Laboratorium 1 – Przygotowanie danych (*preprocessing*)

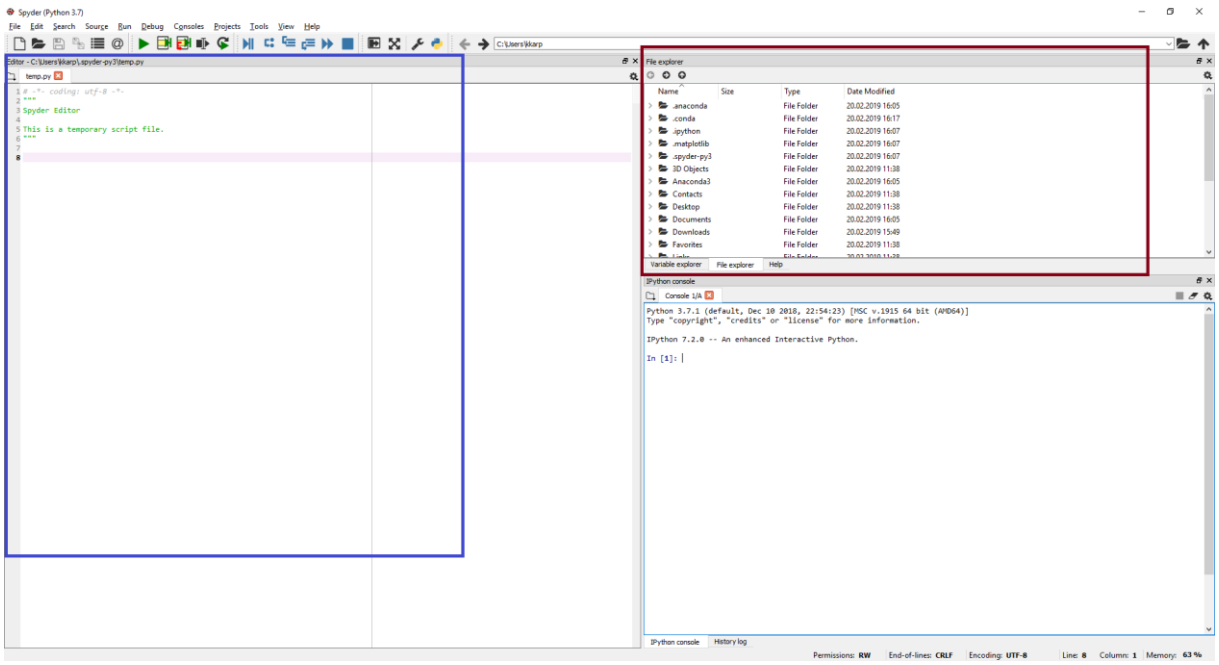
Sztuczna inteligencja (SI) jako dział informatyki zajmuje się tworzeniem modeli zachowań inteligentnych, a także programów komputerowych realizujących takie zachowania. W skład nauk zajmujących się problematyką SI wchodzą systemy uczące się (z ang. *machine learning*). Są to systemy potrafiące doskonalić się przy pomocy zgromadzonego doświadczenia (czyli danych) i nabywania na tej podstawie nowej wiedzy. W konsekwencji tworzone są pewne modele zachowań rozwiązujące problemy, wobec których trudno opracować efektywne algorytmy. Pierwszym i bardzo istotnym etapem uczenia systemu jest przygotowanie danych wejściowych (treningowych). W dalszej części instrukcji do tego laboratorium przedstawiono jak z wykorzystaniem języka python przeprowadzić *preprocessing* danych.

Zadania laboratoryjne:

1. Przygotowanie środowiska do pracy.
   1.1. Pobrać i zainstalować platformę Anaconda (https://www.anaconda.com/distribution/).
   1.2. Uruchomić platformę Anaconda, a następnie aplikację Spyder.
   1.3. Na komputerze utworzyć nowy folder i zapisać w nim otrzymany od prowadzącego plik z danymi (Data.csv).

UWAGA! Pliki mogą być wykorzystywane podczas kolejnych laboratoriów, dlatego po zakończeniu zajęć należy skopiować folder na prywatny nośnik.

   1.4. W programie Spyder ustawić swój folder jako katalog roboczy (obszar zaznaczony czerwonym prostokątem na Rysunku 1.). Zapisać pod dowolną nazwą plik temp.py (widoczny w obszarze zaznaczonym niebieskim prostokątem na Rysunku 1.) w tym samym folderze co plik Data.csv.



Rysunek 1. Interfejs programu Spyder.

2. Utworzenie skryptu przetwarzającego zestaw danych.
   Każdorazowo po dodaniu kodu należy uruchomić skryp. Można to zrobić na kilka sposobów:
   - poprzez wciśnięcie przycisku *Run file*,
   - użycie klawisza F5,
   - zaznaczenie linijek kodu i użycie kombinacji klawiszy ctrl + enter. Jest to najlepszy sposób w przypadku pisania i uruchamiania skryptu linijka po linijce (tak jak to będzie miało miejsce na laboratorium), ponieważ można uruchomić tylko wybrany wiersz.

   2.1. Zaimportować biblioteki, które będą wykorzystywane podczas laboratorium.

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

   2.2. Wczytać zestaw danych.

```
dataset = pd.read_csv('Data.csv')
```

   2.3. Zapoznać się z zawartością pliku Data.csv (zakładka *Variable explorer*). Należy zwrócić uwagę na to, że brakuje danych w wierszu 6 kolumna Age, oraz w wierszu 4 kolumna Salary.

   2.4. Utworzyć macierz trzech niezależnych cech korzystając z danych w pliku Data.csv.

```
X = dataset.iloc[:, :-1].values

# X to macierz trzech niezależnych cech. Zapis [:, :-1].values
oznacza, że pobierane są wartości ze wszystkich wierszy, oraz
wszystkich kolumn oprócz ostatniej. Nie pobieramy ostatniej
kolumny, ponieważ zawiera ona zmienne zależne.
```

   2.5. Utworzyć wektor zmiennych zależnych.

```
y = dataset.iloc[:, 3].values
```

   2.6. Jak można było zaobserwować realizując podpunkt 2.3, w zestawie danych brakuje dwóch informacji. Usunięcie wierszy z brakującymi danymi spowodowałoby utratę informacji, które

mogłyby być wartościowe przy uczeniu systemu. Lepszym rozwiązaniem jest uzupełnienie brakujących komórek np. wartością średnią dla danej kolumny, co należy wykonać dla bieżącego zestawu danych. Uzupełnienie wartości może zostać zrealizowane poprzez wykorzystanie klasy Imputer.

```
from sklearn.preprocessing import Imputer

# ctrl + i w celu uzyskania pomocy dla tej klasy i uzyskania informacji z jakimi parametrami może zostać utworzony obiekt klasy Imputer

imputer = Imputer(missing_values="NaN", strategy="mean", axis=0)

imputer = imputer.fit(X[:, 1:3])

X[:, 1:3] = imputer.transform(X[:, 1:3])
```

2.7. Zakodować dane zawierające kategorie. W tym wypadku są to dane znajdujące się w kolumnach Country i Purchased. Kodowanie odbywa się poprzez przypisywanie danym wartości liczbowych. Zakodowanie danych może zostać zrealizowane poprzez wykorzystanie klasy LabelEncoder.

```
from sklearn.preprocessing import LabelEncoder

labelencoder_X = LabelEncoder()

# [0] - wskazujemy, której kolumny dotyczy operacja, w tym wypadku kolumna o indeksie [0] to kolumna Country

X[:, 0] = labelencoder_X.fit_transform(X[:, 0])

# W tym momencie kraje mają przypisane wartości 0, 1 lub 2.
# Ponieważ są to wartości liczbowe, zostaną błędnie zinterpretowane jako np. 1 > 0, więc Spain jest bardziej istotne od France.
# Należy sprawić, żeby zakodowane nazwy państw nie miały wag.

from sklearn.preprocessing import OneHotEncoder

# [0] - wskazujemy, której kolumny dotyczy operacja

onehotencoder = OneHotEncoder(categorical_features=[0])
X = onehotencoder.fit_transform(X).toarray()

# Kodowanie danych dla kolumny Purchase. Tu nie ma ryzyka wartościowania informacji, więc nie ma potrzeby dalszych modyfikacji danych

labelencoder_y = LabelEncoder()
y = labelencoder_y.fit_transform(y)
```

2.8. Podzielić zbiór danych na zestawy treningowy i testowy.

```
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y,
    test_size = 0.2, random_state = 0)
```

2.9. Przeskalować (dokonać normalizacji) danych. Bez normalizacji kwadrat odległości pomiędzy wartościami danymi z kolumny Salary będzie dominował nad kwadratem odległości pomiędzy wartościami danych z kolumny Age, co wpłynie na proces uczenia się.

```
from sklearn.preprocessing import StandardScaler
sc_X =  StandardScaler()
X_train = sc_X.fit_transform(X_train)
X_test = sc_X.transform(X_test)
```

3. Utworzony skrypt zachować jako template dla potrzeb kolejnych zajęć laboratoryjnych. Przesłać ten plik na serwis moodle w charakterze sprawozdania.