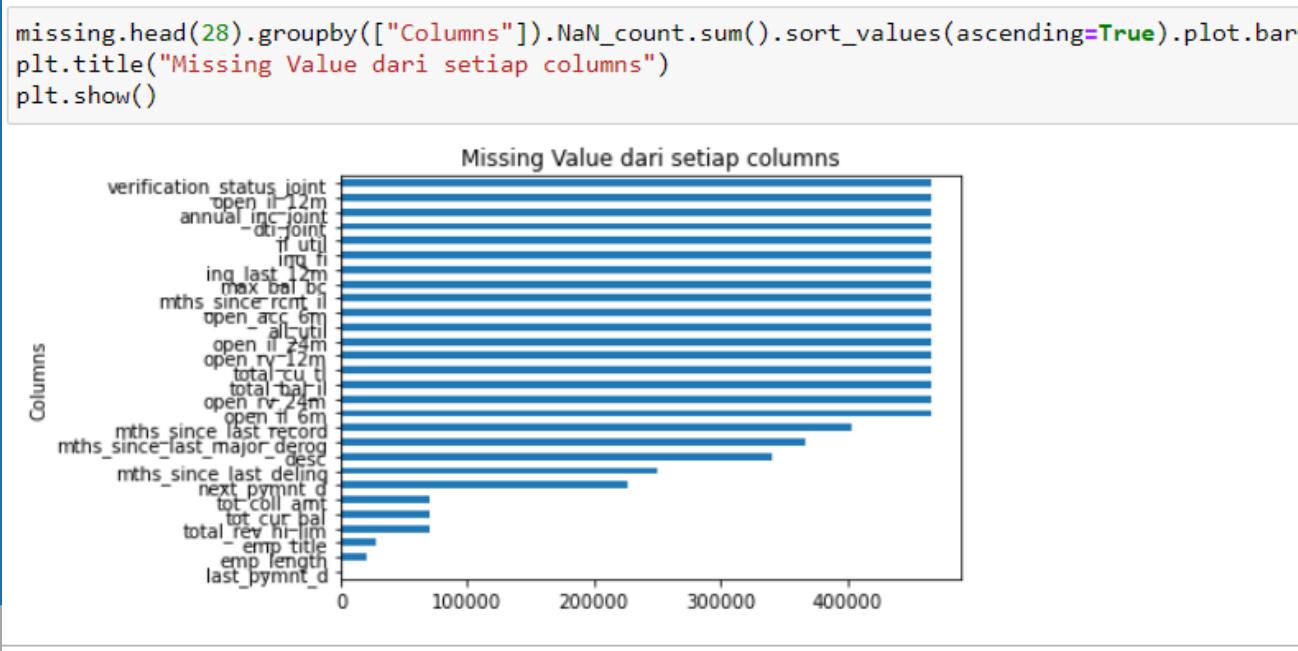
Kami akan mengklasifikasikan akhir pinjaman sebagai berikut:

pinjaman bagus = ["Fully Paid", "Does not meet the credit policy. Status:Fully Paid"]  
Pinjaman berisiko= ["Charged Off", "Late (31-120 days)", "Late (16-30 days)", "Default", "Does not meet the credit policy. Status:Charged Off"]

# 06

# Data Preparation

## Handling Missing value



	Columns	NaN_count	Percentage
48	total_rev_hi_lim	70276	15.071469
47	tot_coll_amt	70276	15.071469
29	revol_util	340	0.072917
44	collections_12_mths_ex_med	145	0.031097
43	last_credit_pull_d	42	0.009007
23	delinq_2yrs	29	0.006219
30	total_acc	29	0.006219
27	pub_rec	29	0.006219
26	open_acc	29	0.006219
25	inq_last_6mths	29	0.006219
24	earliest_cr_line	29	0.006219
19	title	20	0.004289
12	annual_inc	4	0.000858

## Duplicated data

```
print(f'Duplicates in dataframe:{df3.iloc[:,1:]}.\nduplicated().sum()')
Duplicates in dataframe:0, (0.0%)
```

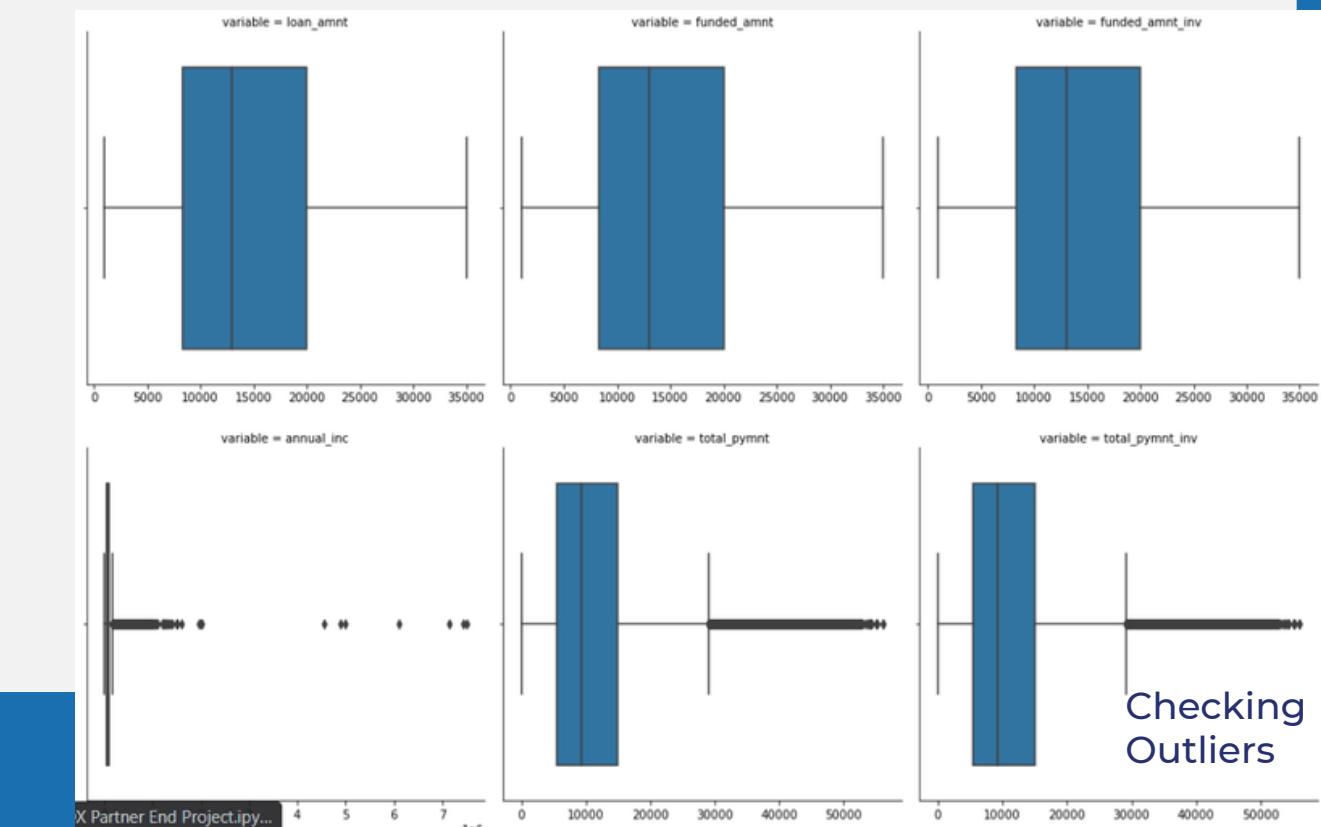
## Data Cleaning

Membuang Kolom yang tidak diperlukan, Handling Missing value, Duplicated data, Checking the outliers.

- Untuk missing value akan dilakukan drop pada columns yang memiliki 100% missing value, selanjutnya untuk column lain yang masih memiliki missing value akan disesuaikan dengan cara fillna(), mean, modus,.
- Untuk duplicated data dapat dilihat tidak terdapat duplicated data.
- Checking Outliers, dengan cara membuat plot, outliers bisa diganti dengan IQR jika mempengaruhi akurasi model.

## Feature Selection

Untuk feature selection, bisa menggunakan Kbest, namun disini saya tidak menggunakan. Karena sedikit banyaknya pasti berpengaruh terhadap model. Cmiw



# 07

# Feature Engineering

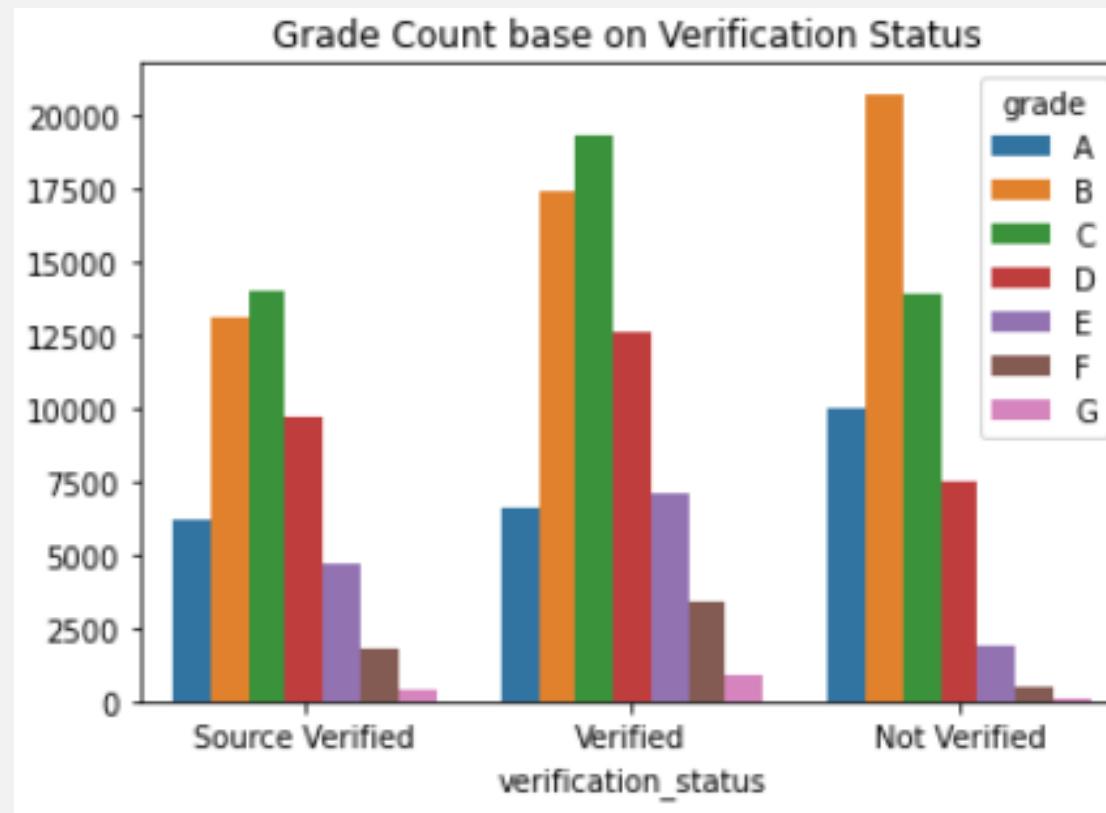
	emp_length	loan_status	mths_since_earliest_cr_line	issue_date	term
42538	10.0	good	230.0	2013-12-01	36
42544	10.0	good	319.0	2013-12-01	36
42546	5.0	good	277.0	2013-12-01	36
42549	10.0	good	347.0	2013-12-01	36
42550	2.0	bad	317.0	2013-12-01	36
...	...	...	...	...	...
466276	5.0	bad	277.0	2014-01-01	60
466277	3.0	bad	233.0	2014-01-01	36
466278	10.0	good	186.0	2014-01-01	36
466281	10.0	bad	246.0	2014-01-01	60
466283	3.0	good	178.0	2014-01-01	36

172157 rows × 5 columns

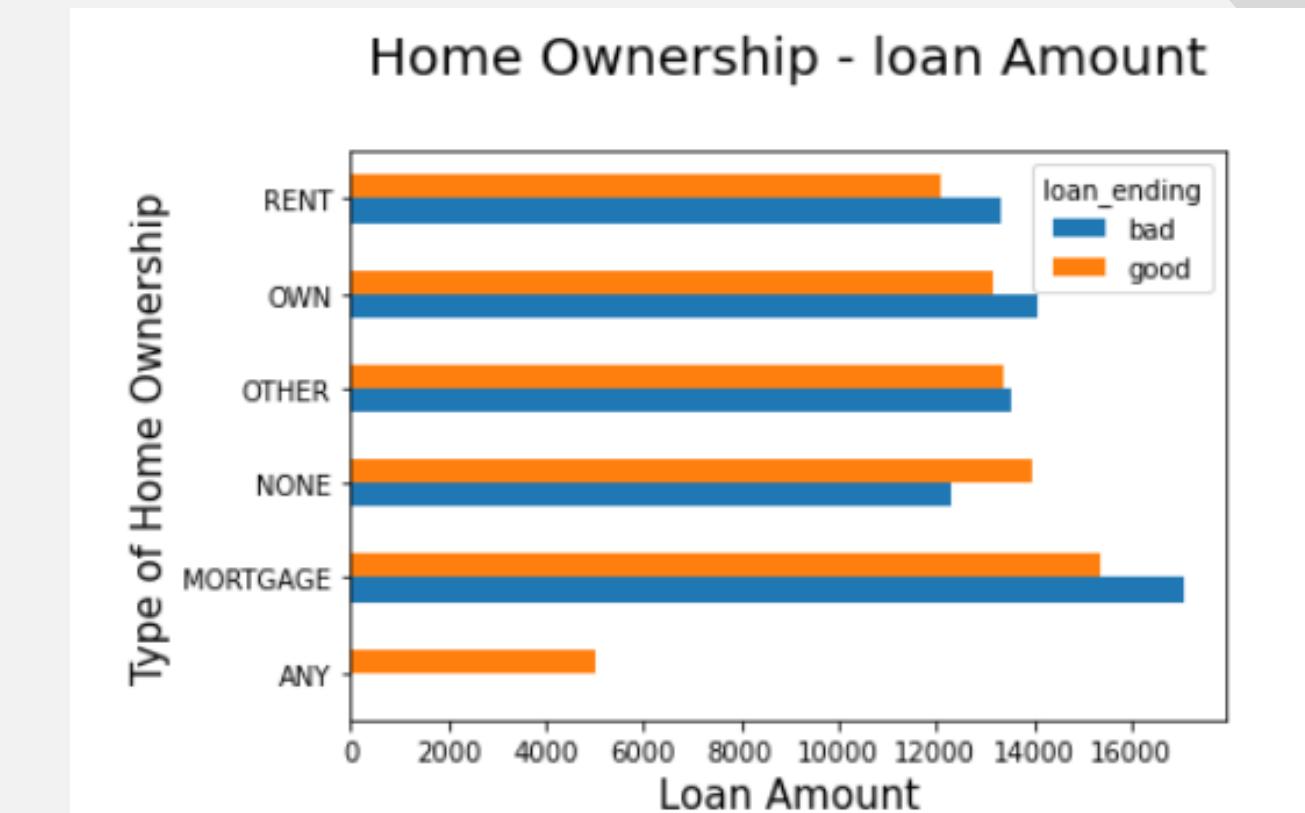
- Feature ‘emp\_title’ ke numerik
- Feature ‘loan\_status’ ke 2 kategori ‘good’ dan ‘bad’
- Feature earliest\_cr\_line ke datetime kemudian ke moths
- Feature ‘Term’ ke numerik
- Feature ‘issue\_date’ ke datetime

# Bussiness Insight

## 08



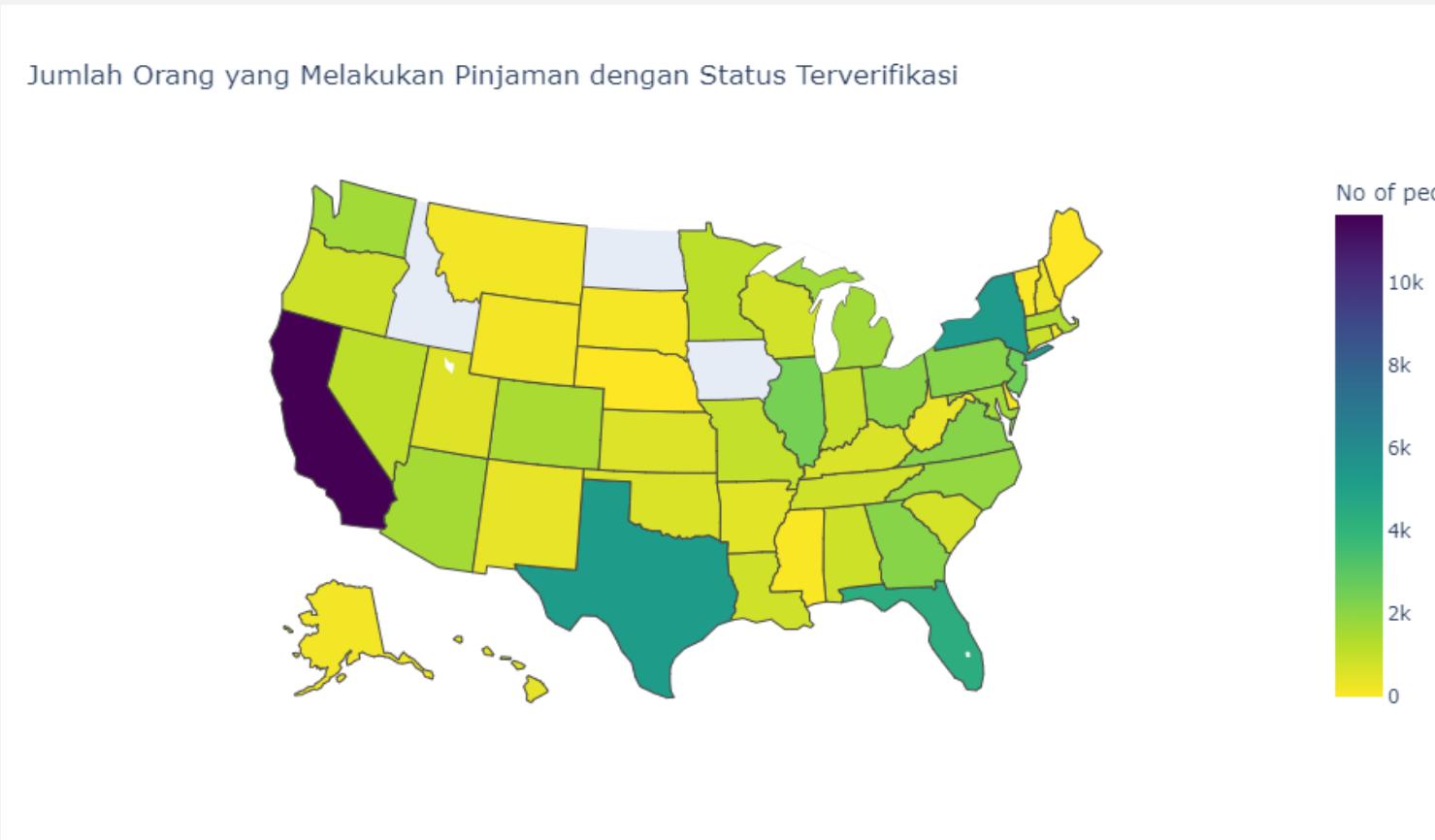
Dapat dilihat dari barchart, bahwa peringkat (Grade) yang ditetapkan oleh Lending Company untuk peminjam berdasarkan statusnya, masih banyak yang tidak terverifikasi padahal gradenya bagus(Grade A:NotVerfied=>20K). Loan grades are set based on both the borrower's credit profile and the nature of the contract. 'A' grade loans represent the lowest risk while 'E' grade loans are the riskiest.



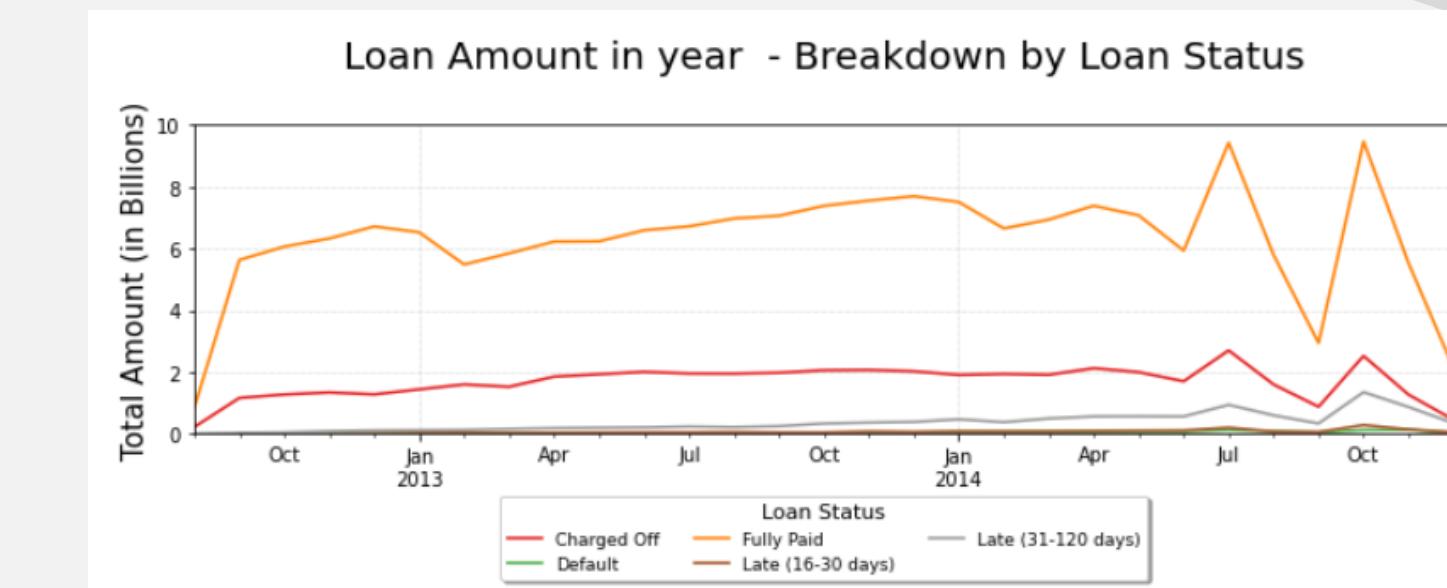
Dapat dilihat dari grafik peminjam dengan tipe rumah MORTGAGE atau yang menggunakan hipotek lebih banyak memiliki resiko untuk melakukan pembayaran pinjaman, untuk langkah selanjutnya dari bidang field collection dapat melakukan tagihan lebih giat kepada peminjam dengan tipe rumah tersebut.

# 08

## Bussiness Insight



Berdasarkan geomap disamping jumlah peminjam terbanyak dan telah diverifikasi berada di negara bagian dari US ada tiga, yaitu California:11.647, New York: 5382, dan Texas:5254 artinya ini menjadi masalah besar bagi pemerintah terkait masalah ekonomi yang ada disana, dan untuk tim dari marketing dari perusahaan pemberi pinjaman dapat menyesuaikan target untuk orang-orang yang berada di wilayah tersebut



Grafik diatas menunjukkan Total pinjaman berdasarkan Status dari peminjam dapat dilihat Untuk tahun 2013 – 2014 awal , tidak terlalu mengalami perubahan signifikan, namun ketikan memasuki pertengahan tahun terdapat 2 kenaikan signifikan yaitu pada bulan Juli dan Oktober, langkah selanjutnya ialah untuk tim marketing Lending Company dapat menggunakan trend ini untuk lebih giat memasarkan produk pinjaman mereka ke peminjam pada bulan Juli dan Oktober dengan persiapan pembuatan konten/ads dan lain lain sebelum bulan tersebut.

# Preprocessing

## 09

### Variabel dengan kategori sedikit

Untuk data dengan nilai unik yang kecil, kita dapat menelusurinya secara visual dengan menggunakan bad loan untuk setiap kategori.

```
# Filtering data with less than 8 unique values
df3.nunique()[df3.nunique() < 8].sort_values()
```

policy_code	1
application_type	1
term	2
pymnt_plan	2
initial_list_status	2
loan_ending	2
verification_status	3
loan_status	5
home_ownership	6
grade	7
collections_12_mths_ex_med	7
dtype: int64	

#### kesimpulan

Untuk kolom yang hanya memiliki kategori <2 maka akan langsung di drop yakni 'policy\_code' & 'application\_type'

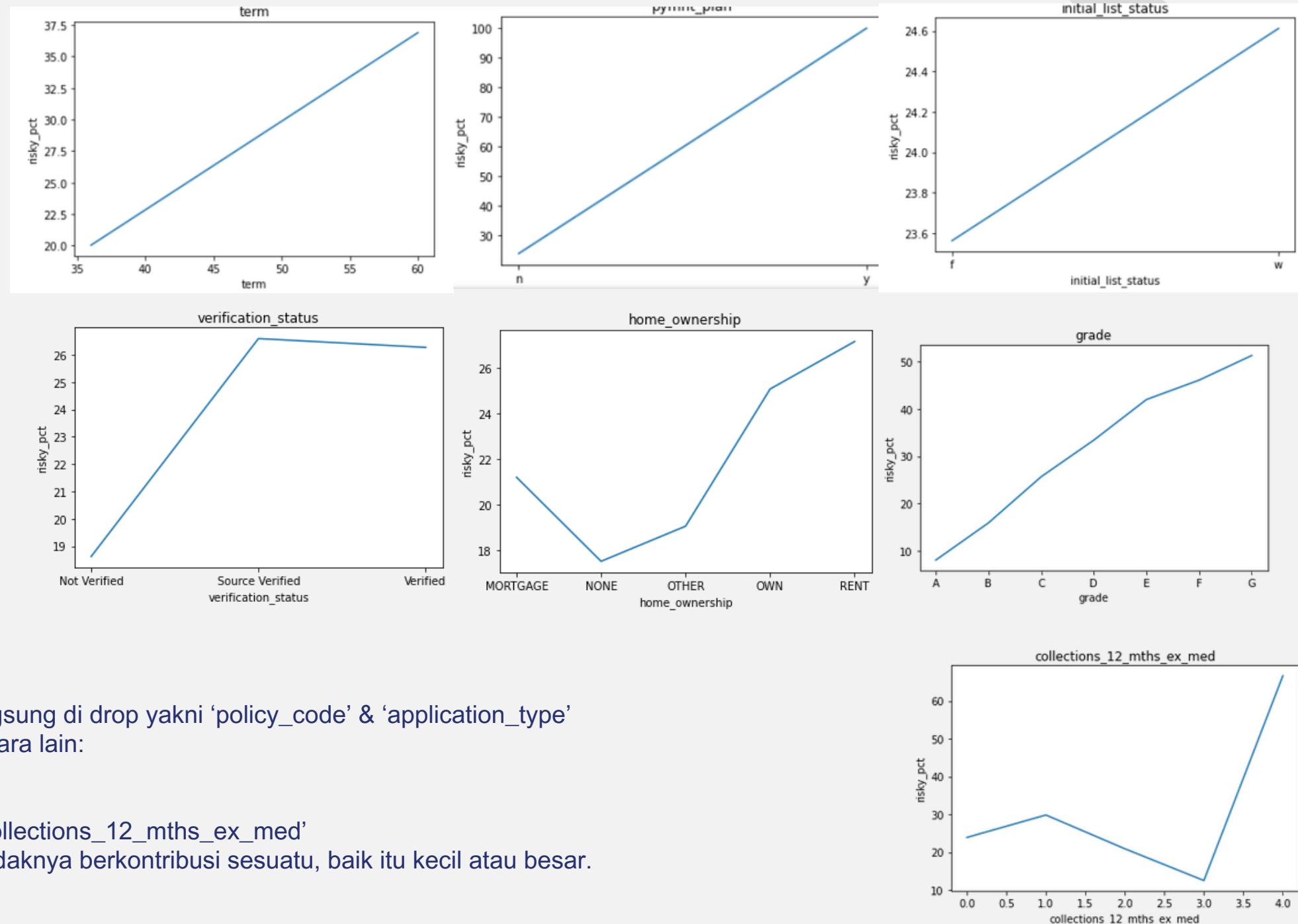
kolom dengan perubahan signifikan rasio 'good' vs 'bad' antara lain:

'term', 'pymnt\_plan', 'initial\_list\_status', 'grade'

kolom dengan sedikit perubahan rasio antara lain:

'home\_ownership', 'verification\_status', 'initial\_list\_status', 'collections\_12\_mths\_ex\_med'

Tapi semuanya bagus, dan bisa dipertahankan, karena setidaknya berkontribusi sesuatu, baik itu kecil atau besar.



# Preprocessing

## 09.1

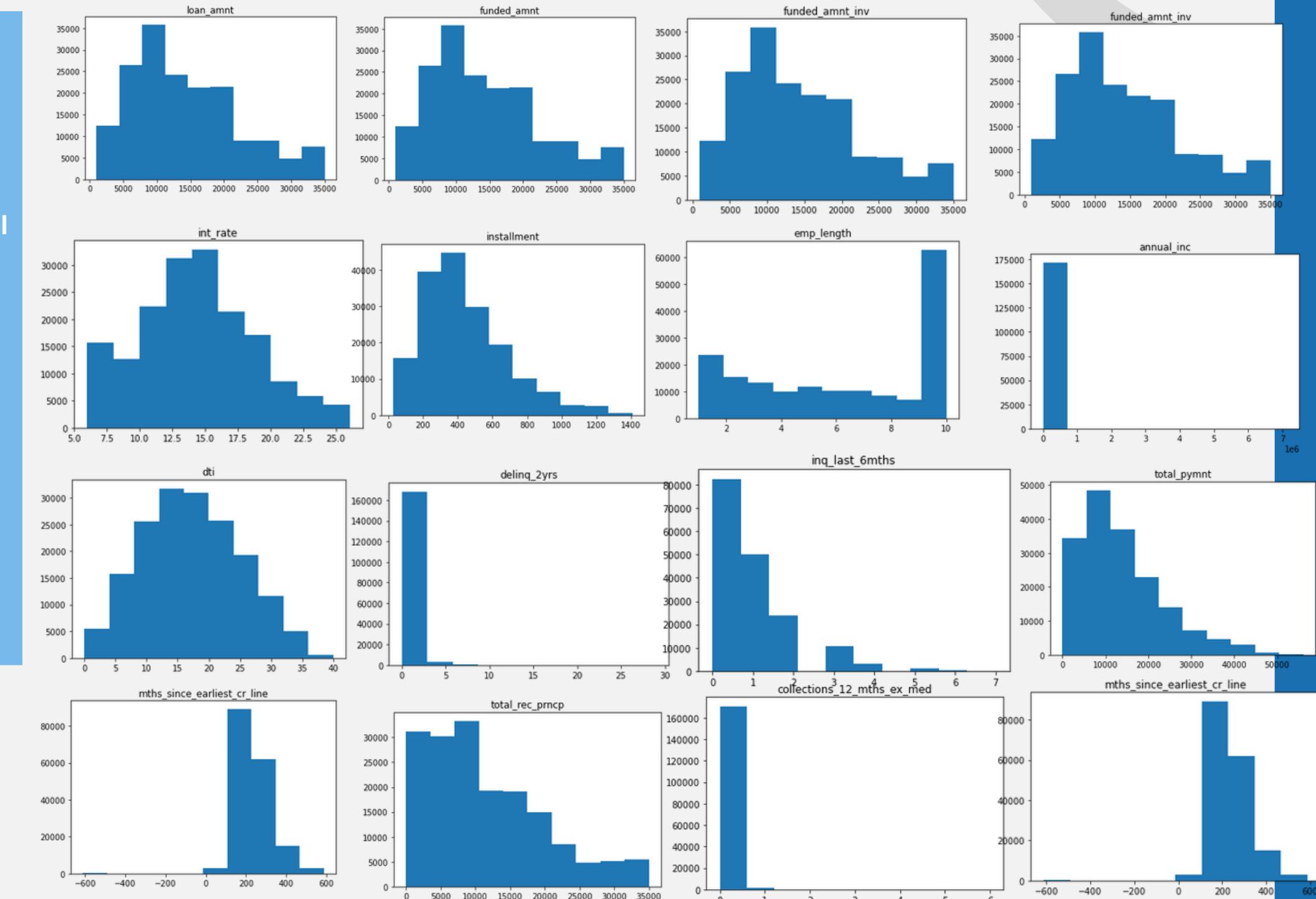
Yang Akan kita lakukan selanjutnya adalah

1. Melihat distribusi dengan membuat histogram
2. Melihat Korrelasi
3. Kemudian membuat pivot terhadap variabel target

```
# numerical
num_data = df3.select_dtypes(exclude='object')
num_data.columns

Index(['loan_amnt', 'funded_amnt', 'funded_amnt_inv', 'term', 'int_rate',
       'installment', 'emp_length', 'annual_inc', 'dti', 'delinq_2yrs',
       'inq_last_6mths', 'open_acc', 'pub_rec', 'revol_bal', 'revol_util',
       'total_acc', 'out_prncp', 'out_prncp_inv', 'total_pymnt',
       'total_pymnt_inv', 'total_rec_prncp', 'total_rec_int',
       'total_rec_late_fee', 'recoveries', 'collection_recovery_fee',
       'last_pymnt_amnt', 'collections_12_mths_ex_med',
       'mths_since_earliest_cr_line'],
      dtype='object')
```

### Numerikal data



# Preprocessing

## 09.2

### Pivot Tabel tehadapt target

loan_heading	annual_inc	collection_recovery_fee	collections_12_mths_ex_med	delinq_2yrs	dti	emp_length	funded_amnt	funded_amnt_inv
bad	66307.401017	80.836508	0.009901	0.322105	19.124398	6.271143	15095.889655	15089.428653
good	75416.645734	0.000000	0.007713	0.273715	16.698700	6.287919	13936.658076	13929.015168

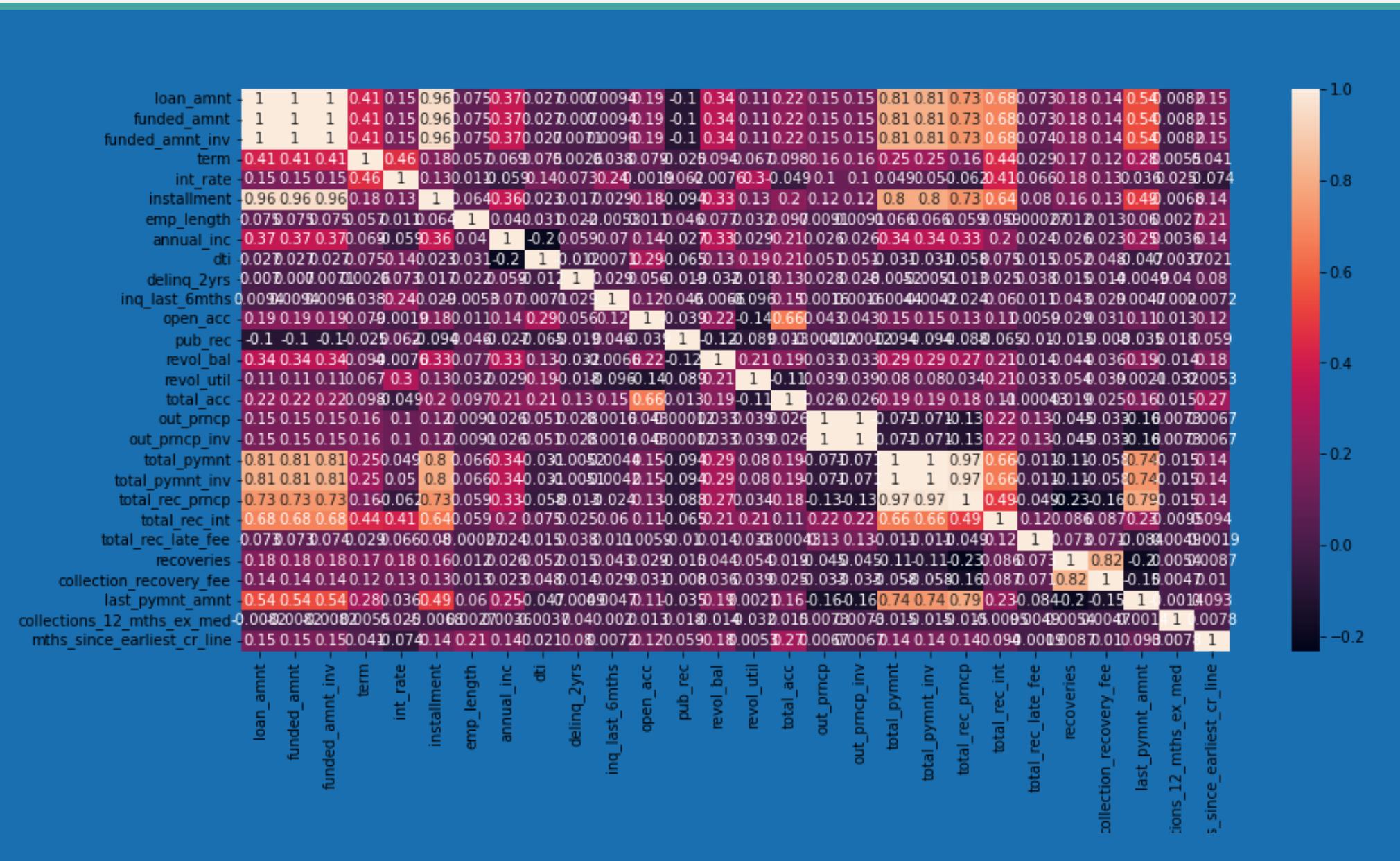
2 rows × 28 columns

inq_last_6mths	installment	...	recoveries	revol_bal	revol_util	term	total_acc	total_pymnt	total_pymnt_inv	total_rec_int	total_rec_late_fee	total_rec_prncp
0.966001	460.138952	...	771.627238	15264.496081	59.980996	44.566894	24.956172	6924.986253	6921.819504	2432.833845	2.685606	3717.839567
0.850538	435.514512	...	0.000000	15741.030286	54.952846	40.611409	26.004292	15933.302043	15924.534205	2012.794098	0.218059	13920.289897

# Preprocessing

## 09.3

### Melihat Korelasi



Di sini, jika ada pasangan fitur-fitur yang memiliki korelasi tinggi maka akan diambil salah satu saja. Nilai korelasi yang dijadikan patokan sebagai korelasi tinggi tidak pasti, umumnya digunakan angka 0.7, namun karena banyaknya data berada diatas angka 0.8 maka dipilih angka 0.8 untuk tidak membuang terlalu banyak feature.

Kesimpulan:

1. Hanya sejumlah kecil data numerik yang terdistribusi secara normal
2. Beberapa data mengandung outlier
3. Seperti yang diharapkan, jumlah angsuran & pinjaman berkorelasi, hampir sempurna. Karena cicilan = jumlah\_pinjaman \* tingkat\_bunga. Meskipun jumlah pinjaman dapat bervariasi, tingkat bunga biasanya tidak terlalu bervariasi.

Berdasarkan tabel pivot, karakteristik pinjaman berisiko:

Berdasarkan catatan pribadi yang buruk:

1. akun delinq yang lebih tinggi
2. kenakalan yang lebih tinggi dalam 2 tahun terakhir
3. pertanyaan yang lebih tinggi dalam 6 bulan terakhir -> pertanyaan sulit dapat memengaruhi penilaian kredit
4. tahun yang lebih rendah sejak pertanyaan terakhir -> lebih rendah = baru-baru ini memiliki pertanyaan kredit
5. Berdasarkan kesulitan pembayaran yang lebih sulit
6. Pendapatan tahunan lebih rendah
7. rasio utang terhadap pendapatan lebih tinggi (dti) -> dti = angsuran bulanan / pendapatan bulanan
8. angsuran & jumlah pinjaman yang lebih tinggi
9. tingkat bunga yang lebih tinggi (biasanya berkorelasi dengan peringkat pinjaman)

# 09.4

# Preprocessing

## Kategorikal data

# Inilah yang akan kita lakukan untuk kategorikal data

1. Untuk 'grade' kita akan menggunakan ordinal encoder atau map
2. dan one hot encoding untuk:

home\_ownership, verification\_status, purpose, addr\_state, initial\_list\_status, initial\_list\_status\_f -> tetapi hanya 1 yang cukup jadi kita akan menjatuhkan 1 kolom dummy

```
cat_data['grade'].unique()
array(['B', 'A', 'E', 'C', 'D', 'F', 'G'], dtype=object)

# 1. transforming grade
grade_map = {
    'A' : 1,
    'B' : 2,
    'C' : 3,
    'D' : 4,
    'E' : 5,
    'F' : 6,
    'G' : 7,
}

cat_data['grade'] = cat_data['grade'].map(grade_map)

cat_data.grade.unique()
array([2, 1, 5, 3, 4, 6, 7], dtype=int64)
```

```
dummies.columns
Index(['home_ownership_ANY', 'home_ownership_MORTGAGE', 'home_ownership_NONE',
       'home_ownership_OTHER', 'home_ownership_OWN', 'home_ownership_RENT',
       'verification_status_Not Verified',
       'verification_status_Source Verified', 'verification_status_Verified',
       'loan_status_Charged Off', 'loan_status_Default',
       'loan_status_Fully Paid', 'loan_status_Late (16-30 days)',
       'loan_status_Late (31-120 days)', 'pymnt_plan_n', 'pymnt_plan_y',
       'purpose_car', 'purpose_credit_card', 'purpose_debt_consolidation',
       'purpose_home_improvement', 'purpose_house', 'purpose_major_purchase',
       'purpose_medical', 'purpose_moving', 'purpose_other',
       'purpose_renewable_energy', 'purpose_small_business',
       'purpose_vacation', 'purpose_wedding', 'addr_state_AL',
       'addr_state_AR', 'addr_state_AZ', 'addr_state_CA', 'addr_state_CO',
       'addr_state_CT', 'addr_state_DC', 'addr_state_DE', 'addr_state_FL',
       'addr_state_GA', 'addr_state_HI', 'addr_state_IA', 'addr_state_IL',
       'addr_state_IN', 'addr_state_KS', 'addr_state_KY', 'addr_state_LA',
       'addr_state_MA', 'addr_state_MD', 'addr_state_ME', 'addr_state_MI',
       'addr_state_MN', 'addr_state_MO', 'addr_state_MS', 'addr_state_MT',
       'addr_state_NC', 'addr_state_NE', 'addr_state_NH', 'addr_state_NJ',
       'addr_state_NM', 'addr_state_NV', 'addr_state_NY', 'addr_state_OH',
       'addr_state_OK', 'addr_state_OR', 'addr_state_PA', 'addr_state_RI',
       'addr_state_SC', 'addr_state_SD', 'addr_state_TN', 'addr_state_TX',
       'addr_state_UT', 'addr_state_VA', 'addr_state_VT', 'addr_state_WA',
       'addr_state_WI', 'addr_state_WV', 'addr_state_WY',
       'initial_list_status_f'],
      dtype='object')
```

# 09.5

## Merge Final Data

# Preprocessing

```
#gabung data numerik dan kategori  
df4 = pd.concat([num_data, cat_data_final], axis = 1).dropna().reset_index().drop('index', axis = 1)  
df4.head()
```

	loan_amnt	term	int_rate	emp_length	annual_inc	dti	delinq_2yrs	inq_last_6mths	open_acc	pub_rec	...	addr_state_TN	addr_state_TX	addr_state_UT
0	12000	36	13.53	10.0	40000.0	16.94	0.0	0.0	7.0	2.0	...	0	0	0
1	3000	36	12.85	10.0	25000.0	24.68	0.0	0.0	5.0	2.0	...	0	0	0
2	28000	36	7.62	5.0	325000.0	18.55	0.0	1.0	15.0	0.0	...	0	0	0
3	24000	36	13.53	10.0	100000.0	22.18	0.0	0.0	14.0	0.0	...	0	0	0
4	8000	36	10.99	2.0	33000.0	15.75	0.0	1.0	9.0	1.0	...	0	0	0

5 rows × 101 columns

```
from sklearn.preprocessing import LabelEncoder  
labelencoder = LabelEncoder()
```

```
df4['loan_heading'] = labelencoder.fit_transform(df4['loan_heading'])
```

```
#pisah variable dependent (y), dependent(X)  
X = df4.drop('loan_heading', axis = 1)  
y = df4['loan_heading']
```

# 10

## Train Test Split

# Modelling

Data dibagi menjadi  
80% Training dan 20%  
Testing

## Modelling

```
: from sklearn.model_selection import train_test_split  
  
: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)  
  
: X_train.shape, X_test.shape, y_train.shape, y_test.shape  
((137725, 100), (34432, 100), (137725,), (34432,))
```

# 10

Karena telah banyak dilakukan pemodelan credit risk, card approval dan nilai accuracy menunjukkan random forest dapat diandalkan maka dengan ini saya menggunakan random forest sebagai algoritma.

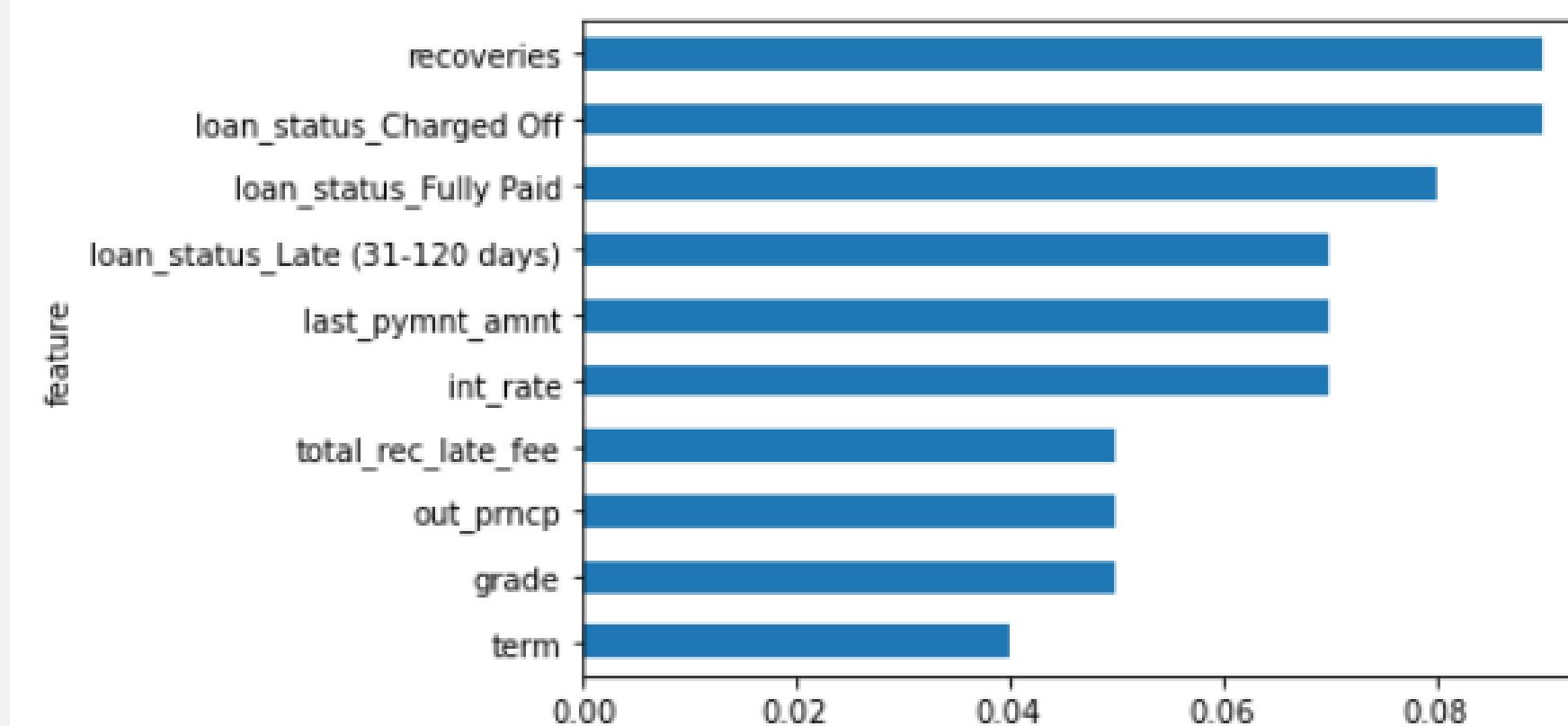
## Random Forest

```
: from sklearn.ensemble import RandomForestClassifier  
# Evaluation  
from sklearn.metrics import confusion_matrix  
from sklearn.metrics import classification_report
```

```
: rfc = RandomForestClassifier(max_depth=1)  
rfc.fit(X_train, y_train)  
pred_y = rfc.predict(X_test)  
print(classification_report(y_test, pred_y))
```

	precision	recall	f1-score	support
0	1.00	0.01	0.03	8156
1	0.77	1.00	0.87	26276
accuracy			0.77	34432
macro avg	0.88	0.51	0.45	34432
weighted avg	0.82	0.77	0.67	34432

## Random Forest



Grafik diatas adalah urutan feature importance dari 10 feature yang digunakan dalam modelling RandomForestClassifier

Jadi, indikasinya adalah ketika recoveries nya besar kemungkinan akan menentukan orang itu dapat dikategorikan baik atau buruk dalam pemberian pinjaman.

Pada gambar disamping kiri, dapat dilihat model ini memiliki accuracy sebesar 0.77(77%)

# Modelling

## Hyper Parameter Turning using RandomSeearchGV

# 11

```
In [132]: rfc_random.fit(X_train, y_train)
```

```
Fitting 5 folds for each of 100 candidates, totalling 500 fits
```

```
Out[132]: RandomizedSearchCV(cv=5, estimator=RandomForestClassifier(max_depth=1),
   n_iter=100, n_jobs=-1,
   param_distributions={'bootstrap': [True, False],
      'max_depth': [10, 20, 30, 40, 50, 60,
                     70, 80, 90, 100, 110,
                     120],
      'max_features': ['auto', 'sqrt'],
      'min_samples_leaf': [1, 3, 4],
      'min_samples_split': [2, 6, 10],
      'n_estimators': [5, 20, 50, 100]},
   random_state=35, verbose=2)
```

```
In [133]: print ('Random grid: ', random_grid, '\n')
# print the best parameters
print ('Best Parameters: ', rfc_random.best_params_, '\n')
```

```
Random grid: {'n_estimators': [5, 20, 50, 100], 'max_features': ['auto', 'sqrt'], 'max_depth': [10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, 120], 'min_samples_split': [2, 6, 10], 'min_samples_leaf': [1, 3, 4], 'bootstrap': [True, False]}
```

```
Best Parameters: {'n_estimators': 20, 'min_samples_split': 10, 'min_samples_leaf': 4, 'max_features': 'sqrt', 'max_depth': 10, 'bootstrap': True}
```

```
: rfbest = pd.DataFrame(rfbest)
rfbest.to_csv('rfbest_param.csv')
```

Menggunakan parameter terbaik

```
: rfc_best = RandomForestClassifier(n_estimators = 20, min_samples_split = 10, min_samples_leaf = 4, max_depth = 10, max_features = 'sqrt', min_samples_leaf = 4, min_samples_split = 10, n_estimators = 20)
```

```
: pred_y_best = rfc_best.predict(X_test)
print(classification_report(y_test, pred_y_best))
```

	precision	recall	f1-score	support
0	1.00	1.00	1.00	8156
1	1.00	1.00	1.00	26276
accuracy			1.00	34432
macro avg	1.00	1.00	1.00	34432
weighted avg	1.00	1.00	1.00	34432

```
: from sklearn.metrics import accuracy_score
print('Training Accuracy :', rfc_best.score(X_train, y_train))
print('Testing Accuracy :', rfc_best.score(X_test, y_test))
```

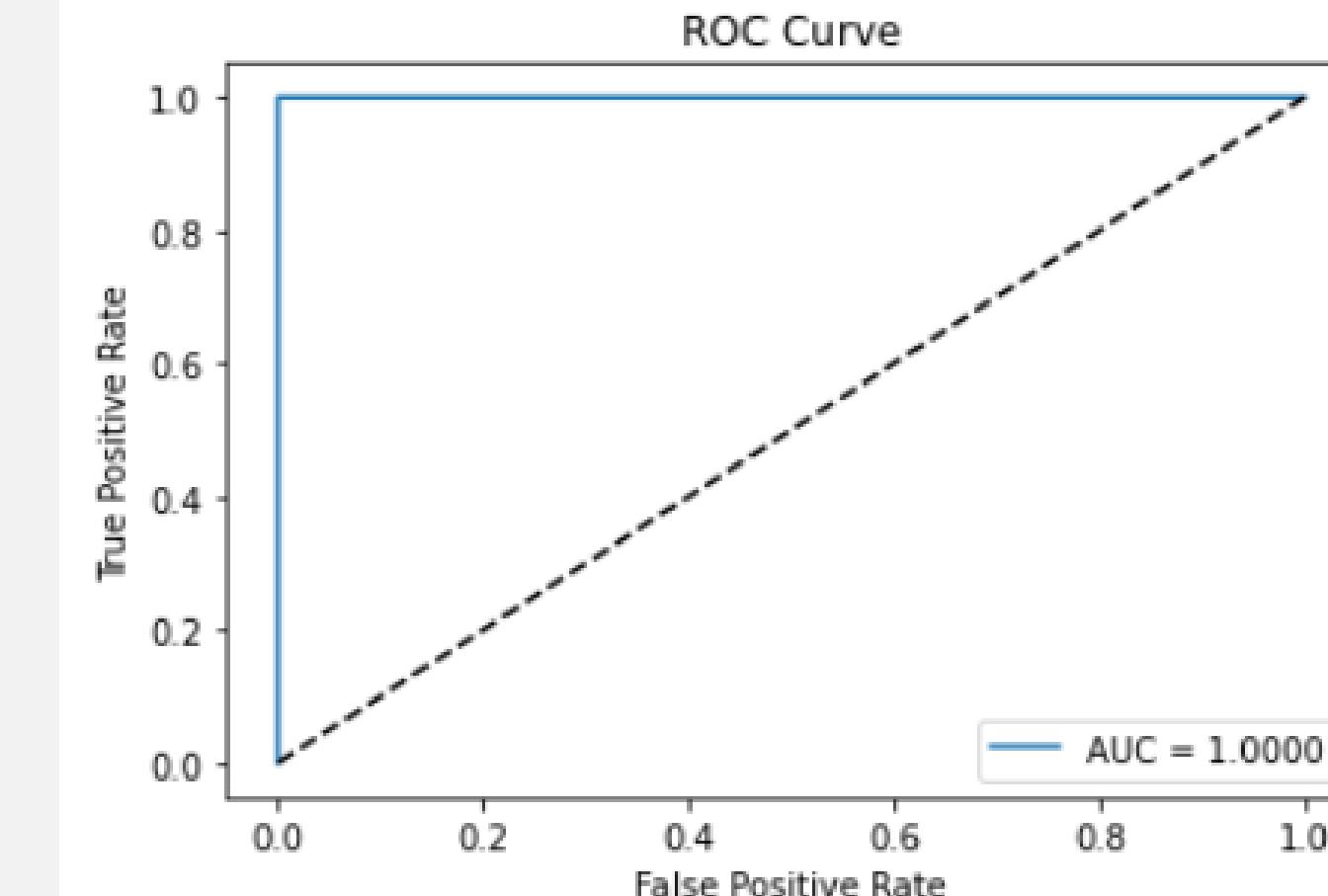
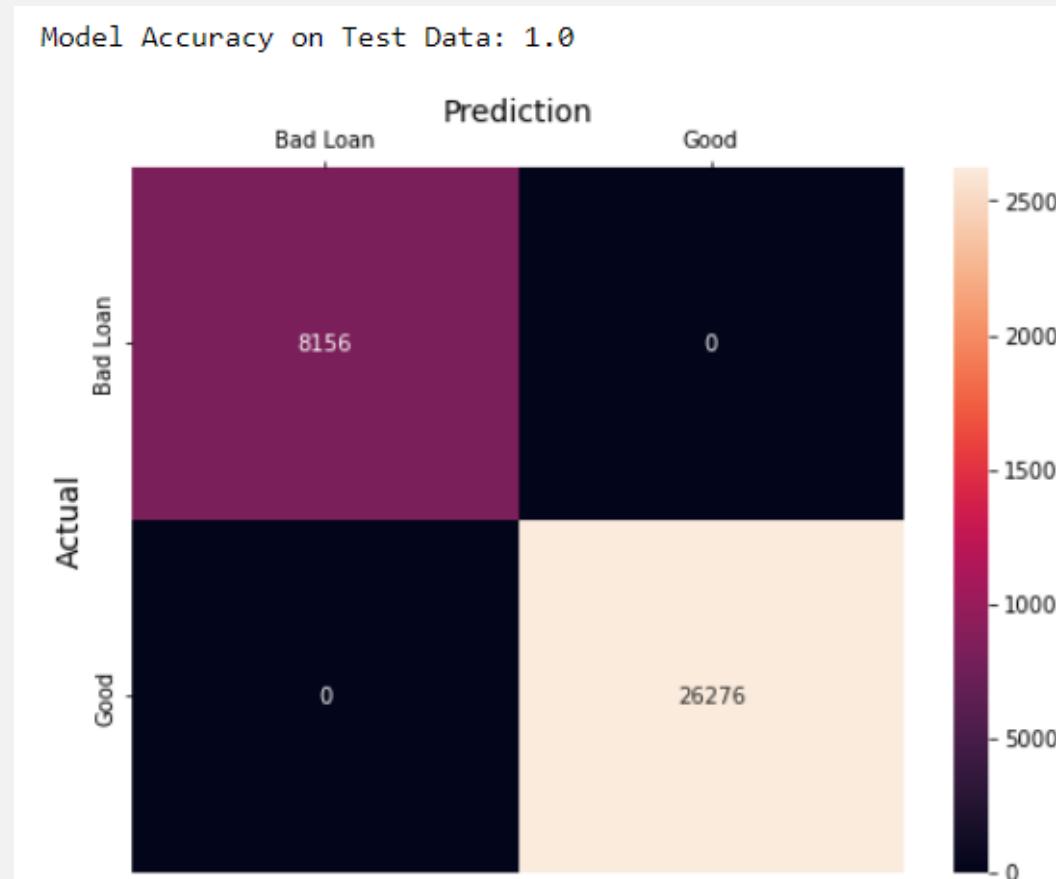
```
Training Accuracy : 1.0
Testing Accuracy : 1.0
```

Setelah dilakukan Hyper parameter Turning didapatkan struktur model terbaik untuk algoritma RandomForestClassifier, sesuai gambar diatas dan digunakan parameter itu untuk modelling selanjutnya dan didapatkan hasil accuracy 1.0 (100%).

# Evaluate Model

# 12

## Confusion Matrix & ROC AUC



- Kesimpulan

Melihat Grafik ROC-AUC dari RandomForestClassifier disamping menghasilkan AUC sebesar 0.96 dimana, model sudah dianggap baik

Dengan penggunaan Algoritma Random Forest:

Hasil yang didapatkan adalah jumlah Loan\_ending/portofolia akan terdiri dari 76.3% pinjaman bagus dan 23.6% pinjaman berisiko, dan sebaiknya Anda akan berinvestasi dalam 100% dari pinjaman bagus yang tersedia.