Анализ данных. Домашнее задание

Задача состоит в работе с реальными данными о работе системы кондиционирования одного из распределительных центров крупной торговой сети.

Описание исходных данных

Система кондиционирования состоит множества практически одинаковых устройств, каждое из которых имеет определенный набор параметров.

Устройство	Тип	Контроллер
111 CT AG	Центральный холодильный агрегат	AK-PC551-0161
11CT G OVZ +5/+8	Холодильная горка	EKC202B-013x
12CT G GSR +2/+4	Холодильная горка	EKC202B-013x
13CT G PBP -1/+1	Холодильная горка	EKC202B-013x
15CT G MSO -1/+1	Холодильная горка	EKC202B-013x
16CT G MSO -1/+1	Холодильная горка	EKC202B-013x
17CT G GSR +2/+4	Холодильная горка	EKC202B-013x
18CT G PTO -1/+1	Холодильная горка	EKC202B-013x
19CT G GSR +2/+4	Холодильная горка	EKC202B-013x
20CT G GSR +2/+4	Холодильная горка	EKC202B-013x
21CT G GSR +2/+4	Холодильная горка	EKC202B-013x
22CT G GSR +2/+4	Холодильная горка	EKC202B-013x
23CT V MSO -1/+1	Холодильная горка	EKC202B-013x
24CT V GSR +2/+4	Холодильная горка	EKC202B-013x
25CT V GSR +2/+4	Холодильная горка	EKC202B-013x
26CT G MLK +2/+4	Холодильная горка	EKC202B-013x
27CT G MLK +2/+4	Холодильная горка	EKC202B-013x
28CT G MLK +2/+4	Холодильная горка	EKC202B-013x
29CT G MLK +2/+4	Холодильная горка	EKC202B-013x
30CT G GSR +2/+4	Холодильная горка	EKC202B-013x
34CT G TRT +2/+4	Холодильная горка	EKC202B-013x
52CT K PTO -1/+1	-	EKC202D-022x
53K VOP POD PTO	-	EKC202D-022x
54K VOP FAS GSR	-	EKC202D-022x
55CT K GSR +2/+4	-	EKC202D-022x
56CT K COF +5/+8	-	EKC202D-022x
57K VOP CXM	-	EKC202D-022x
Gazoanalyzator	Газоанализатор	

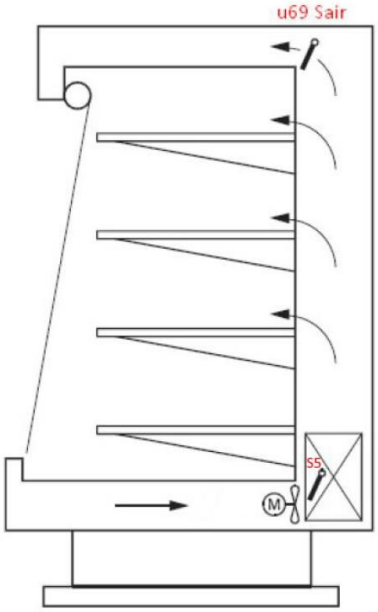
Параметры холодильных установок:

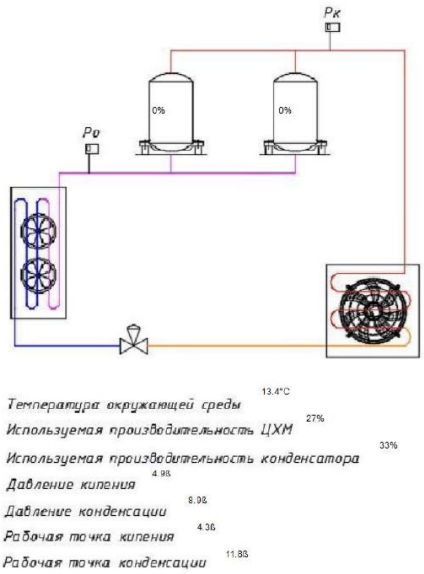
Параметр	Описание	Тип
EKC состояние	Состояние установки	Категориальная
u69 Sair Темп degc	Температура воздуха внутри установки	Непрерывная
u09 S5 Темп degc	Температура воздуха на входе в испаритель	Непрерывная

Параметры центрального холодильного агрегата:

Параметр	Описание	Тип
Cond Requested Cap %	Запрошенная производительность вентиляторов конденсатора	Непрерывная
Cond Ctrl Status	Режим управления конденсатором	Категориальная
Cond Running Cap %	Текущая производительность вентиляторов конденсатора	Непрерывная
Cond Ctrl Pressure Bar	Давление конденсации	Непрерывная
Cond Reference Bar	Уставка давления конденсации	Непрерывная
Comp A Ctrl Status	Режим управления компрессором	Категориальная
Comp A Pressure Bar	Давление на входе в компрессор	Непрерывная
Comp A Reference Bar	Уставка давления на входе в компрессор	Непрерывная
Comp A Capacity %	Производительность компрессоров	Непрерывная
Comp 1A Status	Статус компрессора 1	Категориальная
Comp 2A Status	Стасус компрессора 2	Категориальная
Peregrev u69 Sair Темп degc	Перегрев	Непрерывная
Датч нар воздуха degc	Температура наружного воздуха	Непрерывная

В холодильных установках хранятся продукты, центральный холодильный агрегат обеспечивает установки хладагентом. Внешний вид холодильной установки представлен ниже. Это там, где лежит колбаса и йогурты с ценниками.





Постановка задачи

Все визуализации должны быть подписаны, минимум заголовок и легенда (если требуется).

1. Загрузите исходный датасет

   pd.read_csv()

2. Сделайте необходимые преобразования при загрузке датасета, чтобы данные выглядели корректно.

   https://pandas.pydata.org/docs/reference/api/pandas.read_csv.html, корректно обработайте разделители, NA значения, кодировку, исключите ненужные строки, определите заголовок, при необходимости игнорируйте ошибки в кодировке или подправьте исходный файл.

3. Очистите заголовки колонок от ненужной информации, оставьте только имя устройства и название параметра:

```
data.columns[:10]
```

```
Index(['time', '10CT G PBP -1/+1:   --- ЕКС состояние',
       '10CT G PBP -1/+1:   u09 S5 Темп', '11CT G OVZ +5/+8:   u69 Sair Темп',
       '11CT G OVZ +5/+8:   u09 S5 Темп',
       '11CT G OVZ +5/+8:   --- ЕКС состояние',
       '12CT G GSR +2/+4:   u09 S5 Темп', '12CT G GSR +2/+4:   u69 Sair Темп',
       '12CT G GSR +2/+4:   --- ЕКС состояние',
       '13CT G PBP -1/+1:   u69 Sair Темп'],
      dtype='object')
```

   Если при загрузке датасета у вас получились многоуровневые заголовки, можно удалить незначащие уровни методом df.columns.droplevel():

```
data.columns[:4]
```

```
MultiIndex([(                                'Name', ...),
            ('10CT G PBP -1/+1:   --- ЕКС состояние', ...),
            (      '10CT G PBP -1/+1:   u09 S5 Темп', ...),
            (    '11CT G OVZ +5/+8:   u69 Sair Темп', ...)],
           )
```

```
data.columns = data.columns.droplevel([1,2,3])
data.columns[:4]
```

```
Index(['Name', '10CT G PBP -1/+1:   --- ЕКС состояние',
       '10CT G PBP -1/+1:   u09 S5 Темп', '11CT G OVZ +5/+8:   u69 Sair Темп'],
      dtype='object')
```

   Для переименования индексов можно использовать метод df.rename(columns={<исходное имя>: <новое имя>})

4. Выведите в консоль информацию о датасете: его размер, название колонок, их типы данных

   df.info()

5. Преобразуйте колонку со временем в формат datetime, сделайте ее индексом и отсортируйте (даже если датасет уже отсортирован)

   pd.to_datetime(), df.index = df[<колонка>], df.sort_index()

6. Посчитайте количество пропусков в каждой колонке и визуализируйте их

   df.isna().sum(), для визуализации можно использовать модуль missingno и метод matrix(df)

7. Как оптимизировать типы данных в колонках? Насколько меньше станет размер датафрейма после оптимизации типов?

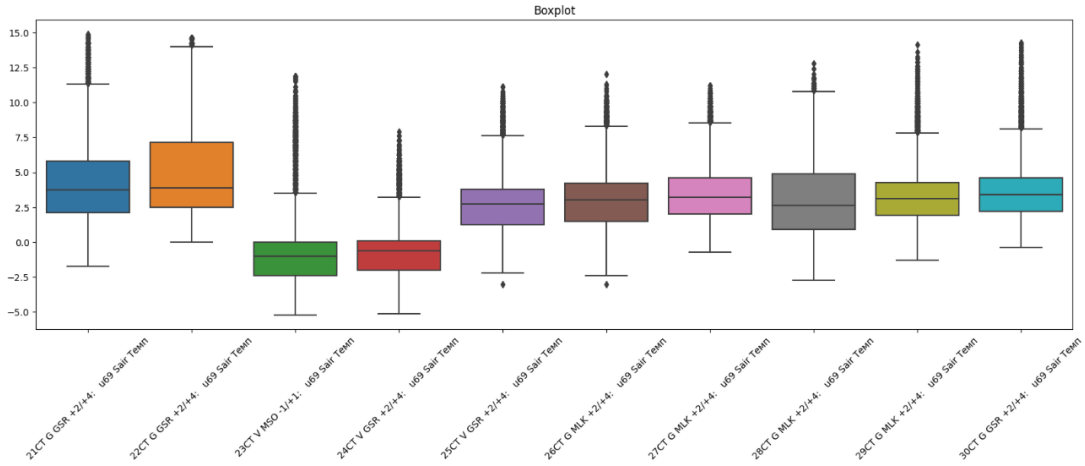
   Приведите к типу int колонки с целыми значениями, используйте df.astype().

   Приведите к типу category колонки с категориальными признаками (признаки, которые принимают ограниченное количество значений, например до 10).

Сравните замеры получившихся датасетов, оригинальный датасет не перезаписывайте.

8. Дайте табличное и графическое статистическое описание признакам, содержащим параметр «Sair» для устройств 21CT, 22CT,…, 30CT.

   df.describe(), sns.boxplot().



   Можно создать отдельный список колонок, которые вы будете анализировать, например «col_Sair». Для его создания можно использования list comprehension

```
col_Sair = [col for col in data.columns if col.find(' ')>-1 and 21 <= int(col[1:3]) <= 30]
col_Sair
```

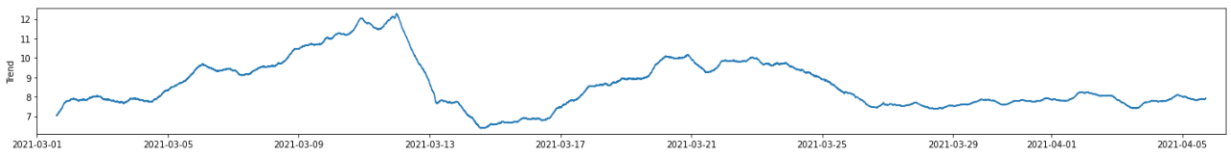
```
['21CT G GSR +2/+4:   u69 Sair Temn',
 '22CT G GSR +2/+4:   u69 Sair Temn',
 '23CT V MSO -1/+1:   u69 Sair Temn',
 '24CT V GSR +2/+4:   u69 Sair Temn',
 '25CT V GSR +2/+4:   u69 Sair Temn',
 '26CT G MLK +2/+4:   u69 Sair Temn',
 '27CT G MLK +2/+4:   u69 Sair Temn',
 '28CT G MLK +2/+4:   u69 Sair Temn',
 '29CT G MLK +2/+4:   u69 Sair Temn',
 '30CT G GSR +2/+4:   u69 Sair Temn']
```

9. Ресемплируйте датасет по медианному значению за 4 минуты, отобразите на линейном графике значение признаков col_Sair. График должен быть читабельным, иметь заголовок и легенду, можете выбрать другое значение ресэмплирования, чтобы график лучше читался.

   Используйте любые модули (df.plot(), sns, plt, plotly). Ресемплированный датасет сохранять не нужно, просто отобразить.

10. Постройте для 2-3 признаков из col_Sair сглаженный график поведения признаков во времени, чтобы был виден тренд

    df.rolling()



11. Постройте на одном графике три графика: оригинальный, ресемплированный и сглаженный.

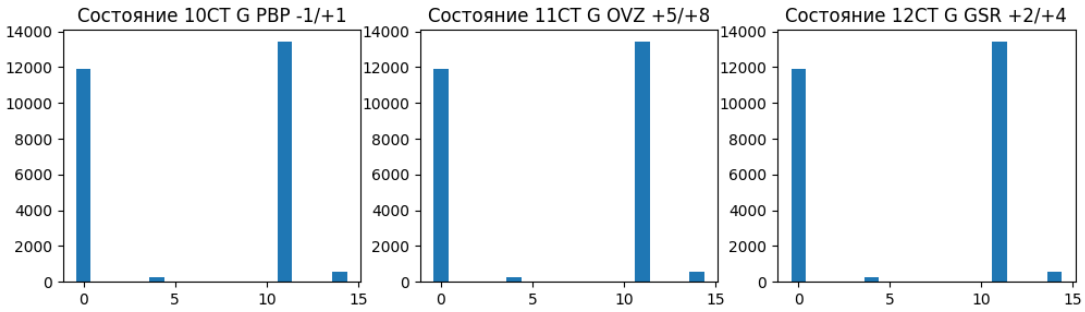
    Параметры ресемплирования и сглаживания выберите любые. Подберите временной интервал, чтобы разница была видна.

12. Отобразите гистограмму по количеству одновременно включенных устройств. Признак состояния представлены в виде «<устройство>: --- EKC состояние», если признак принимает значение 0, то считаем устройство включенным.

```
EKCstate_Map = data[[col for col in data.columns if col.find([illegible]]].applymap(lambda x: 1 if x==0 else 0) тут можно покрасивее сделать
```

```
EKCstate_Map.sum(axis=1).value_counts().sort_index().plot.bar(figsize=(5,5),title='Количество одновременно включенных горок')
```

13. Постройте для нескольких признаков состояния устройств гистограммы, которые покажут распределение этих устройств по состояниям.



Используйте любой plt.bar или sns.barplot() и plt.subplots()

14. Постройте матрицу корреляции для признаков col_Sair

    Получение таблицы с корреляцией: df.corr(), отрисовку удобно делать через sns.heatmap()

15. Творческое: сформулируйте гипотезу (вопрос) и проверьте ее (ответьте на него), подтвердив выводы визуализацией или статистиками.

    Например, можно задаться вопросом, как ведет себя температура **u09 S5** в устройстве при разных состояниях устройства.

```
cols = ['11CT G OVZ +5/+8:   u09 S5 Темп',
        '11CT G OVZ +5/+8:   --- EKC состояние']
```

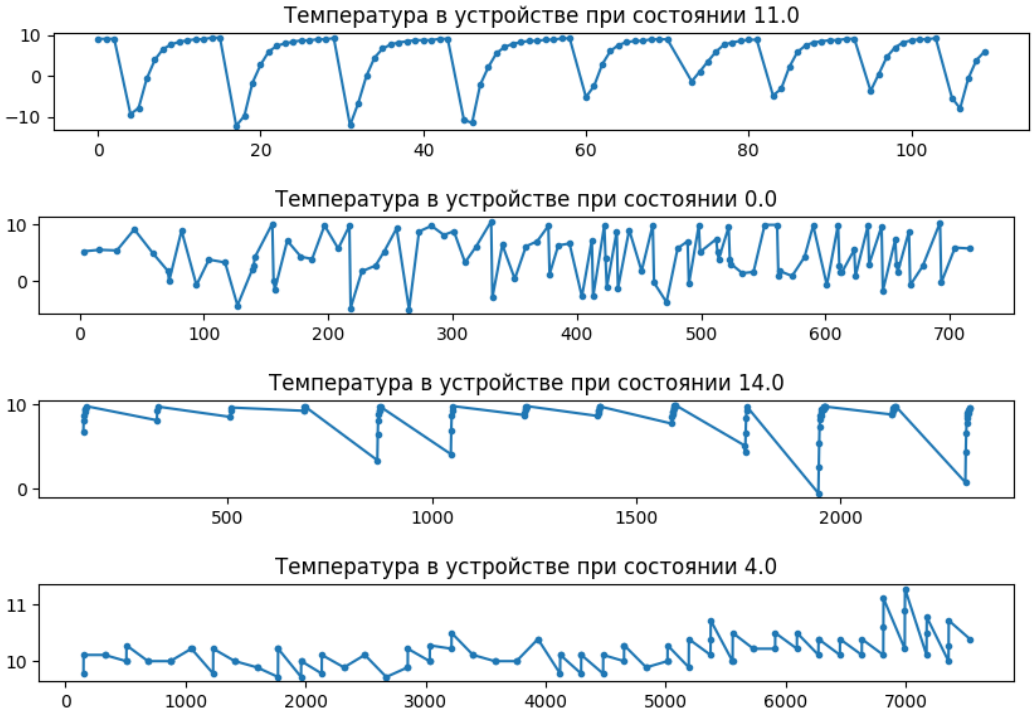
```
data['11CT G OVZ +5/+8:   --- EKC состояние'].value_counts()
```

```
11CT G OVZ +5/+8:   --- EKC состояние
11.0    19682
0.0      5180
14.0      909
4.0       353
10.0       19
Name: count, dtype: int64
```

```
states = data['11CT G OVZ +5/+8:   --- EKC состояние'].value_counts()
```

Только на основании графиков сделать выводы сложно:

```
for state in [11.0, 0.0, 14.0, 4.0]:
    (
        data[data['11CT G OVZ +5/+8:   --- EKC состояние']==state]
        ['11CT G OVZ +5/+8:   u09 S5 Темп'][:100]
        .plot(figsize=(10,1), marker='.')
    )
    plt.title(f"Температура в устройстве при состоянии {state}")
    plt.show()
```



Но статистика говорит о том, что температура в режимах, отличных от 0.0 или растет или меняется незначительно.

```
df_stats = pd.DataFrame(columns = states.index)

for state in states.index:
    df = data[data['11CT G OVZ +5/+8:   --- ЕКС состояние']==state].copy()
    df['index_diff'] = df['index'].diff()
    df['temp_diff'] = df['11CT G OVZ +5/+8:   u09 S5 Темп'].diff()
    df_stats[state] = df.query("index_diff==1.0")['temp_diff'].describe()

df_stats
```

11CT G OVZ +5/+8: --- ЕКС состояние	11.0	0.0	14.0	4.0	10.0
count	16768.000000	2244.000000	764.000000	208.000000	18.000000
mean	2.754904	-7.349211	1.105471	0.229760	0.111111
std	2.457571	4.557101	1.728369	0.193457	0.347595
min	-18.720000	-21.220000	-10.110000	0.000000	-0.280000
25%	1.000000	-10.890000	0.330000	0.000000	0.000000
50%	2.000000	-8.470000	0.670000	0.280000	0.000000
75%	4.390000	-3.890000	1.500000	0.345000	0.000000
max	11.500000	4.610000	10.610000	0.670000	1.390000