**Міністерство Освіти І НАУКИ України**

**Національний університет "Львівська політехніка"**

Інститут **ІКНІ**

Кафедра **СШІ**

**ЗВІТ**

До лабораторної роботи №1

**З дисципліни:**

**Машинне Навчання**

**Виконав:**

ст. гр. КН-308

Матвіїв М.А.

**Прийняв:**

Якимишин Х.М.

Львів – 2020

**Варіант 1**

**Мета:** засвоїти основні відомості про роботу з фреймворком PyTorch. Навчитись розпізнавати не типові елементи в даних.

Теоретичні Відомості

**Те́нзор** — математичний об'єкт, що узагальнює такі поняття як скаляр, вектор, ковектор, лінійний оператор і білінійна форма. Вивченням тензорів займається тензорне числення. В деякому базисі тензор представляється у вигляді багатовимірної таблиці, заповненої числами

Визначення нетипових елементів. Z-score.

**Визначення нетипових елементів**(outliers detection) - процедура визначення елементів в датасеті, які сильно відрізняються від решти елементів цього датасету. Це частий крок який використовують при обробці та аналізі даних. Один за найпростіших алгоритмів визначення - Z-score. Z-score показує наскільки сильно відрізняється елемент даних від типового(середнього). Усі елементи які мають Z-score > 3 вважаються нетиповими.

**Формула Zscore**:

Z =

Формула 1

Де x – score, – середнє значення, - квадратичне відхилення

**Завдання**

Порахувати outliers для датасету і колонок згідно варіантів, а саме:

**1**. Порахувати z-score незалежно для кожної з згаданих цифрових колонок. Для обрахунку використати лише PyTorch.

**2**. Агрегувати пораховані z-score(наприклад усередненням). Візуалізувати датасет і знайдені нетипові дані(ті, для яких z-score > 3) на scatter plot. Нетипові дані і решту датасету звізуалізувати різними кольорами.

**3**. Якщо серед згаданих колонок є категоріальні дані, порахувати z-score для кожної з колонок в межах кожної з можливих категорій. Наприклад для iris датасету колонка ‘species’ може набувати значень 'setosa', 'versicolor', 'virginica'. Потрібно порахувати z-score для решти колонок лише для значень які в колонці ‘species’ мають значення ‘setosa’, після цього повторити те саме для значень які мають ‘versicolor’ і завершити обрахунком z-score для значень що мають ‘virginica’ в полі ‘species’

**4**. Аналогічно до пункту 2, агрегувати отримані значення і визначити нетипові дані.

**Хід Роботи**

1. Порахувати z-score незалежно для кожної з згаданих цифрових колонок. Для обрахунку використати лише PyTorch.

Підключив потрібні бібліотеки

import torch

import seaborn as sns

import numpy as np

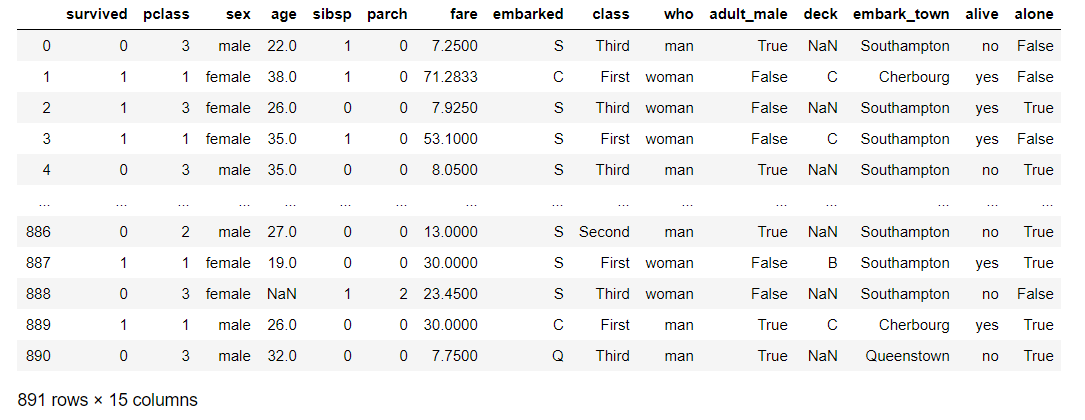
import pandas as pd

import matplotlib.pyplot as plt

вивів датасет згідно варіанту

tit = sns.load\_dataset("titanic")

tit



Створив клас Zscore для обрахунку zscore

class Zscore:

def \_\_init\_\_(self, columns):

self.mean = torch.mean(columns)

self.sigma = torch.std(columns, axis=0)

def get\_score(self, x):

return (x - self.mean)/self.sigma

def get\_avarage\_score(self, x):

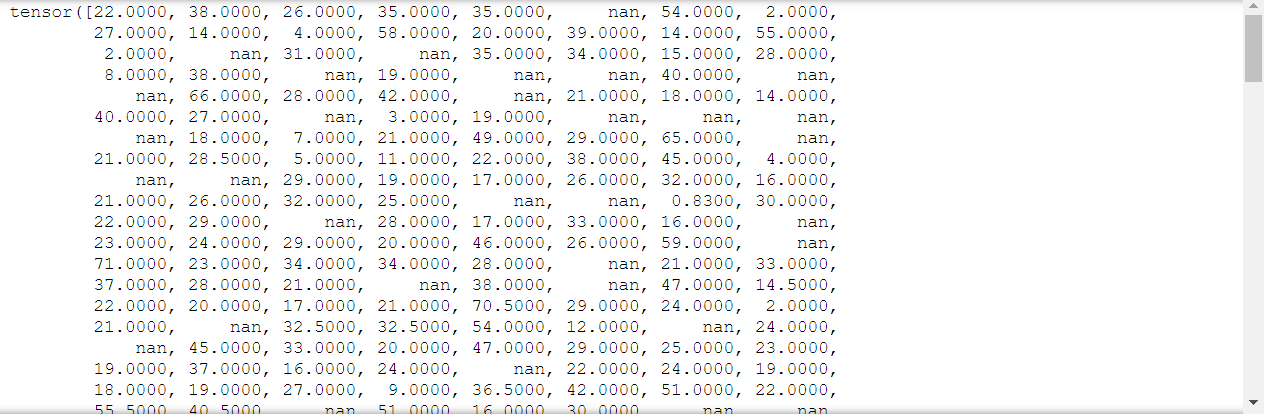
return torch.mean((x - self.mean)/self.sigma)

Зчитав колонки, згідно варіанту, в тензор з типом floаt

age = torch.tensor(tit['age'].values, dtype=torch.float)

fare = torch.tensor(tit['fare'].values, dtype=torch.float)

print(age)



Створив функції для знаходження середнього значення тензора виключаючи nan, заміни nan в тензорі на середнє значення та агрегації

# find average value without nan

def get\_average(array):

sum1 = 0

elements = 0

for i in array:

if not torch.isnan(i):

sum1 += i

elements += 1

return sum1/elements

# replace nan by average value

def replace\_nan\_by\_average(array, average\_number):

for en, i in enumerate(array):

if torch.isnan(i):

array[en] = average\_number

return array

# agregate Zscore

def agregate(score1, score2):

array = []

for i,j in zip(score1, score2):

array.append((i+j)/2)

return array

Замінив nan в тензорах age та fare їхніми середніми значеннями

age\_average = get\_average(age)

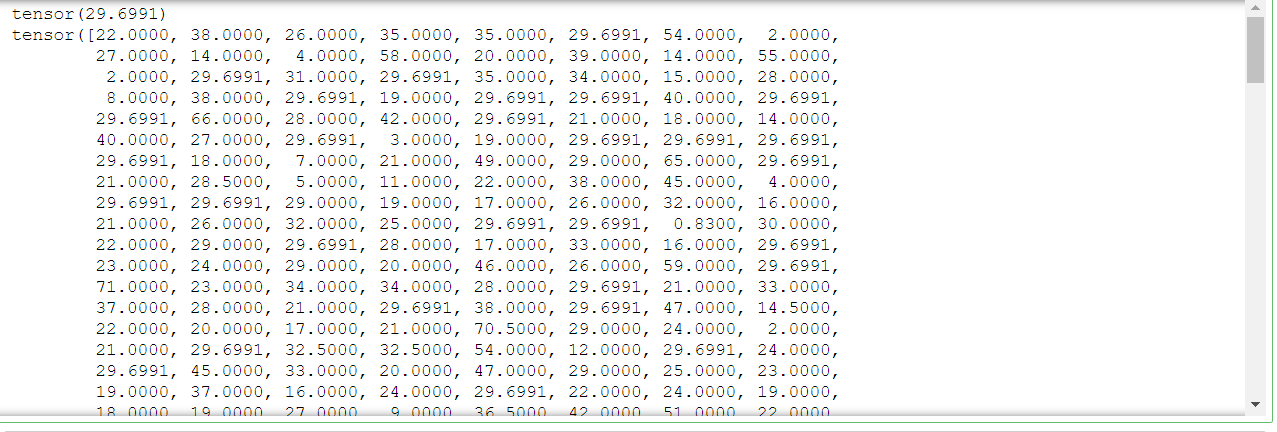
age = replace\_nan\_by\_average(age, age\_average)

fare\_average = get\_average(fare)

fare = replace\_nan\_by\_average(fare, fare\_average)

print(age\_average)

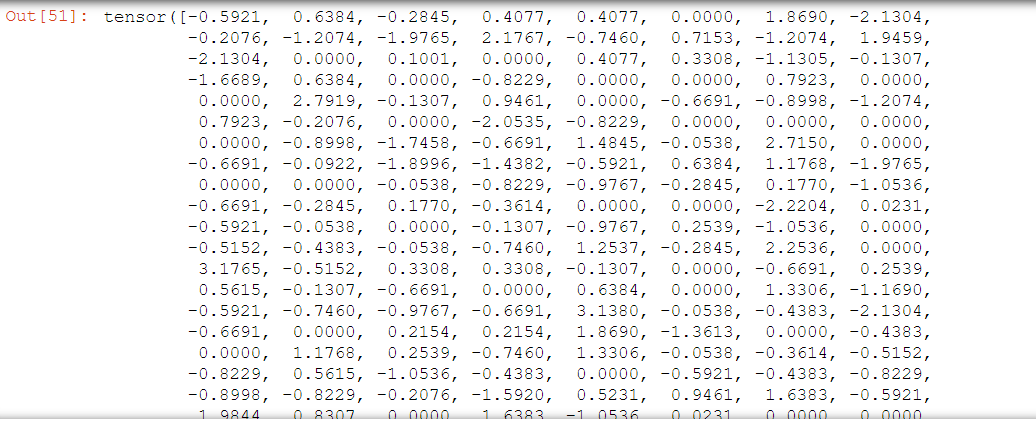
print(age)



Порахував zscore для age та fare

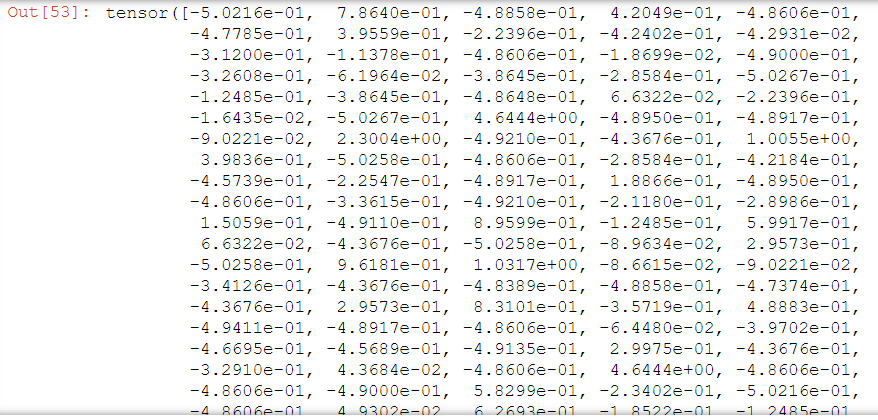
score\_calc = Zscore(age)

age\_score = score\_calc.get\_score(age)



score\_calc = Zscore(fare)

fare\_score = score\_calc.get\_score(fare)



1. Агрегувати пораховані z-score(наприклад усередненням). Візуалізувати датасет і знайдені нетипові дані(ті, для яких z-score > 3) на scatter plot. Нетипові дані і решту датасету звізуалізувати різними кольорами.

Агрегував пораховані zscore методом усереднення

agregated\_score= agregate(age\_score, fare\_score)

Візуалізував датасет до агрегації

colors = []

for i,j in zip(fare\_score, age\_score):

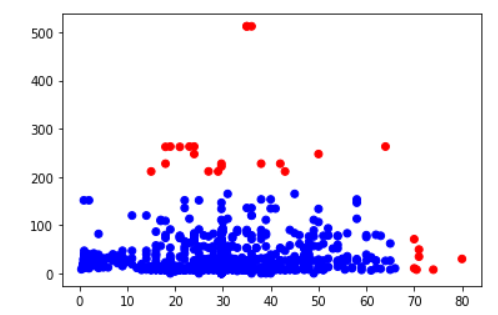
if i > 3 or j > 3:

colors.append('red')

else:

colors.append('blue')

plt.scatter(age, fare, color=colors)



Та після

colors = []

for i in agregated\_score:

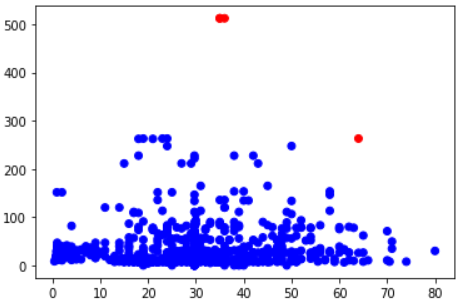
if i > 3:

colors.append('red')

else:

colors.append('blue')

plt.scatter(age, fare, color=colors)



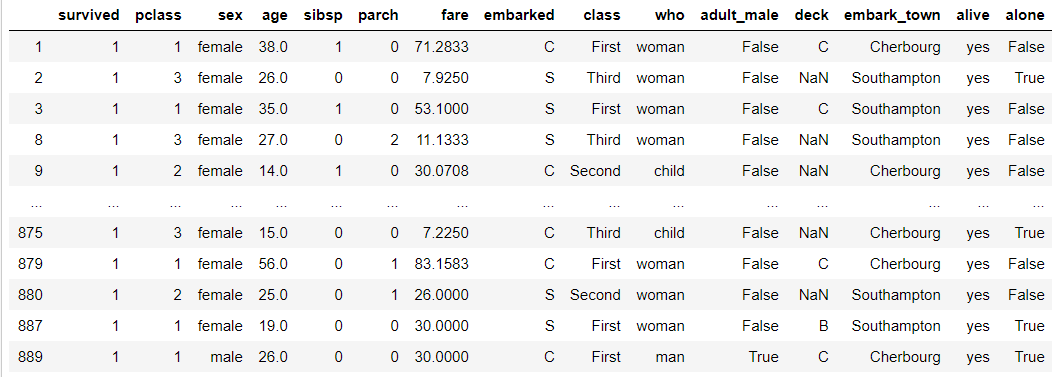
1. Якщо серед згаданих колонок є категоріальні дані, порахувати z-score для кожної з колонок в межах кожної з можливих категорій

Оскільки серед згаданих колонок є категоріальна, а саме survived, порахуємо zscore для кожної.

**Зчитаємо дані де є лише вцілілі та невцілілі**

survived = tit.query("survived =='1'")

not\_survived = tit.query("survived =='0'")



Зчитаємо age та fare для вцілівших людей

age\_survived = torch.tensor(survived['age'].values, dtype=torch.float)

fare\_survived = torch.tensor(survived['fare'].values, dtype=torch.float)

Знайдемо середнє значення та замінимо nan цим значенням

age\_average\_survived = get\_average(age\_survived)

age\_survived = replace\_nan\_by\_average(age\_survived, age\_average\_survived)

fare\_average\_survived = get\_average(fare\_survived)

fare\_survived = replace\_nan\_by\_average(fare\_survived, fare\_average\_survived)

Порахуємо zscore для age\_survived та fare\_survived

score\_calc = Zscore(age\_survived)

age\_survived\_score = score\_calc.get\_score(age\_survived

score\_calc = Zscore(fare\_survived)

fare\_survived\_score = score\_calc.get\_score(fare\_survived)

Та агрегуємо ці значення

agregated\_score\_survived = agregate(age\_survived\_score, fare\_survived\_score)

Візуалізуємо датасет до агрегації

colors = []

for i,j in zip(fare\_survived\_score, age\_survived\_score):

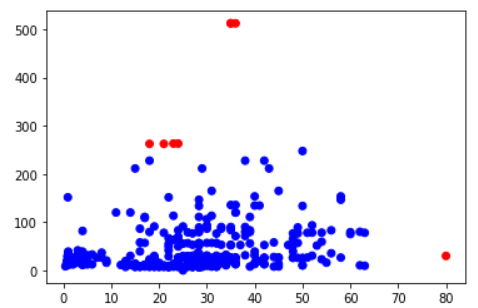
if i > 3 or j > 3:

colors.append('red')

else:

colors.append('blue')

plt.scatter(age\_survived, fare\_survived, color=colors)



Та після

colors = []

for i in agregated\_score\_survived:

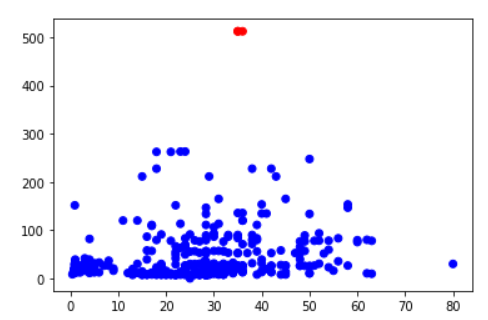
if i > 3:

colors.append('red')

else:

colors.append('blue')

plt.scatter(age\_survived, fare\_survived, color=colors)



**Повторимо ці дії для невцілівших людей**

age\_not\_survived = torch.tensor(not\_survived['age'].values, dtype=torch.float)

fare\_not\_survived = torch.tensor(not\_survived['fare'].values, dtype=torch.float)

age\_average\_not\_survived = get\_average(age\_not\_survived)

age\_not\_survived = replace\_nan\_by\_average(age\_not\_survived, age\_average\_not\_survived)

fare\_average\_not\_survived = get\_average(fare\_not\_survived)

fare\_not\_survived = replace\_nan\_by\_average(fare\_not\_survived, fare\_average\_not\_survived)

score\_calc = Zscore(age\_not\_survived)

age\_not\_survived\_score = score\_calc.get\_score(age\_not\_survived)

score\_calc = Zscore(fare\_not\_survived)

fare\_not\_survived\_score = score\_calc.get\_score(fare\_not\_survived)

agregated\_score\_not\_survived= agregate(age\_not\_survived\_score, fare\_not\_survived\_score)

Візуалізації до агрегації

colors = []

for i,j in zip(fare\_not\_survived\_score, age\_not\_survived\_score):

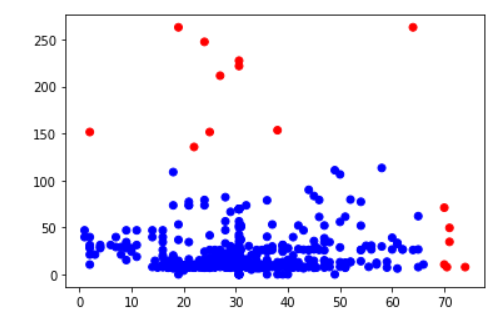
if i > 3 or j > 3:

colors.append('red')

else:

colors.append('blue')

plt.scatter(age\_not\_survived, fare\_not\_survived, color=colors)



Та після агрегації

colors = []

for i in agregated\_score\_not\_survived:

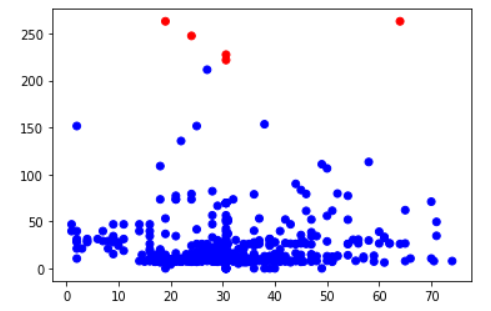
if i > 3:

colors.append('red')

else:

colors.append('blue')

plt.scatter(age\_not\_survived, fare\_not\_survived, color=colors)



**Висновок**

В цій лабораторній роботі я вивчив поняття тензора, алгоритм визначення нетипових даних Z-score та навчився визначати та візуалізовувати нетипові данні за допомогою pytorch