titanic-part1

April 4, 2025

0.1 Inżynieria cech titanic

0.1.1 Roksana Jandura grupa 4 IiAD

[23]:
```
import pandas as pd
import numpy as np
```

Dodaje na_values='?' ponieważ tak są zapisane NaN w tym pliku

[24]:
```
df = pd.read_csv("Zbiór danych Titanic.csv",na_values='?')
```

[25]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1309 entries, 0 to 1308
Data columns (total 14 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   'pclass'     1309 non-null   int64
 1   'survived'   1309 non-null   int64
 2   'name'       1309 non-null   object
 3   'sex'        1309 non-null   object
 4   'age'        1046 non-null   float64
 5   'sibsp'      1309 non-null   int64
 6   'parch'      1309 non-null   int64
 7   'ticket'     1309 non-null   object
 8   'fare'       1308 non-null   float64
 9   'cabin'      295 non-null    object
 10  'embarked'   1307 non-null   object
 11  'boat'       486 non-null    object
 12  'body'       121 non-null    float64
 13  'home.dest'  745 non-null    object
dtypes: float64(3), int64(4), object(7)
memory usage: 143.3+ KB
```

Opis zmiennych

pclass - klasa, odzwierciedla status ekonomiczny podróżującego

survived - czy przeżył/a katastrofę (1 = przeżył, 0 = nie przeżył)

name - imie nazwisko

sex - płeć

age - wiek

sibsp - liczba rodzeństwa/małżonków

parch - liczba dzieci/rodziców

ticket - numer biletu

fare - opłata za bilet

cabin - kabina

embarked - port z którego wypłynął

boat - numer łodzi, którą pasażer opuścił statek, jeśli przeżył

body - numer identyfikacyjny ciała, jeśli pasażer zginął i jego ciało zostało odnalezione

home.dest - cel podróży

Zmienne numeryczne:'pclass', 'survived', 'age', 'sibsp', 'parch', 'fare', 'body'

Zmienne kategoryczne:'name', 'sex', 'ticket', 'cabin', 'embarked', 'boat', 'home.dest'

[26]:
```
print(df.head(20)) #podglądam 20 pierwszych wierszy
```

```
    'pclass'  'survived'                                             'name'  \
0          1           1                    Allen, Miss. Elisabeth Walton
1          1           1                   Allison, Master. Hudson Trevor
2          1           0                     Allison, Miss. Helen Loraine
3          1           0             Allison, Mr. Hudson Joshua Creighton
4          1           0  Allison, Mrs. Hudson J C (Bessie Waldo Daniels)
5          1           1                               Anderson, Mr. Harry
6          1           1                Andrews, Miss. Kornelia Theodosia
7          1           0                             Andrews, Mr. Thomas Jr
8          1           1    Appleton, Mrs. Edward Dale (Charlotte Lamson)
9          1           0                           Artagaveytia, Mr. Ramon
10         1           0                            Astor, Col. John Jacob
11         1           1  Astor, Mrs. John Jacob (Madeleine Talmadge Force)
12         1           1                     Aubart, Mme. Leontine Pauline
13         1           1                      Barber, Miss. Ellen 'Nellie'
14         1           1             Barkworth, Mr. Algernon Henry Wilson
15         1           0                               Baumann, Mr. John D
16         1           0                          Baxter, Mr. Quigg Edmond
17         1           1  Baxter, Mrs. James (Helene DeLaudeniere Chaput)
18         1           1                            Bazzani, Miss. Albina
19         1           0                             Beattie, Mr. Thomson

     'sex'    'age'  'sibsp'  'parch'  'ticket'    'fare'  'cabin' 'embarked'  \
0   female  29.0000        0        0     24160  211.3375       B5          S
```

1	male	0.9167	1	2	113781	151.5500	C22 C26	S
2	female	2.0000	1	2	113781	151.5500	C22 C26	S
3	male	30.0000	1	2	113781	151.5500	C22 C26	S
4	female	25.0000	1	2	113781	151.5500	C22 C26	S
5	male	48.0000	0	0	19952	26.5500	E12	S
6	female	63.0000	1	0	13502	77.9583	D7	S
7	male	39.0000	0	0	112050	0.0000	A36	S
8	female	53.0000	2	0	11769	51.4792	C101	S
9	male	71.0000	0	0	PC 17609	49.5042	NaN	C
10	male	47.0000	1	0	PC 17757	227.5250	C62 C64	C
11	female	18.0000	1	0	PC 17757	227.5250	C62 C64	C
12	female	24.0000	0	0	PC 17477	69.3000	B35	C
13	female	26.0000	0	0	19877	78.8500	NaN	S
14	male	80.0000	0	0	27042	30.0000	A23	S
15	male	NaN	0	0	PC 17318	25.9250	NaN	S
16	male	24.0000	0	1	PC 17558	247.5208	B58 B60	C
17	female	50.0000	0	1	PC 17558	247.5208	B58 B60	C
18	female	32.0000	0	0	11813	76.2917	D15	C
19	male	36.0000	0	0	13050	75.2417	C6	C

	'boat'	'body'	'home.dest'
0	2	NaN	St Louis, MO
1	11	NaN	Montreal, PQ / Chesterville, ON
2	NaN	NaN	Montreal, PQ / Chesterville, ON
3	NaN	135.0	Montreal, PQ / Chesterville, ON
4	NaN	NaN	Montreal, PQ / Chesterville, ON
5	3	NaN	New York, NY
6	10	NaN	Hudson, NY
7	NaN	NaN	Belfast, NI
8	D	NaN	Bayside, Queens, NY
9	NaN	22.0	Montevideo, Uruguay
10	NaN	124.0	New York, NY
11	4	NaN	New York, NY
12	9	NaN	Paris, France
13	6	NaN	NaN
14	B	NaN	Hessle, Yorks
15	NaN	NaN	New York, NY
16	NaN	NaN	Montreal, PQ
17	6	NaN	Montreal, PQ
18	8	NaN	NaN
19	A	NaN	Winnipeg, MN

Przed użyciem poniższych funckji zadbałam, aby przy wczytywaniu za wartości NaN brało '?'

Mamy tutaj 14 cech (features), które charakteryzują danego pasażera statku.

[27]:
```
df.isnull().sum()
```

[27]:
```
'pclass'          0
'survived'        0
'name'            0
'sex'             0
'age'           263
'sibsp'           0
'parch'           0
'ticket'          0
'fare'            1
'cabin'        1014
'embarked'        2
'boat'          823
'body'         1188
'home.dest'     564
dtype: int64
```

[28]:
```
df.isnull().mean()
```

[28]:
```
'pclass'       0.000000
'survived'     0.000000
'name'         0.000000
'sex'          0.000000
'age'          0.200917
'sibsp'        0.000000
'parch'        0.000000
'ticket'       0.000000
'fare'         0.000764
'cabin'        0.774637
'embarked'     0.001528
'boat'         0.628724
'body'         0.907563
'home.dest'    0.430863
dtype: float64
```

Zmienne z wartościami brakującymi: age, fare, cabin, embarked, boat, body, home.dest

Zmienne, gdzie brakuje mnije niż 5% danych: fare, embarked. Prawdopodobnie są to dane MCAR(Missing Data Completely at Random), bo nie mają związku z żadnymi innymi zmiennymi w zestawie danych ani z samymi brakującymi danymi

Zmienne, gdzie brakuje więcej niż 5% danych: age (20%), cabin (77%), boat (63%), body (91%), home.dest (43%)

0.1.2 age (20%)

Sprawdzam rozkład braków w 'age' względem klasy, informacji czy przeżyła osoba, czy nie

[29]:
```
print("\nLiczba osób, z każdej klasy:")
print(df["'pclass'"].value_counts())
```

```
print("Braki w wieku względem klasy (pclass):")
print(df[df["'age'"].isna()]["'pclass'"].value_counts())
```

```
Liczba osób, z każdej klasy:
'pclass'
3    709
1    323
2    277
Name: count, dtype: int64
Braki w wieku względem klasy (pclass):
'pclass'
3    208
1     39
2     16
Name: count, dtype: int64
```

[30]:
```
print("\nProcent braków wieku (age) względem klasy (grupy społecznej):")
#wersja jedno-linijkowa
result_age_by_class  = df.assign(AgeNull=np.where(df["'age'"].isnull(), 1, 0)).
 ↪groupby(["'pclass'"])['AgeNull'].mean()
print(result_age_by_class)
```

```
Procent braków wieku (age) względem klasy (grupy społecznej):
'pclass'
1    0.120743
2    0.057762
3    0.293371
Name: AgeNull, dtype: float64
```

[31]:
```
print("\nLiczba osób, które przeżyły, nie przeżyły (survived):")
print(df["'survived'"].value_counts())
print("\nBraki w wieku względem przeżycia (survived):")
print(df[df["'age'"].isna()]["'survived'"].value_counts())
```

```
Liczba osób, które przeżyły, nie przeżyły (survived):
'survived'
0    809
1    500
Name: count, dtype: int64

Braki w wieku względem przeżycia (survived):
'survived'
0    190
1     73
Name: count, dtype: int64
```

[32]:
```
df['AgeNull'] = np.where(df["'age'"].isnull(), 1, 0)
result_age_by_life = df.groupby(["'survived'"])['AgeNull'].mean()
print("\nProcent braków wieku (age) w każdej grupie przeżycia (survived):")
print(result_age_by_life)
```

```
Procent braków wieku (age) w każdej grupie przeżycia (survived):
'survived'
0    0.234858
1    0.146000
Name: AgeNull, dtype: float64
```

[33]:
```
#Dodatkowo sprawdzam ile osób zginęło w każdej klasie, aby zobaczyć z której␣
 ↪grupy społecznej najwięcej osób straciło życie
survived_to_class = df[df["'survived'"] == 0]["'pclass'"].value_counts().
 ↪sort_index()
print("\nLiczba osób, które zginęły w każdej klasie:")
print(survived_to_class)
```

```
Liczba osób, które zginęły w każdej klasie:
'pclass'
1    123
2    158
3    528
Name: count, dtype: int64
```

Najwięcej braków w kolumnie age występuje wśród pasażerów 3. klasy – 208 przypadków na 709 osób, co stanowi około 29%. Sugeruje to, że pasażerowie tej klasy rzadziej mieli odnotowany wiek, prawdopodobnie ze względu na niższy status społeczny oraz mniej dokładną dokumentację.

Osoby, które nie przeżyły, częściej mają brakującą wartość age (190 z 809 osób – 23%) niż te, które przeżyły (73 z 500 osób – 14,6%). Pokrywa się to z faktem, że najwięcej ofiar pochodziło z 3. klasy, w której notowano najwięcej braków wieku.

Brakujące wartości w kolumnie age można zaklasyfikować jako MAR (Missing At Random), ponieważ ich występowanie częściowo zależy od innych zaobserwowanych zmiennych w zbiorze danych, takich jak pclass (klasa pasażera) oraz survived (czy pasażer przeżył). Nie są to dane losowo brakujące (MCAR), ani też braki wynikające bezpośrednio z wartości zmiennej age (MNAR). Występowanie braków jest raczej związane z kontekstem – sposobem traktowania, dokumentowania i rejestrowania pasażerów różnych klas.

0.1.3 cabin (77%)

[34]:
```
df['CabinNull'] = np.where(df["'cabin'"].isnull(), 1, 0)
result_cabin_by_class = df.groupby(["'pclass'"])['CabinNull'].mean()
print("Procent braków informacji o kabinie (cabin) w każdej klasie:")
print(result_cabin_by_class)
```

```
print("\nBraki w informacji o kabinie względem klasy:")
print(df[df["'cabin'"].isna()]["'pclass'"].value_counts())
```

```
Procent braków informacji o kabinie (cabin) w każdej klasie:
'pclass'
1    0.207430
2    0.916968
3    0.977433
Name: CabinNull, dtype: float64

Braki w informacji o kabinie względem klasy:
'pclass'
3    693
2    254
1     67
Name: count, dtype: int64
```

Najwięcej braków informacji o kabinie (cabin) występuje w 3. klasie – aż 97,7% pasażerów tej klasy (693 osoby) nie ma przypisanej kabiny. W 2. klasie braki występują u 91,7% osób (254 osoby), natomiast 1. klasa ma znacznie mniej braków – tylko 20,7% (67 osób).

Wskazuje to na silny związek między zmienną cabin a klasą (pclass) – brak informacji o kabinie jest zdecydowanie częstszy w niższych klasach, co wynika z faktu, że pasażerowie 2. i 3. klasy często nie mieli przydzielonych indywidualnych kajut lub ich kabiny nie były dokumentowane.

[35]:
```
print("Procent braków kabiny względem liczby rodzeństwa/małżonków (sibsp) i␣
↪klasy:")
print(df.groupby(["'pclass'","'sibsp'"])['CabinNull'].mean())

print("\nProcent braków kabiny względem liczby dzieci/rodziców (parch) i klasy:
↪")
print(df.groupby(["'pclass'","'parch'"])['CabinNull'].mean())
```

```
Procent braków kabiny względem liczby rodzeństwa/małżonków (sibsp) i klasy:
'pclass'  'sibsp'
1         0          0.287879
          1          0.079646
          2          0.125000
          3          0.000000
2         0          0.901099
          1          0.975610
          2          0.750000
          3          1.000000
3         0          0.976517
          1          0.967742
          2          1.000000
          3          1.000000
          4          1.000000
```

```
          5          1.000000
          8          1.000000
Name: CabinNull, dtype: float64

Procent braków kabiny względem liczby dzieci/rodziców (parch) i klasy:
'pclass'  'parch'
1         0          0.252066
          1          0.120000
          2          0.000000
          3          0.000000
          4          0.000000
2         0          0.922330
          1          0.883721
          2          0.960000
          3          0.666667
3         0          0.985560
          1          0.909091
          2          0.983607
          3          1.000000
          4          1.000000
          5          1.000000
          6          1.000000
          9          1.000000
Name: CabinNull, dtype: float64
```

Braki informacji o kabinie (cabin) są silnie zależne od klasy (pclass) oraz liczby towarzyszących osób (sibsp, parch).

W 1. klasie osoby podróżujące z rodziną miały znacznie mniejszy odsetek braków, co sugeruje lepszą dokumentację i przypisywanie wspólnych kabin.

W 2. i 3. klasie braki kabiny były bardzo częste (ponad 90%), niezależnie od tego, czy pasażerowie podróżowali samotnie, czy z rodziną.

Braki cabin są typu MAR (Missing At Random), ponieważ ich występowanie zależy od klasy i struktury rodziny pasażera.

0.1.4 boat (63%)

[36]:
```
df['BoatNull'] = np.where(df["'boat'"].isnull(), 1, 0)
print("\nProcent braków informacji o łodzi (boat) w każdej grupie przeżycia␣
  ↪(survived):")
print(df.groupby(["'survived'"])['BoatNull'].mean())
```

```
Procent braków informacji o łodzi (boat) w każdej grupie przeżycia (survived):
'survived'
0    0.988875
1    0.046000
Name: BoatNull, dtype: float64
```

[37]:
```
print("\nLiczba osób z przypisaną łodzią (boat), w podziale na przeżycie␣
  ↪(survived):")
print(df[df["'boat'"].notna()]["'survived'"].value_counts())
```

```
Liczba osób z przypisaną łodzią (boat), w podziale na przeżycie (survived):
'survived'
1    477
0      9
Name: count, dtype: int64
```

Braki w informacji o łodzi ratunkowej (boat) silnie zależą od tego, czy pasażer przeżył. Aż 98,9% osób, które zginęły, nie ma przypisanego numeru łodzi, podczas gdy wśród osób, które przeżyły, brak ten występuje tylko w 4,6% przypadków.

Z łącznej liczby 486 osób z przypisaną łodzią, aż 477 to osoby, które przeżyły, co wskazuje, że obecność numeru łodzi jest silnie związana z przeżyciem.

Braki te można uznać za dane typu MNAR (Missing Not at Random), ponieważ brak informacji o łodzi wynika bezpośrednio z faktu, że pasażer nie zdążył się do niej dostać i zginął. Innymi słowy, osoby, które nie przeżyły, niemal zawsze nie mają przypisanego numeru łodzi, ponieważ nie zostały ewakuowane – co stanowi mechanizm przyczynowo-skutkowy.

0.1.5 body (91%)

[38]:
```
df['BodyNull'] = np.where(df["'body'"].isnull(), 1, 0)
print("\nProcent braków informacji o ciele (body) w każdej grupie przeżycia␣
  ↪(survived):")
print(df.groupby(["'survived'"])['BodyNull'].mean())
```

```
Procent braków informacji o ciele (body) w każdej grupie przeżycia (survived):
'survived'
0    0.850433
1    1.000000
Name: BodyNull, dtype: float64
```

Brakuje aż 91% danych dotyczących numeru ciała (body). Braki te wynikają z faktu, że osoby, które przeżyły, nie mają przypisanego numeru ciała, a wiele ofiar nie zostało odnalezionych lub nie udało się ich zidentyfikować — co jest związane z tragicznym charakterem katastrofy.

85% osób, które zginęły, nie ma przypisanego numeru ciała, natomiast 100% osób, które przeżyły, ma braki w tej kolumnie. Wynika to z faktu, że numer ciała był przypisywany wyłącznie ofiarom, których ciała zostały odnalezione i zidentyfikowane — osoby, które przeżyły, nie mają podstaw, by taka informacja się pojawiła.

Braki te należy zaklasyfikować jako dane typu MNAR (Missing Not at Random), ponieważ brak informacji o numerze ciała nie jest przypadkowy — zależy bezpośrednio od faktu śmierci oraz możliwości odnalezienia i identyfikacji ofiary. Brak ten jest zatem powiązany z wartością zmiennej i wynika z konkretnego mechanizmu przyczynowo-skutkowego.

[39]:
```
df_filtered = df[(df["'survived'"] == 1) & (df["'body'"].notna())]
print(df_filtered[["'survived'", "'body'"]])
```

```
Empty DataFrame
Columns: ['survived', 'body']
Index: []
```

Potwierdza to, że nie ma ani jednego rekordu, gdzie jednocześnie jest informacja, że ktoś przeżył i jest numer ciała.

0.1.6 home.dest (43%)

[40]:
```
df['HomeDestNull'] = np.where(df["'home.dest'"].isnull(), 1, 0)
print("\nProcent braków informacji o destynacji w każdej klasie:")
print(df.groupby(["'pclass'"])['HomeDestNull'].mean())
```

```
Procent braków informacji o destynacji w każdej klasie:
'pclass'
1    0.105263
2    0.057762
3    0.724965
Name: HomeDestNull, dtype: float64
```

[41]:
```
print("\nProcent braków informacji o destynacji w każdej grupie przeżycia␣
  ↪(survived):")
print(df.groupby(["'survived'"])['HomeDestNull'].mean())
```

```
Procent braków informacji o destynacji w każdej grupie przeżycia (survived):
'survived'
0    0.508035
1    0.306000
Name: HomeDestNull, dtype: float64
```

Aż 72% pasażerów 3 klasy nie miało udokumentowanego miejsca docelowego, ponieważ pasażerowie niższych klas byli słabiej dokumentowani, a dane dotyczące ich miejsca docelowego często nie były rejestrowane lub zaginęły. Braki te są typu MAR (Missing At Random), ponieważ ich występowanie zależy od zmiennych takich jak klasa (pclass), a nie od samej wartości home.dest.

0.1.7 W jaki sposób należy postąpić z brakującymi wartościami?

Braki w zmiennej **age** można uzupełnić medianą, bo jest ona odporna na wartości odstające i będzie lepiej reprezentować typowy wiek pasażera.

Braki w zmiennej **cabin** nie należy uzupełniać na ślepo, wartościami średnimi, bo jest to zmienna tekstowa. Najlepiej stworzyć dodatkową kolumnę binarną i przypisac np. 1 jeśli osoba nie miała kabiny, a 0 jeśli miała kabinę.

Dla braków w zmiennej **boat** nie należy wstawiać innych wartości, bo brak sam w sobie niesie nam informacje. Uważam, że najlepiej byłoby dodać nową kolumnę, gdzie 1 oznaczałoby brak przypisania łodzi (osoba zginęła- nie zdążyła się ewakuować), a 0 łódź przypisana.

Dla zmiennej **body** można by utworzyć zmienną trójkategorialną body_status, która lepiej odzwierciedla rzeczywistość:

1-„żyje, brak ciała" – osoba przeżyła, więc nie przypisano jej numeru ciała,

2-„zmarł, brak ciała" – osoba zginęła, ale jej ciało nie zostało odnalezione,

3-„zmarł, ciało odnalezione" – osoba zginęła, a jej ciało zostało zidentyfikowane.

Dzięki temu podejściu brak danych w kolumnie body nie jest traktowany jednolicie, lecz uwzględnia kontekst i pozwala lepiej analizować strukturę danych typu MNAR.

Braki w zmiennej **home_dest** należy uzupełnić wartością 'Unknown', reprezentującą brak wiedzy o miejscu docelowym.

Braki w zmiennej **fare** można uzupełnić za pomocą średniej lub mediany, w zależności od tego jaki jest rozkład.

Braki z zmiennej **embarked** można uzupełnić najczęstszą wartością.