LOAN
CREDIT RISK MODEL

OLEH: MUH RIVALDI PRASTIANTO

THANK YOU

Muh Rivaldi Prastianto

rivaldi165@gmail.

www.linkedin.com/in/muh-rivaldi

LOAN CREDIT RISK MODEL

Indepth Explanation

LIST

LIST

Of the **466,285 customers** who made loans, there were **10.8% of customers** who were **unable to pay off loans**.

LOAN CREDIT RISK MODEL

Indepth Explanation

LIST

BUSINESS PROBLEM

OBJECTIVE

EXPLORATORY DATA ANALYSIS

MODEL & EVALUATION

BUSINESS SIMULATION

CONCLUSION

- **Predicting** a customer's ability to make **loan repayments** based on the results of Machine Learning
- Provide **recommendations** for **types of loans** based on the results of the Exploratory Data Analysis

LOAN CREDIT RISK MODEL

Indepth Explanation

LIST

BUSINESS PROBLEM

OBJECTIVE

EXPLORATORY DATA ANALYSIS

MODEL & EVALUATION

BUSINESS SIMULATION

CONCLUSION

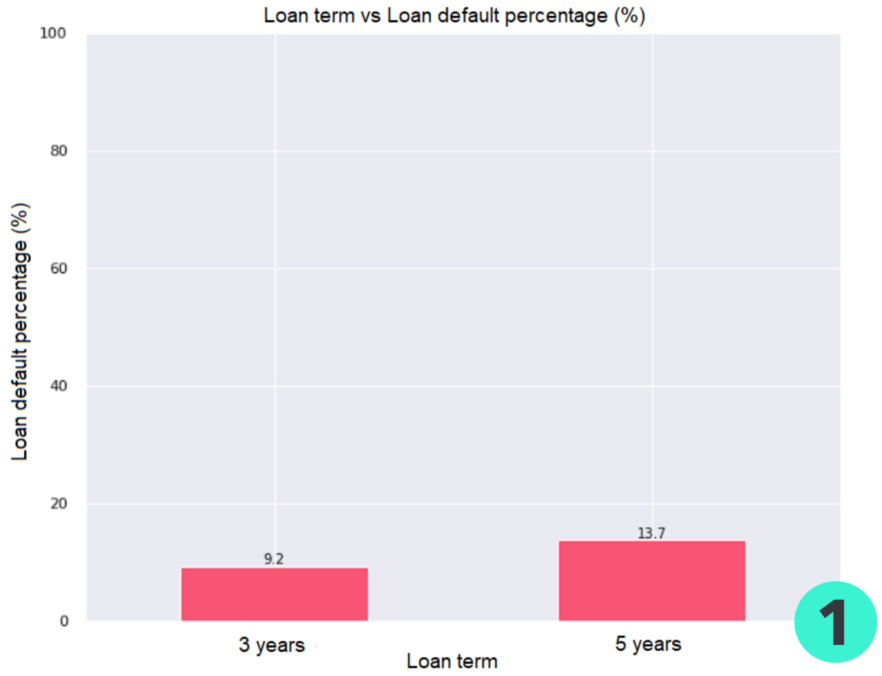
Categorical

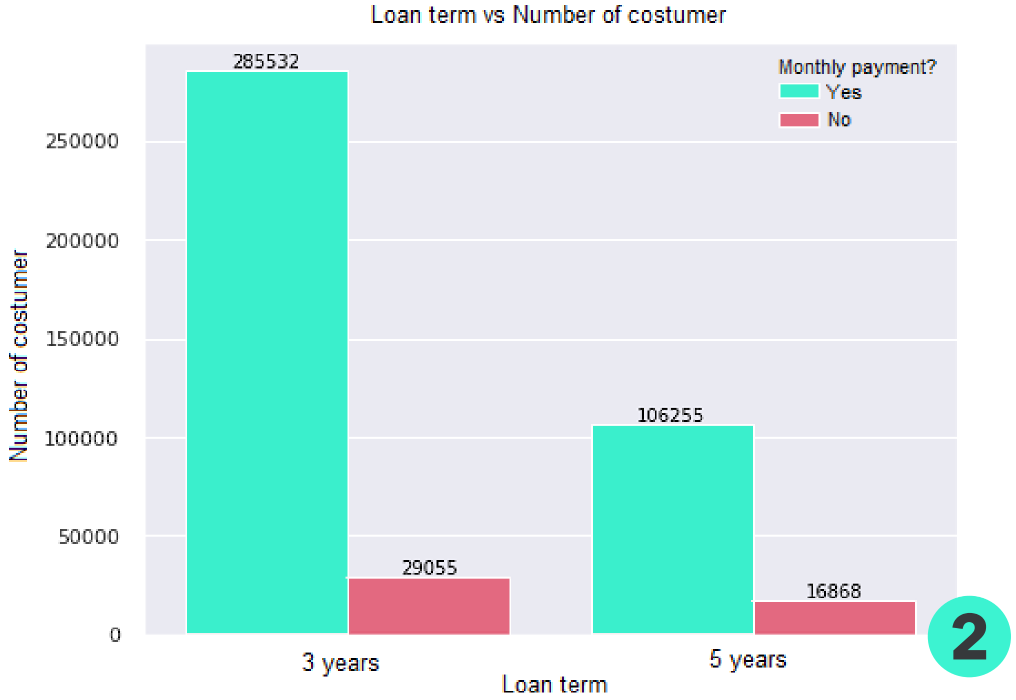
- Loan term (3 year or 5 year)
- Employment length
- Home ownership
- Initial status of loan (whole or fractional)

Numerical

- Loan amount
- Interest rate
- Installment
- Annual income

LOAN TERM



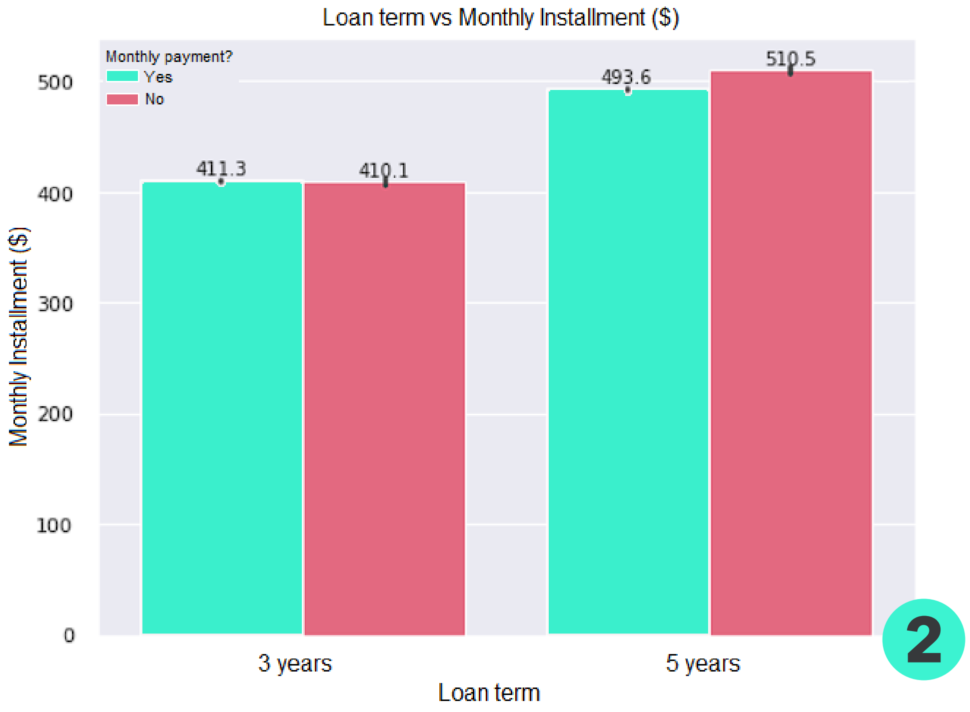


1. Costumer with **longer loan team** will have a **higher percentage** of loan defaults.

2. More costumers have a **3 years** loan term.

LOAN TERM

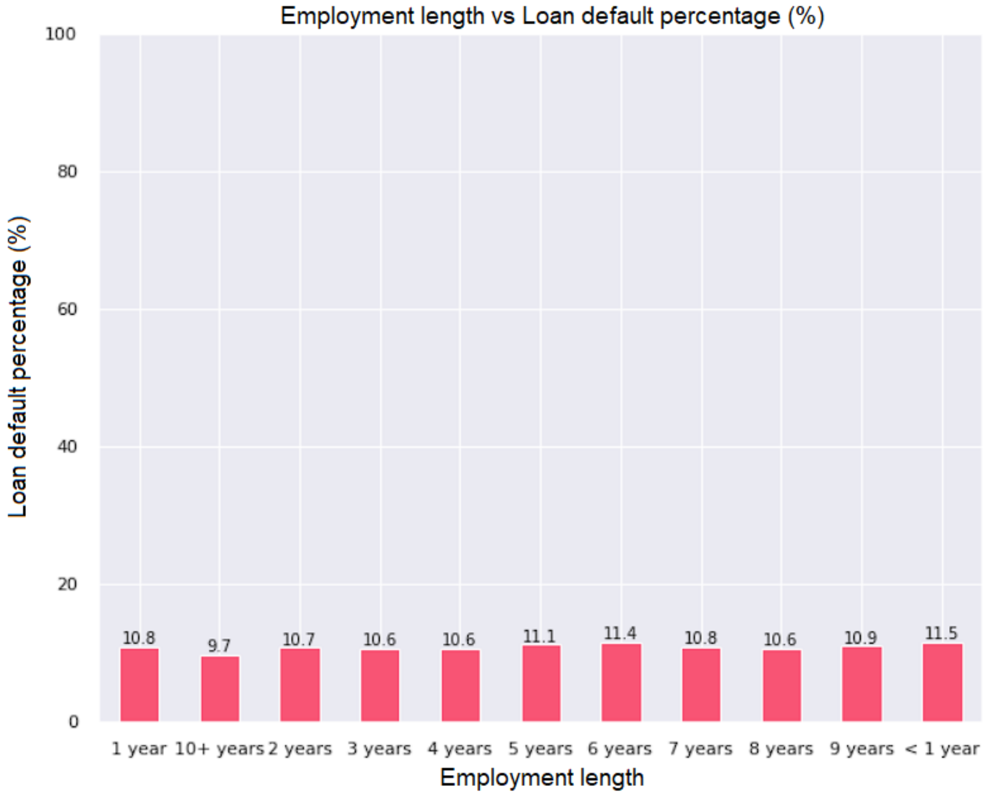
Customers who **fail** to make loan repayments have a **lower average salary**.



However it has almost the **same monthly payment** fees.

Customers with **5 years loan term** have a average **higher salary**, but also have **higher monthly installment.**

EMPLOYMENT LENGTH



There is **no significant difference** in the **percentage of default on debt** payments from how **long the customer has been working**

DATA PREPROCESSING

1. Feature Selection
2. Rank Encoding
One Hot Encoding
3. Train Test Split
4. Modeling
5. Evaluation

1 FEATURE SELECTION

Using Correlation Matrix

```python
plt.figure(figsize=(30,30))
sns.heatmap(df_copy.corr(), annot = True,
            fmt='.2g', cmap='magma')
plt.title('Correlation Matrix',fontsize = 12)
```

Remove feature with high multicollinearity (threshold value **>0.7**)

```python
df_copy.drop(['loan_amnt','funded_amnt','funded_amnt_inv','out_prncp_inv',
              'total_pymnt','total_pymnt_inv','collection_recovery_fee'], axis = 'columns', inplace = True)
```

2 RANK & ONE HOT ENCODING

Rank encoding

```
#rank encoding
df_copy['term'] =df_copy['term'].replace([' 36 months',' 60 months'],[36,60])
df_copy['grade'] =df_copy['grade'].replace(['A','B','C','D','E','F','G'],[1,2,3,4,5,6,7])
df_copy['emp_length'] =df_copy['emp_length'].replace(['< 1 year','1 year','2 years','3 years','4 years','5 years',
                                                     '6 years','7 years','8 years','9 years','10 years','10+ years']
                                                    ,[0,1,2,3,4,5,6,7,8,9,10,11])
df_copy['verification_status'] =df_copy['verification_status'].replace(['Verified','Not Verified'],[1,0])
df_copy['initial_list_status'] =df_copy['initial_list_status'].replace(['f','w'],[1,0])
```

One hot encoding

```
#one hot encoding
df_copy= pd.get_dummies(df_copy,columns=['home_ownership','purpose'])
```

3 TRAIN TEST SPLIT

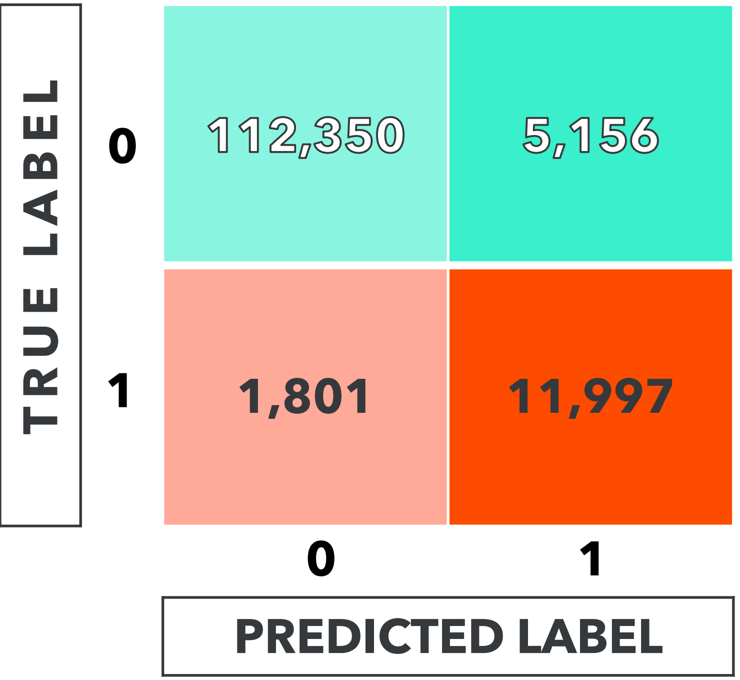
```
X = df_copy.drop('target',axis=1).copy()
y = df_copy['target'].copy()

X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.3,random_state=42)
```

2 MODELING

Logistic Regression

```
logreg_model = LogisticRegression(random_state=42,class_weight='balanced')
eval_train(X_train,y_train,logreg_model)
```



Recall	: 86.9%
Precision	: 69.9%
F1 Score	: 77.5%
ROC-AUC	: 96.2%

Recall is a measure of how many of the **positive correctly predicted**, **over all the positive cases** in the data.

Note : **1** = Costumer **can't** pay loan
0 = Costumer **can** pay loan

2 MODELING

Logistic Regression

```
logreg_model = LogisticRegression(random_state=42,class_weight='balanced')
eval_train(X_train,y_train,logreg_model)
```

TRUE LABEL \ PREDICTED LABEL	0	1
0	112,350	5,156
1	1,801	11,997

Recall : 86.9%
Precision : 69.9%
F1 Score : 77.5%
ROC-AUC : 96.2%

Precision is a measure of how many of the **positive predictions made are correct** (true positives).

Note : **1** = Costumer **can't** pay loan
0 = Costumer **can** pay loan

2 MODELING

Logistic Regression

```
logreg_model = LogisticRegression(random_state=42,class_weight='balanced')
eval_train(X_train,y_train,logreg_model)
```

TRUE LABEL		PREDICTED LABEL 0	PREDICTED LABEL 1
	0	112,350	5,156
	1	1,801	11,997

Recall : 86.9%
Precision : 69.9%
F1 Score : 77.5%
ROC-AUC : 96.2%

F1-Score is a measure **combining** both **precision and recall**.

Note : **1** = Costumer **can't** pay loan
0 = Costumer **can** pay loan

2 MODELING

Naive Bayes

```python
logreg_model = LogisticRegression(random_state=42,class_weight='balanced')
eval_train(X_train,y_train,logreg_model)
```

TRUE LABEL \ PREDICTED LABEL	0	1
0	116,457	1,058
1	6,339	7,459

Recall : 54.0%
Precision : 87.6%
F1 Score : 66.9%
ROC-AUC : 91.6%

Decision Tree

```python
logreg_model = LogisticRegression(random_state=42,class_weight='balanced')
eval_train(X_train,y_train,logreg_model)
```

TRUE LABEL \ PREDICTED LABEL	0	1
0	115,550	1,965
1	1,665	12,133

Recall : 87.9%
Precision : 86.0%
F1 Score : 87.0%
ROC-AUC : 93.1%

Note : **1** = Costumer **can't** pay loan
0 = Costumer **can** pay loan

2 MODELING

Random Forest

```
logreg_model = LogisticRegression(random_state=42,class_weight='balanced')
eval_train(X_train,y_train,logreg_model)
```

TRUE LABEL	PREDICTED LABEL 0	PREDICTED LABEL 1
0	117,290	225
1	2,030	11,718

Recall : 85.0%
Precision : 98.1%
F1 Score : 91.0%
ROC-AUC : 97.4%

Light GBM

```
logreg_model = LogisticRegression(random_state=42,class_weight='balanced')
eval_train(X_train,y_train,logreg_model)
```

TRUE LABEL	PREDICTED LABEL 0	PREDICTED LABEL 1
0	117,416	99
1	1,885	11,913

Recall : 86.3%
Precision : 99.2%
F1 Score : 92.3%
ROC-AUC : 98.1%

Note : **1** = Costumer **can't** pay loan
0 = Costumer **can** pay loan

LIGHT GBM

TRUE LABEL \ PREDICTED LABEL	0	1
0	117416	99
1	1885	11913

Note : **1** = Costumer **can't** pay loan
0 = Costumer **can** pay loan

Recall	: 86,3%
Precision	: 99,2%
F1 Score	: 92,3%
ROC-AUC	: 98,1%

This model is able to catch **11,913** customers who failed to pay off loans out of a total of **13,798** customers.

In other words, the model is capable of lowering the level of the percentage of customers who are unable to pay off loans from **10.8% to 1.6%.**

CUT LOSS CALCULATION

Based on test data

Simulation of using the model against test data to **calculate the reduction of bank losses** from customers who **can't repay loans**.

	Actual Data	Model Result
Total costumers	131,313	119,301
Customers **can** repay loans	117,515	117,416
Customers **can't** repay loans	13,798	1.885

```
#Memfilter nasabah yang gagal membayar
Actual_loss=X_model[(X_model['target_actual']==1)]
#Memfilter nasabah yang gagal membayar pinjaman namun diprediksi dapat membayar pinjaman
Prediction_loss=X_model[(X_model['target_prediction']==0)&(X_model['target_actual']==1)]
```

CUT LOSS CALCULATION

Based on test data

	Actual Data	Model Result
Customers **can't** repay loans	13.798	1.885
The total loan **granted**	$ 205,032,525	$ 29,438,875
Total loan already **paid**	$ 98,213,658	$ 17,796,499
Losses from customers	$ -106,818,866	$ -11,642,375

```
#Perhitungan kerugian dengan mengurangi pembayaran yang sudah dilakukan dengan dana pinjaman yang sudah diberikan
Prediction_loss['loss']=Prediction_loss['total_pymnt']-Prediction_loss['funded_amnt']
Actual_loss['loss']=Actual_loss['total_pymnt']-Actual_loss['funded_amnt']

#Dana yang sudah dibayarkan
round(Actual_loss['total_pymnt'].sum(),1)
#Dana yang sudah diberikan
round(Actual_loss['funded_amnt'].sum(),1)
#Kerugian dari nasabah
round(Actual_loss['loss'].sum(),1)

#Dana yang sudah dibayarkan
round(Prediction_loss['total_pymnt'].sum(),1)
#Dana yang sudah diberikan
round(Prediction_loss['funded_amnt'].sum(),1)
#Kerugian dari nasabah
round(Prediction_loss['loss'].sum(),1)
```

CUT LOSS CALCULATION

Based on test data

	Actual Data	Model Result
Customers **can't** repay loans	13.798	1.885
The total loan **granted**	$ 205,032,525	$ 29,438,875
Total loan already **paid**	$ 98,213,658	$ 17,796,499
Losses from customers	$ -106,818,866	$ -11,642,375

By applying the model, it can **reduce losses** from customers by up to **89.1%**

CONCLUSION

PREDIKSI

•The model is capable of **predicting 86.3%** of customers who **fail to make payments** of loans. With potential **reduction loss** up to **89.1%**.

REKOMENDASI

•Provides a **duration of loan repayment** of **more than 5 years**. In order to **reduce** the cost of **monthly payments** .