Rakamin Academy X id/x partners

Prediction Model

ID/X Partners - Data Scientist

Presented by
Reynald Aryansyah

Yogyakarta

reynalaryansyah22@gmail.com

Reynald Aryansyah

Rakamin Academy

Reynald Aryansyah

Saya adalah seorang profesional di bidang **Data Science** dengan latar belakang pendidikan di bidang **Informatika**. Memiliki keahlian dalam **analisis data**, **pembuatan model machine learning**, serta **pemrosesan data** untuk mendukung pengambilan keputusan berbasis data

***notes: just a sample page**

About Company

ID/X Partners (PT IDX Consulting) didirikan pada tahun 2002 dan telah melayani perusahaan di seluruh wilayah Asia dan Australia dan di berbagai industri, khususnya layanan keuangan, telekomunikasi, manufaktur, dan ritel. ID/X Partners menyediakan layanan konsultasi yang mengkhususkan diri dalam memanfaatkan solusi data analytic and decisioning (DAD) yang dipadukan dengan manajemen risiko dan disiplin pemasaran terintegrasi untuk membantu klien mengoptimalkan profitabilitas portofolio dan proses bisnis. Layanan konsultasi dan solusi teknologi yang komprehensif yang ditawarkan oleh mitra id/x menjadikannya sebagai one-stop service provider

Project Portfolio

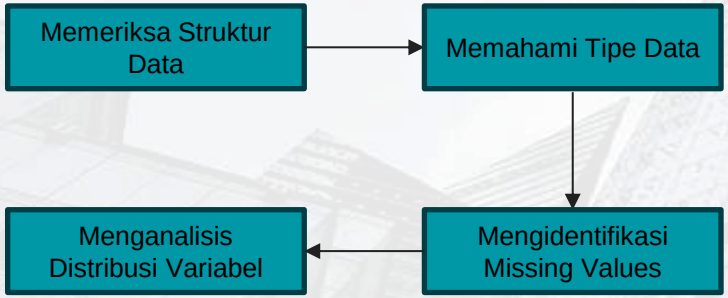
Latar Belakang: Proyek ini bertujuan untuk membangun model **machine learning** yang dapat memprediksi **risiko kredit** atau **loan default** pada pinjaman yang diberikan. Dengan data yang tersedia, model ini diharapkan dapat membantu perusahaan dalam membuat keputusan lebih cepat dan tepat terkait peminjaman kredit, khususnya dalam mengidentifikasi peminjam yang berisiko gagal bayar. Proyek ini sangat relevan mengingat pentingnya mengelola risiko dalam industri perbankan dan keuangan untuk mengurangi potensi kerugian.

Link code
https://drive.google.com/file/d/1hMM1yX3kHWMgXHnNVGQmDMJdd_npGE9M/view?usp=drive_link!

Project explanation video
https://drive.google.com/file/d/1BpBZmua4H-OCP9Sy94trA4beSWxLh8rN/view?usp=drive_link!

1. Data Understanding

Data Understanding adalah tahap awal dalam proyek data science yang bertujuan untuk memahami struktur, karakteristik, dan kualitas data yang akan digunakan. Pada tahap ini, Saya melakukan beberapa langkah penting seperti:

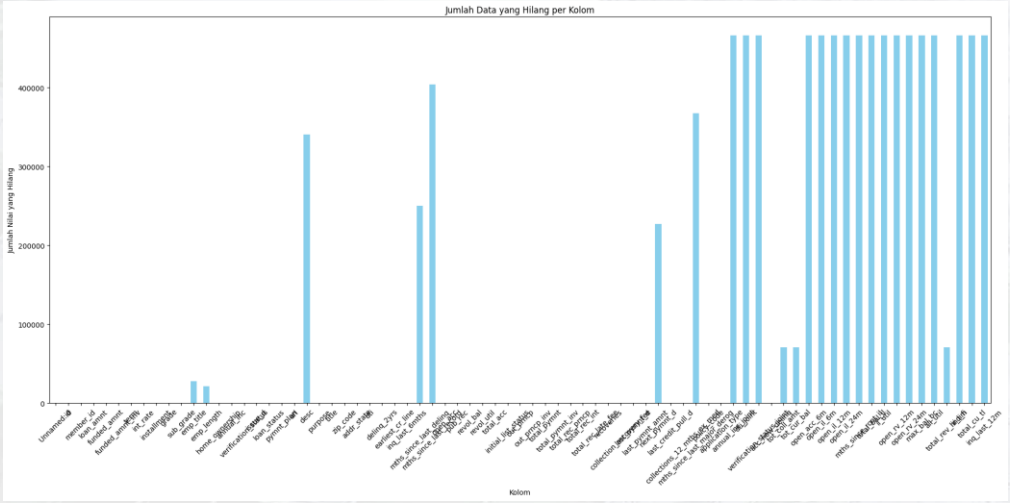


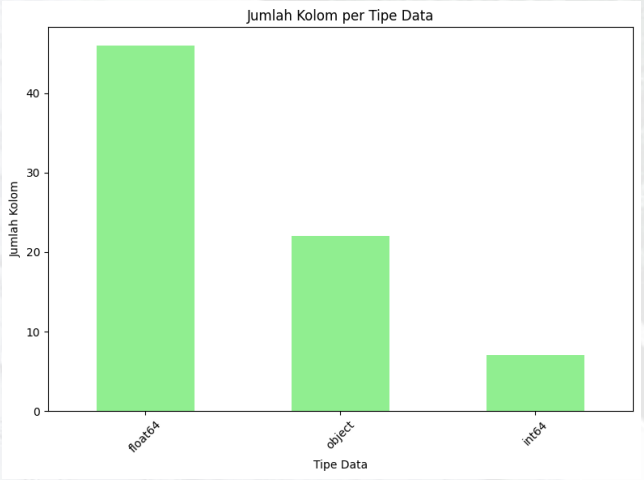
1. Data Understanding

```
df.shape
✓ 0.0s
(466285, 75)
```

```
total_null = df.isnull().sum()
total_null.sum()
✓ 0.4s
9776227
```

- Pada dataset terdiri dari 466285 Baris dan 75 Kolom
- Jumlah total nilai yang hilang dalam dataset adalah 9.776.227
- Dataset berisi 46 kolom bertipe float, 22 kolom bertipe object, 6 kolom bertipe int





2. Feature Engineering

Pada tahap **Feature Engineering**, saya melakukan beberapa langkah penting untuk menyiapkan data agar dapat digunakan dalam modeling. Proses ini mencakup konversi data kategorikal menjadi numerik, pembuatan fitur baru, dan standarisasi data numerik. Berikut adalah langkah-langkah utama yang diambil:

```
df['term'] = df['term'].str.extract('(\d+)').astype(float)

# Memilih hanya kolom numerik
df_numeric = df.select_dtypes(include=['number'])
```

✓ 0.8s

```
# Konversi kolom kategorikal penting ke numerik dengan One-Hot Encoding
categorical_features = ['purpose', 'home_ownership', 'verification_status']
df = pd.get_dummies(df, columns=categorical_features, drop_first=True)
```

✓ 0.3s

1. Mengubah Kolom term

Kolom term berisi teks seperti 36 months dan 60 months, yang diubah menjadi angka.

2. One-Hot Encoding untuk Kolom Kategorikal

Kolom kategorikal seperti grade, purpose, home_ownership diubah menjadi variabel numerik menggunakan **One-Hot Encoding**.

2. Feature Engineering

```
import numpy as np

# Konversi emp_length ke angka
df['emp_length'] = df['emp_length'].replace({'< 1 year': 0, '10+ years': 10})
df['emp_length'] = df['emp_length'].str.extract('(\d+)').astype(float)

# Isi missing values dengan median
df['emp_length'].fillna(df['emp_length'].median(), inplace=True)
```

✓ 0.5s

```
from sklearn.preprocessing import StandardScaler

# Menstandarisasi fitur numerik
scaler = StandardScaler()
df[['loan_amnt', 'funded_amnt', 'int_rate', 'installment', 'dti']] = scaler.fit_transform(
    df[['loan_amnt', 'funded_amnt', 'int_rate', 'installment', 'dti']]
)

# Menampilkan beberapa baris setelah scaling
df[['loan_amnt', 'funded_amnt', 'int_rate', 'installment', 'dti']].head()
```

✓ 0.1s

```
# Konversi loan_status menjadi binary (1 = Charged Off, 0 = Fully Paid)
df['loan_status'] = df['loan_status'].apply(lambda x: 1 if x == "Charged Off" else 0)
```

✓ 0.1s

```
# Membuat fitur baru: funding_ratio dan loan_to_income_ratio
df['funding_ratio'] = df['funded_amnt'] / df['loan_amnt']
df['loan_to_income_ratio'] = df['loan_amnt'] / df['annual_inc']

# Menampilkan beberapa baris untuk memastikan fitur baru
df[['funding_ratio', 'loan_to_income_ratio']].head()
```

✓ 0.0s

3. Mengonversi emp_length menjadi Angka
4. Mengonversi Kolom loan_status Menjadi Biner
5. Membuat Fitur Baru
6. Standarisasi Fitur Numerik

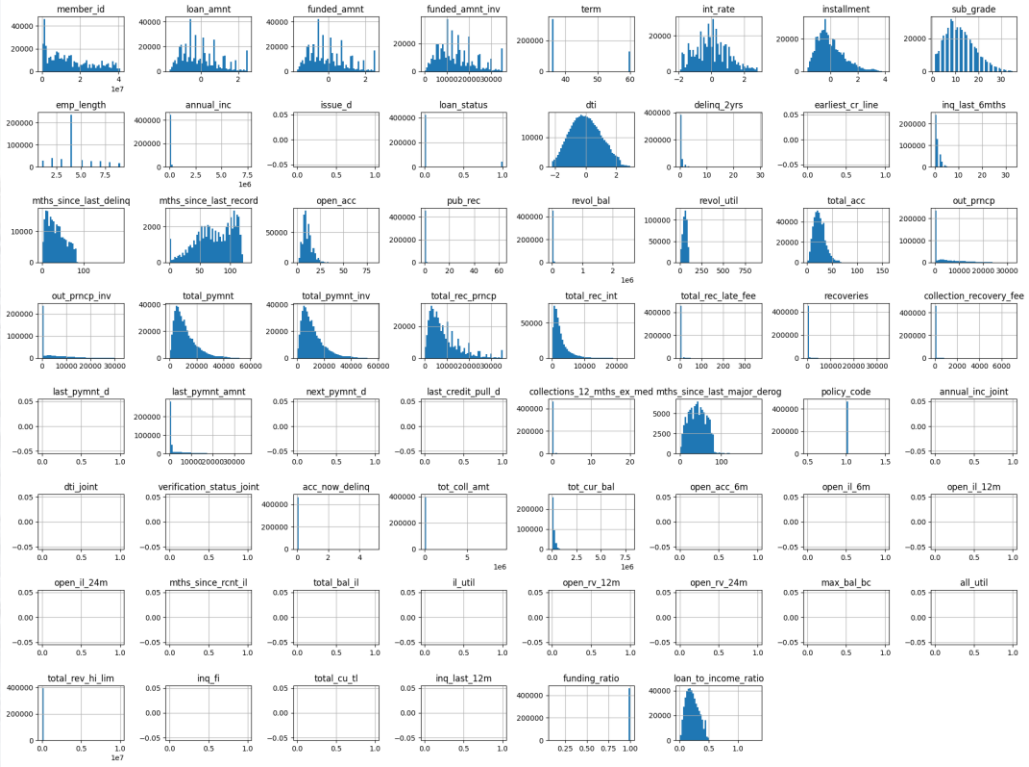
3. Exploratory Data Analysis

Distribusi Tidak Normal: Beberapa fitur seperti **loan_amnt**, **annual_inc**, dan **funding_ratio** menunjukkan distribusi yang **skewed** ke kanan dengan banyak nilai rendah dan beberapa nilai ekstrem.

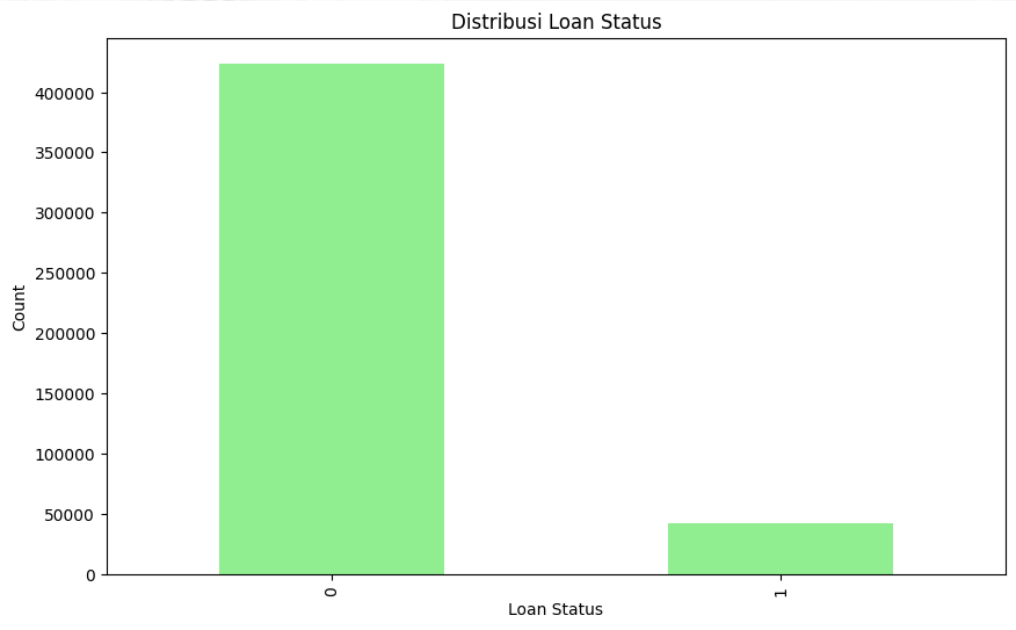
Korelasi Fitur: Terdapat **korelasi positif** antara beberapa fitur, seperti **loan_amnt** dan **funded_amnt**. Beberapa fitur juga menunjukkan hubungan antara **loan_status** dan karakteristik peminjam.

Outliers: Beberapa fitur mengandung nilai ekstrim (outliers), yang dapat mempengaruhi model.

Data Sparsity: Beberapa kolom menunjukkan **nilai nol yang dominan**, yang mengindikasikan adanya banyak data kosong atau tidak relevan.



3. Exploratory Data Analysis



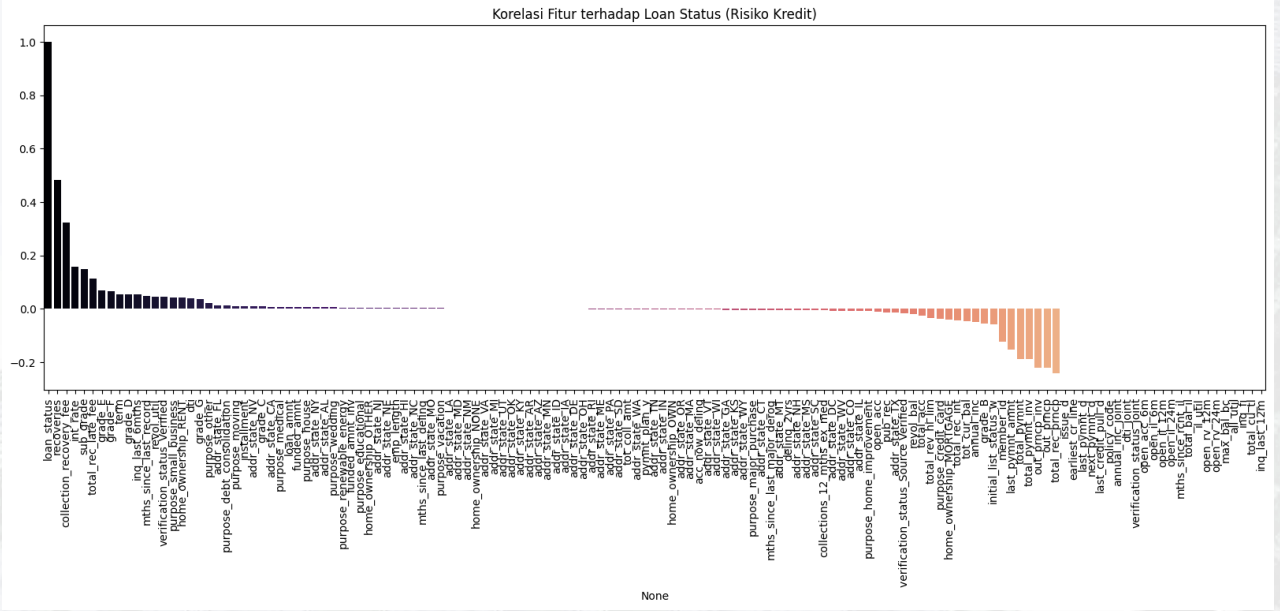
0 (Fully Paid): Pinjaman yang sudah dibayar penuh, yang mendominasi dataset dengan jumlah lebih dari 400.000.

1 (Charged Off): Pinjaman yang gagal bayar, yang jumlahnya sangat sedikit.

3. Exploratory Data Analysis

```
10 Fitur dengan Korelasi Tertinggi terhadap Risiko Kredit:

loan_status                 1.000000
recoveries                  0.482238
collection_recovery_fee     0.323009
int_rate                    0.157621
sub_grade                   0.148525
total_rec_late_fee          0.112825
grade_E                     0.068704
grade_F                     0.066436
term                        0.055570
grade_D                     0.055115
```



4. Data Preparation

Langkah pertama adalah memeriksa jumlah **missing values** di setiap kolom untuk mengidentifikasi data yang hilang.

Kolom dengan **lebih dari 30% missing values** dihapus dari dataset karena dianggap tidak memberikan informasi yang cukup dan dapat mengganggu analisis.

Kolom **numerik** yang memiliki missing values diisi dengan **median** untuk menjaga distribusi data, sedangkan kolom **kategorikal** diisi dengan **modus** (nilai yang paling sering muncul).

```python
# Hapus kolom dengan missing values lebih dari 30%
threshold = 30
cols_to_drop = missing_df[missing_df['Percentage'] > threshold].index
df.drop(cols_to_drop, axis=1, inplace=True)
```

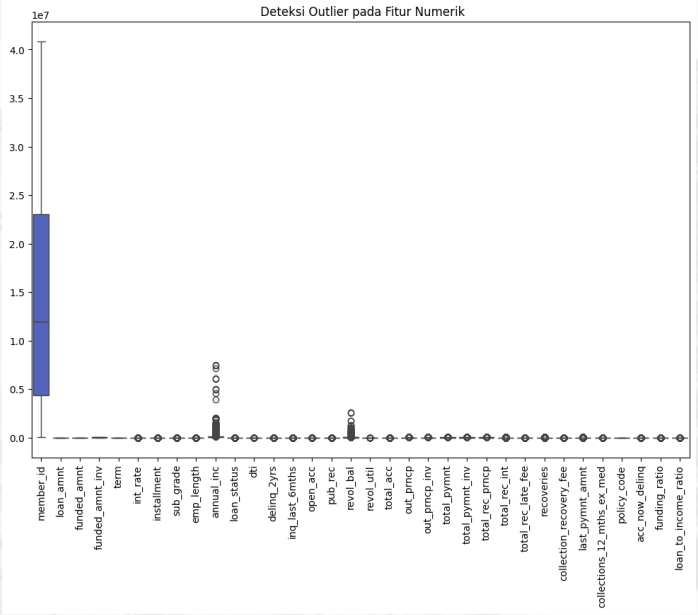
```python
# Isi missing values dengan median (untuk data numerik)
num_cols = ['int_rate', 'dti']
for col in num_cols:
    df[col].fillna(df[col].median(), inplace=True)
```

4. Data Preparation

Outliers atau nilai yang sangat ekstrem dapat mengganggu analisis data dan model. Oleh karena itu, langkah penting adalah mendeteksi dan menangani outliers.

Teknik yang digunakan untuk mendeteksi outliers adalah dengan **Z-Score** atau menggunakan **boxplot**.

Outliers yang terdeteksi dapat dihapus atau diganti dengan nilai yang lebih representatif, seperti median.




```
# Menggunakan capping untuk outlier (mengatur nilai maksimum atau minimum)
for col in numerical_columns:
    q1 = df[col].quantile(0.25)
    q3 = df[col].quantile(0.75)
    iqr = q3 - q1
    lower_limit = q1 - 1.5 * iqr
    upper_limit = q3 + 1.5 * iqr

    # Capping outlier
    df[col] = np.where(df[col] < lower_limit, lower_limit, df[col])
    df[col] = np.where(df[col] > upper_limit, upper_limit, df[col])
```

4. Data Preparation

Data split adalah langkah penting untuk memisahkan data menjadi dua bagian: satu untuk pelatihan (training) dan satu untuk pengujian (testing).

Data dibagi menggunakan **80% untuk training** dan **20% untuk testing**. Teknik **stratified sampling** digunakan untuk memastikan proporsi kelas **loan_status** tetap terjaga di kedua bagian.

```python
from sklearn.model_selection import train_test_split

# Fitur (X) dan target (y)
X = df.drop(columns=['loan_status'])
y = df['loan_status']

# Membagi data menjadi training dan testing (80%-20% split)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Memeriksa ukuran dataset training dan testing
print("Training set size:", X_train.shape)
print("Test set size:", X_test.shape)
```

```
Training set size: (373028, 110)
Test set size: (93257, 110)
```

5. Data Modeling

Memilih model machine learning yang tepat berdasarkan masalah yang dihadapi. Dalam hal ini, saya sedang menangani masalah **klasifikasi biner** (apakah pinjaman gagal bayar atau tidak) dengan target variabel **loan_status**.

Maka model yang saya gunakan adalah Logistic Regresion dan Random Forest.

```python
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, accuracy_score, confusion_matrix

# Membuat model Logistic Regression
logreg_model = LogisticRegression(random_state=42)

# Melatih model dengan data pelatihan
logreg_model.fit(X_train_res, y_train_res)

# Membuat prediksi dengan data pengujian
y_pred_logreg = logreg_model.predict(X_test)

# Evaluasi Model
print("Akurasi Model Logistic Regression: ", accuracy_score(y_test, y_pred_logreg))
print("\nConfusion Matrix (Logistic Regression):\n", confusion_matrix(y_test, y_pred_logreg))
print("\nClassification Report (Logistic Regression):\n", classification_report(y_test, y_pred_logreg))
```

✓ 31.6s

```
Confusion Matrix (Logistic Regression):
 [[84601   161]
 [  113  8382]]

Classification Report (Logistic Regression):
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     84762
           1       0.98      0.99      0.98      8495

    accuracy                           1.00     93257
   macro avg       0.99      0.99      0.99     93257
weighted avg       1.00      1.00      1.00     93257
```

5. Data Modeling

```python
from sklearn.ensemble import RandomForestClassifier

# Membuat model Random Forest
rf_model = RandomForestClassifier(random_state=42)

# Melatih model dengan data pelatihan
rf_model.fit(X_train_res, y_train_res)

# Membuat prediksi dengan data pengujian
y_pred_rf = rf_model.predict(X_test)

# Evaluasi Model
print("Akurasi Model Random Forest: ", accuracy_score(y_test, y_pred_rf))
print("\nConfusion Matrix (Random Forest):\n", confusion_matrix(y_test, y_pred_rf))
print("\nClassification Report (Random Forest):\n", classification_report(y_test, y_pred_rf))
```

✓ 10m 10.0s

```
Akurasi Model Random Forest:  0.9975765894249226

Confusion Matrix (Random Forest):
 [[84676    86]
 [  140  8355]]

Classification Report (Random Forest):
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     84762
           1       0.99      0.98      0.99      8495

    accuracy                           1.00     93257
   macro avg       0.99      0.99      0.99     93257
weighted avg       1.00      1.00      1.00     93257
```

5. Data Modeling

Logistic Regression:

Akurasi: 99.57% – Model berhasil memprediksi dengan sangat baik, meskipun ada sedikit penurunan akurasi antara data latih dan uji.\

Precision untuk kelas 0 (Fully Paid): 1.00 – Model sangat tepat dalam memprediksi pinjaman yang dilunasi.

Recall untuk kelas 1 (Charged Off): 0.97 – Meskipun recall sudah sangat baik, ada sedikit kesalahan dalam memprediksi gagal bayar.

F1-Score: 0.98 untuk kelas 1, menunjukkan keseimbangan yang baik antara precision dan recall untuk pinjaman yang gagal bayar.

ROC-AUC: 0.98 – Model sangat baik dalam memisahkan pinjaman yang dilunasi dan yang gagal bayar.

5. Data Modeling

Random Forest:

Akurasi: 99.77% – Model lebih akurat pada data uji dibandingkan Logistic Regression, tetapi ada overfitting kecil (100% pada data latih).

Precision untuk kelas 1 (Charged Off): 1.00 – Model sangat tepat dalam memprediksi pinjaman gagal bayar.

Recall untuk kelas 1 (Charged Off): 0.98 – Model juga berhasil menangkap sebagian besar pinjaman gagal bayar, meskipun recall untuk kelas 1 sedikit lebih rendah dibandingkan dengan Logistic Regression.

F1-Score: 0.99 – Skor F1 menunjukkan keseimbangan yang sangat baik antara precision dan recall.

ROC-AUC: 0.99 – Random Forest memiliki kemampuan luar biasa dalam memisahkan gagal bayar dan lunas.

6. Evaluation

Pada proyek ini, tujuan utama adalah membangun model **Machine Learning** untuk memprediksi **risiko kredit** pada pinjaman yang diberikan. Data yang digunakan mencakup informasi terkait **jumlah pinjaman**, **status pinjaman** (apakah berhasil dibayar atau tidak), dan berbagai fitur terkait peminjam. Setelah mempersiapkan dan membersihkan data melalui **data preparation** dan **feature engineering**, kami melatih model untuk mengidentifikasi potensi **risiko gagal bayar**.

7. Conclusion

Model ini telah menunjukkan kemajuan yang signifikan dalam memprediksi **risiko kredit**. Dengan evaluasi yang cermat dan penyesuaian lebih lanjut pada **imbalance kelas**, perusahaan dapat memanfaatkan model ini untuk mengambil keputusan pemberian kredit yang lebih terinformasi dan proaktif. Kami menyarankan perusahaan untuk terus mengembangkan model ini dan terus memperbarui dengan data terbaru serta mempertimbangkan model tambahan untuk meningkatkan kinerja lebih lanjut.

Thank You

Rakamin Academy X id/x partners