

# Методы машинного обучения

ИУ5-22М Ким Р.И.

## Рубежный контроль №1

### Вариант 6:

Задача №6:

Для набора данных проведите устранение пропусков для одного (произвольного) числового признака с использованием метода заполнения средним значением.

Задача №26:

Для набора данных для одного (произвольного) числового признака проведите обнаружение и замену (найденными верхними и нижними границами) выбросов на основе правила трех сигм.

### Дополнительные требования по группам:

Для студентов групп ИУ5-22М, ИУ5И-22М - для произвольной колонки данных построить гистограмму.

```
In [1]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import scipy.stats as stats
```

```
In [2]: !ls
```

```
RK1.ipynb                                healthcare-dataset-stroke-data.
csv
```

```
In [3]: data = pd.read_csv('healthcare-dataset-stroke-data.csv')
```

In [4]: `data.head()`

Out [4]:

	id	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type
0	9046	Male	67.0	0	1	Yes	Private	Urban
1	51676	Female	61.0	0	0	Yes	Self-employed	Rural
2	31112	Male	80.0	0	1	Yes	Private	Rural
3	60182	Female	49.0	0	0	Yes	Private	Urban
4	1665	Female	79.0	1	0	Yes	Self-employed	Rural

In [5]: `data = data.drop('id', 1)`  
`data.head()`

/var/folders/s3/\_mzg85qn3cb47rm50fh0tmhc0000gn/T/ipykernel\_32771/3892771371.py:1: FutureWarning: In a future version of pandas all arguments of DataFrame.drop except for the argument 'labels' will be keyword-only  
 data = data.drop('id', 1)

Out [5]:

	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	average_
0	Male	67.0	0	1	Yes	Private	Urban	
1	Female	61.0	0	0	Yes	Self-employed	Rural	
2	Male	80.0	0	1	Yes	Private	Rural	
3	Female	49.0	0	0	Yes	Private	Urban	
4	Female	79.0	1	0	Yes	Self-employed	Rural	

## Задача 6

Для набора данных проведите устранение пропусков для одного (произвольного) числового признака с использованием метода заполнения средним значением.

```
In [6]: data_features = list(zip(
# признаки
[i for i in data.columns],
zip(
# типы колонок
[str(i) for i in data.dtypes],
# проверим есть ли пропущенные значения
[i for i in data.isnull().sum()]
)))
# Признаки с типом данных и количеством пропусков
data_features
```

```
Out[6]: [('gender', ('object', 0)),
('age', ('float64', 0)),
('hypertension', ('int64', 0)),
('heart_disease', ('int64', 0)),
('ever_married', ('object', 0)),
('work_type', ('object', 0)),
('Residence_type', ('object', 0)),
('avg_glucose_level', ('float64', 0)),
('bmi', ('float64', 201)),
('smoking_status', ('object', 0)),
('stroke', ('int64', 0))]
```

```
In [7]: # Доля (процент) пропусков
[(c, data[c].isnull().mean()) for c in data.columns]
```

```
Out[7]: [('gender', 0.0),
('age', 0.0),
('hypertension', 0.0),
('heart_disease', 0.0),
('ever_married', 0.0),
('work_type', 0.0),
('Residence_type', 0.0),
('avg_glucose_level', 0.0),
('bmi', 0.03933463796477495),
('smoking_status', 0.0),
('stroke', 0.0)]
```

Видно, что пропуски имеются в полях age и bmi

```
In [8]: # Заполним пропуски bmi средними значениями
def impute_na(df, variable, value):
    df[variable].fillna(value, inplace=True)
impute_na(data, 'bmi', data['bmi'].mean())
```

```
In [9]: # Удалим данные, где возраст незаполнен, так как таких данных мало,
data.dropna(subset=['age'], inplace=True)
```

```
In [10]: # Убедимся что нет пустых значений
data.isnull().sum()
```

```
Out[10]: gender          0
age          0
hypertension  0
heart_disease 0
ever_married  0
work_type     0
Residence_type 0
avg_glucose_level 0
bmi           0
smoking_status 0
stroke        0
dtype: int64
```

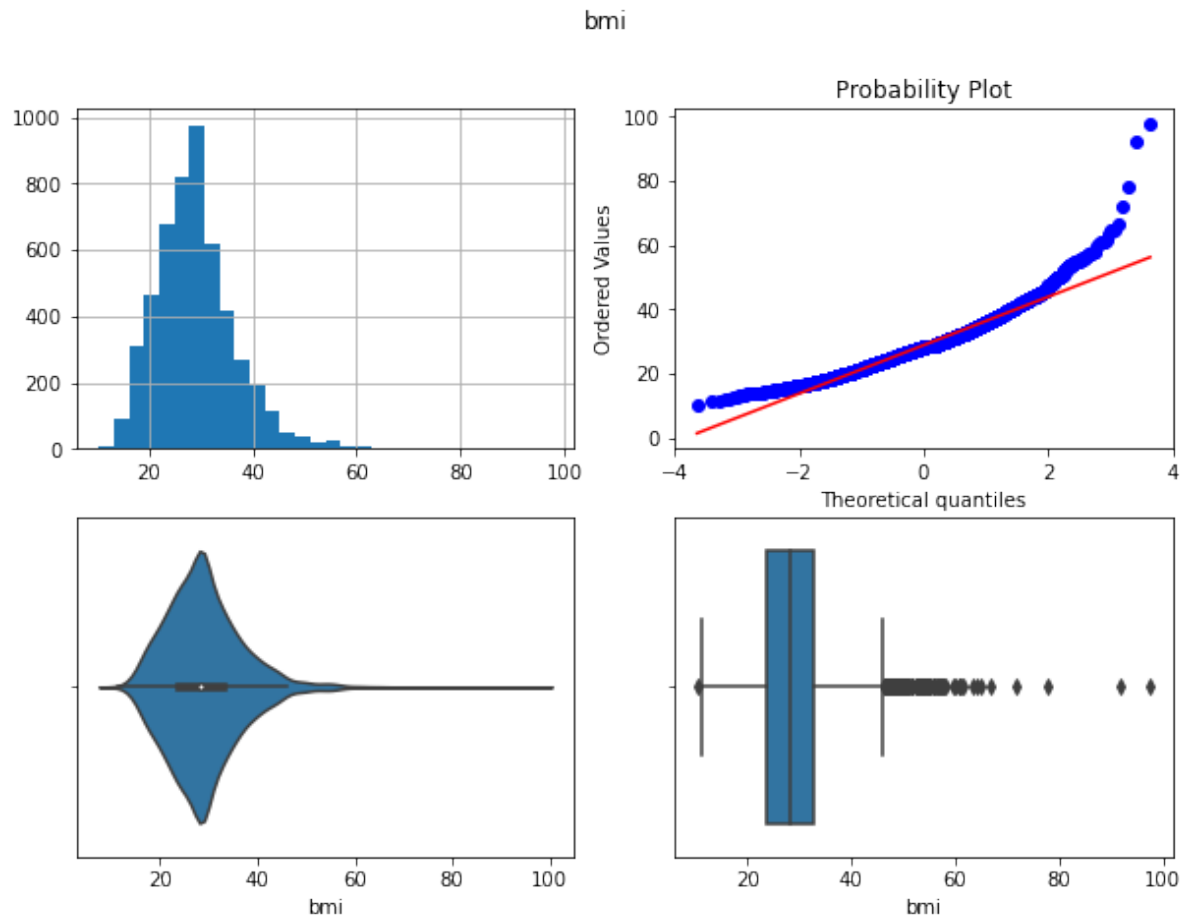
Итого: Провели устранение пропусков в полях Age - возраст и bmi - индекс массы тела

## Задача 26

Для набора данных для одного (произвольного) числового признака проведите обнаружение и замену (найденными верхними и нижними границами) выбросов на основе правила трех сигм.

```
In [11]: def diagnostic_plots(df, variable, title):
fig, ax = plt.subplots(figsize=(10,7))
# гистограмма
plt.subplot(2, 2, 1)
df[variable].hist(bins=30)
## Q-Q plot
plt.subplot(2, 2, 2)
stats.probplot(df[variable], dist="norm", plot=plt)
# ящик с усами
plt.subplot(2, 2, 3)
sns.violinplot(x=df[variable])
# ящик с усами
plt.subplot(2, 2, 4)
sns.boxplot(x=df[variable])
fig.suptitle(title)
plt.show()
```

```
In [12]: diagnostic_plots(data, 'bmi', 'bmi')
```



На графике "Ящик с усами" видно, что много выбросов с левой стороны, устраним их заменой

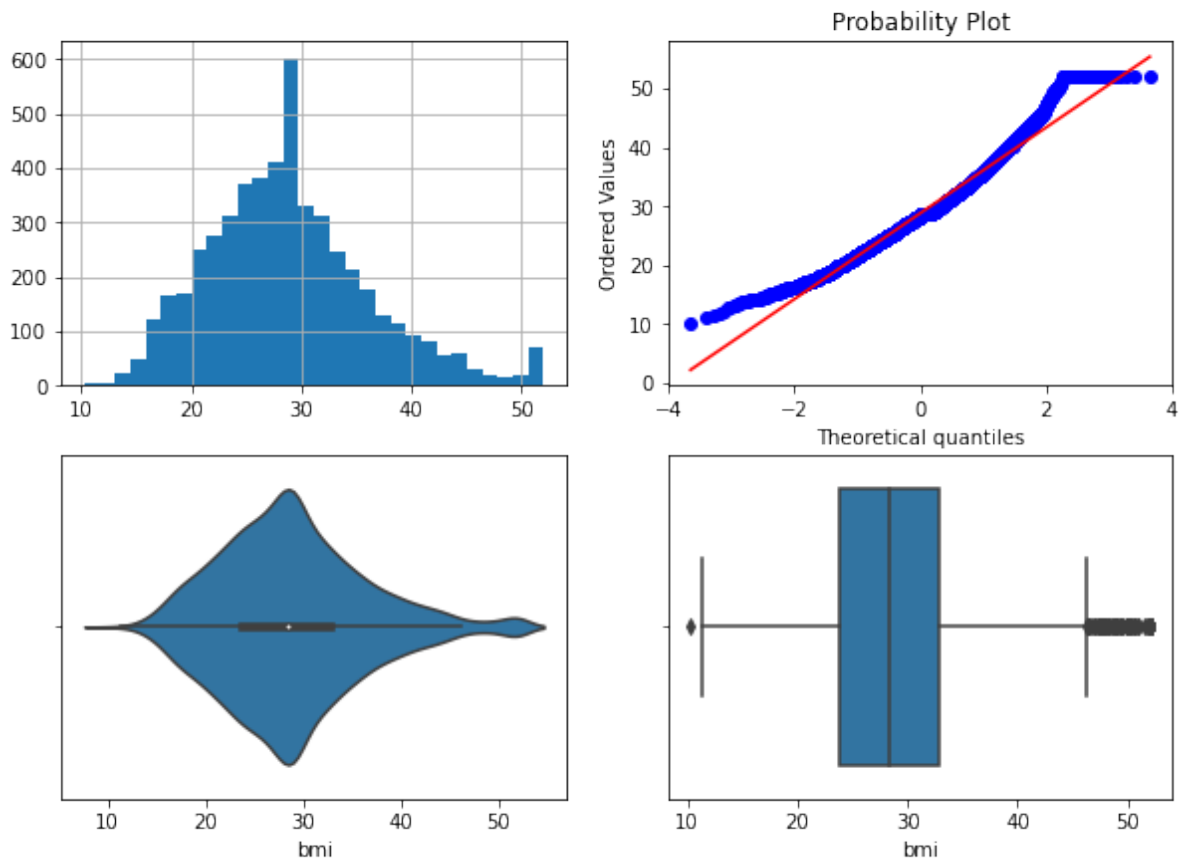
```
In [13]: lower_boundary = data['bmi'].mean() - (3 * data['bmi'].std())
upper_boundary = data['bmi'].mean() + (3 * data['bmi'].std())
print('Нижняя граница', lower_boundary)
print('Верхняя граница', upper_boundary)
```

Нижняя граница 5.7991834312234865

Верхняя граница 51.98729039236595

```
In [17]: col = 'bmi'
data[col] = np.where(data[col] > upper_boundary, upper_boundary,
                    np.where(data[col] < lower_boundar
diagnostic_plots(data, col, 'Анализ BMI')
```

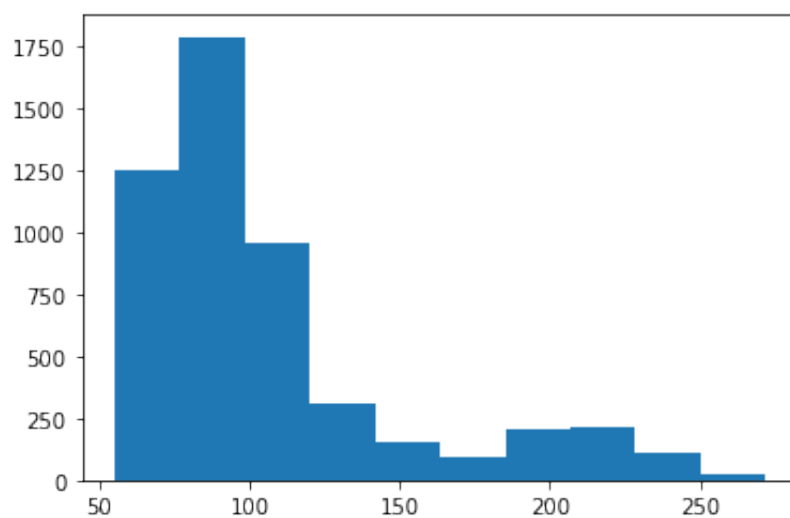
Анализ BMI



Видно, что количество выбросов уменьшилось, но некоторое количество всё же осталось.

## Дополнительное задание

```
In [19]: plt.hist(data['avg_glucose_level'])  
plt.show()
```



Построили гистограмму для колонки 'avg\_glucose\_level'