credit_scoring

September 25, 2022

1 Menganalisis risiko peminjam gagal membayar

Sebagai kredit analyst proyek kami ialah menyiapkan laporan untuk bank bagian kredit. Kami mencari tahu pengaruh status perkawinan seorang nasabah dan jumlah anak terhadap probabilitas ketepatan waktu dalam melunasi pinjaman. Bank sudah memiliki beberapa data mengenai kelayakan kredit nasabah.

Laporan Anda akan dipertimbangkan pada saat membuat **penilaian kredit** untuk calon nasabah. **Penilaian kredit** digunakan untuk mengevaluasi kemampuan calon peminjam untuk melunasi pinjaman mereka.

Tujuan utama dari poject ini adalah untuk mengetahui kelayakan seorang klien untuk mendapatkan kredit berdasarkan status dan keadaan mereka yang tersimpan dalam data kita. Kita juga menguji kapasitas nasabah berdasarkan karakteristik mereka yang kita rangkum berdasarkan kategori-kategori sehingga diperoleh *pattern* untuk memberikan lampu kuning kepada nasabah yang masuk ke dalam kategori tertentu.

Hipotesis project : 1. Apakah terdapat korelasi antara jumlah anak dengan kemampuan melunasi pinjaman tepat waktu? 2. Apakah terdapat korelasi antara status keluarga dengan kemampuan melunasi pinjaman tepat waktu? 3. Apakah terdapat korelasi antara kelas ekonomi dengan kemampuan melunasi pinjaman tepat waktu? 4. Apakah terdapat korelasi antara tujuan kredit dengan kemampuan melunasi pinjaman tepat waktu?

1.1 Membuka *file* data dan menampilkan informasi umumnya.

Kita akan mulai dengan mengimport library dan memuat data.

[1]:
```
# Memuat semua perpustakaan
import pandas as pd
```

[2]:
```
# muat data
df = pd.read_csv('/datasets/credit_scoring_eng.csv')
```

1.2 Soal 1. Eksplorasi Data

Deskripsi Data - *children* - jumlah anak dalam keluarga - *days_employed* - pengalaman kerja dalam hari - *dob_years* - usia klien dalam tahun - *education* - pendidikan klien - *education_id* - tanda pengenal pendidikan - *family_status* - status perkawinan - *family_status_id* - tanda pengenal status perkawinan - *gender* - jenis kelamin klien - *income_type* - jenis pekerjaan - *debt*

- apakah klien memiliki hutang pembayaran pinjaman - *total_income* - pendapatan bulanan - *purpose* - tujuan mendapatkan pinjaman

[3]:
```
# Memeriksa jumlah baris dan kolom dalam dataset
df.shape
```

[3]:
```
(21525, 12)
```

[4]:
```
# Menampilkan 10 baris pertama dalam dataset
df.head(10)
```

[4]:

	children	days_employed	dob_years	education	education_id	\
0	1	-8437.673028	42	bachelor's degree	0	
1	1	-4024.803754	36	secondary education	1	
2	0	-5623.422610	33	Secondary Education	1	
3	3	-4124.747207	32	secondary education	1	
4	0	340266.072047	53	secondary education	1	
5	0	-926.185831	27	bachelor's degree	0	
6	0	-2879.202052	43	bachelor's degree	0	
7	0	-152.779569	50	SECONDARY EDUCATION	1	
8	2	-6929.865299	35	BACHELOR'S DEGREE	0	
9	0	-2188.756445	41	secondary education	1	

	family_status	family_status_id	gender	income_type	debt	total_income	\
0	married	0	F	employee	0	40620.102	
1	married	0	F	employee	0	17932.802	
2	married	0	M	employee	0	23341.752	
3	married	0	M	employee	0	42820.568	
4	civil partnership	1	F	retiree	0	25378.572	
5	civil partnership	1	M	business	0	40922.170	
6	married	0	F	business	0	38484.156	
7	married	0	M	employee	0	21731.829	
8	civil partnership	1	F	employee	0	15337.093	
9	married	0	M	employee	0	23108.150	

	purpose
0	purchase of the house
1	car purchase
2	purchase of the house
3	supplementary education
4	to have a wedding
5	purchase of the house
6	housing transactions
7	education
8	having a wedding
9	purchase of the house for my family

Dari sampel data yang ditampilkan, terdapat beberapa masalah yang bisa dideteksi yaitu tedapat

value negatif dan value yang saya rasa sangat tinggi dari kolom `days_employed` yang saya rasa tidak masuk akal karena pada kolom menampilkan pengalaman kerja dalam hari , serta penulisan huruf kapital yang tidak tidak teratur pada kolom `education`.

[5]:
```
# Mendapatkan informasi seluruh kolom dalam data
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21525 entries, 0 to 21524
Data columns (total 12 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   children          21525 non-null  int64
 1   days_employed     19351 non-null  float64
 2   dob_years         21525 non-null  int64
 3   education         21525 non-null  object
 4   education_id      21525 non-null  int64
 5   family_status     21525 non-null  object
 6   family_status_id  21525 non-null  int64
 7   gender            21525 non-null  object
 8   income_type       21525 non-null  object
 9   debt              21525 non-null  int64
 10  total_income      19351 non-null  float64
 11  purpose           21525 non-null  object
dtypes: float64(2), int64(5), object(5)
memory usage: 2.0+ MB
```

Terdapat nilai yang hilang pada kolom `days_employed` dan `total_income`.

[6]:
```
# Menampilkan nilai yang hilang dalam dataset
df[df['days_employed'].isna()].head(10)
```

[6]:

	children	days_employed	dob_years	education	education_id	\
12	0	NaN	65	secondary education	1	
26	0	NaN	41	secondary education	1	
29	0	NaN	63	secondary education	1	
41	0	NaN	50	secondary education	1	
55	0	NaN	54	secondary education	1	
65	0	NaN	21	secondary education	1	
67	0	NaN	52	bachelor's degree	0	
72	1	NaN	32	bachelor's degree	0	
82	2	NaN	50	bachelor's degree	0	
83	0	NaN	52	secondary education	1	

	family_status	family_status_id	gender	income_type	debt	\
12	civil partnership	1	M	retiree	0	
26	married	0	M	civil servant	0	
29	unmarried	4	F	retiree	0	

```
41            married                   0      F  civil servant     0
55  civil partnership                   1      F        retiree     1
65          unmarried                   4      M       business     0
67            married                   0      F        retiree     0
72            married                   0      M  civil servant     0
82            married                   0      F       employee     0
83            married                   0      M       employee     0

    total_income                                    purpose
12           NaN                        to have a wedding
26           NaN                                education
29           NaN                  building a real estate
41           NaN                second-hand car purchase
55           NaN                        to have a wedding
65           NaN  transactions with commercial real estate
67           NaN       purchase of the house for my family
72           NaN  transactions with commercial real estate
82           NaN                                  housing
83           NaN                                  housing
```

Sejauh ini, dari yang saya lihat saya asumsikan bahwa value yang hilang memang tampak simetris karena berada pada baris yang sama, tapi untuk dapat menyimpulkan apakah nilai yang hilang memang bebar-benar berada pada baris yang sama perlu dilakukan investigasi lebih lanjut.

[7]:
```
# Melakukan beberapa pemfilteran untuk mengetahui jumlah baris dari value yang␣
↪hilang
df_filtered_nan = df[df['days_employed'].isna()]
df_filtered_nan = df_filtered_nan[df_filtered_nan['total_income'].isna()]
df_filtered_nan.shape[0]
```

[7]:
```
2174
```

Data yang haliang dalam dataset kita berjumlah 2174 baris.

[8]:
```
# Sekarang kta akan menghitung rasio value yang hilang dari seluruh dataframe
df_distribution_nan = df_filtered_nan.shape[0] / df.shape[0]
print(f'Distribusi nilai yang hilang sebesar: {df_distribution_nan:0%}')
```

```
Distribusi nilai yang hilang sebesar: 10.099884%
```

Kesimpulan menengah

Kita bisa simpulkan jumlah nilai hilang sama dengan jumlah tabel yang difilter. Artinya nilai yang hilang dari tabel yang difilter simetris.

Persentase data yang hilang dari dataframe sebesar 10%, cukup berpengaruh terhadap data kita bukan?

Selanjutnya saya akan menghitung jumlah persentase dari value yang hilang apakah memiliki dampak yang signifikan terhadap data dalam dataset, sehingga kita akan mengetahui langkah

yang tepat untuk memproses data yang hilang tersebut.

[9]:
```
# Menerapkan filter untuk menampilkan baris yang hilang dari data
df_filtered_nan.head(10)
```

[9]:

	children	days_employed	dob_years	education	education_id	\
12	0	NaN	65	secondary education	1	
26	0	NaN	41	secondary education	1	
29	0	NaN	63	secondary education	1	
41	0	NaN	50	secondary education	1	
55	0	NaN	54	secondary education	1	
65	0	NaN	21	secondary education	1	
67	0	NaN	52	bachelor's degree	0	
72	1	NaN	32	bachelor's degree	0	
82	2	NaN	50	bachelor's degree	0	
83	0	NaN	52	secondary education	1	

	family_status	family_status_id	gender	income_type	debt	\
12	civil partnership	1	M	retiree	0	
26	married	0	M	civil servant	0	
29	unmarried	4	F	retiree	0	
41	married	0	F	civil servant	0	
55	civil partnership	1	F	retiree	1	
65	unmarried	4	M	business	0	
67	married	0	F	retiree	0	
72	married	0	M	civil servant	0	
82	married	0	F	employee	0	
83	married	0	M	employee	0	

	total_income	purpose
12	NaN	to have a wedding
26	NaN	education
29	NaN	building a real estate
41	NaN	second-hand car purchase
55	NaN	to have a wedding
65	NaN	transactions with commercial real estate
67	NaN	purchase of the house for my family
72	NaN	transactions with commercial real estate
82	NaN	housing
83	NaN	housing

Value yang hilang tampak simetris dari table yang ditampilkan.

[10]:
```
# Memeriksa distribusi
print(df_filtered_nan['income_type'].value_counts())
print()
df_filtered_nan['income_type'].value_counts() / df['income_type'].
 ↪value_counts() * 100
```

```
employee         1105
business          508
retiree           413
civil servant     147
entrepreneur        1
Name: income_type, dtype: int64
```

[10]:
```
business                        9.990167
civil servant                  10.075394
employee                        9.937944
entrepreneur                   50.000000
paternity / maternity leave          NaN
retiree                        10.710581
student                              NaN
unemployed                           NaN
Name: income_type, dtype: float64
```

Cukup terpola disini tetapi memang ada beberapa kategori yang valuenya tidak bisa dikatakan valid untuk dilakukan perhitungan mengingat jumlahnya hanya sedikit sekitar `1 - 2` data saja jumlahnya seperti kategori `unemployed, paternity / maternity leave, student,` dan `entrepreneur`. Secara overall data yang hilang terdistribusi sebesar `10%` pada setiap kategori.

Kemungkinan penyebab hilangnya nilai dalam data

Belum dapat ditentukan penyebab nilai yang hilang, kita harus mempertimbangkan kemungkinan-kemungkinan lain penyebab nilai yang hilai apakah nilai yang memiliki karakteristik tertentu seperti dari klien yang sudah menikah, jumlah anak, ataupun klien yang memiliki tunggakan pembayaran kredit.

[11]:
```
# Memeriksa distribusi di seluruh dataset
df.isna().sum() / df.shape[0] * 100
```

[11]:
```
children            0.000000
days_employed      10.099884
dob_years           0.000000
education           0.000000
education_id        0.000000
family_status       0.000000
family_status_id    0.000000
gender              0.000000
income_type         0.000000
debt                0.000000
total_income       10.099884
purpose             0.000000
dtype: float64
```

Kesimpulan menengah

Jumlah nilai yang hilang dalam dataset mirip dengan tabel yang difilter. Dan jumlah data yang

hilang pada kolom `days_employed` sama dengan `total_income` yang artinya error hanya terjadi pada dua kolom tersebut.

Hal ini menimbulkan pertanyaan apakah nilai yang hilang membentuk suatu pola atau terjadi secara acak? Untuk mejawab pertannyaan ini mari kita lakukan analisa lebih lanjut.

[12]:
```
# Memeriksa apakah ada pola lain yang menyebabkan hilangnya data dari kolom
 ↪'family_status'
print(df_filtered_nan['family_status'].value_counts())
print()
df_filtered_nan['family_status'].value_counts() / df['family_status'].
 ↪value_counts() * 100
```

```
married              1237
civil partnership     442
unmarried             288
divorced              112
widow / widower        95
Name: family_status, dtype: int64
```

[12]:
```
married               9.991922
civil partnership    10.581757
unmarried            10.238180
divorced              9.372385
widow / widower       9.895833
Name: family_status, dtype: float64
```

Kesimpulan menengah

Cukup menarik bahwa nilai yang hilang terdistribusi secara merata dari kategori di kolom `family_status` yaitu sekitar 10%. Tetapi juga dikatahui tidak ada ketegori khusus yang mengakibatkan data hilang disebabkan oleh salah satu kategori saja.

[13]:
```
# Check for relation both 'gender' and missing value
print(df_filtered_nan['gender'].value_counts())
print()
df_filtered_nan['gender'].value_counts() / df['gender'].value_counts() * 100
```

```
F    1484
M     690
Name: gender, dtype: int64
```

[13]:
```
F      10.424276
M       9.467618
XNA          NaN
Name: gender, dtype: float64
```

Pada kategori gender data yang hilang masing-masing terdistribusi sebesar 9% dan 10% artinya data yang hilang tidak hanya terdapat pada satu kategori saja.

[14]:
```
# Memeriksa pendapatan dari beberapa pekerjaan yang kemungkinan tidak memiliki␣
↪'income'
df[df['income_type'].isin(['unemployed', 'paternity / maternity leave',␣
↪'student', 'entrepreneur'])]
```

[14]:

	children	days_employed	dob_years	education	education_id	\
3133	1	337524.466835	31	secondary education	1	
5936	0	NaN	58	bachelor's degree	0	
9410	0	-578.751554	22	bachelor's degree	0	
14798	0	395302.838654	45	Bachelor's Degree	0	
18697	0	-520.848083	27	bachelor's degree	0	
20845	2	-3296.759962	39	SECONDARY EDUCATION	1	

	family_status	family_status_id	gender	\
3133	married	0	M	
5936	married	0	M	
9410	unmarried	4	M	
14798	civil partnership	1	F	
18697	civil partnership	1	F	
20845	married	0	F	

	income_type	debt	total_income	\
3133	unemployed	1	9593.119	
5936	entrepreneur	0	NaN	
9410	student	0	15712.260	
14798	unemployed	0	32435.602	
18697	entrepreneur	0	79866.103	
20845	paternity / maternity leave	1	8612.661	

	purpose
3133	buying property for renting out
5936	buy residential real estate
9410	construction of own property
14798	housing renovation
18697	having a wedding
20845	car

Dari tabel diatas kita bisa mendapatkan informasi bahwa nilai yang hilang tidak selalu diakibatkan oleh pekerjaan yang biasanya tidak memiliki penghasilan, mungkin ada beberapa alasan bagaimana cara mereka mendapatkan penghasilan meskipun tidak sedang memiliki pekerjaan yang regular. Seperti pada baris 3133 klien yang `unemployed` tetapi mengajukan pinjaman untuk membeli properti untuk desewakan, meskipun belum diketahui darimana penghasilannya, Klien 9410 seorang `student` apakah dia seorang pekerja part time atau memiliki *rich parents* tetapi cukup aneh mengingat kegunaannya tidak untuk melanjutkan pendidikan melainkan membangun properti. Selain itu di beberapa negara juga memilki peraturan bahwa `paternity / maternity leave`

still getting paid.

Kesimpulan

Konklusi yang bisa kita ambil mengenai penyebab data yang hilang adalah *human error* karena data yang hilang tidak terjadi pada kategori tertentu saja melainkan pada beberapa kategori.

Dari beberapa pengujian yang sudah saya lakukan saya menemukan bahwa dari kolom `family_status` dan `income_type` saya menemukan bahwa setiap kategori dalam masing-masing kolom tersebut memiliki nilai yang hilang hampir sama di setiap kategori yang sekitar `10%` artinya nilai yang hilang terdistribusi secara merata di setiap kategori dan nilainya simetris dengan pengujian yang kita lakukan sebelumnya dengan mencari distribusi nilai yang hilang di seluruh dataset yang menghasilkan angka juga sebesar `10%`.

Untuk beberapa masalah seperti: 1. Untuk nilai yang hilang saya akan membuat beberapa kateghori berdasarkan usia untuk mencari rata-rata `days_employed` dan `total_income` yang akan digunakan untuk mengisi value yang hilang. 2. Untuk mengatasi register yang berbeda pada kolom `education` saya akan mengubah semua huruf menjadi `lower`. 3. Kita bisa melakukan `drop` untuk nilai duplikat.

1.3 Transformasi data

Memeriksa kolom 'education'

[15]:
```
# Memeriksa ejaan pada kolom 'education' yang terindikasi memiliki perbebedaan␣
↪penulisan dengan makna yang sama
df_education_value = df.pivot_table(index='education', values='days_employed',␣
↪aggfunc= 'count')
df_education_value
```

[15]:

education	days_employed
BACHELOR'S DEGREE	251
Bachelor's Degree	243
GRADUATE DEGREE	1
Graduate Degree	1
PRIMARY EDUCATION	16
Primary Education	14
SECONDARY EDUCATION	705
SOME COLLEGE	22
Secondary Education	646
Some College	40
bachelor's degree	4222
graduate degree	4
primary education	231
secondary education	12342
some college	613

[16]:
```

```

```
# Kemudian kita akan memeriksa nilai duplikat sebelum kita mengangani register␣
↪yang tidak teratur
df.duplicated().sum()
```

[16]: 54

```
# Memperbaiki penulisan
df['education'] = df['education'].str.lower()
```

```
# Memeriksa kembali nilai duplikat setelah dilakukan perbaikan
df.duplicated().sum()
```

[18]: 71

Nilai duplikat bertambah dari sebelumnya 54 baris menjadi 71 baris, artinya ada baris yang memilki nilai yang sama dalam data dengan input `education` yang berbeda.

```
# Memeriksa kembali apakah penulisan kolom 'education' telah diperbaiki
df_education_value = df.pivot_table(index='education', values='days_employed',␣
↪aggfunc= 'count')
df_education_value
```

[19]:

education	days_employed
bachelor's degree	4716
graduate degree	6
primary education	261
secondary education	13693
some college	675

Masalah pada kolom `education` telah kita atasi selanjutnya kita akan memeriksa kolom `children`.

Memeriksa kolom `children`

```
# Menampilkan distribusi nilai pada kolom `children`
df_children_value = df.pivot_table(index='children', values='days_employed',␣
↪aggfunc= 'count')
df_children_value
```

[20]:

children	days_employed
-1	44
0	12710
1	4343
2	1851
3	294
4	34
5	8

```
20                    67
```

Terdapat value yang menunjukkan angka -1 dan 20 yang saya nilai tidak mungkin. Kita asumsikan nilai yang bermasalah terjadi karena kesalahan input/typo. Kita akan me-replace nilai -1 menjadi 1 dan 20 menjadi 2.

[21]:
```
# Memeriksa jumlah dan persentase data yang bermasalah pada kolom 'children'
df_child_problem_value = (df['children'] == -1).sum() + (df['children'] == 20).
 ↪sum()
print('Jumlah nilai yang tidak wajar:', df_child_problem_value)
df_child_prob_percent = df_child_problem_value / df.shape[0]
print(f'Persentase data yang bermasalah adalah: {df_child_prob_percent:.2%}')
```

```
Jumlah nilai yang tidak wajar: 123
Persentase data yang bermasalah adalah: 0.57%
```

Data dengan nilai yang tidak wajar tidak sampai 1% dalam dataset kita meskipun begitu lebih baik kita perbaiki alih-alih melakukan *drop* pada data tersebut.

[22]:
```
# Memperbaiki nilai yang bermasalah
df.loc[df['children'] == -1, 'children'] = 1
df.loc[df['children'] == 20, 'children'] = 2
df.duplicated().sum()
```

[22]:
```
71
```

[23]:
```
# Periksa kembali kolom `children` untuk memastikan semua telah diperbaiki
df_children_value = df.pivot_table(index='children', values='days_employed',␣
 ↪aggfunc= 'count')
df_children_value
```

[23]:

children	days_employed
0	12710
1	4387
2	1918
3	294
4	34
5	8

[24]:
```
# Mari kita periksa kembali nilai duplikat
df.duplicated().sum()
```

[24]:
```
71
```

Belum ada perubahan pada nilai duplikat.

Memeriksa kolom `days_employed`

[25]:
```
# Kita akan melihat nilai yang tidak wajar pada kolom 'days_employed'
df[df['income_type'] == 'retiree']
```

[25]:
```
        children  days_employed  dob_years            education  education_id  \
4              0  340266.072047         53  secondary education             1
12             0            NaN         65  secondary education             1
18             0  400281.136913         53  secondary education             1
24             1  338551.952911         57  secondary education             1
25             0  363548.489348         67  secondary education             1
…            …              …        …                    …             …
21505          0  338904.866406         53  secondary education             1
21508          0  386497.714078         62  secondary education             1
21509          0  362161.054124         59    bachelor's degree             0
21518          0  373995.710838         59  secondary education             1
21521          0  343937.404131         67  secondary education             1

            family_status  family_status_id gender income_type  debt  \
4       civil partnership                 1      F     retiree     0
12      civil partnership                 1      M     retiree     0
18        widow / widower                 2      F     retiree     0
24              unmarried                 4      F     retiree     0
25                married                 0      M     retiree     0
…                     …               …    …         …   …
21505   civil partnership                 1      M     retiree     0
21508             married                 0      M     retiree     0
21509             married                 0      M     retiree     0
21518             married                 0      F     retiree     0
21521             married                 0      F     retiree     0

        total_income                                  purpose
4          25378.572                  to have a wedding
12               NaN                  to have a wedding
18          9091.804              buying a second-hand car
24         46487.558  transactions with commercial real estate
25          8818.041                    buy real estate
…                …                                  …
21505      12070.399                  to have a wedding
21508      11622.175                           property
21509      11684.650           real estate transactions
21518      24618.344                  purchase of a car
21521      24959.969                  purchase of a car

[3856 rows x 12 columns]
```

[26]:
```
# Menampilkan statistik deskriptif pada kolom 'days_employed'
df['days_employed'].describe()
```

[26]:
```
count     19351.000000
mean      63046.497661
std      140827.311974
min      -18388.949901
25%       -2747.423625
50%       -1203.369529
75%        -291.095954
max      401755.400475
Name: days_employed, dtype: float64
```

Terdapat angka-angka negatif dan nilai yang tidak wajar, perlu diingat kolom ini menampilkan jumlah hari klien telah bekerja.

[27]:
```
# Menampilkan nilai rata-rata 'days_employed'
df_days_employed_median = df.groupby(['income_type'])['days_employed'].median()
df_days_employed_median
```

[27]:
```
income_type
business                      -1547.382223
civil servant                 -2689.368353
employee                      -1574.202821
entrepreneur                   -520.848083
paternity / maternity leave    -3296.759962
retiree                       365213.306266
student                         -578.751554
unemployed                    366413.652744
Name: days_employed, dtype: float64
```

Kita tidak bisa melakukan proses uji kelayakan credit sebelum memperbaiki nilai pada kolom ini, nilai negatif sama dengan nilai yang hilang.

[28]:
```
df_days_employed_problem_value = (df['days_employed'] < 0).sum() +␣
↪(df['days_employed'] > 21600).sum()
print(df_days_employed_problem_value)
df_days_employed_prob_percent = df_days_employed_problem_value / df.shape[0]
print(f'Persentase data yang bermasalah adalah: {df_days_employed_prob_percent:.
↪2%}')
```

```
19351
Persentase data yang bermasalah adalah: 89.90%
```

Terdapat masalah yang sangat tinggi pada kolom `days_employed`, jika kita melihat nilai negatif sudah berdampak besar dalam dataset karena itu sama saja dengan nilai yang hilang, lalu bagaimana dengan nilai yang terlampau tinggi, tidak ada seorangpun bekerja melebihi usianya sendiri. Karena nilai mungkin terjadi karena kesalah teknis dalam penambahan tanda `-` jadi solusi yang dapat saya lakukan adalah dengan dengan mengganti angka negatif menjadi positif dengan fungsi `absolut`. Kemudian untuk nilai yang terlampau tinggi yang saya asumsikan hanya terjadi pada pada `kategori`, `retiree` dan `unemployed` maka saya putuskan untuk mengganti nilai tersebut dengan 0 dengan batas `21600 hari atau 59 tahun`.

[29]:
```
# Menerapkan fungsi 'absolut' untuk nilai negatif dan me-'replace' nilai yang␣
↪tidak wajar dengan angka '0'
df['days_employed'] = df['days_employed'].abs()
df.loc[df['days_employed'] > 21600, 'days_employed'] = 0
```

[30]:
```
# Memastikan nilai telah diperbaiki
print(df['days_employed'].describe())
print()
print((df['days_employed'] < 0).sum())
```

```
count    19351.000000
mean      1934.115623
std       2274.751213
min          0.000000
25%        291.095954
50%       1203.369529
75%       2747.423625
max      18388.949901
Name: days_employed, dtype: float64

0
```

Terdapat masalah pada kolom `usia` yaitu tidak seseorang yang bekerja sejak usia `0`

Memeriksa kolom `dob_years`

[31]:
```
# Memeriksa kolom 'dob_years' apakah memiliki nilai anomali
df_dob_years_value = df.pivot_table(index='dob_years', values='days_employed',␣
↪aggfunc= 'count')
df_dob_years_value
```

[31]:

dob_years	days_employed
0	91
19	13
20	46
21	93
22	166
23	218
24	243
25	334
26	373
27	457
28	446
29	495
30	482
31	495
32	473

33	530
34	534
35	553
36	492
37	484
38	544
39	522
40	543
41	548
42	532
43	463
44	503
45	447
46	427
47	421
48	492
49	458
50	463
51	398
52	431
53	415
54	424
55	395
56	433
57	404
58	405
59	410
60	338
61	317
62	314
63	240
64	228
65	174
66	163
67	151
68	90
69	80
70	62
71	53
72	31
73	7
74	6
75	1

[32]:
```
# Melihat persentase data anomali
df_days_employed_prob_percent = (df['dob_years'] == 0).sum() / df.shape[0]
```

```
print(f'Persentase data yang bermasalah adalah: {df_days_employed_prob_percent:.
↪2%}')
```

```
Persentase data yang bermasalah adalah: 0.47%
```

Saya akan mengganti value 0 di kolom **age**, dengan **median** dari usia. Karena tiap klien yang memiliki usia 0 memiliki karakteristik yang berbeda.

[33]:
```
# Mengganti angka '0' dengan rata-rata
avg_dob_years = df['dob_years'].median()
df.loc[df['dob_years'] == 0, 'dob_years'] = avg_dob_years
```

[34]:
```
# Memastikai nilai telah diperbaiki
df[df['dob_years'] == 0].shape[0]
```

[34]:
```
0
```

Memeriksa kolom family_status

[35]:
```
# Memeriksa kolom 'family_status'
df_family_status_value = df.pivot_table(index='family_status',␣
↪values='days_employed', aggfunc= 'count')
df_family_status_value
```

[35]:

family_status	days_employed
civil partnership	3735
divorced	1083
married	11143
unmarried	2525
widow / widower	865

Tidak ditemukan masalah pada kolom family_status

Memeriksa kolom gender

[36]:
```
# Check 'gender' column
df_gender_value = df.pivot_table(index='gender', values='days_employed',␣
↪aggfunc= 'count')
df_gender_value
```

[36]:

gender	days_employed
F	12752
M	6598
XNA	1

Terdapat satu masalah yaitu pada value XNA Sulit untuk mengidentifikasi untuk mengganti value tersebut, dan saya putuskan untuk menghapus 1 baris tersebut

[37]:
```
# Menghapus value `XNA`
df = df.loc[df["gender"] != 'XNA']
```

[38]:
```
# Memastikan kolom telah diperbaiki
df_gender_value = df.pivot_table(index='gender', values='days_employed',␣
 ↪aggfunc= 'count')
df_gender_value
```

[38]:

gender	days_employed
F	12752
M	6598

Memeriksa kolom `income_type`

[39]:
```
# Mari kita lihat nilai dalam kolom
df_income_type_value = df.pivot_table(index='income_type',␣
 ↪values='days_employed', aggfunc= 'count')
df_income_type_value
```

[39]:

income_type	days_employed
business	4576
civil servant	1312
employee	10014
entrepreneur	1
paternity / maternity leave	1
retiree	3443
student	1
unemployed	2

Tidak ada masalah pada kolom `income_type`

Terdapat nilai duplikat sebesar `72` atau hanya berdampak `0.3%` dalam dataset sehingga saya putuskan untuk melakukan `drop` untuk nilai duplikat, karena nilai tersebut masih dapat diterima.

[40]:
```
# Memeriksa duplikat
print(df.duplicated().sum())
print()
df_duplicated_percent = df.duplicated().sum() / df.shape[0]
print(f'Distribusi nilai duplikat sebesar: {df_duplicated_percent:.2%}')
```

```
72

Distribusi nilai duplikat sebesar: 0.33%
```

[41]:
```
# Atasi duplikat, jika ada
df = df.drop_duplicates().reset_index(drop=True)
```

[42]:
```
# Terakhir periksa apakah kita memiliki duplikat
df.duplicated().sum()
```

[42]:
```
0
```

[43]:
```
# Periksa ukuran dataset yang sekarang Anda miliki setelah manipulasi pertama␣
↪yang Anda lakukan
df.shape
```

[43]:
```
(21452, 12)
```

[44]:
```
old_df_shape = 21525
df_shape_percentage = (old_df_shape - df.shape[0]) / old_df_shape
print(f'Persentase dari perubahan dalam dataset adalah: {df_shape_percentage:.
↪2%}')
```

```
Persentase dari perubahan dalam dataset adalah: 0.34%
```

Kita telah memperbaiki beberapa masalah dalam dataset seperti: 1. Nilai negatif dan nilai yang terlampau tinggi dalam kolom `days_employed`. 2. Register yang tidak teratur dalam kolom `education`. 3. Menghapus nilai duplikat. 4. Beberapa masalah yang terjadi di kolom `children` dan `gender`.

Sehingga kita mendapatkan perubahan sebesar `0.34%` yang artinya nilai tersebut tidak begitu berdampak dalam dataset karena kurang dari `1%`.

2 Bekerja dengan nilai yang hilang

Saya memasukkan dictionary `numpy` untuk mempercepat pekerjaan saya yang akan digunakan untuk me-`replace` nilai `0` pada kolom `days_employed` setelah membuat beberapa kategori usia.

[45]:
```
# Import dictionary
import numpy as np
```

2.0.1 Memperbaiki nilai yang hilang di `total_income`

Kita akan mulai untuk mengatasi nilai yang hilang pada kolom`total_income`.

Pertama-tama kita akan membuat beberapa kategori berdasarkan rentang usia dari kolom `dob_years`, yang diharapkan dapat membantu kita dalam mencari nilai rata-rata dari setiap rentang usia yang akan kita gunakan untuk mengisi nilai yang hilang dari masing-masing kategori.

[46]:
```
# Menulis fungsi untuk menghitung kategori usia
def age_group(age):

    if age <= 30:
        return '19-30'
    elif age <= 40:
```

```
        return '31-40'
    elif age <= 50:
        return '41-50'
    elif age <= 60:
        return '51-60'
    else:
        return '+60'
```

[47]:
```
# Melakukan pengujian apakah fungsi bekerja atau tidak
print(age_group(23))
print(age_group(38))
print(age_group(46))
print(age_group(55))
print(age_group(70))
```

```
19-30
31-40
41-50
51-60
+60
```

[48]:
```
# Membuat kolom baru berdasarkan fungsi
df['age_group'] = df['dob_years'].apply(age_group)
```

[49]:
```
# Memeriksa bagaimana nilai di dalam kolom baru
df.tail(10)
```

[49]:

	children	days_employed	dob_years	education	education_id	\
21442	1	467.685130	28.0	secondary education	1	
21443	0	914.391429	42.0	bachelor's degree	0	
21444	0	404.679034	42.0	bachelor's degree	0	
21445	0	0.000000	59.0	secondary education	1	
21446	1	2351.431934	37.0	graduate degree	4	
21447	1	4529.316663	43.0	secondary education	1	
21448	0	0.000000	67.0	secondary education	1	
21449	1	2113.346888	38.0	secondary education	1	
21450	3	3112.481705	38.0	secondary education	1	
21451	2	1984.507589	40.0	secondary education	1	

	family_status	family_status_id	gender	income_type	debt	\
21442	married	0	F	employee	1	
21443	married	0	F	business	0	
21444	civil partnership	1	F	business	0	
21445	married	0	F	retiree	0	
21446	divorced	3	M	employee	0	
21447	civil partnership	1	F	business	0	
21448	married	0	F	retiree	0	

```
21449  civil partnership                1      M    employee     1
21450            married                0      M    employee     1
21451            married                0      F    employee     0

       total_income                    purpose age_group
21442     17517.812          to become educated     19-30
21443     51649.244    purchase of my own house     41-50
21444     28489.529           buying my own car     41-50
21445     24618.344           purchase of a car     51-60
21446     18551.846  buy commercial real estate     31-40
21447     35966.698        housing transactions     41-50
21448     24959.969           purchase of a car       +60
21449     14347.610                    property     31-40
21450     39054.888           buying my own car     31-40
21451     13127.587                to buy a car     31-40
```

Beberapa faktor yang dapat memengaruhi pendapatan diantaranya `education`, `days_employed` (pengalaman kerja dalam hari), dan `income_type` (jenis pekerjaan). Untuk menentukan apakah kita akan menggunakan `mean` atau `median` untuk mengisi nilai yang hilang, kita melakukan eksplorasi lebih dalam untuk mengetahui bagaimana distribusi nilai `mean` dan `median` berdasarkan kategori yang kita kelompokkan terdistribusi secara normal atau tidak.

[50]:
```
# Membuat tabel tanpa nilai yang hilang dan menampilkan beberapa barisnya
df_clean = df[df.notna()]
df_clean.head(10)
```

[50]:
```
   children  days_employed  dob_years            education  education_id  \
0         1    8437.673028       42.0    bachelor's degree             0
1         1    4024.803754       36.0  secondary education             1
2         0    5623.422610       33.0  secondary education             1
3         3    4124.747207       32.0  secondary education             1
4         0       0.000000       53.0  secondary education             1
5         0     926.185831       27.0    bachelor's degree             0
6         0    2879.202052       43.0    bachelor's degree             0
7         0     152.779569       50.0  secondary education             1
8         2    6929.865299       35.0    bachelor's degree             0
9         0    2188.756445       41.0  secondary education             1

       family_status  family_status_id gender income_type  debt  total_income  \
0            married                 0      F    employee     0     40620.102
1            married                 0      F    employee     0     17932.802
2            married                 0      M    employee     0     23341.752
3            married                 0      M    employee     0     42820.568
4  civil partnership                 1      F     retiree     0     25378.572
5  civil partnership                 1      M    business     0     40922.170
6            married                 0      F    business     0     38484.156
7            married                 0      M    employee     0     21731.829
```

```
8  civil partnership                1      F    employee     0     15337.093
9            married                0      M    employee     0     23108.150

                                 purpose age_group
0                purchase of the house     41-50
1                          car purchase     31-40
2                purchase of the house     31-40
3              supplementary education     31-40
4                    to have a wedding     51-60
5                purchase of the house     19-30
6                 housing transactions     41-50
7                            education     41-50
8                      having a wedding     31-40
9  purchase of the house for my family     41-50
```

[51]:
```
# Menampilkan nilai 'mean' dari 'total_income' berdasarkan 'age_group'
df_clean.groupby(['age_group'])['total_income'].mean()
```

[51]:
```
age_group
+60      23057.777452
19-30    25815.651899
31-40    28376.735148
41-50    28332.806009
51-60    25482.856294
Name: total_income, dtype: float64
```

[52]:
```
# Menampilkan nilai 'median' dari 'total_income' berdasarkan 'age_group'
df_clean.groupby(['age_group'])['total_income'].median()
```

[52]:
```
age_group
+60      19637.0560
19-30    22955.4740
31-40    24825.1865
41-50    24563.6500
51-60    22056.7710
Name: total_income, dtype: float64
```

Pendapatan rata-rata berdasarkan kategori dari `mean` memiliki rata-rata yang lebih besar untuk setiap kategori, dengan demikian saya akan memilih `median` karena lebih merepresentasikan rata-rata berdasarkan kategori kelompok umur.

[53]:
```
# Saatnya kita mengisi nilai yang hilang
df['total_income'] = df.groupby(['age_group'])['total_income'].transform(lambda␣
↪x: x.fillna(x.median()))
```

[54]:
```
# Memeriksa statistik deskriptif dari 'total_income'
df['total_income'].describe()
```

[54]:
```
count     21452.000000
mean      26447.583422
std       15690.140709
min        3306.762000
25%       17217.441750
50%       23234.038000
75%       31328.693750
max      362496.645000
Name: total_income, dtype: float64
```

[55]:
```
# Memeriksa apakah nilai hilang sudah terisi
df['total_income'].isna().sum()
```

[55]:
```
0
```

Diketahui data yang hilang belum terisi pada kolom `total_income` itu karena kita belum memasukkan nilai dari kolom baru yang kita buat untuk mengganti nilai yang hilang dari kolom tersebut. Untuk mengisi kolom tersebut kita akan menggantinya dengan metode `fillna` dari kolom baru yang kita buat yaitu `total_revenue`.

Seperti yang kita lihat sudah tidak terdapat nilai yang hilang dari kolom `total_income`. Saatnya memeriksa apakah jumlah kolom `total_income` sama dengan jumlah kolom lainnya.

[56]:
```
# Memeriksa jumlah entri kolom dari dataset
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21452 entries, 0 to 21451
Data columns (total 13 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   children          21452 non-null  int64
 1   days_employed     19350 non-null  float64
 2   dob_years         21452 non-null  float64
 3   education         21452 non-null  object
 4   education_id      21452 non-null  int64
 5   family_status     21452 non-null  object
 6   family_status_id  21452 non-null  int64
 7   gender            21452 non-null  object
 8   income_type       21452 non-null  object
 9   debt              21452 non-null  int64
 10  total_income      21452 non-null  float64
 11  purpose           21452 non-null  object
 12  age_group         21452 non-null  object
dtypes: float64(3), int64(4), object(6)
memory usage: 2.1+ MB
```

Dari informasi di atas kita menemukan bahwa jumlah pada kolom `total_income` sudah memiliki nilai yang sama dengan kolom lainnya. Selanjutnya kita akan memperbaiki nilai pada kolom

days_employed.

2.0.2 Memperbaiki nilai di days_employed

Tentunya faktor yang paling berpengaruh di dalam kolom days_employed adalah age_category bayangkan seseorang tidak mungkin berkerja melebihi usia minimal seseorang untuk bekerja secara legal atau bahkan pengalaman kerja seseorang melebihi jumlah usianya sendiri, itu merupakan hal yang mustahil.

[57]:
```
# Menampilkan nilai 'median' dari 'days_employed' berdasarkan 'age_group'
df_clean.groupby(['age_group'])['days_employed'].median()
```

[57]:
```
age_group
+60         0.000000
19-30    1041.692949
31-40    1601.812856
41-50    1931.819208
51-60     551.486703
Name: days_employed, dtype: float64
```

[58]:
```
# Menampilkan nilai 'median' dari 'days_employed' berdasarkan 'age_group'
df_clean.groupby(['age_group'])['days_employed'].mean()
```

[58]:
```
age_group
+60       799.550825
19-30    1276.700146
31-40    2075.768902
41-50    2685.878763
51-60    1937.100928
Name: days_employed, dtype: float64
```

Saya rasa mean bisa mewakili nilai untuk kolom days_employed, selain karena median memiliki nilai 0 yang mungkin terjadi karena kita mengubah nilai yang trelampau tinggi pada pengujian sebelumnya, tetapi apakah seseorang yang telah mencapai usia pensiun tidak memiliki pengalaman bekerja sama sekali? Atau mungkin sesorang yang telah pensiun pengalamannya akan dihitung sebagai nilai 0.

[59]:
```
# Waktunya untuk kita mengganti nilai yang hilang dan nilai '0' dengan rata-rata
df.loc[df['days_employed'] == 0, 'days_employed'] = np.NaN
df['days_employed'] = df.groupby(['age_group'])['days_employed'].
 ↪transform(lambda x: x.fillna(x.mean()))
```

[60]:
```
# Menampilkan statistik deskriptif dari kolom 'days_employed'
df['days_employed'].describe()
```

[60]:
```
count    21452.000000
mean      2560.787849
std       2049.095043
```

```
min         24.141633
25%       1023.688788
50%       2228.980549
75%       3332.487952
max      18388.949901
Name: days_employed, dtype: float64
```

[61]:
```
# Memeriksa apakah nilai yang hilang telah teratasi
df['days_employed'].isna().sum()
```

[61]:
```
0
```

Memeriksa jumlah kolom dari seluruh dataset.

[62]:
```
# Memeriksa informasi seluruh dataset
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21452 entries, 0 to 21451
Data columns (total 13 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   children          21452 non-null  int64
 1   days_employed     21452 non-null  float64
 2   dob_years         21452 non-null  float64
 3   education         21452 non-null  object
 4   education_id      21452 non-null  int64
 5   family_status     21452 non-null  object
 6   family_status_id  21452 non-null  int64
 7   gender            21452 non-null  object
 8   income_type       21452 non-null  object
 9   debt              21452 non-null  int64
 10  total_income      21452 non-null  float64
 11  purpose           21452 non-null  object
 12  age_group         21452 non-null  object
dtypes: float64(3), int64(4), object(6)
memory usage: 2.1+ MB
```

Semua kolom termasuk `days_employed` telah memiliki nilai yang sama, saya rasa pekerjaan kita untuk *data cleansing* telah selesai. Sekarang mari kita coba mengeksplorasi data lebih lanjut untuk mendapatkan hal menarik dalam data maupun sebagai acuan kita dalam mengambil keputusan.

2.1 Pengkategorian Data

Sepertinya saya menemukan hal menarik di kolom `purpose` yaitu banyak sekali peng-kategorian data yang saya rasa bisa kita sederhanakan menjadi lebih general, sehingga kita akan lebih mudah dalam melakukan investigasi dalam mengambil keputusan.

[63]:
```
# Menampilkan kolom 'purpose' untuk dikategorikan dengan lebih umum
df['purpose'].value_counts()
```

[63]:
```
wedding ceremony                            791
having a wedding                            767
to have a wedding                           765
real estate transactions                    675
buy commercial real estate                  661
housing transactions                        652
buying property for renting out             651
transactions with commercial real estate    650
housing                                     646
purchase of the house                       646
purchase of the house for my family         638
construction of own property                635
property                                    633
transactions with my real estate            627
building a real estate                      624
purchase of my own house                    620
buy real estate                             620
building a property                         619
housing renovation                          607
buy residential real estate                 606
buying my own car                           505
going to university                         496
car                                         494
second-hand car purchase                    486
buying a second-hand car                    478
cars                                        478
to own a car                                478
to buy a car                                471
car purchase                                461
supplementary education                     460
purchase of a car                           455
university education                        452
education                                   447
to get a supplementary education            446
getting an education                        442
profile education                           436
getting higher education                    426
to become educated                          408
Name: purpose, dtype: int64
```

Memeriksa value unique.

[64]:
```
# Memeriksa nilai unik
df['purpose'].sort_values().unique()
```

[64]:
```
array(['building a property', 'building a real estate',
       'buy commercial real estate', 'buy real estate',
       'buy residential real estate', 'buying a second-hand car',
       'buying my own car', 'buying property for renting out', 'car',
       'car purchase', 'cars', 'construction of own property',
       'education', 'getting an education', 'getting higher education',
       'going to university', 'having a wedding', 'housing',
       'housing renovation', 'housing transactions', 'profile education',
       'property', 'purchase of a car', 'purchase of my own house',
       'purchase of the house', 'purchase of the house for my family',
       'real estate transactions', 'second-hand car purchase',
       'supplementary education', 'to become educated', 'to buy a car',
       'to get a supplementary education', 'to have a wedding',
       'to own a car', 'transactions with commercial real estate',
       'transactions with my real estate', 'university education',
       'wedding ceremony'], dtype=object)
```

Dari pengamatan saya sebelumnya memang benar bahwa peng-kategorian yang terjadi di kolom `purpose` bisa generalisasikan ke dalam beberapa kategori yang umum dan mudah dipahami, sperti `car`, `property`, `education`, dan `wedding`. Mari kita buat fungsinya di bawah ini:

[65]:
```python
# Menulis fungsi untuk mengkategorikan data berdasarkan topik umum
def purpose_common(purpose):

    if 'property' in purpose:
        return 'property'
    elif 'estate' in purpose:
        return 'property'
    elif 'hous' in purpose:
        return 'property'
    elif 'car' in purpose:
        return 'car'
    elif 'educ' in purpose:
        return 'education'
    elif 'univ' in purpose:
        return 'education'
    elif 'wedd' in purpose:
        return 'wedding'
```

[66]:
```python
# Memuat kolom dengan kategori dan menghitung nilainya
df['general_purpose'] = df['purpose'].apply(purpose_common)
print(df['general_purpose'].value_counts())
print()
df['general_purpose'].count()
```

```
property     10810
car           4306
education     4013
```

```
wedding       2323
Name: general_purpose, dtype: int64
```

[66]:
```
21452
```

Kita akan membuat kategori berdasarkan `total_income` ke dalam beberapa kelas.

[67]:
```
# Melihat kolom 'total_income' untuk dikategorikan
df['total_income'].value_counts().sort_index()
```

[67]:
```
3306.762      1
3392.845      1
3418.824      1
3471.216      1
3503.298      1
             ..
273809.483    1
274402.943    1
276204.162    1
352136.354    1
362496.645    1
Name: total_income, Length: 19348, dtype: int64
```

[68]:
```
# Mendapatkan kesimpulan statistik untuk kolom 'total_income'
df['total_income'].describe()
```

[68]:
```
count     21452.000000
mean      26447.583422
std       15690.140709
min        3306.762000
25%       17217.441750
50%       23234.038000
75%       31328.693750
max      362496.645000
Name: total_income, dtype: float64
```

Kita mengkategorikan `total_income` ke dalam beberapa kelas kategori berdasarkan klasifikasi tingkat ekonomi. Hal ini akan memudahkan kita dalamn proses pengambilan keputusan kedepannya. Mari kita buat fungsinya di bawah ini:

[69]:
```
# Membuat fungsi untuk pengkategorian menjadi kelompok kelas
def income_class(total_income):

    if total_income <= 32000:
        return 'poor'
    elif total_income <= 53000:
        return 'lower middle class'
```

```
    elif total_income <= 106000:
        return 'middle class'
    elif total_income <= 373000:
        return 'upper middle class'
    else:
        return 'rich'
```

[70]:
```
# Membuat kolom baru dengan kategori
df['economic_class'] = df['total_income'].apply(income_class)
```

[71]:
```
# Menghitung distribusi
df['economic_class'].value_counts()
```

[71]:
```
poor                 16387
lower middle class    3985
middle class          1000
upper middle class      80
Name: economic_class, dtype: int64
```

2.2 Memeriksa Hipotesis

Apakah terdapat korelasi antara memiliki anak dengan membayar kembali tepat waktu?

[72]:
```
# Memeriksa apakah kolom 'children' berpengaruh terhadap 'debt'
pivot_table_children = df.pivot_table(index='children', columns= 'debt',␣
 ↪values='days_employed', aggfunc='count')
pivot_table_children
```

[72]:
```
debt            0       1
children
0         13026.0  1063.0
1          4410.0   445.0
2          1926.0   202.0
3           303.0    27.0
4            37.0     4.0
5             9.0     NaN
```

[73]:
```
# Memeriksa persentase untuk mendapatkan konklusi
pivot_table_children['percent_1'] = pivot_table_children[1] /␣
 ↪(pivot_table_children[1] + pivot_table_children[0]) * 100
pivot_table_children
```

[73]:
```
debt            0       1  percent_1
children
0         13026.0  1063.0   7.544893
1          4410.0   445.0   9.165808
```

```
2          1926.0   202.0   9.492481
3           303.0    27.0   8.181818
4            37.0     4.0   9.756098
5             9.0     NaN        NaN
```

Kesimpulan

Dari data di atas kita temukan bahwa: 1. Klien yang memliki 1 sampai 4 anak memiliki persentase yang hampir sama di angka 8% sampai 9%. 2. Klien yang memiliki 5 anak tidak memiliki hutang pembayaran pinjaman tatapi tidak bisa kita jadikan acuan karena jumlah data terlalu sedikit. 3. Klien yang **tidak** memilki anak memiliki rasio yang paling kecil untuk hutang pembayaran pinjaman sebesar 7% hal ini mungkin terjadi karena mereka memiliki tanggungan yang lebih sedikit dibandingkan dengan klien yang telah memilki anak.

Hal ini tentu akan memudahkan dalam pengambilan keputusan kita dalam memberikan kredit kepada klien yang belum memiliki anak karena kemampuan mereka dalam melunasi kredit mereka.

Apakah terdapat korelasi antara status keluarga dengan membayar kembali tepat waktu?

[74]:
```
# Memeriksa data 'family_status' memengaruhi kolom 'debt'
pivot_table_family_status = df.pivot_table(index='family_status', columns=␣
↪'debt', values='days_employed', aggfunc='count')
pivot_table_family_status
```

[74]:
```
debt                  0    1
family_status
civil partnership   3761  388
divorced            1110   85
married            11408  931
unmarried           2536  274
widow / widower      896   63
```

[75]:
```
# Menghitung rasio untuk mencari kseimpulan
pivot_table_family_status['percent_1'] = pivot_table_family_status[1] /␣
↪(pivot_table_family_status[1] + pivot_table_family_status[0]) * 100
pivot_table_family_status
```

[75]:
```
debt                  0    1  percent_1
family_status
civil partnership   3761  388   9.351651
divorced            1110   85   7.112971
married            11408  931   7.545182
unmarried           2536  274   9.750890
widow / widower      896   63   6.569343
```

Kesimpulan

Ada beberapa hal menarik yang kita temukan disini: 1. Klien yang `unmarried` dan `civil partnership` memiliki persentase yang cukup tinggi sebesar 9%. 2. Klien yang `divorced` dan

`married` memiliki rasio sebesar `7%` yang artinya lebih kecil daripada klien yang belum menikah apakah karena mereka dapat menggabungkan penghasilan dengan pasangan mereka, hal ini tentu berbanding terbalik dengan pengujian sebelumnya bahwa klien yang belum memiliki anak memiliki persentase hutang lebih kecil mengingat seseorang yang telah menikah punya kecenderungan untuk memiliki anak. 3. Klien dengan status `widow / widower` memiliki pensentase hutang paling kecil sebesar `6%`. Apakah kita akan mempertimbangkan status sebagai dasar dalam pengambilan keputusan?

Apakah terdapat korelasi antara tingkat pendapatan dengan membayar kembali tepat waktu?

[76]:
```
# Memeriksa apakah 'economic_class' memiliki korelasi dengan 'debt'
pivot_table_economic_class = df.pivot_table(index='economic_class', columns=␣
↪'debt', values='days_employed', aggfunc='count')
pivot_table_economic_class
```

[76]:
```
debt                    0     1
economic_class
lower middle class   3704   281
middle class          928    72
poor                15004  1383
upper middle class     75     5
```

[77]:
```
# Menghitung persentase distribusi untuk menarik kesimpulan
pivot_table_economic_class['percent_1'] = pivot_table_economic_class[1] /␣
↪(pivot_table_economic_class[1] + pivot_table_economic_class[0]) * 100
pivot_table_economic_class
```

[77]:
```
debt                    0     1  percent_1
economic_class
lower middle class   3704   281   7.051443
middle class          928    72   7.200000
poor                15004  1383   8.439617
upper middle class     75     5   6.250000
```

Kesimpulan

Beberapa hal yang kita temukan dari manipulasi data di atas diantaranya: 1. Klien dengan tingkat ekonomi `lower middle class` dan `middle class` memilki persentase yang seimbang yaitu sebesar `7%` untuk klien yang memiliki hutang pembayaran kredit. 2. Klien dengan tingkat ekonomi `poor` memiliki kemungkinan untuk menunggak lebih besar yaitu `8%`. Apakah ini dapat memengaruhi keputusan kita untuk tidak memberikan kredit mengingat jumlah mereka yang paling banyak. 3. Klien dengan pendapatan `upper middle class` yang memiliki risiko paling kecil sebesar `6%`.

Bagaimana tujuan kredit memengaruhi tarif otomatis?

[78]:
```
# Memeriksa persentase 'general_purpose' terhadap 'debt' untuk menggali konklusi
pivot_table_general_purpose = df.pivot_table(index='general_purpose', columns=␣
↪'debt', values='days_employed', aggfunc='count')
```

```
pivot_table_general_purpose['percent_1'] = pivot_table_general_purpose[1] /␣
 ↪(pivot_table_general_purpose[1] + pivot_table_general_purpose[0]) * 100
pivot_table_general_purpose
```

[78]:

debt	0	1	percent_1
general_purpose			
car	3903	403	9.359034
education	3643	370	9.220035
property	10028	782	7.234043
wedding	2137	186	8.006888

Kesimpulan

Berdasarkan penganganan yang kita lakukan hal-hal yang bisa kita dapatkan: 1. Klien yang melakukan pinjaman untuk keperluan `car` dan `education` memiliki risiko gagal bayar paling tinggi sebesar `9%`. 2. Klien yang menggunakan pinjamannya untuk `wedding` memiliki persentase hutang pinjaman yang lebih rendah dibandigkan kriteria sebelumnya. 3. Klien yang membayar tepat waktu yaitu untuk kegunaan `property` dengan risiko hutang tak lancar sebesar `7%`.

3 Kesimpulan Umum

Kita telah melakukan proses *cleansing data* untuk memperbaiki data-data yang bermasalah dalam dataset kita. Pembersihan yang kita lakukan meliputi mengisi value yang hilang, menghapus nilai duplikat, memperbaiki register yang tak beraturan, nilai yang terlalu besar, hingga mengganti nilai yang tidak wajar, sehingga kita mendapati dataset yang dapat kita olah untuk proses analisa kredit.

Temuan yang kita dapatkan setelah melakukan beberapa eksplorasi kita mendapati bahwa terdapat korelasi antara jumlah anak dan status perkawinan dalam risiko pemayaran kredit, klien yang tidak memiliki anak akan lebih mudah dalam melunasi hutangnya dibandingkan dengan klien yang memiliki anak. Klien yang menikah atau pernah memiliki pasangan memiliki risiko lebih rendah gagal bayar daripada klien dengan status *single* maupun tinggal bersama. Klien yang memiliki penghasilan lebih rendah akan lebih tinggi untuk memiliki hutang pinjaman, dan klien yang menggunakan uangnya untuk keperluan rumah akan lebih besar persentase mereka untuk dapat melunasi hutangnya.

Tetapi apakah semua manipulasi data yang kita lakukan dapat kita gunakan dalam proses *decision making* sehingga akan meminimalisir risiko yang akan terjadi di kemudian hari?