Предобработка данных и статистические тесты

Предобработка данных

Данные, которые содержат пропуски, дубликаты, неверный формат хранения и т.д. называются «грязными». Прежде чем начинать работу с данными, необходимо их очистить, то есть обработать пропуски, дубликаты, выбросы, и при этом не потерять важную информацию.

Далее рассмотрим основные методы поиска и исправления:

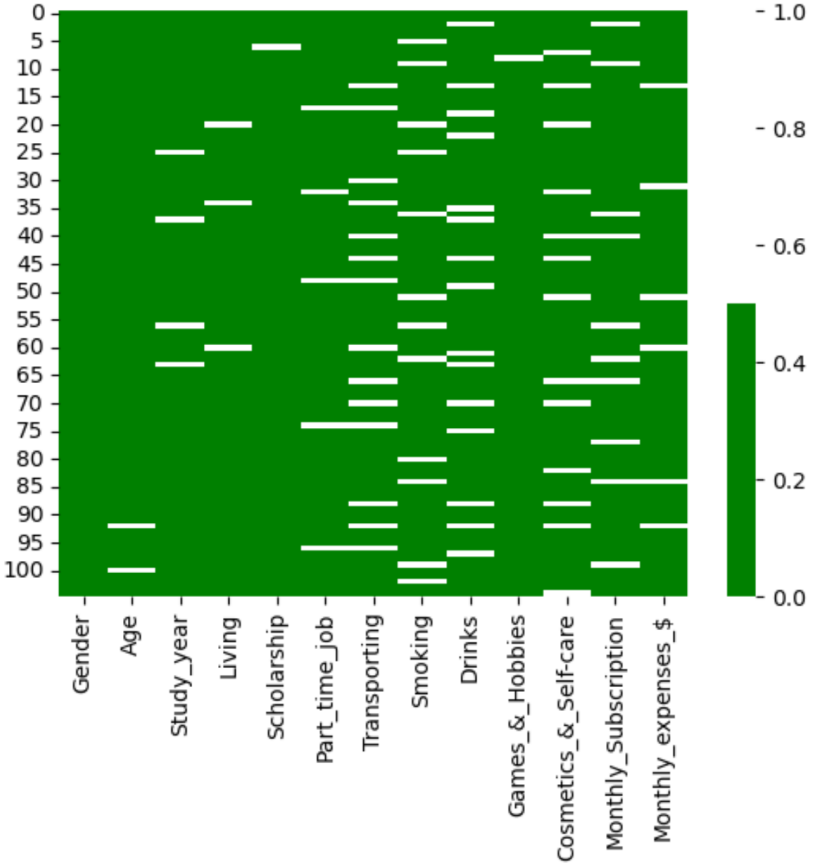
- отсутствующих данных;
- нетипичных данных – выбросов;
- неинформативных данных – дубликатов;
- несогласованных данных – одних и тех же данных, представленных в разных регистрах или форматах.

Рассмотрим три метода обнаружения пропусков в данных.

Уже используемый нами в предыдущих работах – функция isna(), которая возвращает True, если значение пропущено. Обратной ей является функция notna(), которая возвращает True, если значение не пропущено.

Еще один метод обнаружения пропусков – это построение тепловой карты. Пример с использованием библиотеки seaborn:

```
colors = ['green','white'] # пропущенные значения - белые
sns.heatmap(data.isna(), cmap=sns.color_palette(colors))
plt.show()
```



Тепловую карту удобно использовать, когда признаков в данных не очень много.

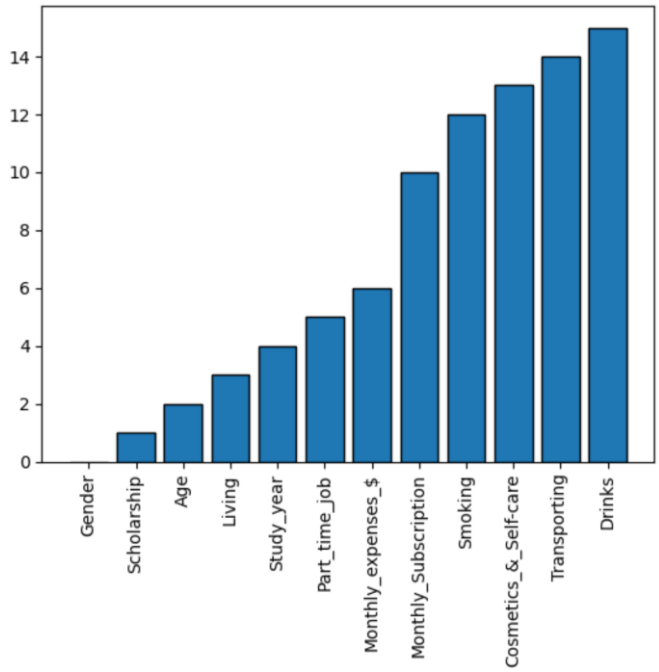
Следующий способ – это процентное содержание пропусков. Удобно использовать, когда данных очень много.

```
for column in data.columns:
    missing = np.mean(data[column].isna()*100)
    print(f" {column} : {round(missing,1)}%")
```

```
Gender : 0.0%
Age : 1.9%
Study_year : 3.8%
Living : 2.9%
Scholarship : 1.0%
Part_time_job : 4.8%
Transporting : 13.3%
Smoking : 11.4%
Drinks : 14.3%
Games_&_Hobbies : 1.0%
Cosmetics_&_Self-care : 12.4%
Monthly_Subscription : 9.5%
Monthly_expenses_$ : 5.7%
```

Еще один хороший способ визуализации для наборов с большим количеством признаков – столбчатая диаграмм пропусков. Можно изобразить количество пропусков по признакам, а можно наоборот – количество значений без пропусков. Пример вывода пропущенных значений в признаках:

```
d = dict()
for column in data.columns: #проход по столбцам
    missing = data[column].isna().sum() #кол-во пропущенных значений в столбце
    without_missing = len(data[column]) - missing #кол-во значений в столбце
    d[column] = missing #словарь признак : кол-во значений
sorted_dict = sorted([(value, key) for (key, value) in d.items()])
sort = dict(sorted_dict)
plt.bar(sort.values(),sort.keys(), edgecolor = 'black')
plt.xticks(rotation = 90)
plt.show()
```



Для каждого конкретного набора данных используются подходящие именно ему методы работы с пропусками или используются комбинации методов.

Далее рассмотрим самые распространенные методы работы с пропущенными значениями в данных.

Самый простой, но не самый лучший способ – это просто удалить все объекты, которые содержат пропуски. В таком случае можно потерять большое количество полезной информации. Удалить записи можно с помощью метода dropna().

Так же можно удалить признак, который содержит много пропусков, только если признак неинформативен для решаемой задачи. Пример:

```
data.drop('Drinks', axis = 1,inplace=True)
```

Следующий способ – это заполнение пропусков в данных. Это можно сделать несколькими способами, например: для числовых признаков можно заполнить пропуски средним значением или медианным, для категориальных данных пропуски можно заполнить самым часто встречающимся значением.

Заполнить пропуски можно с помощью метода fillna(). Пример заполнения признака возраста медианным значением:

```
median_age = data.Age.median()
data.Age.fillna(median_age, inplace=True)
```

Пример заполнения самым часто встречающимся значением категориального признака:

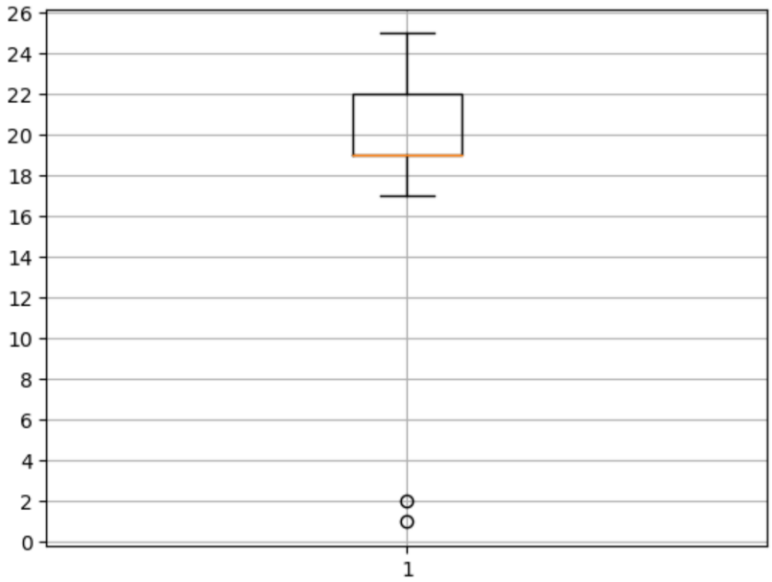
```
living = data.Living.describe().top
data.Living.fillna(living, inplace=True)
```

Можно заменить недостающие значения некоторым значением по умолчанию, таким образом мы сохраняем информацию о пропусках, она может иметь свою ценность.

Еще один метод заполнения пропусков – интерполяция. Интерполировать данные можно с использованием метода датафрема interpolate().

Далее рассмотрим методы поиска выбросов в данных. Выбросы – это значения, которые сильно отличаются от остальных объектов. Выбросы могут бать как реальными значениями, например, как рост самого высокого человека в мире, а могут быть просто ошибками.

Для нахождения выбросов можно использовать гистограмму и boxplot. Используем boxplot для определения выбросов признака возраста у студентов.



Определяются два выброса, которые являются ошибками, так как исходные данные – это данные о студентах, студенту не может быть 1 или 2 года. Можно использовать метод describe(). Для категориальных признаков можно построить гистограмму и по ней выяснить, есть ли выбросы.

Так же для поиска выбросов используется кластеризация. Алгоритмы кластеризации будут рассмотрены на следующих практических занятиях.

Работа с выбросами похожа на работу с пропущенными значениями. Все зависит от специфики задачи. Можно выбросы удалить, можно заменить на некоторое значение.

Для нахождения дублированных строк в наборе данных можно использовать метод duplicated(), который возвращает True, если строка дублируется. При необходимости удалить дублированные строки можно методом drop_dublicates().

Форматы записей в наборе данных могут отличаться. Например, форма записей дат, опечатки при вводе данных, разные регистры. Все это необходимо приводить к единому формату.

Найти такие ошибки можно с помощью метода value_counts(). Он подсчитывает количество уникальных значений.

```
1  data.Gender.value_counts()
```

```
Male       32
Female     29
Mal         2
Femal       1
```

Также используется метод unique(). Он возвращает уникальные значения.

```
1  data.Gender.unique()
```

```
array(['Female ', 'Male ', 'Femal ', 'Mal '], dtype=object)
```

Очевидно, что гендера всего два, это Female и Male. Два оставшихся значения являются опечатками, их необходимо заменить на верные варианты записи. Также с помощью метода unique() видим, что присутствует в записи лишний пробел в конце. Для удаления пробелов можно использовать метод str.strip(). Можно указать набор символов, которые необходимо удалить. Если не передавать в эту функцию аргументы, то он удалит пробелы в начале и в конце строки.

```
1 data['Gender']=data['Gender'].str.strip()
```

```
1 data.Gender.unique()
```

```
array(['Female', 'Male', 'Femal', 'Mal'], dtype=object)
```

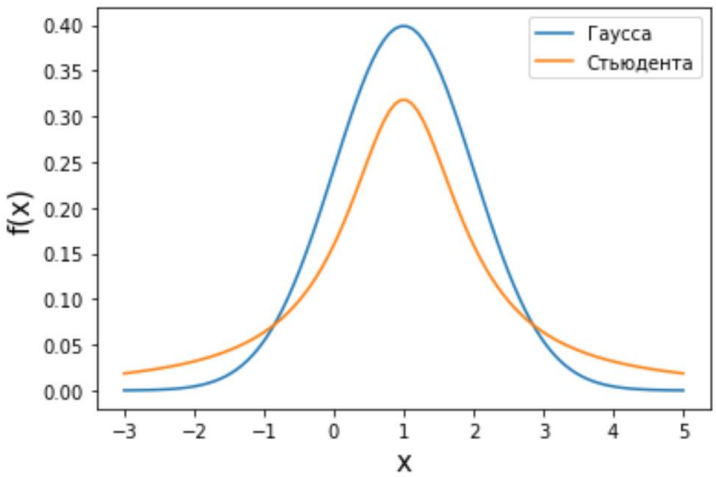
Еще используется .str.replace(‘ ‘,’’), в котором указывается какой символ на какой можно заменить.

Когда очень много уникальных элементов в признаке, удобно использовать поиск опечаток с помощью расстояния между словами. Чем меньше расстояние между словами, тем больше они похожи.

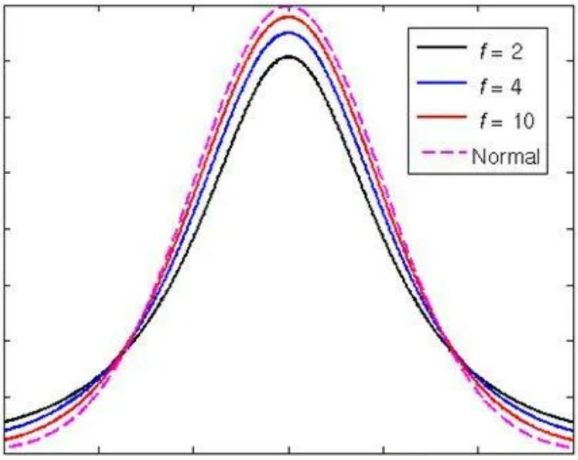
Тесты

Основываясь на центральной предельной теореме, мы узнали, как ведет себя распределение выборочных средних для разных длин выборок n. Чем больше элементов в выборке, тем лучше среднее выборочных средних будет приближено к среднему генеральной совокупности. Чем меньше выборка, тем чаще мы получаем выборочные средние, которые далеко отклоняются от среднего генеральной совокупности. Если $n < 30$, то нарушается предположение о том, что выборочные средние будут иметь нормальное распределение.

Когда $n < 30$, используется распределение Стьюдента. Распределение Стьюдента было представлено в предыдущей практической работе, рассмотрим его чуть более подробно.



t-распределение имеет более высокие хвосты, это значит, что наблюдения с большей вероятностью попадают за пределы $\pm 2\sigma$ (2 стандартных отклонения для выборок) от среднего генеральной совокупности, чем в нормальном распределении. Когда берем средние для маленьких выборок, большие отклонения от среднего генеральной совокупности мы получаем с большей вероятностью. Важным параметром распределения Стьюдента является число степеней свободы df, которое зависит от длины выборки n и вычисляется по формуле $df = n - 1$. Чем больше df (т.е. чем больше длина выборки), тем больше распределение Стьюдента стремиться к нормальному.



Если мы используем распределение Стьюдента, то значения p-уровня значимости будут выше, нежели при нормальном распределении, что не дает нам отклонять нулевую гипотезу в различных исследованиях. Число степеней свободы – это количество элементов выборке, которые могут варьироваться при расчете некоторого статистического показателя. Например, мы имеем выборку из 5 элементов и знаем ее среднее значение, тогда нам достаточно знать среднее значение и только 4 элемента выборки, чтобы однозначно определить, чему равно 5-е значение. То есть 4 элемента могут варьироваться, а 5 значение всегда будет однозначно определено. Важно понимать, сколько независимых элементов мы использовали для расчета того или иного показателя. Например, для t-распределения с 30 степенями свободы и для t-распределения с 3 степенями свободы результаты проверки гипотез будут кардинально отличаться.

Очень частый метод, применяемый в статистическом анализе – сравнение средних значений двух выборок. Целью данного метода является понимание того, действительно ли средние значения выборок отличаются друг от друга или же это статистические колебания. Самая популярная мера – это **t-критерий Стьюдента**, которая оценивает, есть ли разницы между средними значениями двух выборок. При сравнении средних выборок длины n < 30,

важным условием является нормальность распределения двух выборок. Так же необходимо, чтобы дисперсии двух выборок были примерно одинаковы (гомогенность дисперсий). Если длина выборок большая, то t-критерий Стьюдента дает достаточно точные результаты, не смотря на то, что распределение выборок отличается от нормального.

Существует три основных типа **t-теста,** которые применяются в зависимости от исходных данных:

1) **Одновыборочный t-критерий**, где сравнивается среднее одной выборки с эталонным значением.
2) **t-критерий Стьюдента для независимых выборок.** Сравнивается среднее значение двух несвязанных выборок. Например, мы взяли 50 котов и разделили их на две группы по 25 котов, одна группа будет есть новый корм, а другая нет. Это пример несвязанных выборок.
3) **Парный t-критерий Стьюдента (для зависимых выборок).** Сравниваются средние значения двух зависимых выборок. Например, у нас есть 25 котов в момент времени t, далее этим котам дается корм и в момент времени t+dt измеряется их средний вес. Получаем две выборки, одну в момент времени t, а вторую в момент времени t+dt. Эти выборки зависимы.

Рассмотрим пример сравнения среднего веса групп котов по критерию t-Стьюдента, где одна группа потребляла новый корм, а вторая нет.

Нулевая гипотеза – новый корм не работает, исследуемые выборки принадлежат к одной генеральной совокупности.

Альтернативная гипотеза – прием корма повлиял на котов, выборки принадлежат к разным генеральным совокупностям.

	Среднее	Стандартное отклонение	Кол-во котов в группе
Группа 1	8	2.1	25
Группа 2	5.5	1.9	25

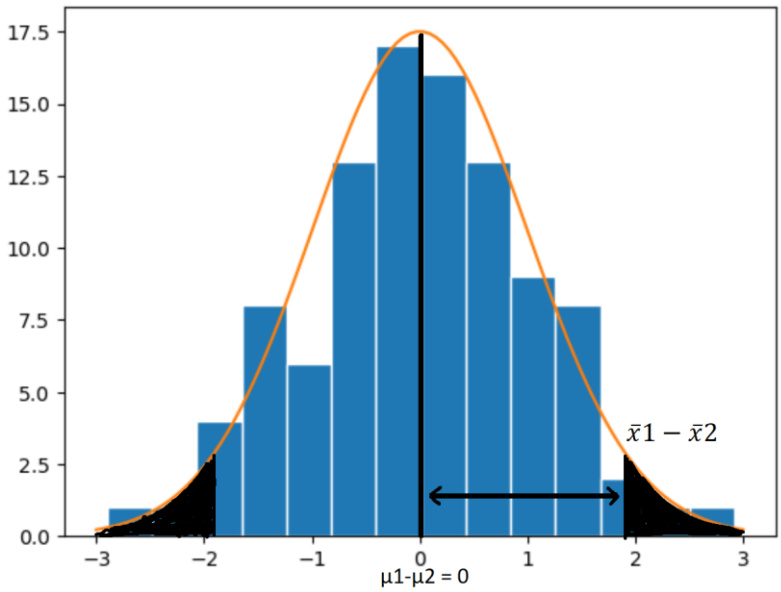
Являются ли эти различия статистически значимыми?

Помня правило центральной предельной теоремы, мы можем построить разности выборок и получить нормальное распределение средних весов котов. Но мы учитываем, что выборки у нас маленькие, тогда правильнее будет сказать, что мы получим распределение Стьюдента с числом степеней свободы $df = n_1 + n_2 - 2$. Учитывая, что изначально принимается нулевая гипотеза и выборки принадлежат к одной генеральной совокупности, тогда среднее такого распределения будет равно 0. Так как разность средних выборок стремится к разности средних двух генеральных совокупностей, а так как это одна и та же генеральная совокупность, то разность будет равна 0. Стандартное отклонение sd или стандартная ошибка SE такого распределения рассчитывается по следующей формуле:

$$\sqrt{\frac{sd_1^2}{n_1} + \frac{sd_2^2}{n_2}}$$

Это стандартное отклонение первой выборки в квадрате, деленное на количество элементов в выборке, плюс стандартное отклонение второй выборки в квадрате, деленное на количество элементов во второй выборке, и все это под корнем. То есть обе выборки влияют на стандартную ошибку.

Основываясь на всем этом, мы можем рассчитать, на сколько разность средних наших выборок отклонилась от разности средних генеральной совокупности. То есть найти p-уровень значимости – вероятность того, что мы получим такие или еще более выраженные различия разности средних значений при условии того, что верна нулевая гипотеза.



Формула t-значения выглядит следующим образом:

$$t = \frac{(\bar{x}_1 - \bar{x}_2) - (\mu_1 - \mu_2)}{\sqrt{\frac{sd_1^2}{n_1} + \frac{sd_2^2}{n_2}}} = \frac{(\bar{x}_1 - \bar{x}_2)}{\sqrt{\frac{sd_1^2}{n_1} + \frac{sd_2^2}{n_2}}}$$

где μ_1 и μ_2 – среднее значение генеральных совокупностей. Так как это одна и та же генеральная совокупность, то $\mu_1 - \mu_2 = 0$.

Эта формула аналогична формуле стандартизации выборки:

$$z_i = \frac{x_i - \bar{x}}{sd}$$

В стандартизированной выборке стандартное отклонение равно 1. Если внимательно посмотреть формулу t-значения, можно увидеть, что мы также отнимаем от значения выборки среднее, и делим на стандартное отклонение, которое является стандартной ошибкой на основе двух выборок. То есть в результате мы получаем отклонение от среднего значения в стандартных отклонениях.

Основываясь на числе степеней свободы и на полученном t-значении, мы можем рассчитать p-уровень значимости, который скажет нам какова вероятность получить такое или еще более выраженное различие между средними выборок, если верна нулевая гипотеза.

Вернемся к котам. Рассчитаем число степеней свободы и t-значение.

$$df = 25 + 25 - 2 = 48$$

$$t = \frac{8 - 5.5}{\sqrt{\frac{2.1^2}{25} + \frac{1.9^2}{25}}} = \frac{2.5}{\sqrt{0.177 + 0.14}} \approx 4$$

Получается, что мы отклонились от разности средних генеральной совокупности на 4 сигмы. Для того, чтобы найти p-уровень значимости можно

воспользоваться сайтом https://gallery.shinyapps.io/dist_calc/. p-значение равно 0.000218, следовательно, мы обнаружили статистически значимые различия, нулевая гипотеза отклоняется и коты принадлежат к разным группам, то есть корм оказывает влияние на их вес. Если разность средний достаточно большая, а стандартная ошибка маленькая, то значение t-критерия будет весьма большим. Это говорит о том, что средние выборок сильно отличаются друг от друга.

Проверка распределений на нормальность (критерий Шапиро-Уилка) с использованием scipy.stats:

```
import scipy.stats as sts
```

```
res1 = sts.shapiro(a)
res2 = sts.shapiro(b)
print(res1,'\n',res2)
```

```
ShapiroResult(statistic=0.9675695896148682, pvalue=0.5842580795288086)
 ShapiroResult(statistic=0.9567145705223083, pvalue=0.35293325781822205)
```

Тест дает p-значение выше 0.05, следовательно, выборки имеют нормальное распределение.

Проверка гомогенности дисперсии. Критерий Бартлетта – статистический критерий, позволяющий проверять равенство дисперсий нескольких (двух и более) выборок. Нулевая гипотеза предполагает, что рассматриваемые выборки получены из генеральных совокупностей, обладающих одинаковыми дисперсиями. Так же предполагается, что выборки распределены нормально.

```
res = sts.bartlett(a,b)
print(res)
```

```
BartlettResult(statistic=0.1991075475776091, pvalue=0.6554421742428356)
```

p-уровень превышает 0.05, следовательно, дисперсии выборок примерно одинаковы. Можем переходить к t критерию Стьюдента.

```
t_res = sts.ttest_ind(a,b)
print(t_res)
```

```
Ttest_indResult(statistic=-11.320889622349442, pvalue=3.739392839235914e-15)
```

p-значение намного ниже 0.05, следовательно нулевая гипотеза отвергается, выборки принадлежат к разным генеральным совокупностям и их средние значения различны.

Если распределение выборки очень сильно отличается от нормального, то можно использовать непараметрический критерий Манна-Уитни.

Биномиальный тест

Биномиальный тест показывает вероятность выполнения предполагаемой гипотезы при двух возможных исходах. Одно из распространенных применений биномиального теста – это случай, когда нулевая гипотеза

состоит в том, что две категории имеют одинаковую вероятность (например, подбрасывание монеты).

Рассмотрим пример.

Мы хотим узнать, знает ли о нашем магазине 50% населения города? Половину всех жителей опросить мы не можем. Но мы можем провести опрос небольшой выборки людей, например, 20. Из 20 человек, 6 человек посещали магазин (0.3), 14 человек впервые о нем слышат (0.7). Вопрос, является ли доля 0.3 нормальным результатом при условии, что о магазине знают 50 % населения?

Нулевая гипотеза – 50 % населения города знают о магазине.

Воспользуемся библиотекой

```
1 bin_test = sts.binom_test(6, n = 20, p = 0.5)
2 print(bin_test)
```

```
0.11531829833984375
```

p-значение больше, чем 0.05, следовательно, нулевая гипотеза не отклоняется.

Критерий хи-квадрат (χ^2) Пирсона

Ранее мы работали с нормальным распределением, но на практике точно не известно по какому закону распределены данные. В связи с этим решается вопрос о соответствии эмпирического распределения (полученная выборка) к теоретическому распределению вероятностей (которые мы определяем самостоятельно) и называется такое соответствие **критерием согласия**.

Одним из первых был изобретён критерий $\boldsymbol{\chi^2}$ **(хи квадрат)**, который остаётся популярным до сих пор. В основном представленный метод применяется для анализа таблиц сопряжённости, которые содержат категориальные данные (пол, производитель смартфона и т.д.).

Введём обозначение частот:

- Наблюдаемые частоты (**O**bserved) — количество объектов в каждой категории (данные из выборки)
- Ожидаемые частоты (**E**xpected) — количество наблюдений, при условии выполнения нашего предположения о распределении.

Критерий $\boldsymbol{\chi^2}$ Пирсона — это непараметрический метод, который позволяет оценить статистическую значимость различий двух или нескольких относительных показателей (частот, долей).

Ниже представлена формула Хи-квадрат для набора значений с набором эталонов:

$$\chi_n^2 = \sum_{i=1}^{n} \frac{(O_i - E_i)^2}{E_i},$$

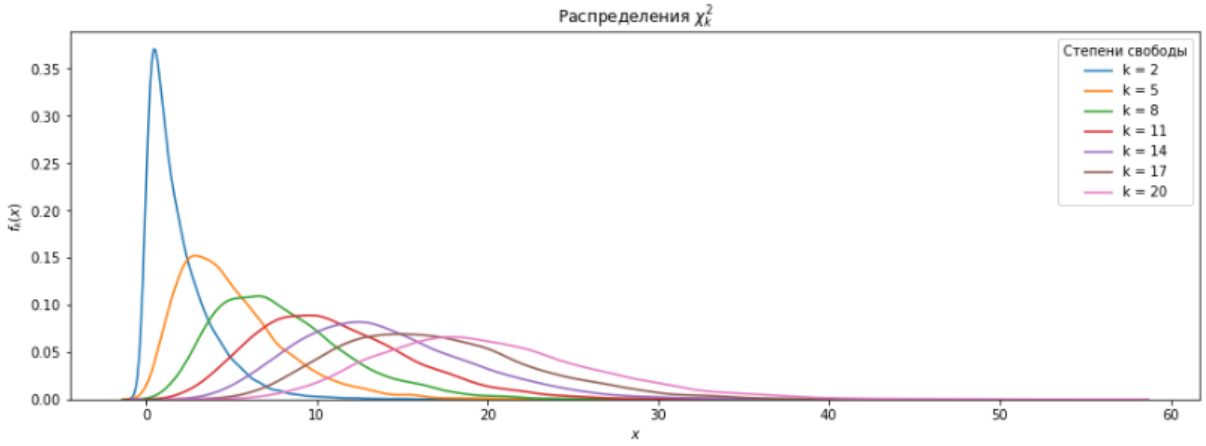
где O — наблюдаемое значение, E — ожидаемое значение.

С её помощью мы можем охарактеризовать одним числом на сколько сильно наблюдаемые частоты признака отклоняются от ожидаемых частот признака — это называется **расстояние Пирсона**.

Если частоты действительно соответствуют ожидаемым, то значение статистики Хи-квадрат будет относительно не большим (отклонения находятся близко к нулю). Большое значение статистики свидетельствует в пользу существенных различий между частотами.

Распределение хи квадрат – это семейство распределений, каждое из которых зависит от параметра степеней свободы. Или более формально: **распределение хи-квадрат с k степенями свободы** — это распределение суммы квадратов k независимых стандартных нормальных случайных величин.

На графике ниже представлено распределение хи-квадрат при многократном вычислении расстояния Пирсона и откладыванием его по оси абсцисс, а по оси ординат записывается частота встречаемости подсчитанного значения. С увеличением степеней свободы распределение хи-квадрат стремится к нормальному. Это объясняется действием центральной предельной теоремы, согласно которой сумма большого количества независимых случайных величин имеет нормальное распределение.



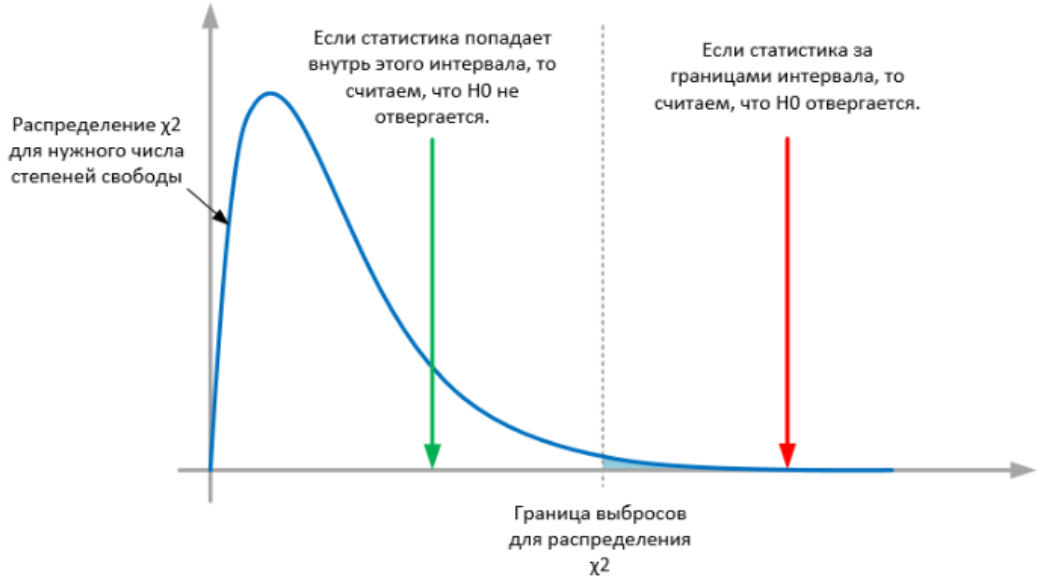
Критерий Пирсона применяется в тестах:

- На **гомогенность** — непараметрический, одновыборочный тест, который сопоставляет эмпирическое распределение признака с теоретическим распределением;
- На **независимость** — непараметрический, одновыборочный тест, который проверяет наличие связи между двумя категориальными переменными.

В тесте на гомогенность ставятся следующие гипотезы:

- H_0 между наблюдаемым распределением и эталонным различий нет
- H_1 между наблюдаемым распределением и эталонным есть различия

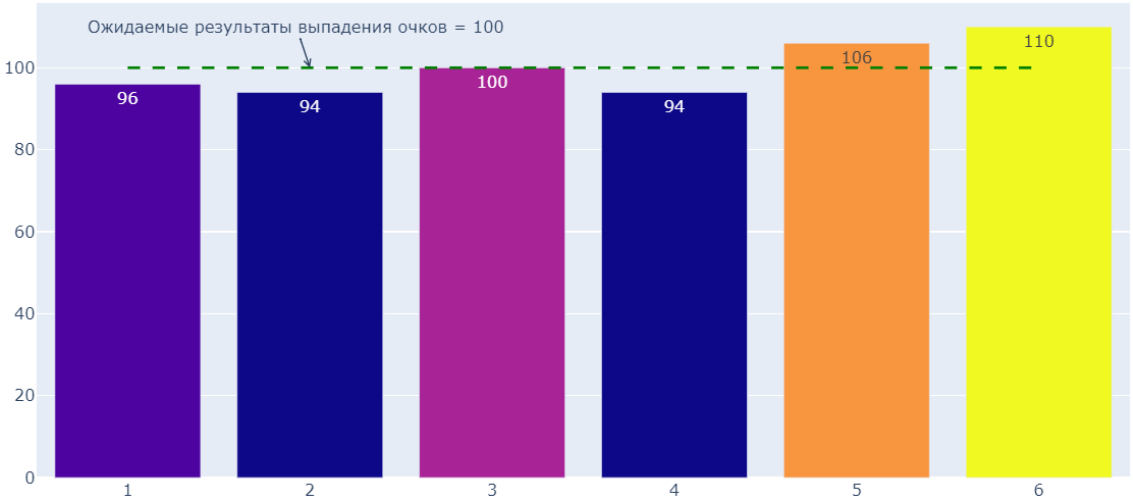
Уже на данном этапе можно воспользоваться специальной таблицей значений хи-квадрат для различных p, в зависимости от числа степеней свободы. То есть подсчитать расстояние Пирсона, сопоставить его с числом степеней свободы и отбросить или применить нулевую гипотезу. Или рассчитать p-уровень значимости.



Рассмотрим простой пример с игральной костью. К примеру, мы подбрасываем кубик 600 раз, тогда вероятность выпадения любой стороны равна 1/6, следовательно ожидание выпадения каждой из сторон равна 100.

```python
attempts = 600
np.random.seed(2718281828)
throws = pd.DataFrame(np.random.randint(1,7,attempts))
print(throws.head(),'\n') #вывод первых 5 бросков кубика
throws = throws.groupby(throws.columns.tolist(),as_index=False).size()
throws = throws.rename({0:'Points','size':'Observed'}, axis = 'columns')
throws['Expected'] = 100
throws.head(6) #количество выпавших значений кубика
```

	Points	Observed	Expected
0	1	96	100
1	2	94	100
2	3	100	100
3	4	94	100
4	5	106	100
5	6	110	100




```python
import scipy.stats as st
#попробуем расчитать критерий Хи-вадрат самостоятельно:
#Рассчитываем число степеней свободы
k = len(throws.Points) - 1 #так как в кубике 6 вариантов выпадения значений
#то число степеней свободы 6-1 = 5

#следуя формуле Хи-квадрат:
difference = throws.Observed - throws.Expected
throws['Difference'] = difference #записываем разность между полученными и ожидаемыми значениями
throws['SquaredDif'] = throws['Difference']**2 #возводим их в квадрат
throws['SquaredDif/Epxected'] = throws['SquaredDif']/throws.Expected #делим полученное значение на ожидание
print(throws)
statistic = throws['SquaredDif/Epxected'].sum() #суммируем последний столбец, получая статистику
print('\nСтатистика: ',statistic)

#воспользуемся функцией распределения cdf и получим значение
#вероятности получить похожее значение или выше найденной статистики
pval = 1 - st.chi2.cdf(statistic, k)
print('Вероятность получить похожее значение или выше, исходя из полученной статистики: ',pval)

st.chisquare(throws['Observed'], throws['Expected']) #встроенный метод Хи квадрат
```

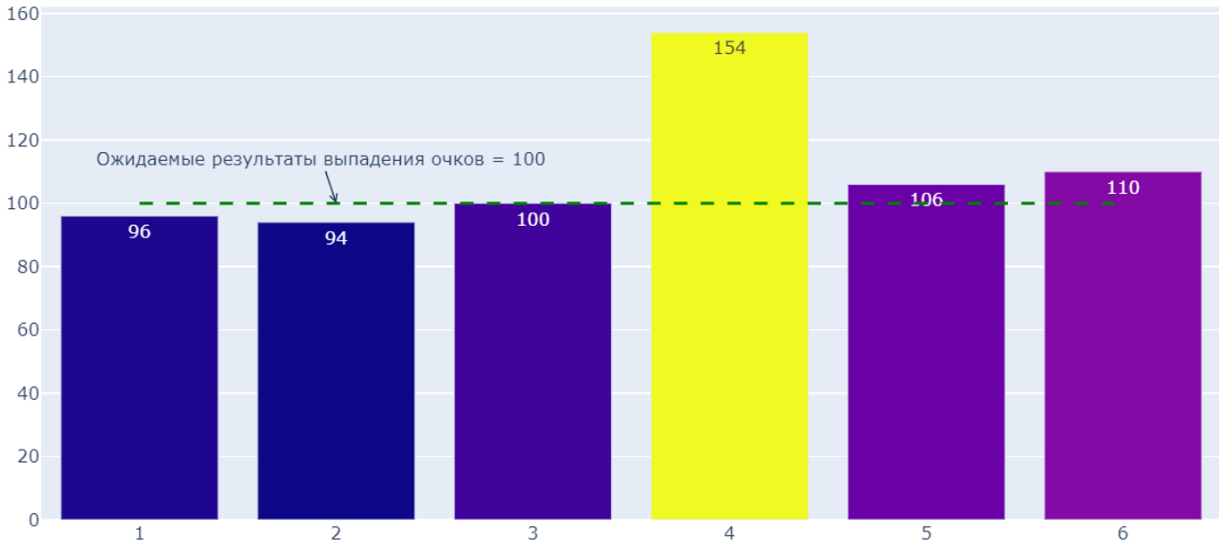
```
   Points  Observed  Expected  Difference  SquaredDif  SquaredDif/Epxected
0       1        96       100          -4          16                 0.16
1       2        94       100          -6          36                 0.36
2       3       100       100           0           0                 0.00
3       4        94       100          -6          36                 0.36
4       5       106       100           6          36                 0.36
5       6       110       100          10         100                 1.00

 Статистика:  2.24
Вероятность получить похожее значение или выше, исходя из полученной статистики:  0.8150376319793067

Power_divergenceResult(statistic=2.24, pvalue=0.8150376319793067)
```

Основываясь на p-value делаем вывод, что распределение равномерное.

Представим, что кубик «заговорён» на более частое выпадение числа 4:

```
#представим, что кубик "заговорён" на выпадение  4 в большей степени.
throws4 = throws.copy(deep=True)
throws4 = throws4.drop(['Difference','SquaredDif','SquaredDif/Epxected'], axis = 1)
throws4.at[3,'Observed'] = throws4['Observed'][3] + 60
throws4['Expected'] = 110
print(throws4)
```

```
   Points  Observed  Expected
0       1        96       110
1       2        94       110
2       3       100       110
3       4       154       110
4       5       106       110
5       6       110       110
```




```
import scipy.stats as st

st.chisquare(throws4['Observed'], throws4['Expected'])
#встроенный метод Хи квадрат
```

```
Power_divergenceResult(statistic=22.763636363636365, pvalue=0.0003745547927853111)
```

В таком случае p-value оказывается 0,0003, что гораздо меньше, чем 0,05. Исходя из этого мы видим, что распределение становится не равномерным и даёт повод сомневаться в достоверности игральной кости.

Хи-квадрат работает корректно в случае, когда количество всех частот превышает 50, а минимальное ожидаемое значение частоты не меньше 5. Если в какой-либо категории ожидаемая частота менее 5, но при этом сумма всех частот превышает 50, то такую категорию объединяют с ближайшей, чтобы их общая частота превысила 5. Если это сделать невозможно, или сумма частот меньше 50, то следует использовать более точные методы проверки гипотез.

Тест χ^2-тест на независимость отличается от предыдущего теста постановкой гипотез:

- H0: категориальные переменные A и B независимы;
- H1: категориальные переменные A и B связаны между собой.

Пример:

```
#χ²-тест на независимость
#составим таблицу сопряжённости
'''На четырёх видах оборудования разработаны детали трёх уровней качества «высокое», «среднее», «низкое».
Данные приведены в таблице'''

data = pd.DataFrame({'Низкое': [2, 1, 3, 2],
                     'Среднее': [10, 10, 15, 13],
                     'Высокое': [20, 21, 22, 20]})
data.index = ['№1','№2','№3','№4']

#Нужно проверить зависит ли распределение качества деталей от вида оборудования, на котором оно было сделано.
data.head(6)
```

	Низкое	Среднее	Высокое
№1	2	10	20
№2	1	10	21
№3	3	15	22
№4	2	13	20

```
st.chi2_contingency(data)[:3]
```

```
(1.3971398988812402, 0.9660321696141388, 6)
```

p-значение больше 0.05, качество детали не зависит от станка, на котором деталь сделана.

Точный критерий Фишера

Если выборка очень мала, и нет возможности набрать еще данных, например, если исследуется очень редкое заболевание, то некорректно использовать критерий χ^2. Для таких выборок используется точный критерий Фишера. Особое место отводится точному критерию Фишера в медицине. Это важный метод обработки медицинских данных, нашедший свое применение во многих научных исследованиях. В данном тесте рассчитывается вероятность получить таблицу сопряженности с такими или с еще более выраженными отклонениями при условии нулевой гипотезы. Если p-значение больше критического, принимается нулевая гипотеза и делается вывод об отсутствии статистически значимых различий частоты исхода в зависимости от наличия некоторого фактора.

Точный критерий Фишера можно реализовать, используя функцию scipy.stats.fisher_exact().

Практическая работа

1. Загрузить данные из файла “ECDCCases.csv”.
2. Проверить в данных наличие пропущенных значений. Вывести количество пропущенных значений в процентах. Удалить два признака, в которых больше всех пропущенных значений. Для оставшихся признаков обработать пропуски: для категориального признака использовать заполнение значением по умолчанию (например, «other»), для числового признака использовать заполнение медианным значением. Показать, что пропусков больше в данных нет.
3. Посмотреть статистику по данным, используя describe(). Сделать выводы о том, какие признаки содержат выбросы. Посмотреть, для каких стран количество смертей в день превысило 3000 и сколько таких дней было.
4. Найти дублирование данных. Удалить дубликаты.
5. Загрузить данные из файла “ bmi.csv”. Взять оттуда две выборки. Одна выборка – это индекс массы тела людей с региона northwest, вторая выборка – это индекс массы тела людей с региона southwest. Сравнить средние значения этих выборок, используя t-критерий Стьюдента. Предварительно проверить выборки на нормальность (критерий Шопиро-Уилка) и на гомогенность дисперсии (критерий Бартлетта).
6. Кубик бросили 600 раз, получили следующие результаты:

N	Количество выпадений
1	97
2	98
3	109
4	95
5	97
6	104

С помощью критерия Хи-квадрат проверить, является ли полученное распределение равномерным. Использовать функцию scipy.stats.chisquare().

7. С помощью критерия Хи-квадрат проверить, являются ли переменные зависимыми.

Создать датафрейм, используя следующий код:

```
data = pd.DataFrame({'Женат': [89,17,11,43,22,1],
                     'Гражданский брак': [80,22,20,35,6,4],
                     'Не состоит в отношениях': [35,44,35,6,8,22]})
data.index = ['Полный рабочий день','Частичная занятость','Временно не работает','На домохозяйстве','На пенсии','Учёба']
```

Использовать функцию scipy.stats.chi2_contingency().

Влияет ли семейное положение на занятость?

8. Оформить отчет о проделанной работе, написать выводы.