**Министерство науки и высшего образования Российской Федерации**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**



**«Московский государственный технический университет имени Н.Э. Баумана**

**(национальный исследовательский университет)»**

**(МГТУ им. Н.Э. Баумана)**

ФАКУЛЬТЕТ Информатика, искусственный интеллект и системы управления

КАФЕДРА Системы обработки информации и управления

**Методические указания к лабораторным работам по курсу «Машинное обучение»**

**Лабораторная работа №3**

**«Обработка признаков(часть 2)**

Выполнила Зозуля О.А. (ИУ5-25М)

Москва, 2023 г.

# ЗАДАНИЕ

**Цель лабораторной работы:** изучение продвинутых способов предварительной обработки данных для дальнейшего формирования моделей.

1. Выбрать один или несколько наборов данных (датасетов) для решения следующих задач. Каждая задача может быть решена на отдельном датасете, или несколько задач могут быть решены на одном датасете. Просьба не использовать датасет, на котором данная задача решалась в лекции.
2. Для выбранного датасета (датасетов) на основе материалов лекций решить следующие задачи:
   1. масштабирование признаков (не менее чем тремя способами);
   2. обработку выбросов для числовых признаков (по одному способу для удаления выбросов и для замены выбросов);
   3. обработку по крайней мере одного нестандартного признака (который не является числовым или категориальным);
   4. отбор признаков:
      * один метод из группы методов фильтрации (filter methods);
      * один метод из группы методов обертывания (wrapper methods);
      * один метод из группы методов вложений (embedded methods).

Для данной лабораторной работы выберем датасет: babies.csv

Импортируем нужные нам библиотеки и выведем:

# РЕШЕНИЕ

import numpy as np  
import pandas as pd  
import seaborn as sns  
import matplotlib.pyplot as plt  
from sklearn.impute import SimpleImputer  
from sklearn.impute import MissingIndicator  
from sklearn.impute import KNNImputer  
from sklearn.preprocessing import StandardScaler  
from sklearn.linear\_model import Lasso  
from sklearn.pipeline import Pipeline  
from sklearn.model\_selection import GridSearchCV  
from sklearn.ensemble import RandomForestRegressor  
from sklearn.experimental import enable\_iterative\_imputer  
from sklearn.impute import IterativeImputer  
from IPython.display import Image  
import matplotlib  
from mpl\_toolkits import mplot3d  
import pandas as pd  
from matplotlib import pyplot as plt  
import numpy as np  
import seaborn as sns  
import os  
%matplotlib inline  
sns.set(style="ticks")

path=os.environ["userprofile"]+"\\"+".atom"+"\\"+"babies.csv"  
print(path)

C:\Users\657432343536\.atom\babies.csv

data = pd.read\_csv(path)  
print(data)

case bwt gestation parity age height weight smoke

0 1 120 284.0 0 27.0 62.0 100.0 0.0

1 2 113 282.0 0 33.0 64.0 135.0 0.0

2 3 128 279.0 0 28.0 64.0 115.0 1.0

3 4 123 NaN 0 36.0 69.0 190.0 0.0

4 5 108 282.0 0 23.0 67.0 125.0 1.0

... ... ... ... ... ... ... ... ...

1231 1232 113 275.0 1 27.0 60.0 100.0 0.0

1232 1233 128 265.0 0 24.0 67.0 120.0 0.0

1233 1234 130 291.0 0 30.0 65.0 150.0 1.0

1234 1235 125 281.0 1 21.0 65.0 110.0 0.0

1235 1236 117 297.0 0 38.0 65.0 129.0 0.0

[1236 rows x 8 columns]

data.shape

(1236, 8)

hdata=data

list(zip(hdata.columns, [i for i in hdata.dtypes]))

[('case', dtype('int64')),

('bwt', dtype('int64')),

('gestation', dtype('float64')),

('parity', dtype('int64')),

('age', dtype('float64')),

('height', dtype('float64')),

('weight', dtype('float64')),

('smoke', dtype('float64'))]

**Определим колонки с пропусками:**

# Колонки с пропусками  
hcols\_with\_na = [c for c in hdata.columns if hdata[c].isnull().sum() > 0]  
hcols\_with\_na

['gestation', 'age', 'height', 'weight', 'smoke']

data.shape

(1236, 8)

***Устранение пропусков***

[(c, hdata[c].isnull().sum()) for c in hcols\_with\_na]

[('gestation', 13), ('age', 2), ('height', 22), ('weight', 36), ('smoke', 10)]

# Доля (процент) пропусков  
[(c, hdata[c].isnull().mean()\*100) for c in hcols\_with\_na]

[('gestation', 1.051779935275081),

('age', 0.16181229773462785),

('height', 1.779935275080906),

('weight', 2.912621359223301),

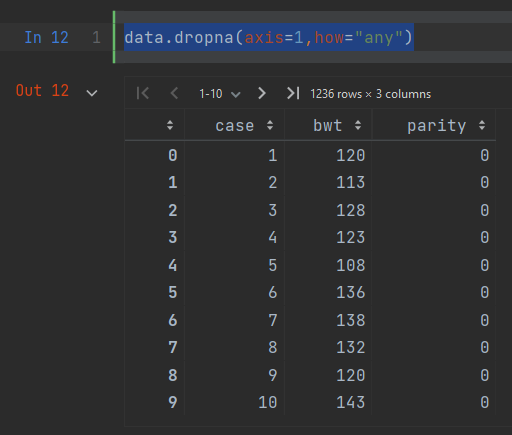
('smoke', 0.8090614886731391)]

# Колонки для которых удаляются пропуски  
hcols\_with\_na\_temp = ['gestation', 'age', 'height', 'weight','smoke']

# Удаление пропусков  
hdata\_drop = hdata[hcols\_with\_na\_temp].dropna()  
hdata\_drop.shape

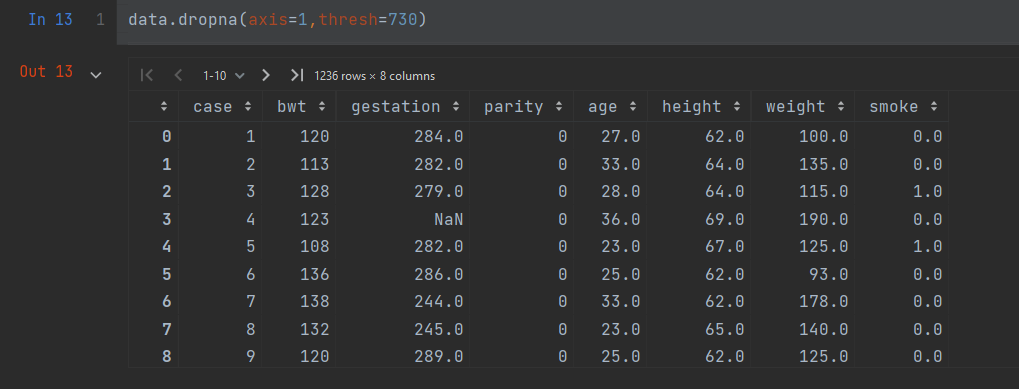
(1174, 5)Удаление колонки, которая содержат пустые значения

data.dropna(axis=1,how="any")



Удалим колонки с высоким процентом (более 50%) пропусков:

data.dropna(axis=1,thresh=730)

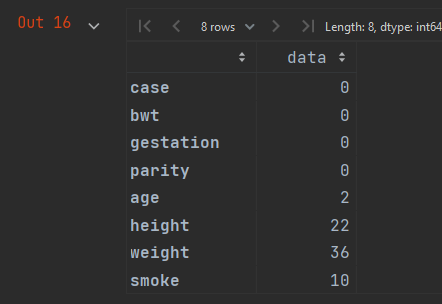


Заполним пропуски возраста средними значениями:

def impute\_na(df,variable,value):  
 df[variable].fillna(value,inplace=True)  
impute\_na(data,'gestation',data['gestation'].mean())

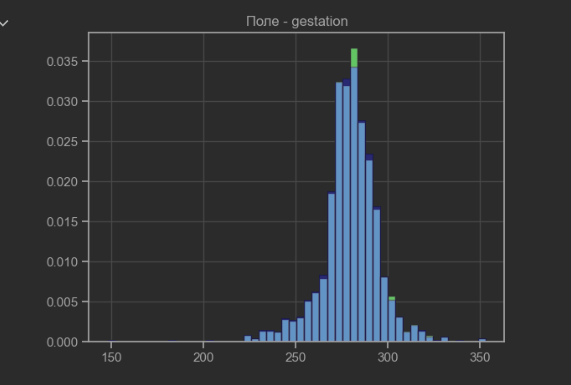
Убедимся,что признак ‘gestation’ не имеет пустых значений:

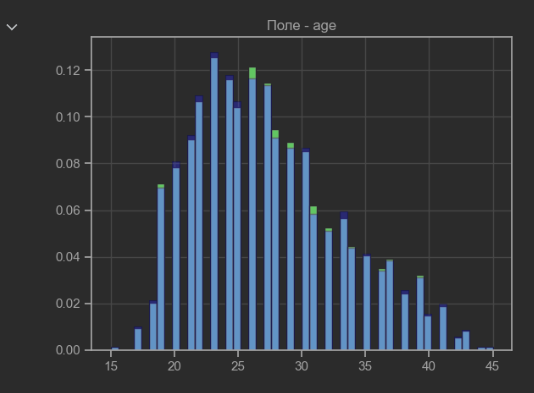
data.isnull().sum()

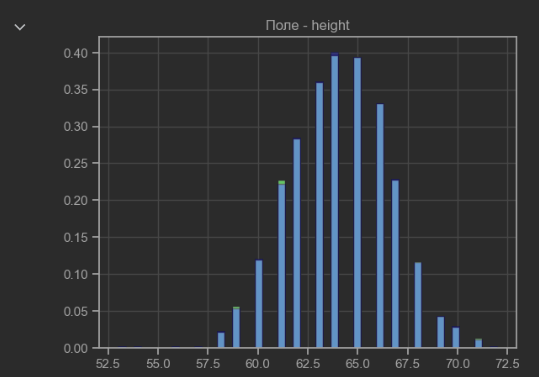


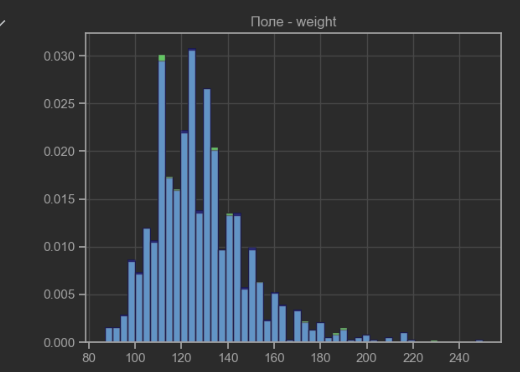
def plot\_hist\_diff(old\_ds, new\_ds, cols):  
 *"""  
 Разница между распределениями до и после устранения пропусков  
 """* for c in cols:  
 fig = plt.figure()  
 ax = fig.add\_subplot(111)  
 ax.title.set\_text('Поле - ' + str(c))  
 old\_ds[c].hist(bins=50, ax=ax, density=True, color='green')  
 new\_ds[c].hist(bins=50, ax=ax, color='blue', density=True, alpha=0.5)  
 plt.show()

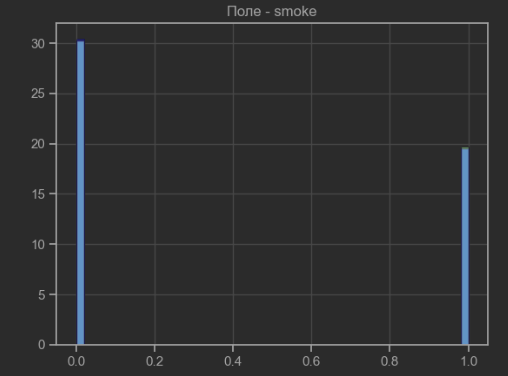
plot\_hist\_diff(hdata, hdata\_drop, hcols\_with\_na\_temp)









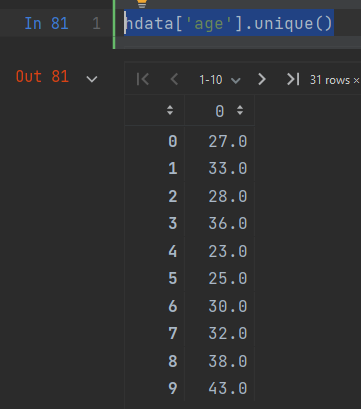


**Кодирование категориальных признаков**

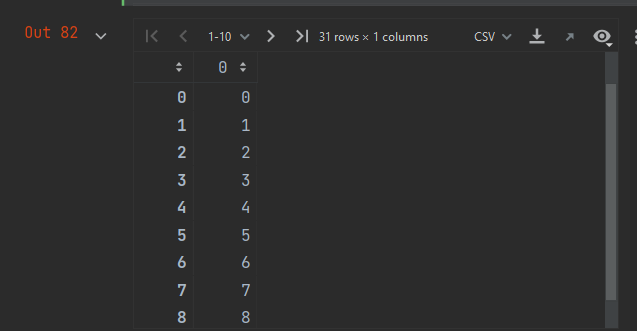
from sklearn.preprocessing import LabelEncoder

le=LabelEncoder()  
cat\_enc\_le=le.fit\_transform(hdata['age'])

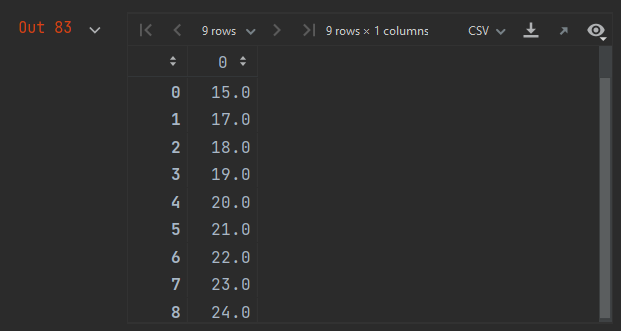
hdata['age'].unique()



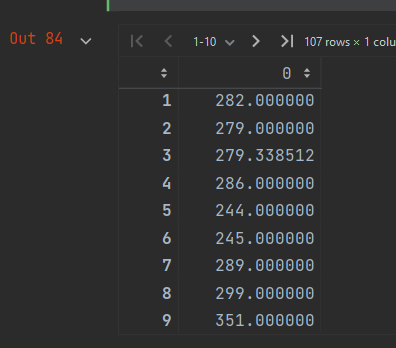
np.unique(cat\_enc\_le)



le.inverse\_transform([0,1,2,3,4,5,6,7,8])



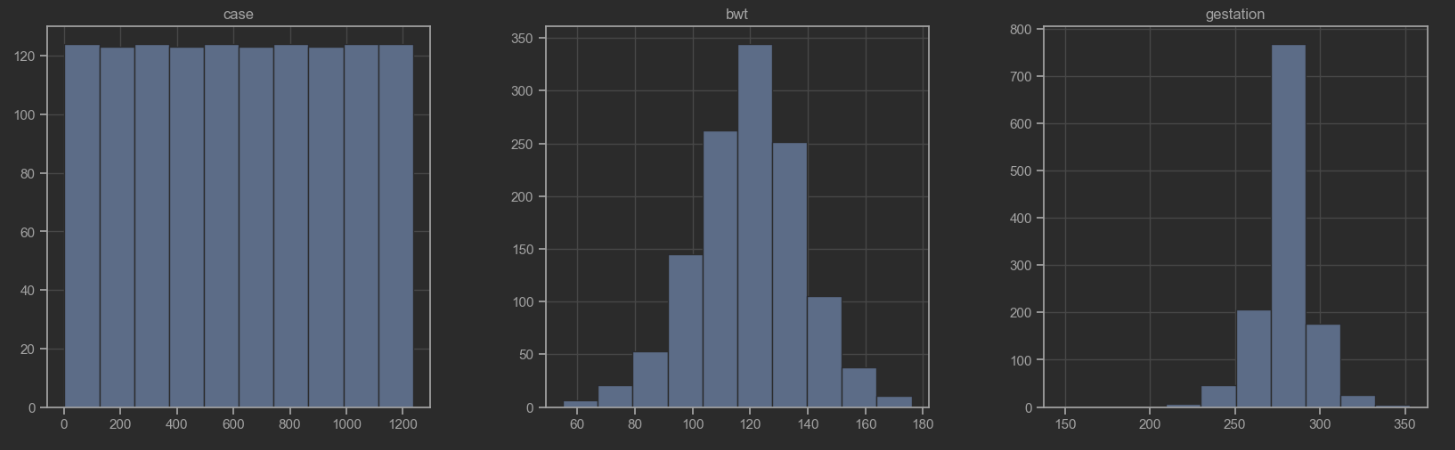
hdata['gestation'].unique()

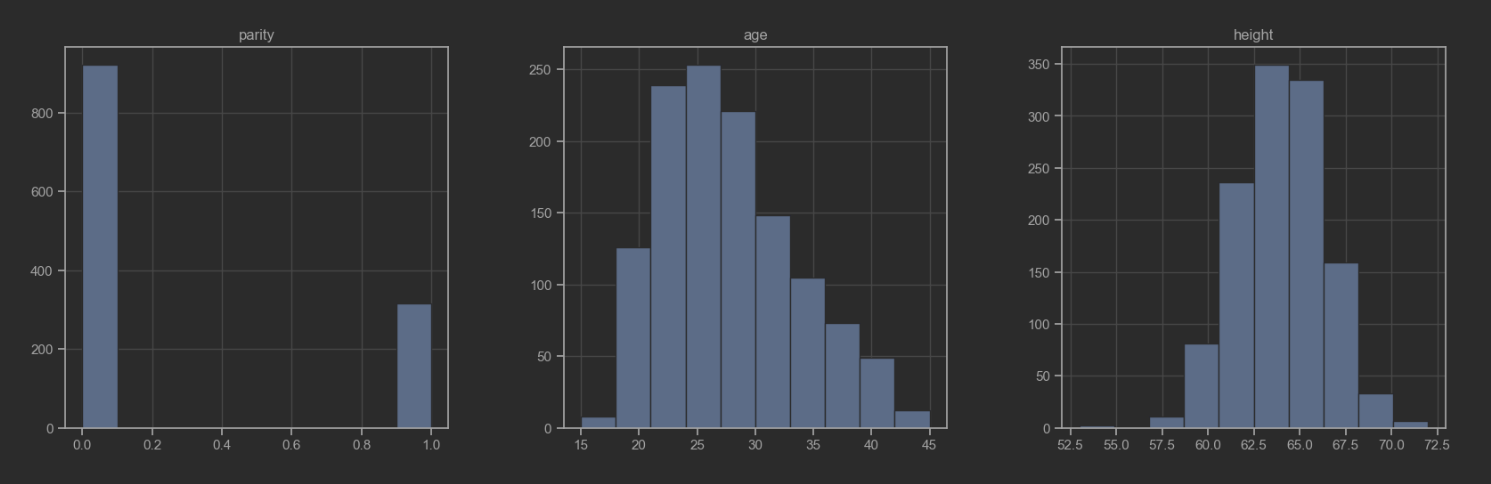


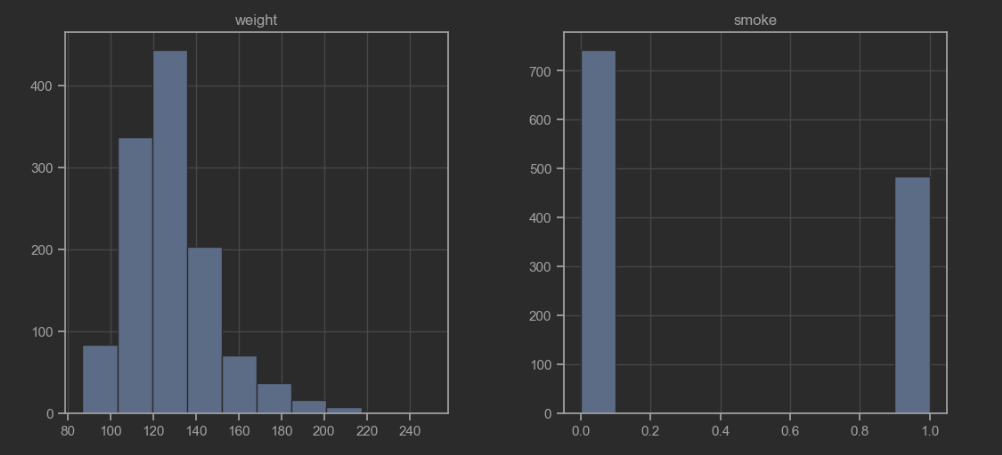
**Нормализация цифровых признаков**

def diagnostic\_plots(df,variable):  
 plt.figure(figsize=(15,6))  
 #гистрограмма  
 plt.subplot(1,2,1)  
 df[variable].hist(bins=30)  
 ##Q-Q-plot  
 plt.subplot(1,2,2)  
 stats.probplot(df[variable],dist="norm",plot=plt)  
 plt.show()

hdata.hist(figsize=(20,20))  
plt.show()







**Нормализация числовых признаков**

normalized\_data=np.log(hdata['gestation'])  
  
plt.violinplot([data["gestation"],normalized\_data,])

