Rakamin Academy X id/x partners

Virtual Internship Experience

Data Scientist

Presented by
Yoga Aria Sena

Yoga Aria Sena

About Me

I am an enthusiastic individual deeply intrigued by the world of data science. My journey is fueled by a profound interest in unraveling insights from data and transforming them into meaningful solutions. Aspiring to become a skilled data scientist, I am dedicated to expanding my expertise in programming, statistical analysis, and machine learning algorithms. My aspiration is to harness the power of data to create innovative solutions that make a tangible impact on the world.

Rakamin Academy

My Experience

- Project Based Virtual Internship: Data Scientist Kalbe Nutritionals x Rakamin Academy
  (August 2023 – September 2023)

- Project Based Virtual Internship: Data Scientist ID/X Partners x Rakamin Academy
  (September 2023 – Present)

About Rakamin Academy

Rakamin Academy adalah Platform akselerasi karir, tempat bagi siapapun untuk belajar dan membangun karir di bidang digital dan teknologi.

Rakamin Academy menyediakan layanan Pendidikan yang paling terjangkau (efisien secara model bisnis), berdampak (dihubungkan langsung dengan ahli, praktik standar industri), dan bermakna (mengajarkan soft skills, membantu pengembangan karir, membangun komunitas).

About ID/X Partners

id/x partners didirikan pada tahun 2002 oleh mantan banker dan konsultan manajemen yang memiliki pengalaman luas dalam siklus kredit dan manajemen proses, pengembangan skor, dan manajemen kinerja. Pengalaman gabungan kami telah melayani perusahaan di berbagai wilayah Asia dan Australia, serta dalam berbagai industri, khususnya layanan keuangan, telekomunikasi, manufaktur, dan ritel.

id/x partners menyediakan layanan konsultasi yang mengkhususkan diri dalam penggunaan solusi analitik data dan pengambilan keputusan (DAD) yang dikombinasikan dengan disiplin manajemen risiko terintegrasi dan pemasaran untuk membantu klien mengoptimalkan profitabilitas portofolio dan proses bisnis.

Layanan konsultasi komprehensif dan solusi teknologi yang ditawarkan oleh id/x partners menjadikannya sebagai penyedia layanan satu atap.

Case Study

- **Latar Belakang**

Sebagai bagian dari tugas akhir kontrak sebagai intern Data Scientist di ID/X Partners, proyek ini melibatkan kerjasama dengan Lending Company.

- **Alat**

Python, Jupyter Notebook

- **Tujuan:**

Membangun model prediksi risiko kredit menggunakan dataset berisi data peminjaman yang diterima dan ditolak.

Menyajikan solusi secara visual kepada klien.

Menyediakan solusi teknologi yang dapat membantu perusahaan mengoptimalkan portofolio dan proses bisnis mereka.

Dataset dan Tujuan

Dataset dan Tujuan

	id	member_id	loan_amnt	funded_amnt	funded_amnt_inv	term	int_rate
0	1077501	1296599	5000	5000	4975.0	36 months	10.65
1	1077430	1314167	2500	2500	2500.0	60 months	15.27
2	1077175	1313524	2400	2400	2400.0	36 months	15.96
3	1076863	1277178	10000	10000	10000.0	36 months	13.49
4	1075358	1311748	3000	3000	3000.0	60 months	12.69

5 rows x 74 columns

- **Dataset**

Terdiri dari data status peminjaman yang mencakup berbagai status, seperti Charged Off, Current, Default, Does not meet the credit policy. Status:Charged Off, Does not meet the credit policy. Status:Fully Paid, Fully Paid, In Grace Period, Late (16-30 hari), dan Late (31-120 hari)

- **Tujuan**

Membangun peta risiko kredit yang akurat dan informatif.
Membantu dalam pengambilan keputusan terkait peminjaman dari setiap risiko kredit.

Data Cleaning

Data Cleaning

```
# Mengecek kolom-kolom duplikat

df.duplicated().sum()
```

```
# Menghapus kolom-kolom yang tidak diperlukan

unnecessary_columns = ['id', 'member_id', 'url', 'pymnt_plan', 'policy_code', 'application_type', 'acc_now_delinq']

df.drop(columns=unnecessary_columns, axis=1, inplace=True)
```

```
# Melakukan perubahaan tipe data untuk kolom-kolom tanggal

df['term_months'] = df['term'].str.split(' ').str[1].astype(int)
df.drop('term', axis=1, inplace=True)

df['earliest_cr_line'] = pd.to_datetime(df['earliest_cr_line'], format='%b-%y', errors='coerce').dt.year

date_columns = ['issue_d', 'last_pymnt_d', 'last_credit_pull_d']
for column in date_columns:
    df[column] = pd.to_datetime(df[column], format='%b-%y', errors='coerce').dt.month
```

Data Cleaning

```python
# Menghapus kolom-kolom dengan missing value lebih dari 20%

columns_to_drop = []

# Loop melalui setiap kolom dalam DataFrame
for column in df.columns:
    total_missing = df[column].isnull().sum()
    percentage_missing = (total_missing / len(df)) * 100
    if percentage_missing > threshold:
        columns_to_drop.append(column)

# Drop kolom-kolom yang memiliki persentase missing value lebih dari 20%
df = df.drop(columns=columns_to_drop)
```

```python
# Imputasi missing values kolom-kolom numerical

numeric_columns = df.select_dtypes(exclude='object').columns

for column in numeric_columns:
    df[column].fillna(df[column].mean(), inplace=True)
```

Data Cleaning

```python
# Imputasi missing values kolom-kolom kategorikal

categorical_columns = df.select_dtypes(include='object').columns

for column in categorical_columns:
    df[column].fillna(df[column].mode()[0], inplace=True)
```

Data:First Five rows

	loan_amnt	funded_amnt	funded_amnt_inv	int_rate	installment	grade	sub_g
0	5000	5000	4975.000000	10.650000	162.870000	B	
1	2500	2500	2500.000000	15.270000	59.830000	C	
2	2400	2400	2400.000000	15.960000	84.330000	C	
3	10000	10000	10000.000000	13.490000	339.310000	C	
4	3000	3000	3000.000000	12.690000	67.790000	B	

Data: Unique Value Counts In Each Column

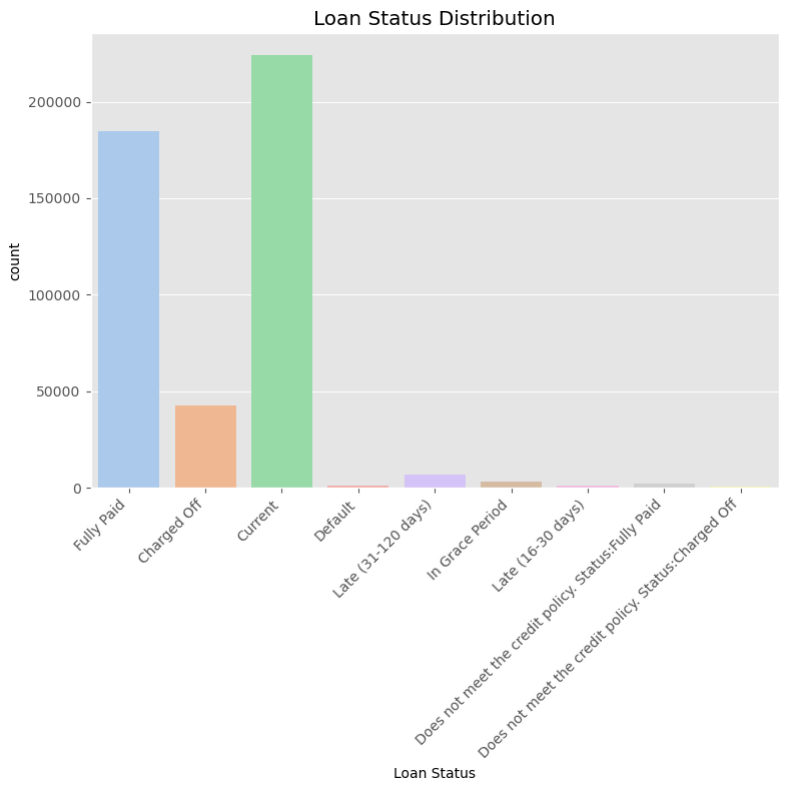
	loan_amnt	funded_amnt	funded_amnt_inv	int_rate	installment	grade	su
Unique Value Count	1352	1354	9854	506	55622	7	

Data:Columns With Nan

	loan_amnt	funded_amnt	funded_amnt_inv	int_rate	installment	grade	sub_gra
0	0	0	0	0	0	0	

Exploratory Data Analysis

Loan Status Distribution



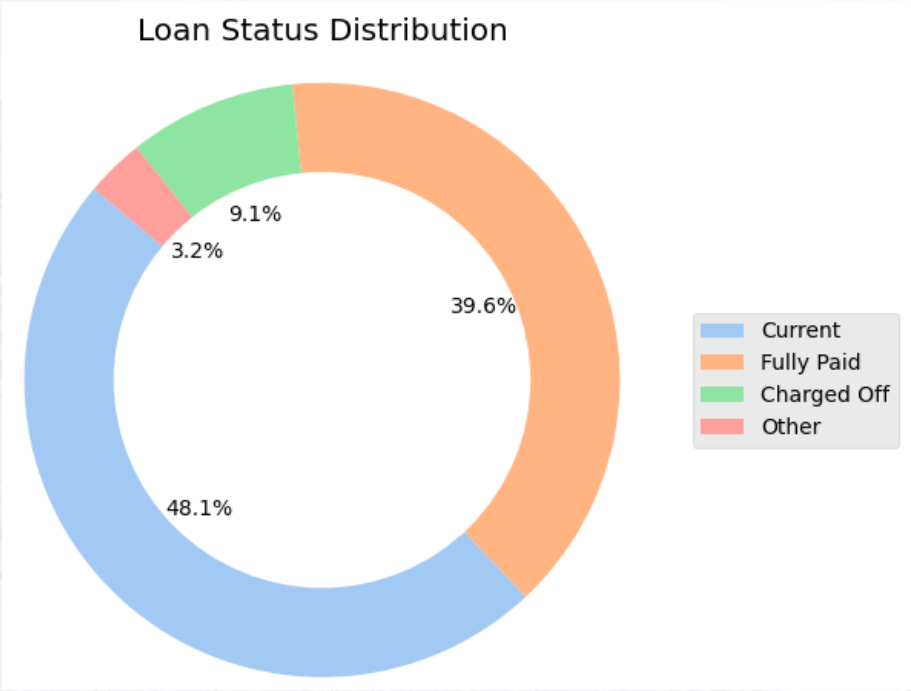
Distribusi Status Pinjaman

- Current sejumlah 224,226
- Fully Paid sejumlah 184,739
- Charged Off sejumlah 42,475
- Late (31-120 days) sejumlah 6,900
- In Grace Period sejumlah 3,146
- Does not meet the credit policy. Status: Fully Paid sejumlah 1,988
- Late (16-30 days) sejumlah 1,218
- Default sejumlah 832
- Does not meet the credit policy. Status: Charged Off sejumlah 761

Low Risk **Medium Risk** **High Risk**

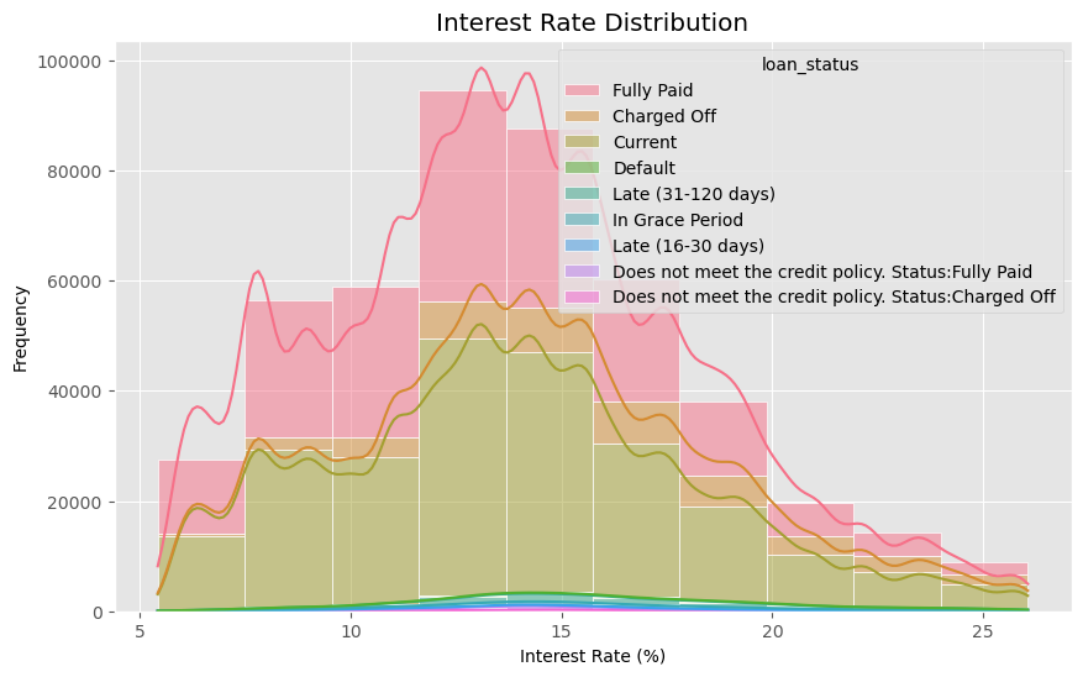
Loan Status Distribution

- Current sebesar 48.1%
- Fully Paid sebesar 39.6%
- Charged Off sebesar 9.1%
- Dan sisanya sebesar 3.2%



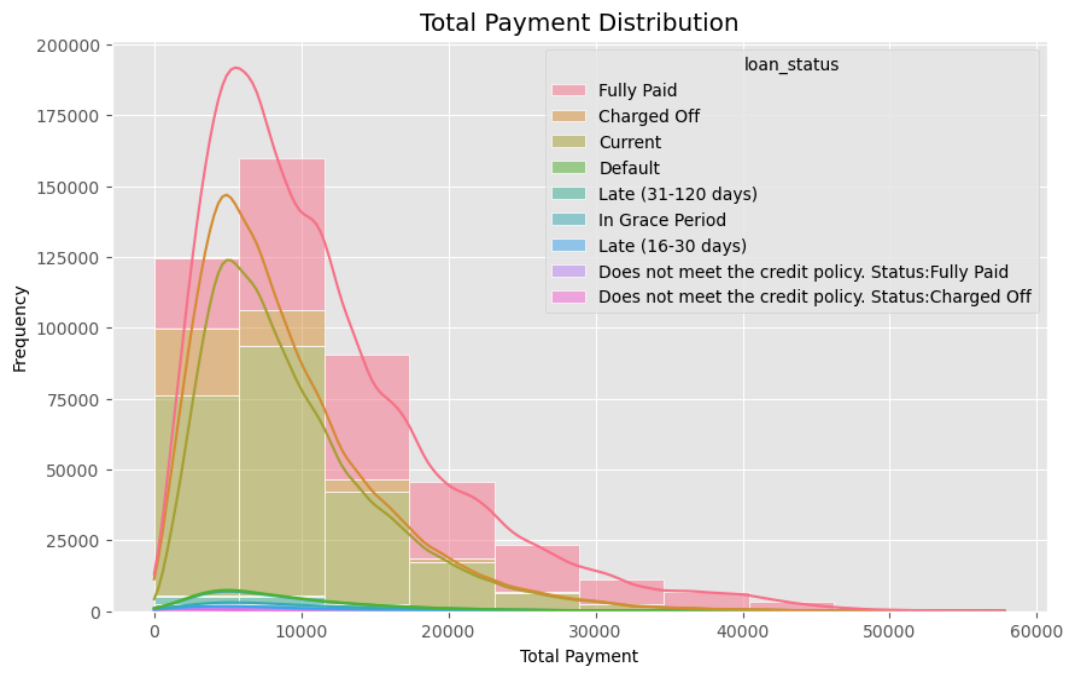
Low Risk **Medium Risk** **High Risk**

Interest Rate Distribution



- Analisis distribusi tingkat bunga pada dataset ini mengindikasikan bahwa sebagian besar tingkat bunga berkisar antara 12% hingga 18%.

Total Payment Distribution



- Analisis distribusi total pembayaran pada dataset ini menggambarkan variasi jumlah pembayaran. Terlihat bahwa puncak distribusi terjadi di sekitar 5,000 USD.

Data Preprocessing

Data Preprocessing

```
1 from sklearn.model_selection import train_test_split
2 from sklearn.preprocessing import StandardScaler, LabelEncoder
3 from sklearn.decomposition import PCA
```

Mengimport preprocessing module yang diperlukan

```
1 X = df.drop('loan_status', axis=1)
2 y = df['loan_status']
```

```
1 X_train, X_test, y_train, y_test = train_test_split(X, y, test_
```

Melakukan train-test split dengan variable target

```
1 mapping = {label: index for index, label in enumerate(label_enc
2
3 print("Mapping hasil Label Encoding:")
4 for label, value in mapping.items():
5     print(f"{label} : {value}")
```

```
Charged Off : 0
Current : 1
Default : 2
Does not meet the credit policy. Status:Charged Off : 3
Does not meet the credit policy. Status:Fully Paid : 4
Fully Paid : 5
In Grace Period : 6
Late (16-30 days) : 7
Late (31-120 days) : 8
```

Data Preprocessing

```python
from sklearn.base import BaseEstimator, TransformerMixin

class MultiColumnLabelEncoder(BaseEstimator, TransformerMixin):
    def __init__(self, columns=None):
        self.columns = columns

    def fit(self, X, y=None):
        return self

    def transform(self, X):
        output = X.copy()
        if self.columns is not None:
            for col in self.columns:
                output[col] = LabelEncoder().fit_transform(outp
        else:
            for colname, col in output.iteritems():
                output[colname] = LabelEncoder().fit_transform(
        return output
```

```python
encoder = MultiColumnLabelEncoder(columns=X.select_dtypes(inclu
X_train = encoder.fit_transform(X_train)
X_test = encoder.transform(X_test)
```

Melakukan enkode label pada semua variabel kategorikal untuk memastikan konsistensi dalam pemrosesan data.

Data Preprocessing

```
X_num_cols = df.select_dtypes(exclude='object').columns
```

```
scaler = StandardScaler()

X_train[X_num_cols] = scaler.fit_transform(X_train[X_num_cols])
X_test[X_num_cols] = scaler.transform(X_test[X_num_cols])
```

Melakukan standarisasi pada variabel-variabel numerik untuk memastikan data memiliki distribusi yang normal.

```
from imblearn.over_sampling import RandomOverSampler

ros = RandomOverSampler(sampling_strategy={8: 10000, 6: 10000,

X_train_oversampled, y_train_oversampled = ros.fit_resample(X_t

print("Distribusi kelas setelah Oversampling:", Counter(y_train
```

Melakukan oversampling untuk kelas-kelas minority pada variable target menjadi 10,000 sampel

Data Preprocessing

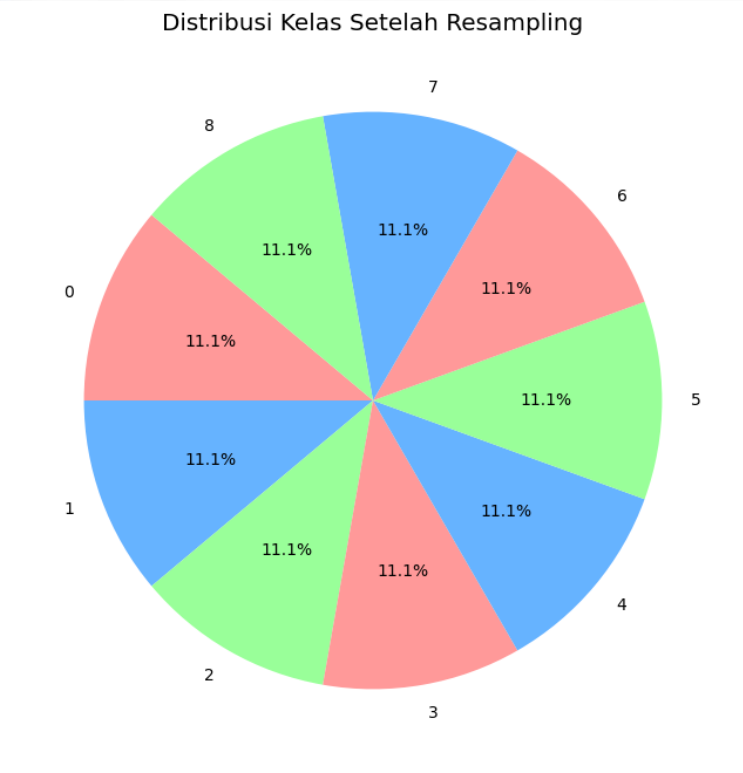
```python
from imblearn.under_sampling import RandomUnderSampler

rus = RandomUnderSampler(sampling_strategy={1: 10000, 5: 10000,

X_train_resampled, y_train_resampled = rus.fit_resample(X_train

print("Distribusi kelas setelah undersampling:", Counter(y_trai
```

Melakukan undersampling untuk kelas-kelas majority pada variable target menjadi 10,000 sampel untuk mengatasi imbalance data.



Model Building and Evaluation

1. XGB Classifier

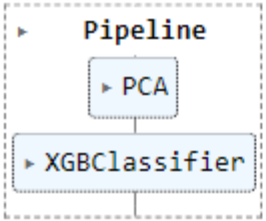
```python
from sklearn.pipeline import make_pipeline
from sklearn.decomposition import PCA
from xgboost import XGBClassifier
```

Import modul yang diperlukan

```python
pipeline = make_pipeline(
    PCA(),
    XGBClassifier(random_state=42, num_class=9)
)
```

Membangun model menggunakan pipeline

```python
pipeline.fit(X_train_resampled, y_train_resampled)
```




```python
y_pred = pipeline.predict(X_test)
```

XGB Classifier Model Evaluation

```
from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred))
```

	precision	recall	f1-score	support
0	0.96	0.97	0.96	8495
1	0.97	0.82	0.89	44845
2	0.25	0.25	0.25	166
3	0.66	0.68	0.67	152
4	0.36	0.92	0.51	398
5	0.99	0.95	0.97	36948
6	0.04	0.34	0.08	629
7	0.02	0.10	0.03	244
8	0.26	0.73	0.38	1380
accuracy			0.88	93257
macro avg	0.50	0.64	0.53	93257
weighted avg	0.95	0.88	0.91	93257

Model memiliki kinerja yang baik dalam mengklasifikasikan kategori risiko rendah (low risk) seperti **"Fully Paid"** dan **"Current"**, dengan presisi dan recall yang tinggi. Kinerja sedang terlihat pada kategori risiko menengah (medium risk) seperti **"Does not meet the credit policy. Status: Fully Paid"** dan **"In Grace Period"**. Model memiliki kinerja yang baik dalam mengklasifikasikan kategori risiko tinggi (high risk) seperti **"Charged Off"**, namun kinerja rendah untuk **"Default"**, **dan "Does not meet the credit policy. Status: Charged Off".** Dan model ini memiliki performa yang buruk Dalam memprediksi status pinjaman yang mengalami keterlambatan.

2. Random Forest Classifier

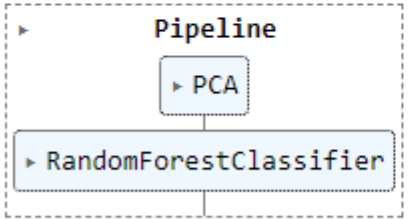
```python
from sklearn.ensemble import RandomForestClassifier
```

Import modul yang diperlukan

```python
pipeline2 = make_pipeline(
    PCA(),
    RandomForestClassifier(random_state=42)
)
```

Membangun model menggunakan pipeline

```python
pipeline2.fit(X_train_resampled, y_train_resampled)
```




```python
y_pred_2 = pipeline2.predict(X_test)
```

Random Forest Classifier Model Evaluation

```
1 from sklearn.metrics import classification_report
2 print(classification_report(y_test, y_pred_2))
```

	precision	recall	f1-score	support
0	0.91	0.92	0.91	8495
1	0.96	0.81	0.87	44845
2	0.00	0.00	0.00	166
3	0.75	0.37	0.49	152
4	0.32	0.82	0.46	398
5	0.94	0.93	0.94	36948
6	0.05	0.11	0.07	629
7	0.00	0.00	0.00	244
8	0.13	0.75	0.22	1380
accuracy			0.86	93257
macro avg	0.45	0.52	0.44	93257
weighted avg	0.92	0.86	0.88	93257

Model ini berhasil dengan baik dalam mengklasifikasikan risiko rendah (low risk) dengan tingkat presisi dan recall yang tinggi pada kategori **'Fully Paid'** dan **'Current'**. Kategori risiko menengah (medium risk) juga berhasil diidentifikasi, terutama pada **'Does not meet the credit policy. Status: Fully Paid'** dan **'In Grace Period'**. Kinerja model terlihat baik dalam mengklasifikasikan risiko tinggi (high risk) seperti **'Charged Off'**. Namun, terdapat penurunan kinerja pada **'Default'** dan **'Does not meet the credit policy. Status: Charged Off'**. Paling menonjol, model mengalami kesulitan memprediksi status pinjaman yang mengalami keterlambatan **(late status)** dengan akurasi yang rendah.

Rakamin Academy

Connect With Me!

GitHub

yogaariasena (Yoga Aria Sena) (github.com)

https://www.linkedin.com/in/yogaariasena


Thank You
Rakamin
Academy
X
id/x partners