22 marca 2020

Politechnika Wrocławska

# Pakiety statystyczne – las losowy

Wydział Matematyki Politechniki Wrocławskiej

Łukasz Łaszczuk

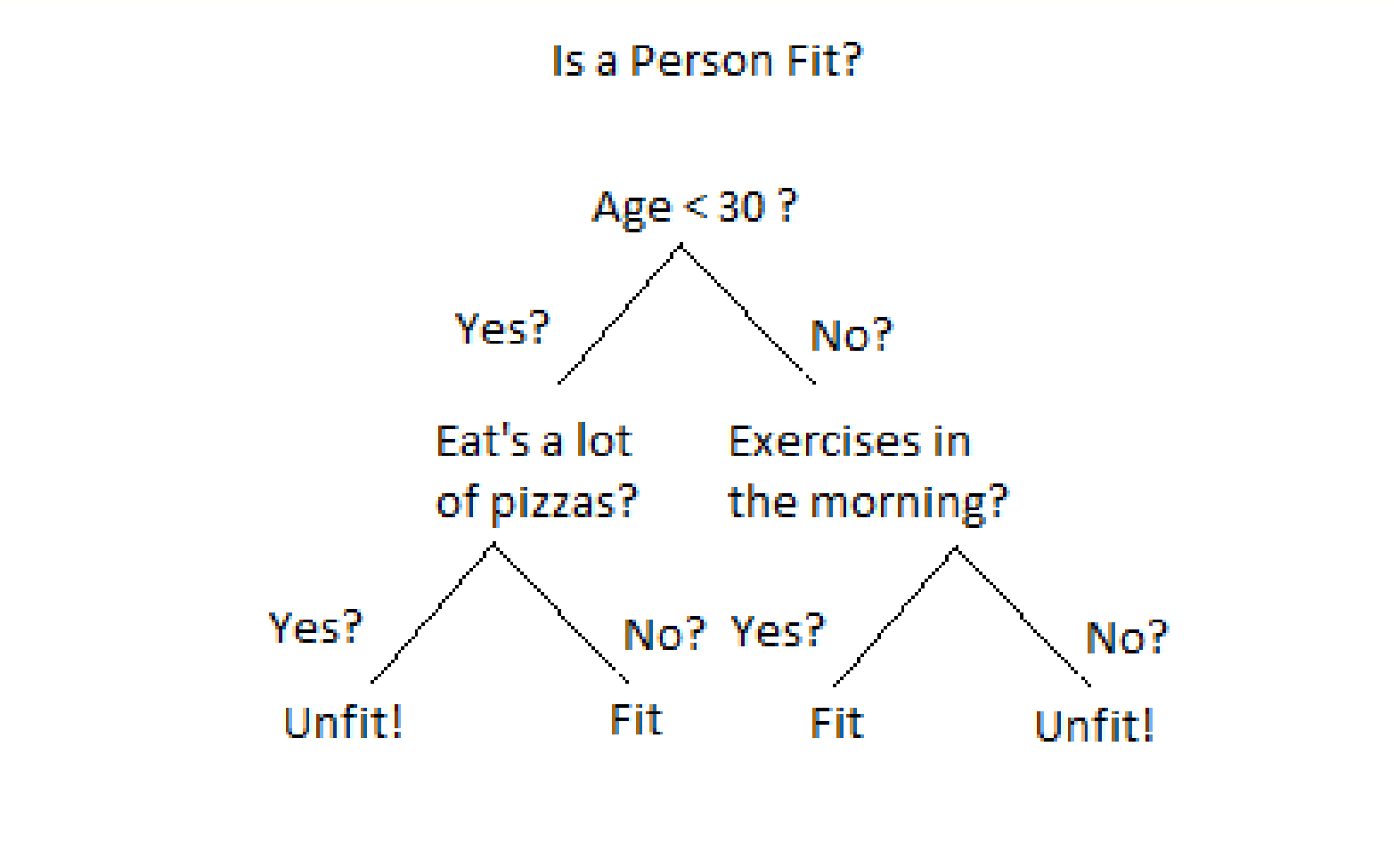
# Plan prezentacji

- Opis algorytmu
  - Co to są drzewa decyzyjne?
  - Drzewa decyzyjne w problemach klasyfikacyjnych i regresyjnych
  - W jaki sposób drzewa decyzyjne są wykorzystywane podczas tworzenia lasu losowego
- Modelowanie za pomocą lasu losowego
  - Eksploracyjna analiza danych
  - Omówienie metryk do ewaluacji
  - Budowa modelu, dobór hiperparametrów
  - Porównanie z innymi modelami

# Drzewa decyzyjne

- Drzewa decyzyjne same w sobie nie są skutecznym algorytmem, ale stanowią podstawę efektywnych modeli ensemble, np. lasów losowych, czy gradient boostingu;
- Intuicja: zbiór pytań (reguł eksperckich) oddzielających od siebie grupy (np. dobrych i złych klientów banku);
- Jest algorytmem glass-box (w łatwy sposób możemy zrozumieć jego działanie oraz istotne zmienne - im wcześniej pojawi się w drzewie, tym jest istotniejsza).

# Drzewa decyzyjne

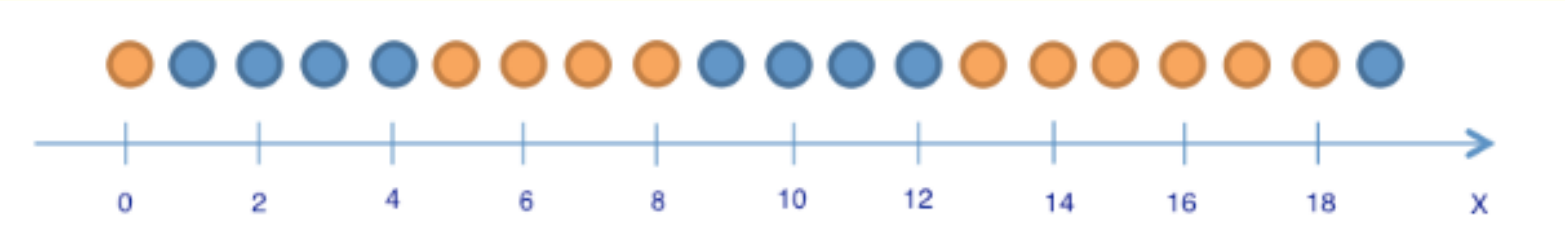



https://www.xoriant.com/blog/product-engineering/decision-trees-machine-learning-algorithm.html

# Przydatne pojęcia - entropia Shannona

Intuicja: miara "chaosu" naszego zbioru.

Entropia Shannona: $S = -\sum_{i=1}^{N} p_i \log_2 p_i$



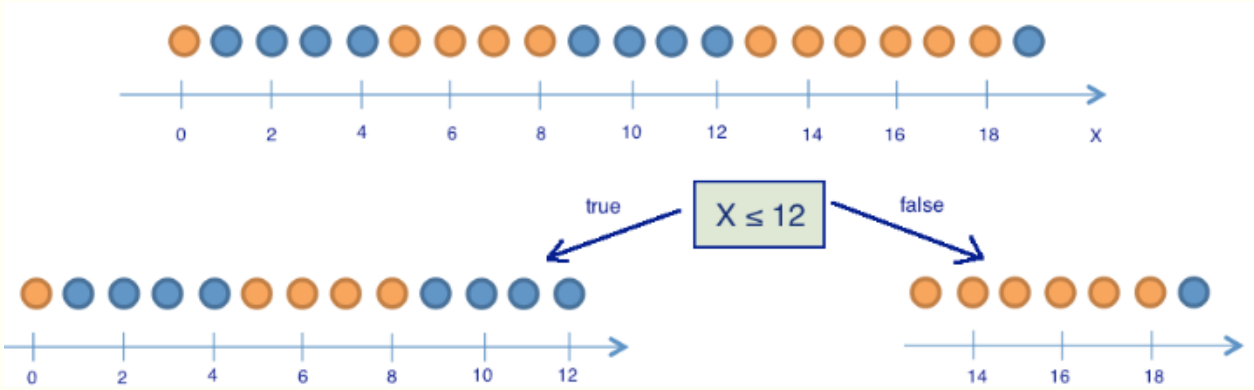

$S = -\frac{9}{20} \log_2 \frac{9}{20} - \frac{11}{20} \log_2 \frac{11}{20} \approx 1$

`https://mlcourse.ai/articles/topic3-dt-knn/`

# Przydatne pojęcia - przyrost informacji

Intuicja: miara zmniejszenia "chaosu" po dokonaniu podziału danych (stworzenia rozgałęzienia). Chcemy ją maksymalizować przy kolejnych podziałach.

Przyrost informacji (Information Gain): $IG = S_0 - \sum_{i=1}^{q} \frac{N_i}{N} S_i$



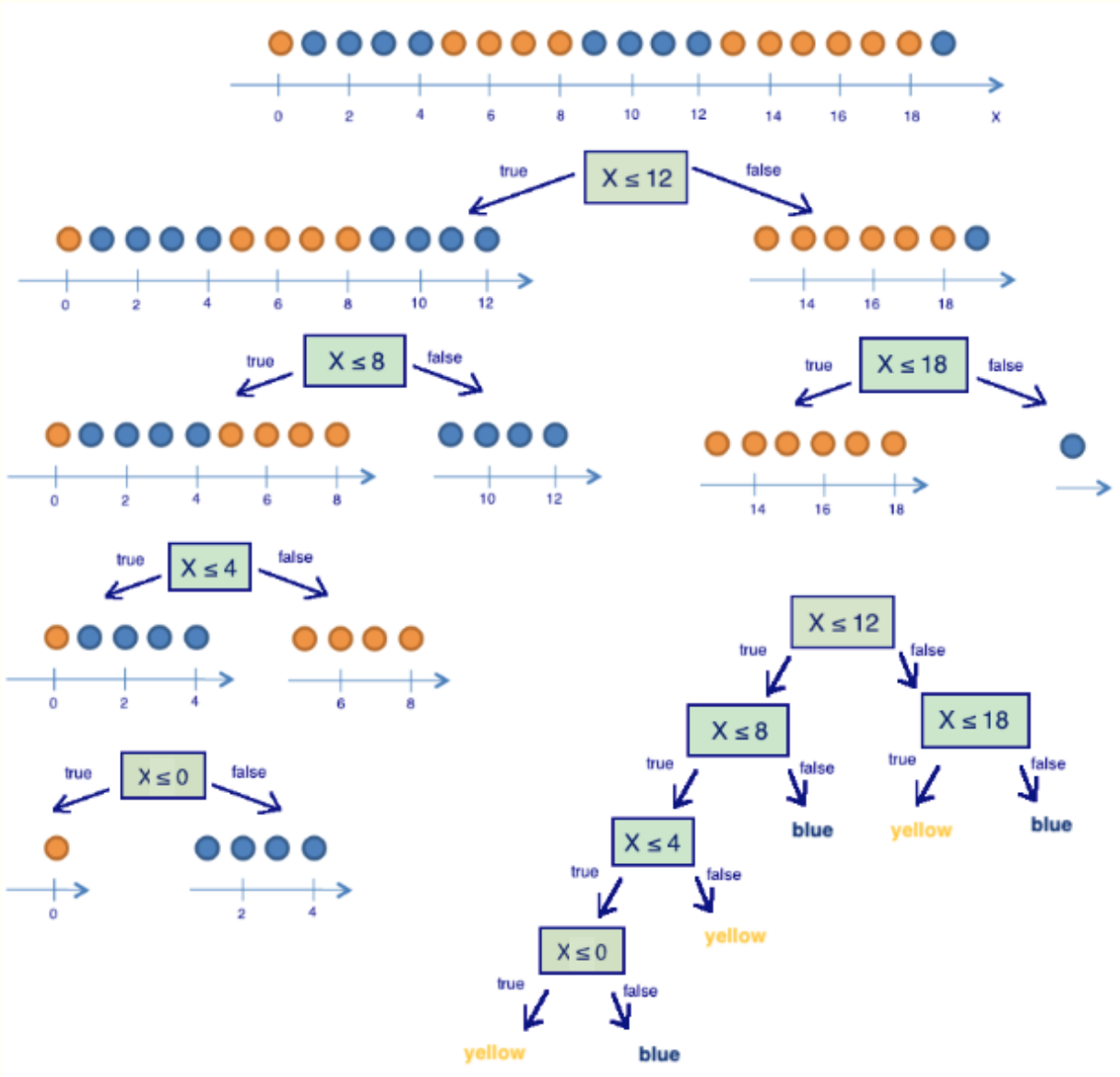

$$IG = S_0 - \frac{13}{20} S_1 - \frac{7}{20} S_2 \approx 0,16$$

`https://mlcourse.ai/articles/topic3-dt-knn/`

# Kroki algorytmu

1. Drzewo zaczyna od pojedynczego węzła reprezentującego cały zbiór treningowy;
2. Jeżeli wszystkie przykłady należą do jednej klasy decyzyjnej, to zbadany węzeł staje się liściem i jest on etykietowany tą decyzją;
3. W przeciwnym przypadku algorytm wykorzystuje miarę entropii (funkcja przyrostu informacji) jako heurystyki do wyboru atrybutu, który najlepiej dzieli zbiór przykładów treningowych;
4. Dla każdego wyniku testu tworzy się jedno odgałęzienie i przykłady treningowe są odpowiednio rozdzielone do nowych węzłów (poddrzew);
5. Algorytm działa dalej w rekurencyjny sposób dla zbiorów przykładów przydzielonych do poddrzew.;
6. Algorytm kończy się, gdy kryterium stopu jest spełnione.

# Cały algorytm - wizualizacja



# Drzewa decyzyjne - regresja

W problemie regresji sposób tworzenia drzew pozostaje taki sam, zmienia się jednak kryterium podziału (na wariancję):

$$D = \frac{1}{n} \sum_{i=1}^{n} \left( y_i - \frac{1}{n} \sum_{j=1}^{n} y_j \right)^2$$

Końcową predykcją jest średnia zmiennej objaśnianej w danym liściu.

# Problemy związane z drzewami decyzyjnymi

- Drzewa decyzyjne w łatwy sposób jest przeuczyć (nie potrafią generalizować swoich predykcji na zbiorze testowym) - duża wariancja;
- Można częściowo temu zapobiec poprzez wcześniejsze zatrzymanie algorytmu (ustawienie maksymalnej głębokości lub minimalnej ilości obserwacji, która musi znaleźć się w liściu)

# Twierdzenie Condorceta

- Cel: podjęcie decyzji (tak lub nie);
- Głosuje $N$ jurorów, końcowa decyzja zapada większością głosów;
- Jurorzy głosują niezależnie od siebie, każdy z nich podejmuje właściwą decyzję z prawdopodobieństwem $p$;
- Prawdopodobieństwo podjęcia właściwej decyzji końcowej: $\mu = \sum_{i=\frac{N}{2}+1}^{N} \binom{N}{i} p^i (1-p)^{N-i}$
- Jeśli $p > 0,5$, to przy $N \to \infty$, $\mu \to 1$

# Las losowy

Intuicja: Zbudowanie wielu "słabych" modeli (drzew), które po połączeniu (w przypadku lasu losowego po wyciągnięciu średniej) dadzą lepszy wynik niż pojedynczy model.

Problem: Sprawienie, aby decyzje kolejnych drzew były niezależne oraz jak najdokładniejszse;

# Lasy losowe - algorytm

- Z próby o wielkości $N$ (nasze dane) losujemy jednostajnie ze zwracaniem $N$ elementów, tworząc nową próbę (bootstrapping);
- Ze zbioru stworzoną metodą bootstrap, wybieramy losowo $m$ spośród $p$ zmiennych (kolumn; przyjmuje się domyślną wartość $m$ jako $\lfloor \sqrt{p} \rfloor$ (klasyfikacja) lub $\lfloor \frac{p}{3} \rfloor$ (regresja));
- Na próbie tej budujemy nasz podstawowy model (drzewo decyzyjne - model, który jest przetrenowany);
- Powtarzamy poprzednie kroki $B$ razy;
- Tworzymy końcowy model metodą głosowania większościowego (majority voting - klasyfikacja), bądź wyciągamy średnią (regresja) - dostajemy model lepszy niż drzewo decyzyjne (bagging).

# Obserwacje OOB (out of bag)

- Obserwacje OOB – obserwacje, które nie znalazły się w zbiorze, na którym budowaliśmy drzewo decyzyjne;
- Prawdopdobieństwo zdarzenia, że losowo wybrana obserwacja z próby o wielkości $l$ nie znajdzie się w zbiorze bootstrap: $(1 - \frac{1}{l})^l$;
- Gdy $l \to \infty$, prawdopodobieństwo to wynosi $\frac{1}{e} \approx 37\%$;
- Obserwacje OOB stanowią reprezentatywną próbę zbioru, możemy na nich testować nasz model (alternatywa do walidacji krzyżowej, ang. cross-validation).

# Zalety lasów losowych

- Jest skuteczniejszy od algorytmów liniowych oraz drzew decyzyjnych;
- Odporny na wartości odstające (dzięki baggingowi) oraz na wartości None;
- Działa skutecznie bez optymalizacji hiperparametrów (dzięki temu mamy dobry punkt odnosienia przy budowaniu innych modeli, np. xgboosta);
- Obserwacjom z różnych klas możemy przypisywać wagi (szczególnie przydatne przy niezbalansowanych zbiorach danych);
- Jest algorytmem, który się nie przeucza.

# Wady lasów losowych

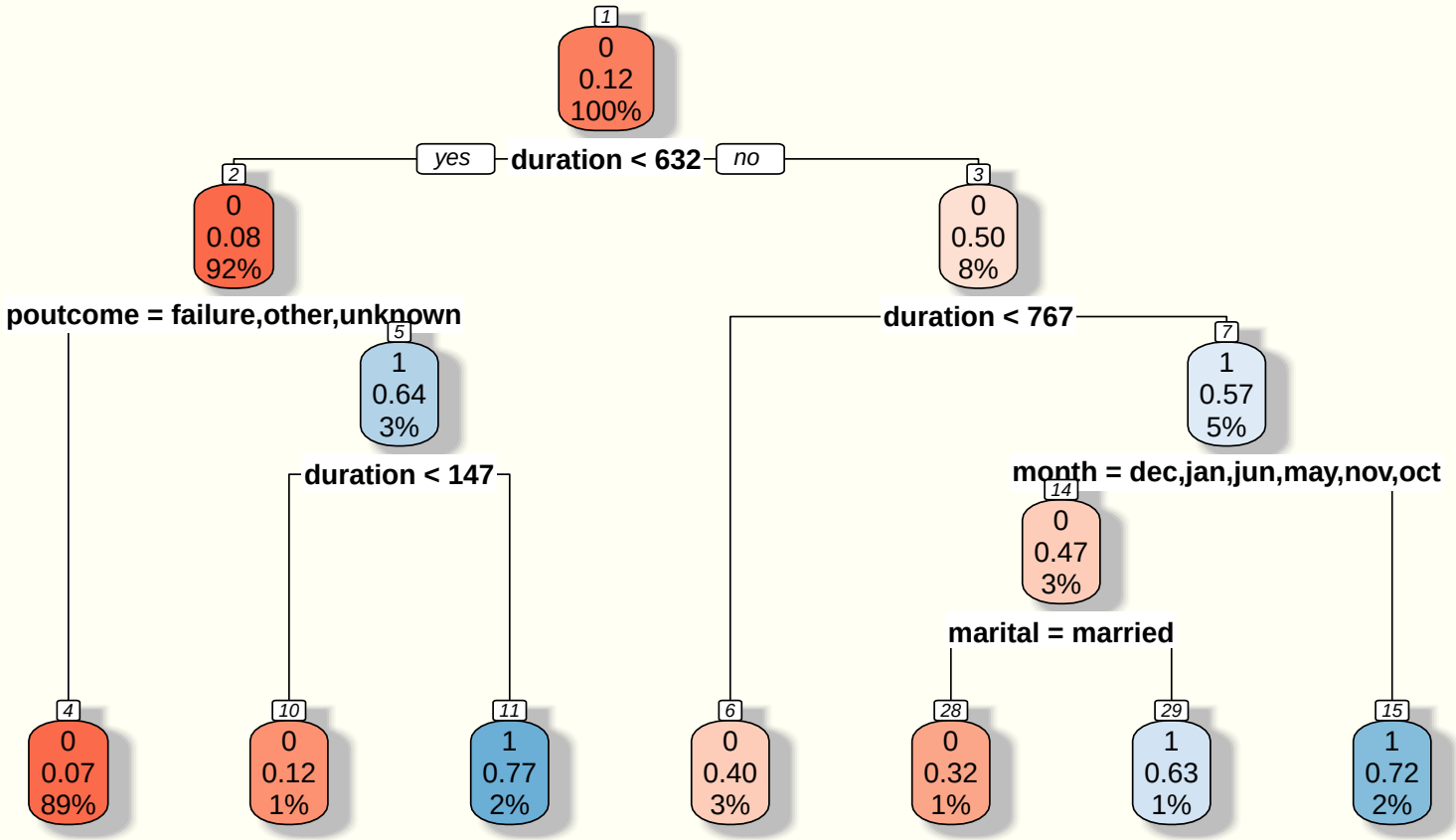
- Trudniejszy w interpretacji niż drzewo decyzyjne (istnieją jednak metody wyjaśniania);
- Nie radzi sobie z danymi, które są rzadkie (np. zastosowanie one hot encodingu zmiennych kategorycznych może pogorszyć rezultaty) oraz które mają bardzo dużą ilość zmiennych;
- W przeciwieństwie do modeli liniowych ekstrapolacja nie jest możliwa;
- Modele lasów losowych potrzebują więcej pamięci niż modele xgboost (ponieważ trenują głębsze drzewa decyzyjne).

# Przedstawienie danych

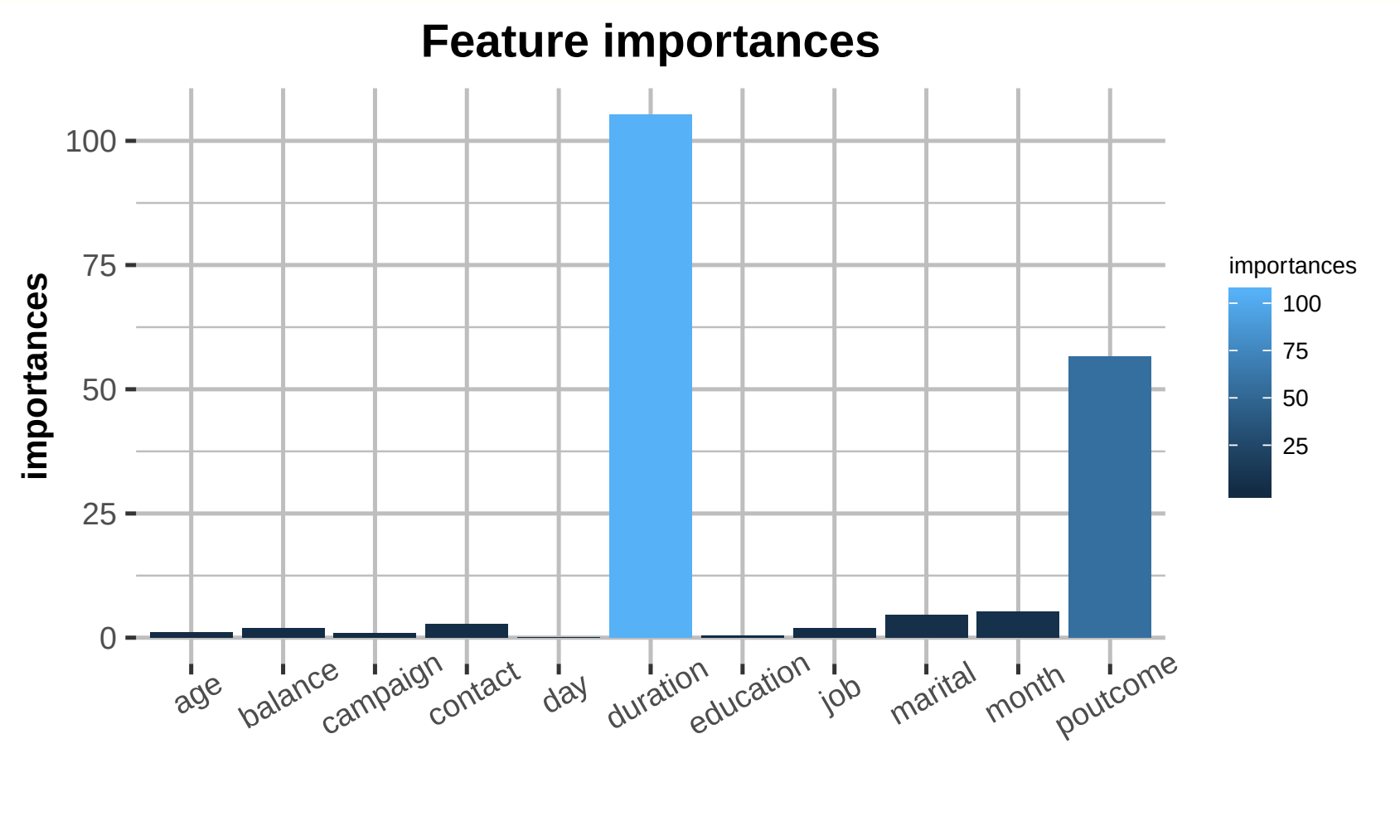
- Bank Marketing Data Set;
- 17 kolumn (9 kategorycznych, 7 numerycznych oraz binarna zmienna objaśniana)
- 4521 obserwacji;
- Zmienna objaśniana (y) zawiera informację, czy klient skorzystał z oferty banku (lokaty terminowej).

# Pierwszy model - drzewo decyzyjne

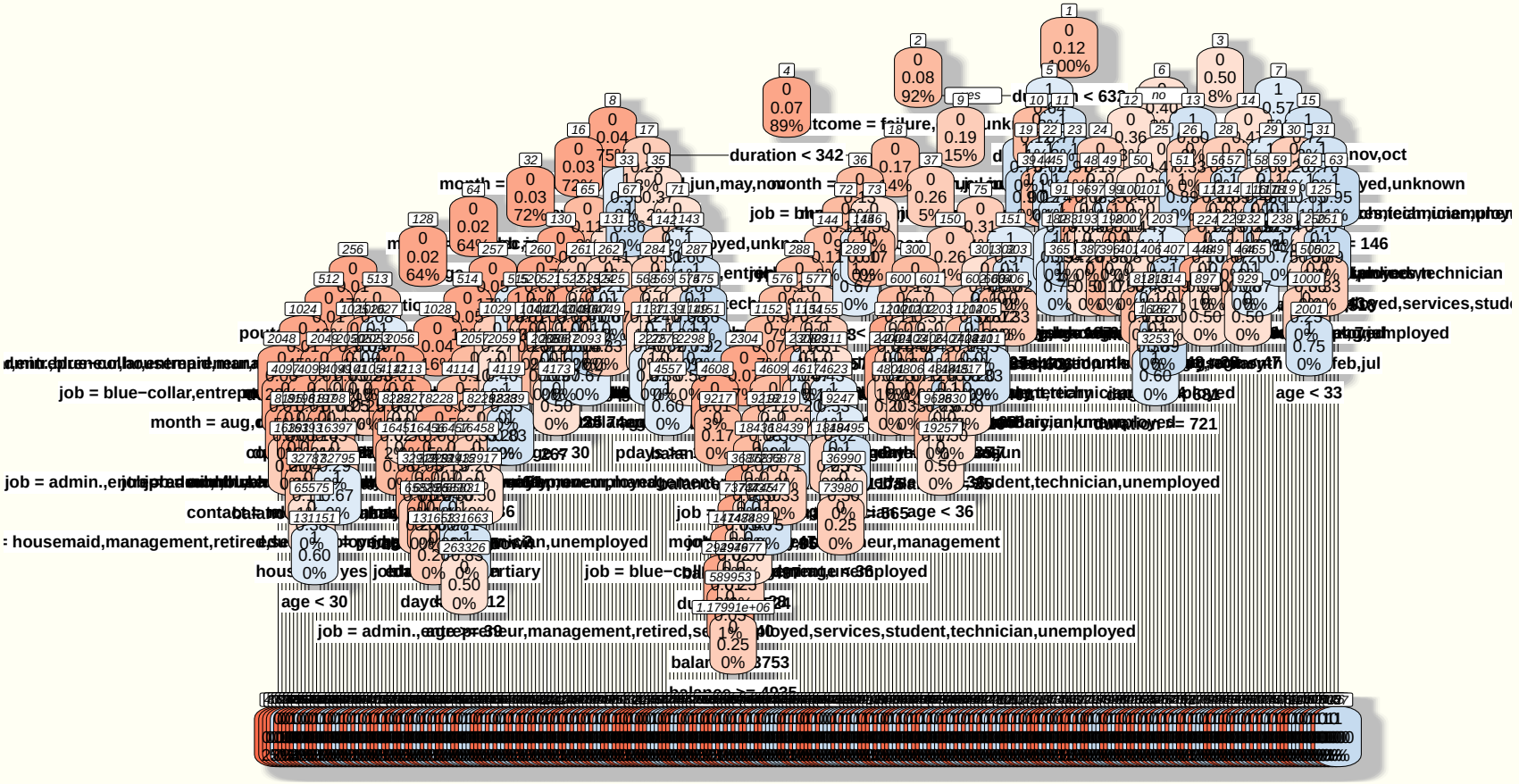
```
tree <- rpart(y ~ ., data = df_train, cp = 0.02)
rpart.plot(tree, box.palette = "RdBu",
           shadow.col = "gray", nn = TRUE)
```



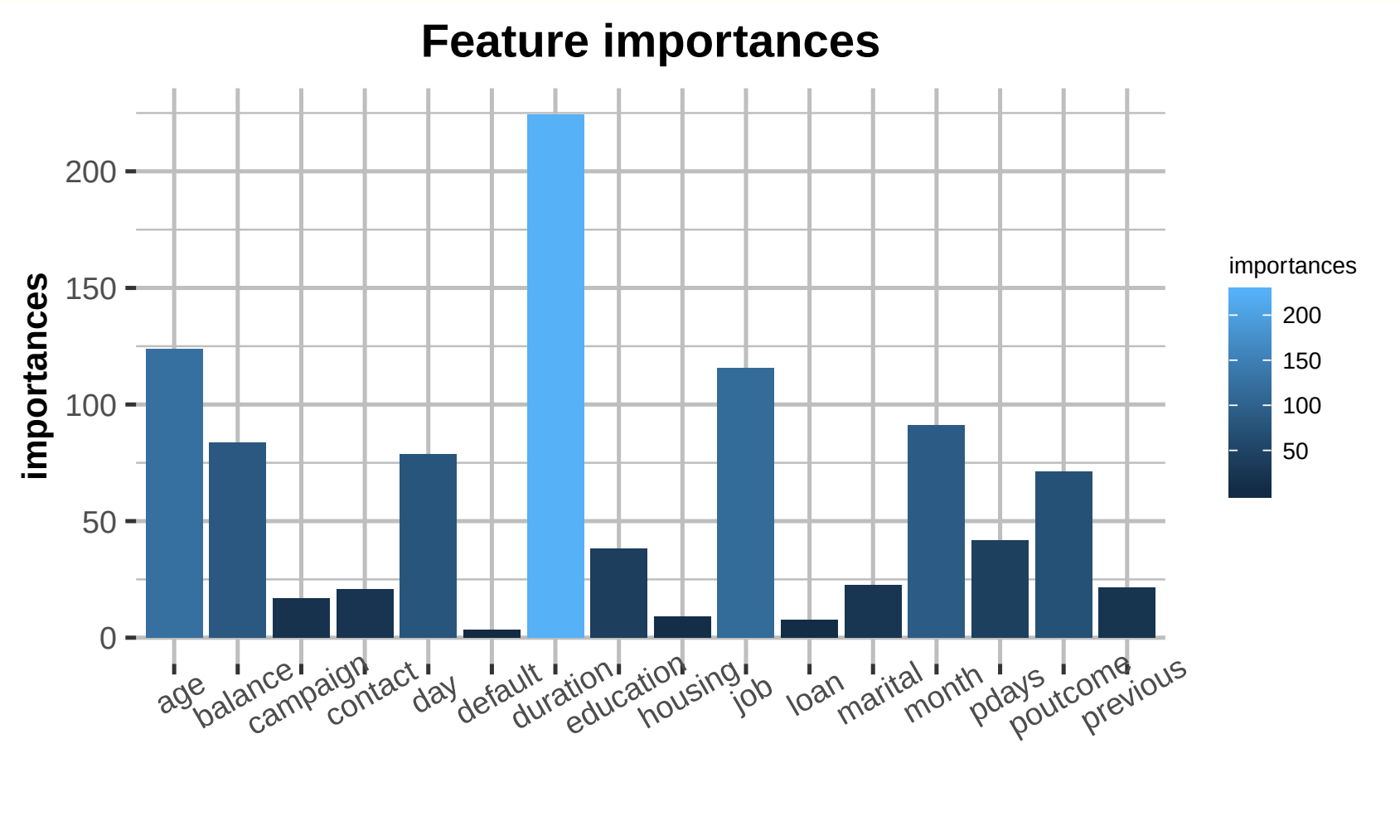
# Pierwszy model - drzewo decyzyjne



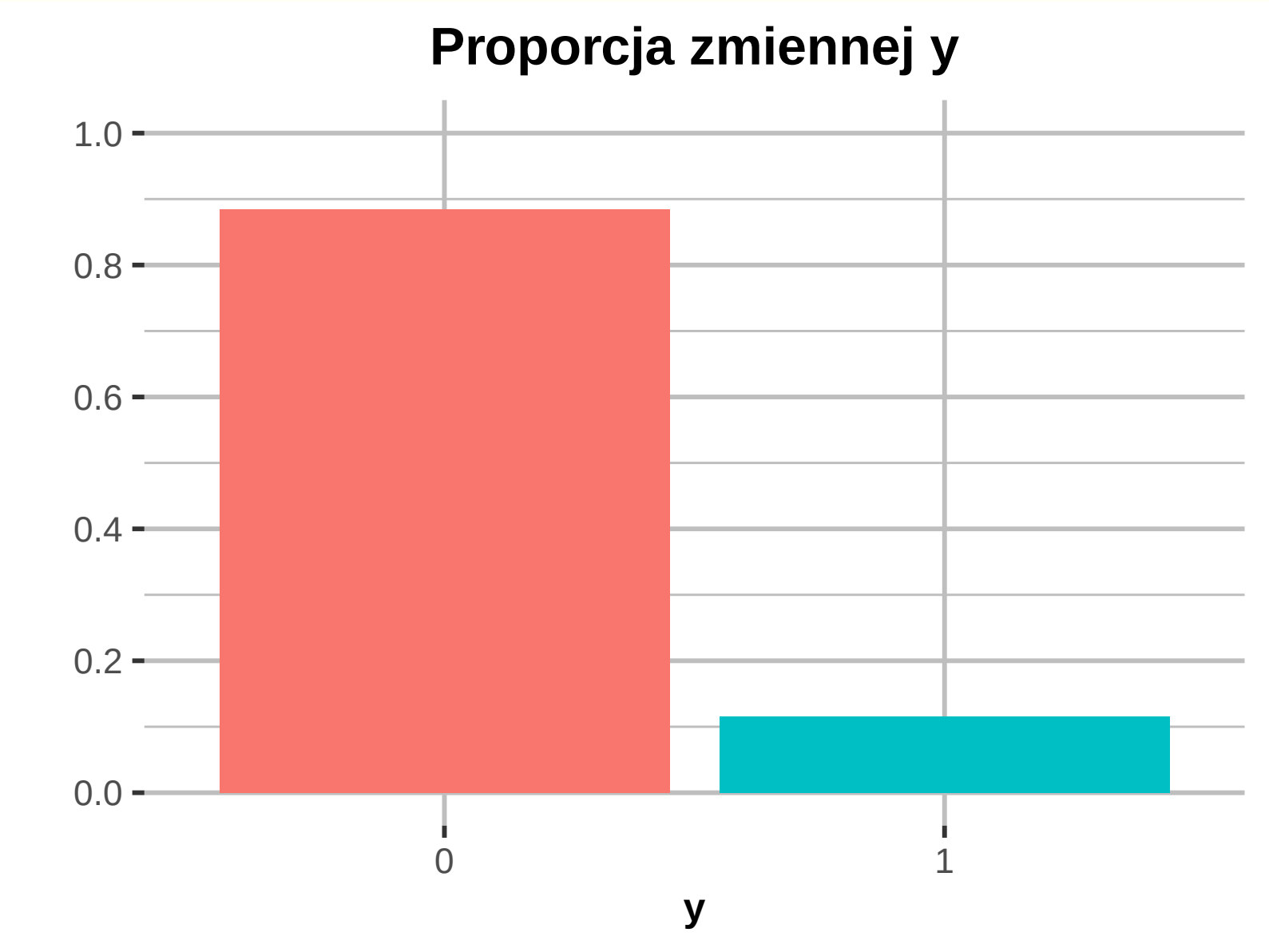
# Przeuczone drzewo :D



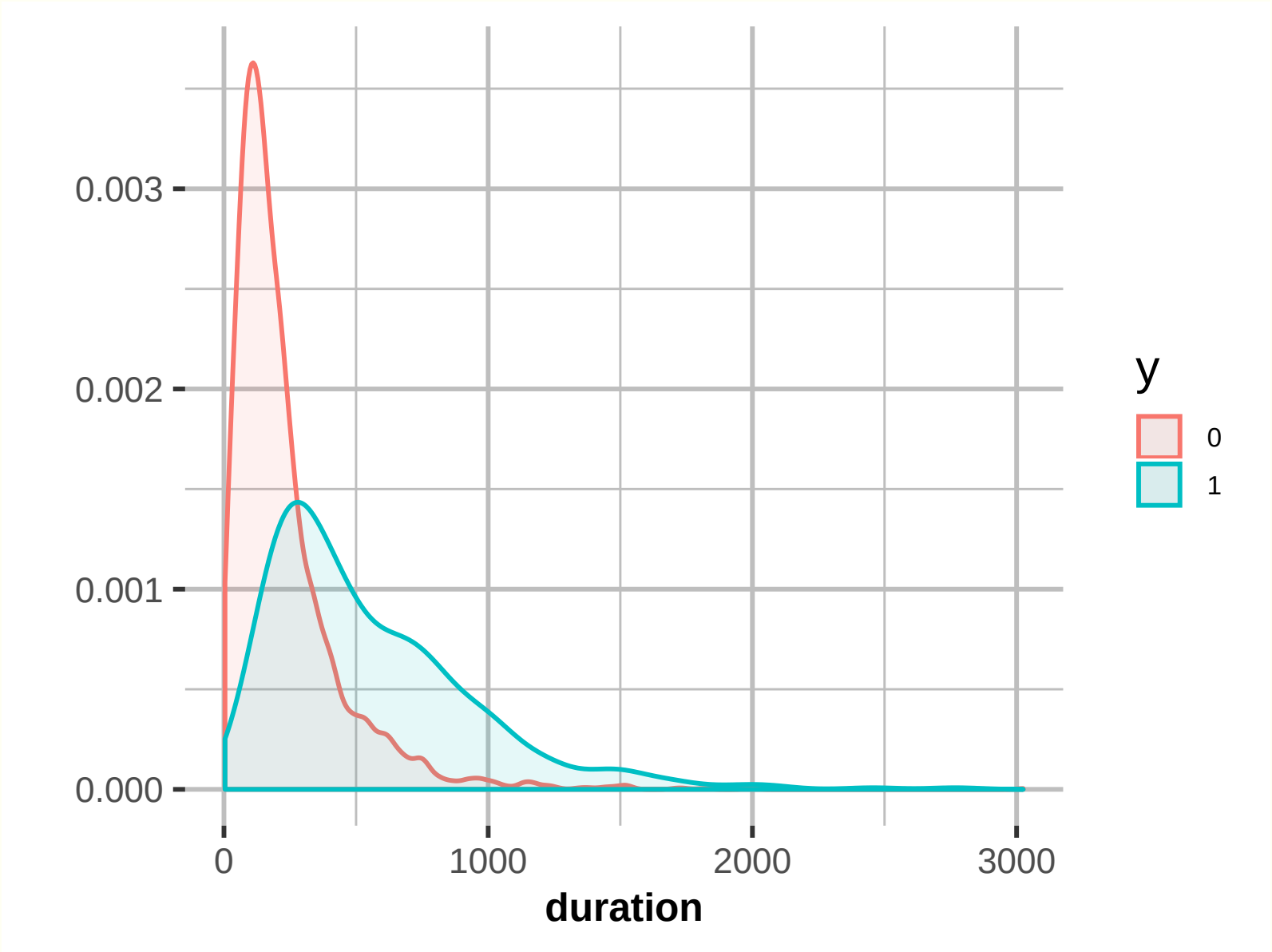
# Pierwszy model - drzewo decyzyjne



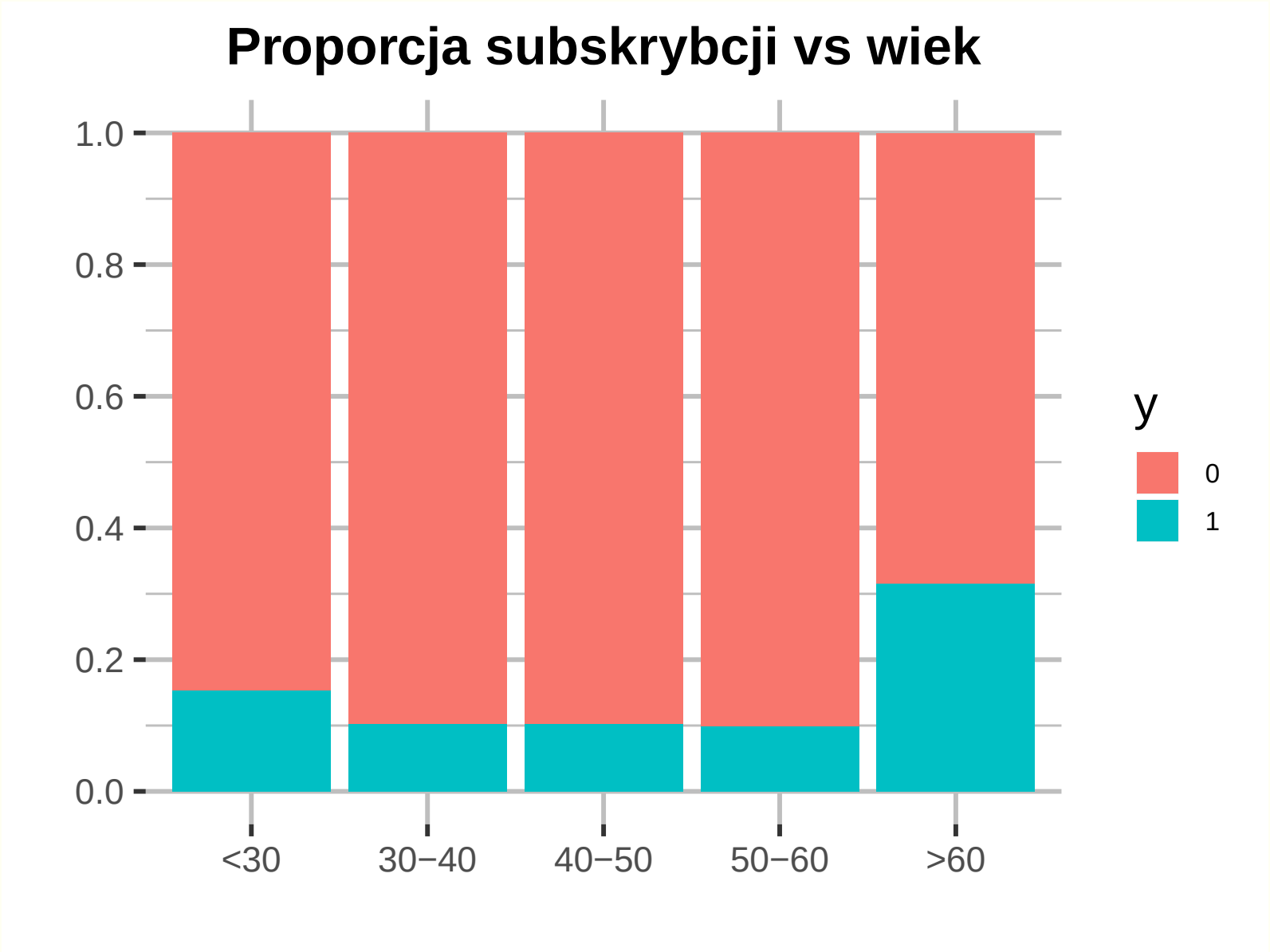
# EDA



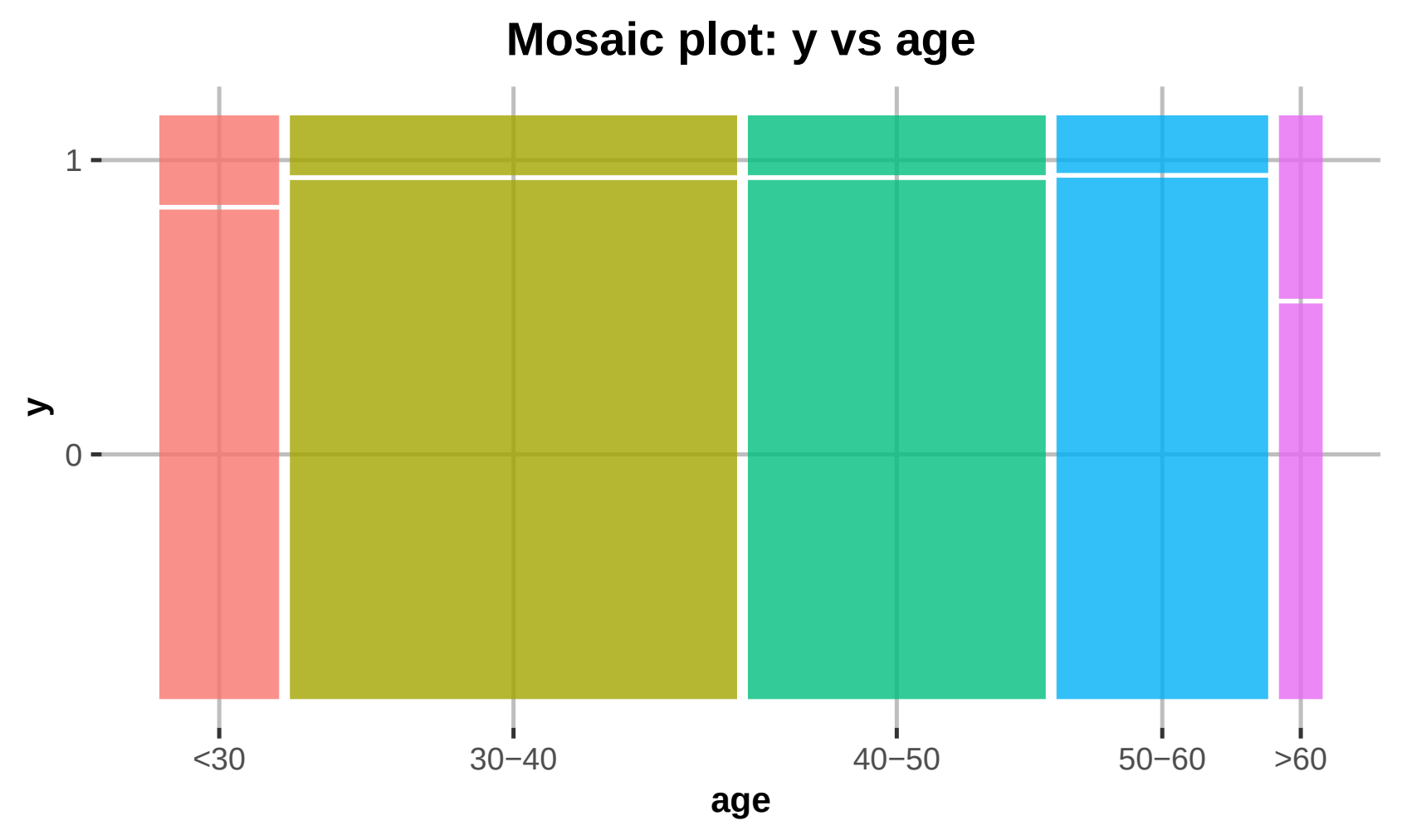
# EDA



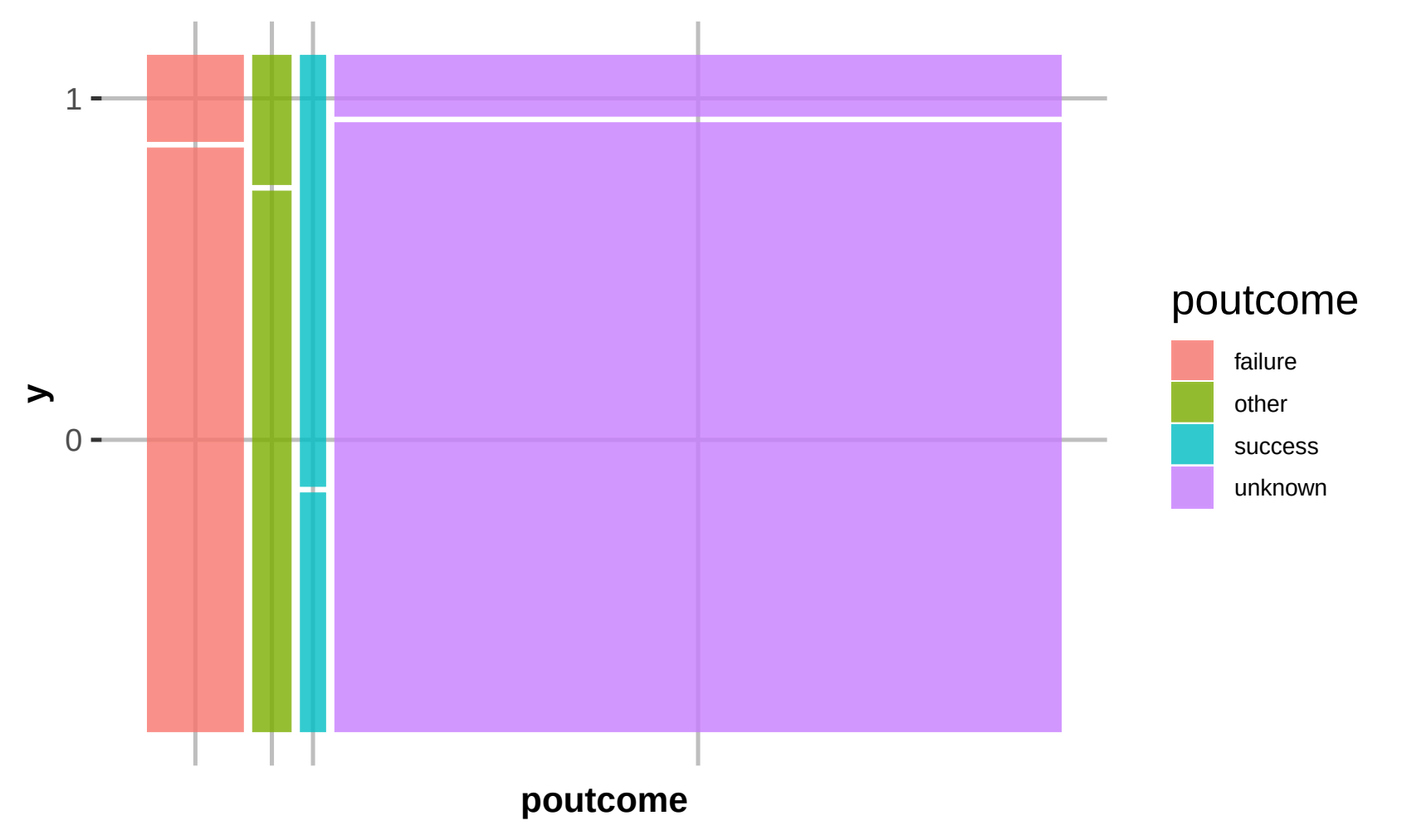
# Czemu mosaic plot?



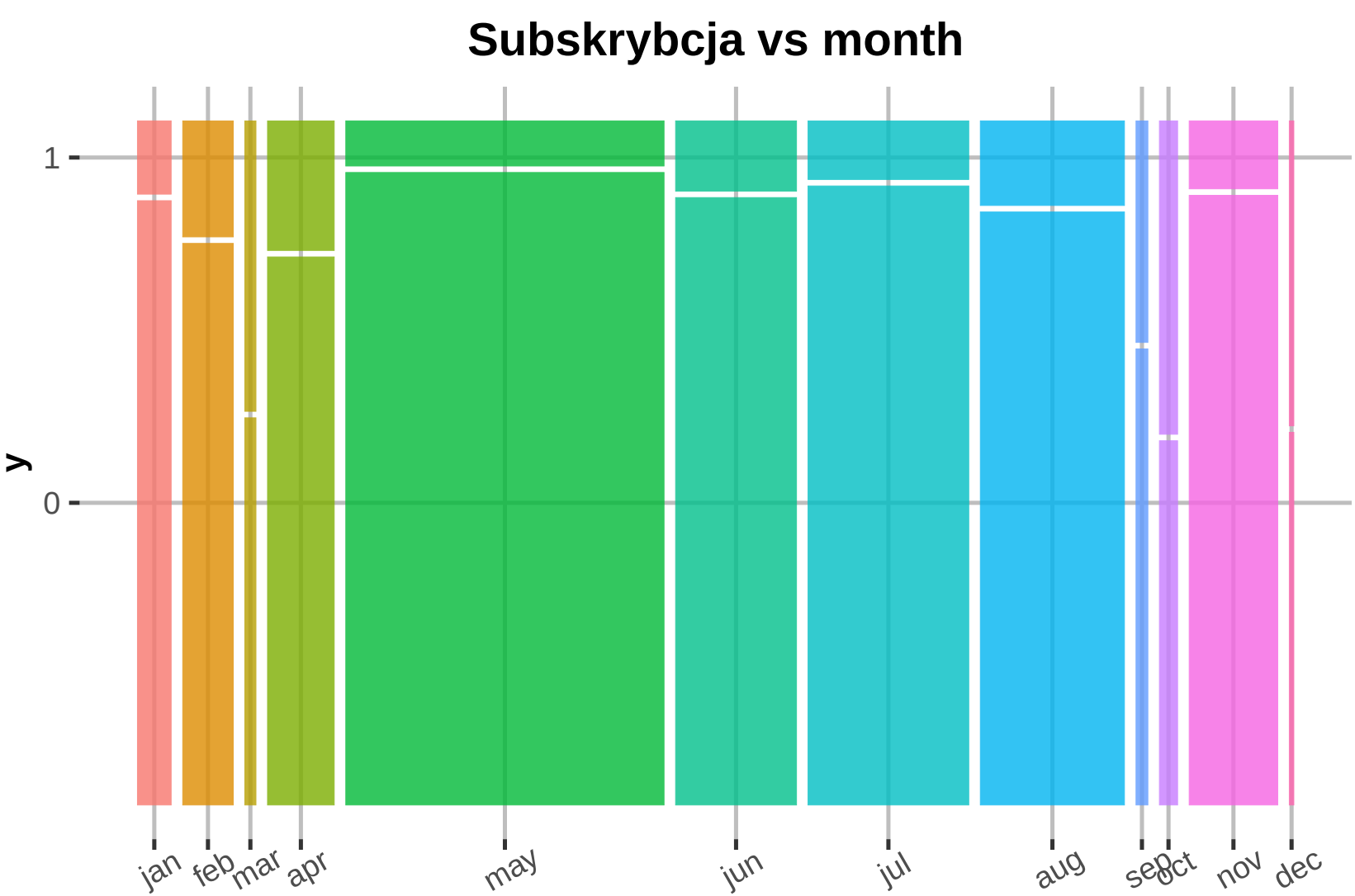
# Czemu mosaic plot?



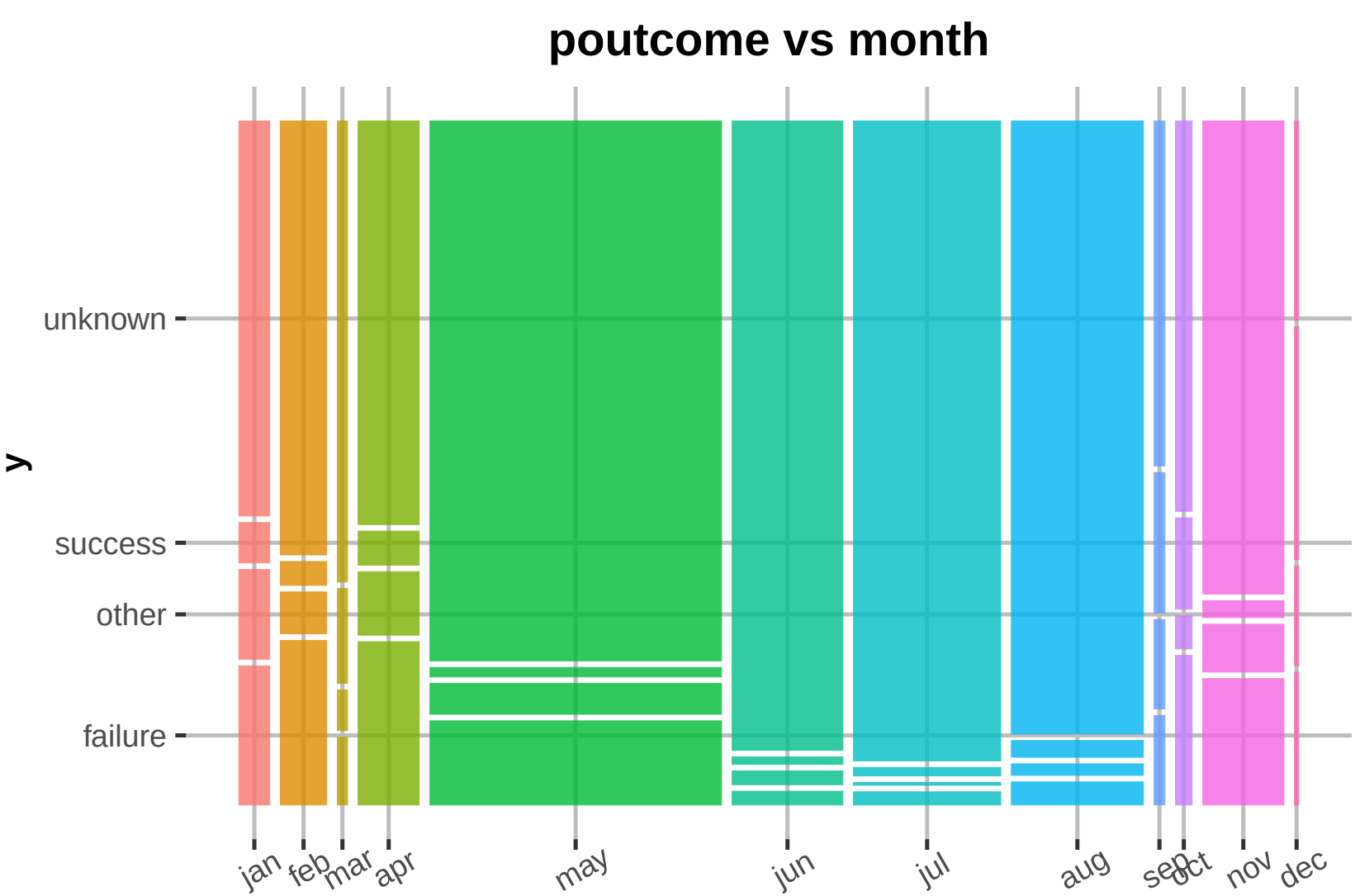
# EDA



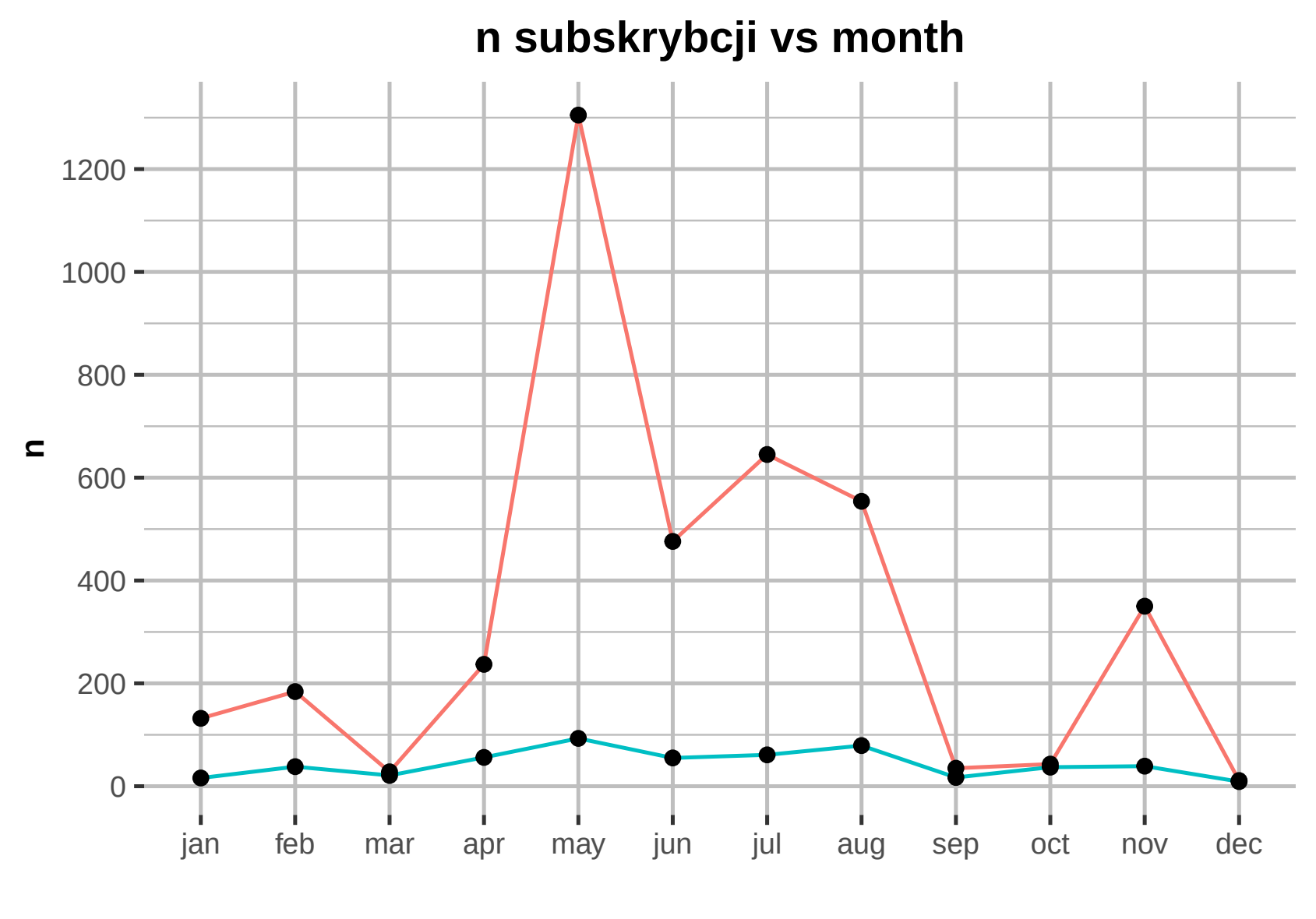
# EDA


Subskrybcja vs month
y
1
0
jan
feb
mar
apr
may
jun
jul
aug
sep
oct
nov
dec

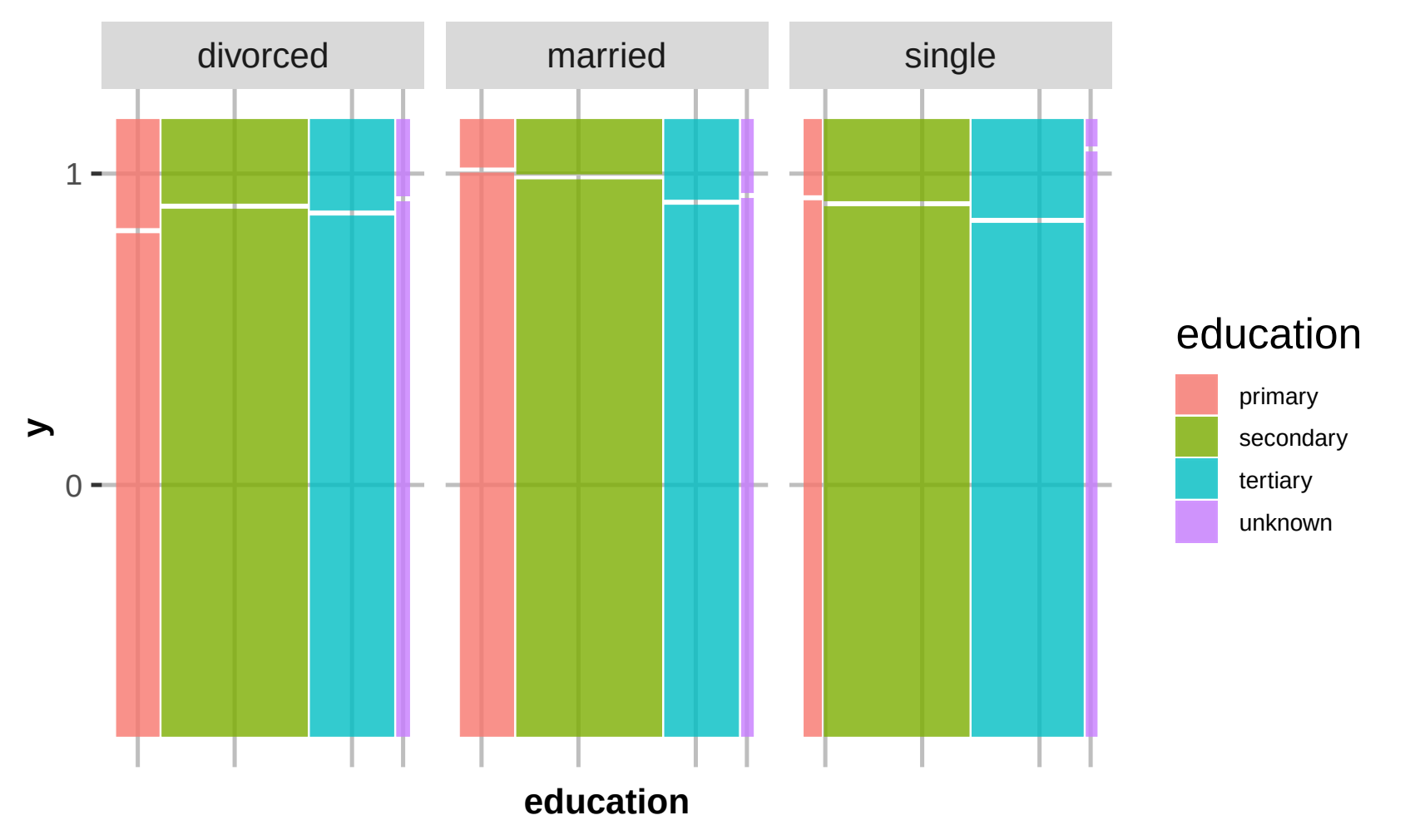
# EDA


poutcome vs month
y
unknown
success
other
failure
jan
feb
mar
apr
may
jun
jul
aug
sep
oct
nov
dec

# EDA


n subskrybcji vs month
n
1200
1000
800
600
400
200
0
jan
feb
mar
apr
may
jun
jul
aug
sep
oct
nov
dec

# EDA

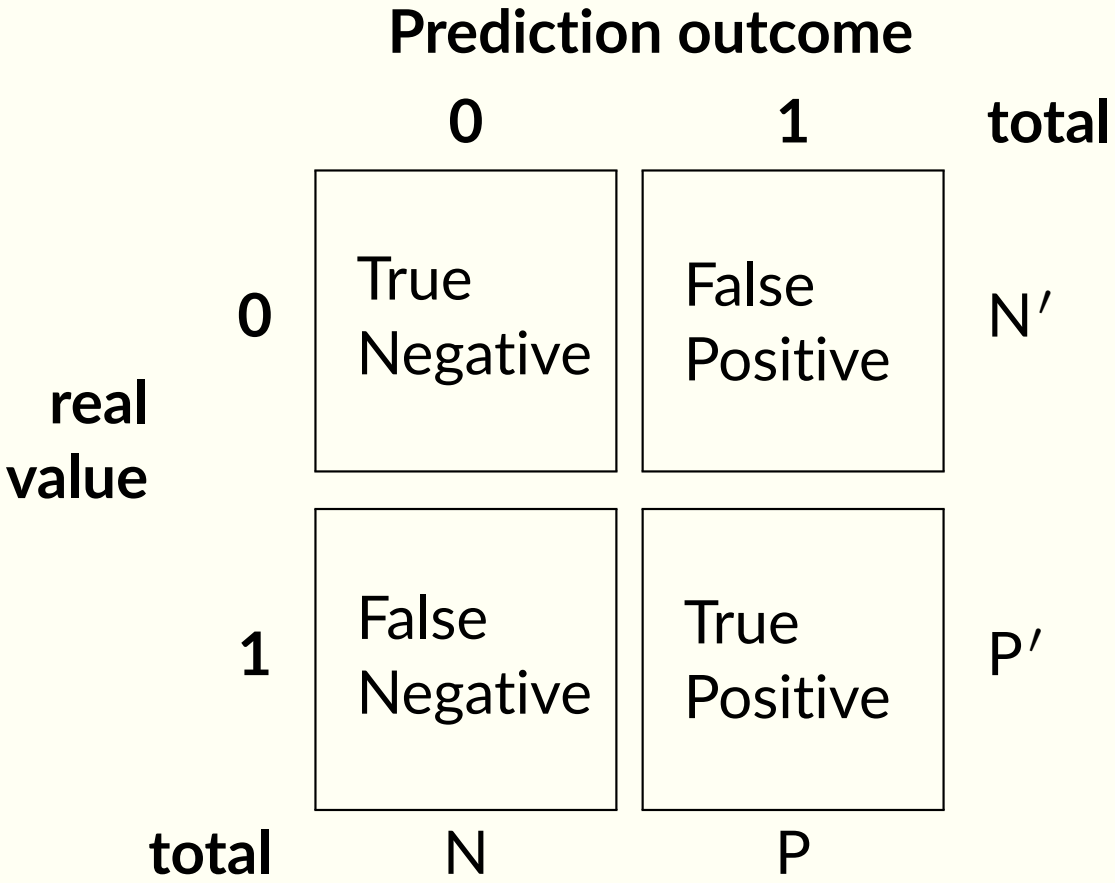

divorced
married
single
1
0
y
education
education
primary
secondary
tertiary
unknown

# Metryki do ewaluacji modeli

- Niezbalansowany zbiór danych (około 10% to obserwacje pozytywne);
- Dokładność (accuracy) nie jest odpowiednią metryką - przewidując same 0 osiągniemy 90% dokładności;
- Nasza metryka: miara F1 (F1 score)

$$F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

# Macierz pomyłek (confusion matrix)




$$precision = \frac{TP}{TP + FP}; \quad recall = \frac{TP}{TP + FN}$$

$$accuracy = \frac{TP + TN}{TP + TN + FN + FP}$$

# Porównanie drzewa przeuczonego i regularyzowanego

MIARA $F_1$

| | train_score | test_score |
|---|---|---|
| drzewo regularyzowane | 0.5291607 | 0.4754098 |
| drzewo przeuczone | 1.0000000 | 0.4395604 |

# Las losowy - budowa modelu

- Skorzystamy z pakietu ranger;
- Przedstawię wyniki czterech modeli: na surowych danych (bez optymalizacji hiperparametrów), na surowych danych (z optymalizacją hiperparametrów), po przekształceniach danych oraz model h2o automl (jako tło);
- We wszystkich modelach przypisywałem klasom wagi (stosunek 1:9);
- Model h2o próbuje trenować różne modele (zarówno drzewiaste jak i sieci neuronowe); po wytrenowaniu zadanej liczby modeli tworzy model ensemble ze stworzonych modeli; końcowym modelem jest model ensemble lub najlepszy z pojedynczych modeli

# Las losowy - wyniki

MIARA $F_1$

| | train_score | test_score |
|---|---|---|
| ranger podstawowy | 0.9637953 | 0.5481481 |
| ranger po optymalizacji | 0.9799499 | 0.5734266 |
| ranger po transformacji | 0.9523810 | 0.5179283 |
| h2o automl model | 0.7887888 | 0.5314685 |

# Siatka parametrów modelu ranger

- mtry: $\{1, 3, 5, 7\}$
- min.node.size: $\{1, 4, \ldots, 19\}$
- num.trees: $\{100, 200, 500, 1000\}$
- wytrenowano losowo 20 modeli;
- najlepsze parametry: mtry – 3, min.node.size – 1, num.trees – 500.

# Podsumowanie

- Transformacje danych nie poprawiały wyników;
- One Hot Encoding pogarszał wyniki;
- Wszystkie kody można znaleźć na:
  `https://github.com/lukaszlaszczuk/pakiety-statystyczne-random-forest`