jupyter Lab2 (автосохранение)

File Edit View Insert Cell Kernel Widgets Help

Logout

Доверенный

Python 3

Markdown

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 и системы управления»**

**Кафедра ИУ5 «Системы обработки информации в управлении»**

**Технологии машинного обучения**

**Отчет по лабораторной работе № 2**

Выполнил:
студент группы ИУ5-64Б
Видяев Е.А.

2021 г.
Москва

# Загрузка и первичный анализ данных

Используем dataset прогнозирования рейтинга ресторанов Tripadvisor в Европе.

https://www.kaggle.com/datascienceshortcut/tripadvisor-european-restaurants-dataset

In [1]:
```
# импортируем библиотеки
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style="ticks")
```

In [2]:
```
# преобразование данных
Country                 object
Real coffee             object
Instant coffee          int64
Tea                     int64
Sweetener               float64
Biscuits                int64
Powder soup             int64
Tin soup                int64
Potatoes                int64
Frozen veg              int64
Frozen veggies          int64
Apples                  int64
Oranges                 int64
Tinned fruit            int64
Jam                     int64
Garlic                  int64
Butter                  int64
Margarine               int64
Olive oil               int64
Yoghurt                 int64
Crisp bread             int64
dtype: object
```

In [3]:
```
# пробуем есть по пропущенным значениям
data.isnull().sum()
```

Out[3]:
```
Country                 0
Real coffee             0
Instant coffee          0
Tea                     0
Sweetener               1
Biscuits                1
Powder soup             0
Margarine               0
Olive oil               0
Yoghurt                 0
Crisp bread             0
dtype: int64
```

In [12]:
```
data.head(10)
```

Out[12]:

| | Country | Real coffee | Instant coffee | Tea | Sweetener | Biscuits | Powder soup | Tin soup | Potatoes | Frozen veg | ... | Apples | Oranges | Tinned fruit | Jam | Garlic | Butter | Margarine | Olive oil | Yoghurt | Crisp bread |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Germany | 90 | 49 | 88 | 19.0 | 57.0 | 51 | 19 | 21 | 27 | ... | 81 | 75 | 44 | 71 | 22 | 91 | 85 | 7 | 91 | 64 |
| 1 | Italy | 62 | 10 | 60 | 2.0 | 55.0 | 41 | 3 | 2 | 4 | ... | 67 | 71 | 9 | 46 | 80 | 66 | 24 | 9 | 39 | 5 |
| 2 | France | 88 | 42 | 63 | 4.0 | 76.0 | 53 | 11 | 23 | 11 | ... | 67 | 84 | 40 | 45 | 88 | 54 | 47 | 3 | 89 | 14 |
| 3 | Holland | 98 | 62 | 98 | 52.0 | 62.0 | 67 | 43 | 7 | 14 | ... | 83 | 88 | 61 | 81 | 15 | 31 | 97 | 1 | 93 | 8 |
| 4 | Belgium | 94 | 38 | 48 | 11.0 | 74.0 | 37 | 23 | 9 | 13 | ... | 76 | 76 | 42 | 57 | 29 | 84 | 80 | 8 | 93 | 6 |
| 5 | Luxembourg | 91 | 61 | 86 | 28.0 | 79.0 | 73 | 12 | 7 | 20 | ... | 95 | 94 | 83 | 20 | 91 | 94 | 94 | 6 | 95 | 1 |
| 6 | England | 27 | 88 | 99 | 22.0 | 91.0 | 55 | 76 | 17 | 20 | ... | 76 | 88 | 89 | 91 | 11 | 95 | 94 | 5 | 73 | 9 |
| 7 | Portugal | 72 | 28 | 77 | 2.0 | 22.0 | 34 | 1 | 5 | 20 | ... | 22 | 51 | 8 | 16 | 89 | 65 | 76 | 9 | 50 | 8 |
| 8 | Austria | 51 | 91 | 95 | 15.0 | 95.0 | 29 | 1 | 5 | 15 | ... | 46 | 42 | 14 | 41 | 51 | 51 | 72 | 3 | 95 | 2 |
| 9 | Switzerland | 73 | 72 | 85 | 25.0 | 31.0 | 69 | 10 | 17 | 19 | ... | 79 | 70 | 46 | 61 | 64 | 52 | 48 | 6 | 92 | 4 |
| 10 | Sweden | 97 | 72 | 93 | 31.0 | NaN | 43 | 43 | 30 | 54 | ... | 93 | 78 | 53 | 75 | 5 | 68 | 32 | 9 | 93 | 7 |
| 11 | Denmark | 98 | 17 | 92 | 35.0 | 66.0 | 32 | 17 | 11 | 51 | ... | 81 | 72 | 50 | 64 | 11 | 52 | 91 | 3 | 93 | 1 |
| 12 | Norway | 92 | 57 | 83 | 13.0 | 62.0 | 51 | 4 | 17 | 30 | ... | 81 | 72 | 34 | 51 | 11 | 63 | 94 | 2 | 92 | 2 |

```
shape_data = data.shape
print('Всего строк: {}'.format(shape_data[0]))
```

```
Всего строк: 10
```

# Обработка пропусков в данных

In [9]:
```
# Уникальные значения, содержащие пропуски значений
data_new_1 = data.dropna(axis=1, how='any')
(data.shape, data_new_1.shape)
```

Out[9]:
```
((26, 21), (26, 18))
```

In [10]:
```
# Уникальные строки, содержащие пропуски значений
data_new_2 = data.dropna(axis=0, how='any')
(data.shape, data_new_2.shape)
```

Out[10]:
```
((26, 21), (15, 21))
```

In [11]:
```
data.head(1)
```

Out[11]:

| | Country | Real coffee | Instant coffee | Tea | Sweetener | Biscuits | Powder soup | Tin soup | Potatoes | Frozen veg | ... | Apples | Oranges | Tinned fruit | Jam | Garlic | Butter | Margarine | Olive oil | Yoghurt | Crisp bread |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Germany | 90 | 49 | 88 | 19.0 | 57.0 | 51 | 19 | 21 | 27 | ... | 81 | 75 | 44 | 71 | 22 | 91 | 85 | 7 | 91 | 64 |
| 1 | Italy | 62 | 10 | 60 | 2.0 | 55.0 | 41 | 3 | 2 | 4 | ... | 67 | 71 | 9 | 46 | 80 | 66 | 24 | 9 | 39 | 5 |
| 2 | France | 88 | 42 | 63 | 4.0 | 76.0 | 53 | 11 | 23 | 11 | ... | 67 | 84 | 40 | 45 | 88 | 54 | 47 | 3 | 89 | 14 |
| 3 | Holland | 98 | 62 | 98 | 52.0 | 62.0 | 67 | 43 | 7 | 14 | ... | 83 | 88 | 61 | 81 | 15 | 31 | 97 | 1 | 93 | 8 |
| 4 | Belgium | 94 | 38 | 48 | 11.0 | 74.0 | 37 | 23 | 9 | 13 | ... | 76 | 76 | 42 | 57 | 29 | 84 | 80 | 8 | 93 | 6 |
| 5 | Luxembourg | 91 | 61 | 86 | 28.0 | 79.0 | 73 | 12 | 7 | 20 | ... | 95 | 94 | 83 | 20 | 91 | 94 | 94 | 6 | 95 | 1 |
| 6 | England | 27 | 88 | 99 | 22.0 | 91.0 | 55 | 76 | 17 | 20 | ... | 76 | 88 | 89 | 91 | 11 | 95 | 94 | 5 | 73 | 9 |
| 7 | Portugal | 72 | 28 | 77 | 2.0 | 22.0 | 34 | 1 | 5 | 20 | ... | 22 | 51 | 8 | 16 | 89 | 65 | 76 | 9 | 50 | 8 |
| 8 | Austria | 51 | 91 | 95 | 15.0 | 95.0 | 29 | 1 | 5 | 15 | ... | 46 | 42 | 14 | 41 | 51 | 51 | 72 | 3 | 95 | 2 |
| 9 | Switzerland | 73 | 72 | 85 | 25.0 | 31.0 | 69 | 10 | 17 | 19 | ... | 79 | 70 | 46 | 61 | 64 | 52 | 48 | 6 | 92 | 4 |
| 10 | Sweden | 97 | 72 | 93 | 31.0 | NaN | 43 | 43 | 30 | 54 | ... | 93 | 78 | 53 | 75 | 5 | 68 | 32 | 9 | 93 | 7 |
| 11 | Denmark | 98 | 17 | 92 | 35.0 | 66.0 | 32 | 17 | 11 | 51 | ... | 81 | 72 | 50 | 64 | 11 | 52 | 91 | 3 | 93 | 1 |
| 12 | Norway | 92 | 57 | 83 | 13.0 | 62.0 | 51 | 4 | 17 | 30 | ... | 81 | 72 | 34 | 51 | 11 | 63 | 94 | 2 | 92 | 2 |

13 rows × 21 columns

## Внедрение значений

### Обработка пропусков в числовых данных

In [14]:
```
# Выберем числовые колонки с пропущенными значениями
# Цикл по колонкам dataset
num_cols = []
for col in data.columns:
    # Количество пустых значений
    temp_null_count = data[data[col].isnull()].shape[0]
    dtype = str(data[col].dtype)
    if temp_null_count > 0 and (dtype.find('float') or dtype.find('int')):
        num_cols.append(col)
        temp_null_ratio = round(temp_null_count / total_count) * 100.0, 2)
        print('Колонка {}. Тип данных {}. Количество пустых значений {}, {}%.'.format(col, dtype, temp_null_count, temp_null_ratio))
```

```
Колонка Sweetener. Тип данных float64. Количество пустых значений 1, 0.25%.
Колонка Biscuits. Тип данных float64. Количество пустых значений 1, 0.25%.
Колонка Yoghurt. Тип данных float64. Количество пустых значений 1, 0.25%.
```

In [15]:
```
# Вывод по колонкам с пропущенными значениями
data_num = data[num_cols]
data_num
```

Out[15]:

| | Sweetener | Biscuits | Yoghurt |
|---|---|---|---|
| 0 | 19.0 | 57.0 | 91.0 |
| 1 | 2.0 | 55.0 | 9.0 |
| 2 | 4.0 | 76.0 | 57.0 |
| 10 | 31.0 | NaN | 2.0 |
| 11 | 35.0 | 66.0 | 11.0 |
| 12 | 13.0 | 62.0 | 2.0 |
| 13 | 25.0 | 64.0 | NaN |
| 14 | NaN | 62.0 | 16.0 |
| 15 | 11.0 | 80.0 | 3.0 |

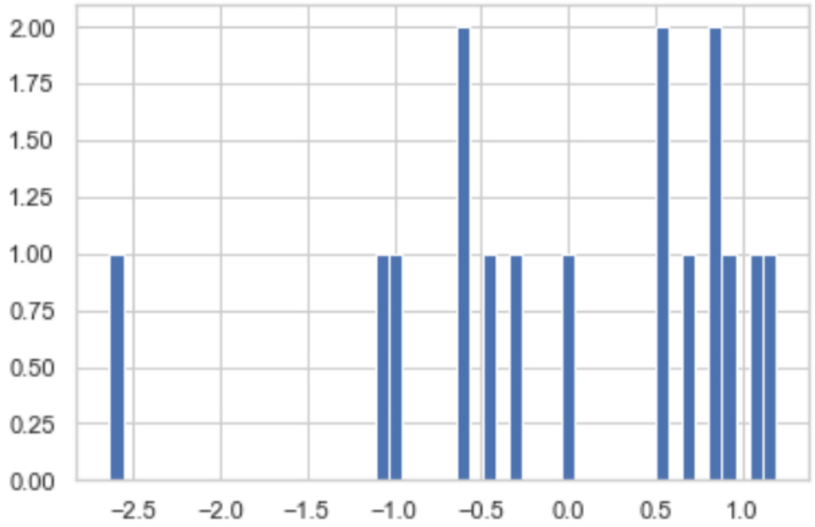
In [16]:
```
# Гистограмма по признакам
for col in data_num:
    plt.hist(data[col], 50)
    plt.xlabel(col)
    plt.show()
```

Будем использовать встроенные средства библиотеки Библиотеки scikit-learn – https://scikit-learn.org/stable/modules/impute.html

In [18]:
```
from sklearn.impute import SimpleImputer
from sklearn.impute import MissingIndicator
```

In [20]:
```
# Выберем для примера количественные колонки dataset
data_num_latestTradeData = data_num[['Biscuits']]
indicator = MissingIndicator(
    features=
    ['Biscuits',
    'Yoghurt',
    'True',
    'False',
    'False',
    'False
```

## Масштабирование данных на основе Z-оценки - StandardScaler

В [61]:
```
sc2 = StandardScaler()
sc2_data = sc2.fit_transform(data[['Apples']])
```

В [62]:
```
plt.hist(sc2_data, 50)
plt.show()
```