# Laboratorium 6
14 listopada 2017

## Analiza Danych w Matlabie

### Titanic

(1) Ze strony:
`https://www.kaggle.com/c/titanic/data`
Pobieramy pliki: train.csv oraz test.csv

(2) Importujemy dane i patrzymy, jak wyglądają:

```
Train = readtable('train.csv','Format','%f%f%f%q%C%f
   %f%f%q%f%q%C');
Test = readtable('test.csv','Format','%f%f%q%C%f%f%f
   %q%f%q%C');
disp(Train(1:5,[2:3 5:8 10:11]))
```

(3) Sprawdźmy, jaki wpływ na przeżycie miała płeć:

```
disp(grpstats(Train(:,{'Survived','Sex'}), 'Sex'))
```

Jeśli w naszym modelu przewidujemy, że przeżyły kobiety, a mężczyźni nie, dokładność tego modelu wyniesie $78,68\%$. Dostaniemy jeszcze niższy wynik na test.csv.

```
gendermdl = grpstats(Train(:,{'Survived','Sex'}), {'
   Survived','Sex'})
all_female = (gendermdl.GroupCount('0_male') +
   gendermdl.GroupCount('1_female'))...
    / sum(gendermdl.GroupCount)
```

(4) Missing values

Zapewne zauważyliście, że w kolumnie Cabin brakuje niektórych wierszy.
Zobaczmy, czy są jakieś inne zmienne z brakującymi danymi.
Przy okazji zajmiemy się wartościami dziwnymi (jak np. zerowa opłata za rejs).

```
Train.Fare(Train.Fare == 0) = NaN;       % treat 0
   fare as NaN
Test.Fare(Test.Fare == 0) = NaN;         % treat 0
   fare as NaN
vars = Train.Properties.VariableNames;   % extract
   column names
```

```
figure
imagesc(ismissing(Train))
ax = gca;
ax.XTick = 1:12;
ax.XTickLabel = vars;
ax.XTickLabelRotation = 90;
title('Missing Values')
```

Mamy zatem 177 pasażerów z nieznanym wiekiem.
Jest parę sposobów na radzenie sobie z brakującymi wartościami. Czasem po prostu usuwa się całe wiersze. Tutaj jednak zastąpimy je średnią:

```
avgAge = nanmean(Train.Age)               % get
   average age
Train.Age(isnan(Train.Age)) = avgAge;     % replace
   NaN with the average
Test.Age(isnan(Test.Age)) = avgAge;       % replace
   NaN with the average
```

Mamy 15 pasażerów z nieznaną opłatą za przejazd. Rozsądne jest założenie, że opłaty różniły się w zależności od klasy.

```
fare = grpstats(Train(:,{'Pclass','Fare'}),'Pclass')
   ;   % get class average
disp(fare)
for i = 1:height(fare) % for each |Pclass|
    % apply the class average to missing values
    Train.Fare(Train.Pclass == i & isnan(Train.Fare)
       ) = fare.mean_Fare(i);
    Test.Fare(Test.Pclass == i & isnan(Test.Fare)) =
        fare.mean_Fare(i);
end
```

Jeśli idzie o numer Cabin, zauważmy, że tylko pasażerowie pierwszej klasy mieli kilka pokoi. Brakujące wartości będziemy tu więc traktowali jako 0. Podobnie z klasą trzecią, tam numery kabin są nieregularne.

```
% tokenize the text string by white space
train_cabins = cellfun(@strsplit, Train.Cabin, '
   UniformOutput', false);
test_cabins = cellfun(@strsplit, Test.Cabin, '
   UniformOutput', false);
```

```
% count the number of tokens
Train.nCabins = cellfun(@length, train_cabins);
Test.nCabins = cellfun(@length, test_cabins);

% deal with exceptions - only the first class people
    had multiple cabins
Train.nCabins(Train.Pclass ~= 1 & Train.nCabins >
   1,:) = 1;
Test.nCabins(Test.Pclass ~= 1 & Test.nCabins > 1,:)
   = 1;

% if |Cabin| is empty, then |nCabins| should be 0
Train.nCabins(cellfun(@isempty, Train.Cabin)) = 0;
Test.nCabins(cellfun(@isempty, Test.Cabin)) = 0;
```

Dla dwóch pasażerów nie wiemy, skąd wypłynęli. Użyjemy najczęście występującego miejsca. Zmienimy to też na wartość liczbową, by łatwiej z tego później korzystać:

```
% get most frequent value
freqVal = mode(Train.Embarked);

% apply it to missling value
Train.Embarked(isundefined(Train.Embarked)) =
   freqVal;
Test.Embarked(isundefined(Test.Embarked)) = freqVal;

% convert the data type from categorical to double
Train.Embarked = double(Train.Embarked);
Test.Embarked = double(Test.Embarked);
```

Podobnie zrobimy z płcią:

```
Train.Sex = double(Train.Sex);
Test.Sex = double(Test.Sex);
```

Usuńmy zmienne, którch używać nie będziemy, bo zawierają dane unikatowe lub brakujące:

```
Train(:,{'Name','Ticket','Cabin'}) = [];
Test(:,{'Name','Ticket','Cabin'}) = [];
```

(5) Eksploracja danych i wzualizacja

Przeanalizujmy rozkład danych.
Jest to zadanie bardzo czasochłonne, ale i ważne.
Tu zawęzimy się do zmiennej wieku:

```
figure
histogram(Train.Age(Train.Survived == 0))    % age
   histogram of non-survivers
hold on
histogram(Train.Age(Train.Survived == 1))    % age
   histogram of survivers
hold off
legend('Didn''t Survive', 'Survived')
title('The Titanic Passenger Age Distribution')
```

(6) Cechy – wykorzystujemy powyższe wizualizacje
Dzielimy dane na grupy wzgl. danej zmiennej, czyli u nas np. wieku.

```
% group values into separate bins
Train.AgeGroup = double(discretize(Train.Age,
   [0:10:20 65 80], ...
    'categorical',{'child','teen','adult','senior'})
      );
Test.AgeGroup = double(discretize(Test.Age, [0:10:20
    65 80], ...
    'categorical',{'child','teen','adult','senior'})
      );
```

Popatrzmy teraz na opłatę za bilet.

```
figure
histogram(Train.Fare(Train.Survived == 0));
   % fare histogram of non-survivers
hold on
histogram(Train.Fare(Train.Survived == 1),0:10:520)
   % fare histogram of survivers
hold off
legend('Didn''t Survive', 'Survived')
title('The Titanic Passenger Fare Distribution')

% group values into separate bins
```

```
Train.FareRange = double(discretize(Train.Fare,
   [0:10:30, 100, 520], ...
    'categorical',{'<10','10-20','20-30','30-100','
      >100'}));
Test.FareRange = double(discretize(Test.Fare,
   [0:10:30, 100, 520], ...
    'categorical',{'<10','10-20','20-30','30-100','
      >100'}));
```

źródło: https://blogs.mathworks.com/loren/2015/06/18/getting-started-with-kaggle-da