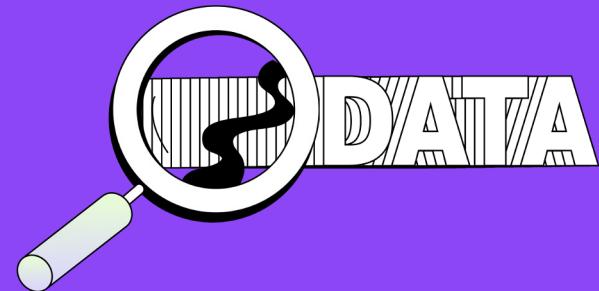


Модификация таблиц с Pandas

Урок 3





Что будет на уроке сегодня

- 📌 Создание, изменение и удаление признаков
- 📌 Группировки данных
- 📌 Объединение таблиц
- 📌 Встроенные визуализации





Работа с признаками



Создание признака

Самый простой способ - заполнить новый признак одинаковыми числами

```
users['new_feature'] = 0
users.head()
```

	CustomerId	Surname	Geography	Gender	Age	EstimatedSalary	new_feature
0	15634602	Hargrave	France	Female	42	101348.88	0
1	15647311	Hill	Spain	Female	41	112542.58	0
2	15619304	Onio	France	Female	42	113931.57	0
3	15701354	Boni	France	Female	39	93826.63	0
4	15737888	Mitchell	Spain	Female	43	79084.10	0



Создание признака

Можно создать новый признак, основываясь на уже имеющихся

```
users['Age (days)'] = users['Age'] * 365
users.head()
```

	CustomerId	Surname	Geography	Gender	Age	EstimatedSalary	new_feature	Age (days)
0	15634602	Hargrave	France	Female	42	101348.88	0	15330
1	15647311	Hill	Spain	Female	41	112542.58	0	14965
2	15619304	Onio	France	Female	42	113931.57	0	15330
3	15701354	Boni	France	Female	39	93826.63	0	14235
4	15737888	Mitchell	Spain	Female	43	79084.10	0	15695



Создание признака

Можем пользоваться методом `apply`, который ускоряет работу со строками. Для этого нужно реализовать функцию (или же пользоваться анонимными функциями) и передать её в метод `apply`

```
def age_to_days(x):
    return x * 365

users['Age (days) 3'] = users['Age'].apply(age_to_days)
users.head()
```

	CustomerId	Surname	Geography	Gender	Age	EstimatedSalary	new_feature	Age (days)	Age (days) 2	Age (days) 3
0	15634602	Hargrave	France	Female	42	101348.88	0	15330	15330	15330
1	15647311	Hill	Spain	Female	41	112542.58	0	14965	14965	14965
2	15619304	Onio	France	Female	42	113931.57	0	15330	15330	15330
3	15701354	Boni	France	Female	39	93826.63	0	14235	14235	14235
4	15737888	Mitchell	Spain	Female	43	79084.10	0	15695	15695	15695



Удаление признака

Чтобы удалить столбец из таблицы можем пользоваться методом `drop()`

```
users.drop(columns='new_feature', inplace=True)  
users.head()
```

	CustomerId	Surname	Geography	Gender	Age	EstimatedSalary	Age (days)	Age (days) 2	Age (days) 3
0	15634602	Hargrave	France	Female	42	101348.88	15330	15330	15330
1	15647311	Hill	Spain	Female	41	112542.58	14965	14965	14965
2	15619304	Onio	France	Female	42	113931.57	15330	15330	15330
3	15701354	Boni	France	Female	39	93826.63	14235	14235	14235
4	15737888	Mitchell	Spain	Female	43	79084.10	15695	15695	15695



Изменение признака

Чтобы изменить существующий признак пользуйтесь фильтрацией с помощью .loc

```
users.loc[users['Geography'] == 'France', 'target'] = 1
users.head()
```

	CustomerId	Surname	Geography	Gender	Age	EstimatedSalary	target
0	15634602	Hargrave	France	Female	42	101348.88	1
1	15647311	Hill	Spain	Female	41	112542.58	0
2	15619304	Onio	France	Female	42	113931.57	1
3	15701354	Boni	France	Female	39	93826.63	1
4	15737888	Mitchell	Spain	Female	43	79084.10	0



Изменение признака

В метод `replace()` можем передать словарь, состоящий из старого значения и нового

```
users['Gender'].replace({'Female': 'F', 'Male': 'M'}, inplace=True)
users.head()
```

	CustomerId	Surname	Geography	Gender	Age	EstimatedSalary	target
0	15634602	Hargrave	France	F	42	101348.88	1
1	15647311	Hill	Spain	F	41	112542.58	0
2	15619304	Onio	France	F	42	113931.57	1
3	15701354	Boni	France	F	39	93826.63	1
4	15737888	Mitchell	Spain	F	43	79084.10	0



Методы агрегации



Методы агрегации

Агрегация, или агрегирование – процесс объединения элементов в одно значение

Первый способ - вызвать метод `agg()` у `pd.Series` и передать в него список желаемых агрегаций

```
users['Age'].agg(['min', 'max'])
```

```
min    18
max    92
Name: Age, dtype: int64
```



Методы агрегации

Второй способ - вызвать метод `agg()` у датафрейма. В вызов этого метода можно передать словарь, где

- ❖ ключ - название признака
- ❖ значение - список желаемых агрегаций

```
users.agg({  
    'Age': ['min', 'max'],  
    'EstimatedSalary': 'mean'  
})
```

	Age	EstimatedSalary
min	18.0	NaN
max	92.0	NaN
mean	NaN	100090.239881



Методы агрегации

Третий способ - вызвать метод `agg()` у датафрейма и передать аргументы (которые будут названиями строк) со значениями кортежа

```
users.agg(  
    min_age=('Age', 'min'),  
    max_age=('Age', 'max'),  
    mean_salary=('EstimatedSalary', 'mean')  
)
```

Age **EstimatedSalary**

min_age	18.0	NaN
----------------	------	-----

max_age	92.0	NaN
----------------	------	-----

mean_salary	NaN	100090.239881
--------------------	-----	---------------



Методы объединения



Метод merge

С помощью метода `merge()` можно объединить две таблицы по столбцам. У первого датафрейма вызываем метод `merge()` и в него передаем:

- 📌 второй датафрейм
- 📌 столбец с левого датафрейма (он же первый), по которому нужно объединение (аргумент `left_on`)
- 📌 столбец с правого датафрейма (он же второй), по которому нужно объединение (аргумент `right_on`)

```
merged = users.merge(bank, left_on='CustomerId', right_on='CustomerId')
merged.head(5)
```

	CustomerId	Surname	Geography	Gender	Age	EstimatedSalary	target	CreditScore	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	Exi
0	15634602	Hargrave	France	F	42	101348.88	1	619	2	0.00	1	1	1	
1	15647311	Hill	Spain	F	41	112542.58	0	608	1	83807.86	1	0	1	
2	15619304	Onio	France	F	42	113931.57	1	502	8	159660.80	3	1	0	
3	15701354	Boni	France	F	39	93826.63	1	699	1	0.00	2	0	0	
4	15737888	Mitchell	Spain	F	43	79084.10	0	850	2	125510.82	1	1	1	



Метод join

Данный метод объединяет две таблицы по индексам, поэтому для успешного объединения нужно указать индексы

```
users_id = users.set_index('CustomerId')
users_id.head()
```

	Surname	Geography	Gender	Age	EstimatedSalary	target
CustomerId						
15634602	Hargrave	France	F	42	101348.88	1
15647311	Hill	Spain	F	41	112542.58	0
15619304	Onio	France	F	42	113931.57	1
15701354	Boni	France	F	39	93826.63	1
15737888	Mitchell	Spain	F	43	79084.10	0

```
bank_id = bank.set_index('CustomerId')
bank_id.head()
```

	CreditScore	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	Exited
CustomerId							
15597909	652	7	128135.99	1	1	0	0
15687913	501	7	93244.42	1	0	1	0
15619087	762	1	102520.37	1	1	1	0
15596552	535	5	134542.73	1	1	1	1
15741417	624	7	119656.45	2	1	1	0



Метод join

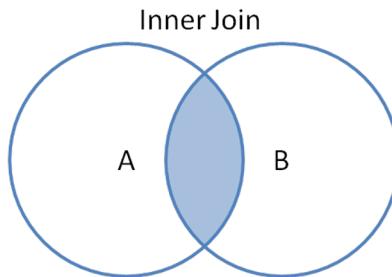
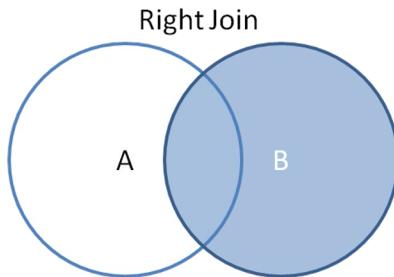
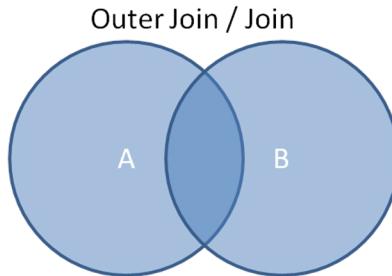
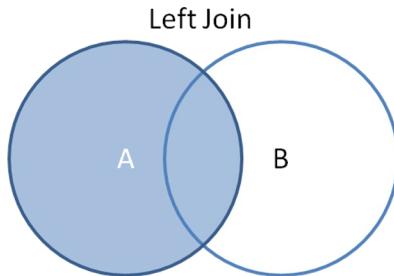
И затем вызвать метод join у одного из датафреймов

```
bank_id.join(users_id).head()
```

	CreditScore	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	Exited	Surname	Geography	Gender	Age	EstimatedSalary	target
Customerid													
15597909	652	7	128135.99		1	1	0	0	Johnstone	Germany	M	33.0	158437.73
15687913	501	7	93244.42		1	0	1	0	Mai	Germany	F	34.0	199805.63
15619087	762	1	102520.37		1	1	1	0	Taylor	France	M	53.0	170195.40
15596552	535	5	134542.73		1	1	1	1	Stephens	Germany	M	48.0	58203.67
15741417	624	7	119656.45		2	1	1	0	Chibuzo	Spain	F	35.0	4595.05



Атрибут how



В методах `merge` и `join` есть атрибут `how`, который позволяет указать способ объединения таблиц.

- 📌 `left` - остаются все объекты с левого датафрейма и ищутся совпадения из правого
- 📌 `right` - остаются все объекты с правого датафрейма и ищутся совпадения из левого
- 📌 `inner` - остаются объекты, которые есть и в левом датафрейме, и в правом
- 📌 `outer` - остаются все объекты из двух датафреймов

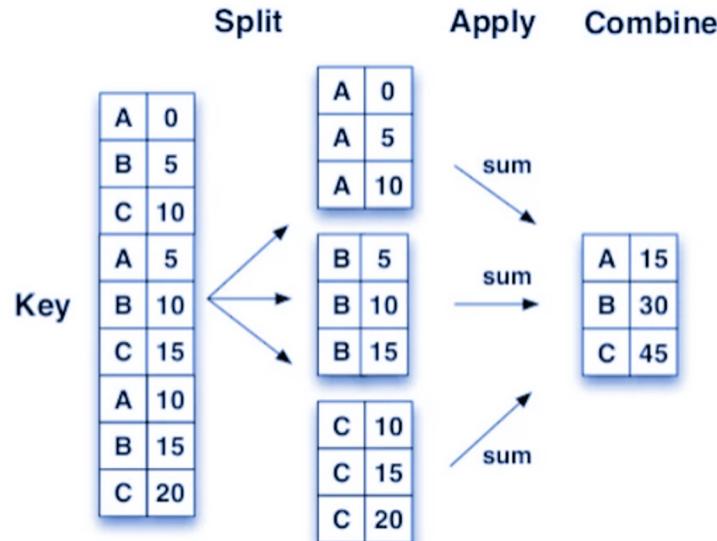


Методы группировок



Метод groupby

В данном методе вначале происходит разбиение на группы, а затем можно сделать агрегацию по любой агрегирующей функции





Метод pivot_table

pivot_table (сводная таблица) - это мощный инструмент для обобщения и представления данных.

Параметры pivot_table():

- 📌 index – столбец, который будет использован для строк
- 📌 columns – столбец, который будет использован для столбцов
- 📌 values – столбец обрабатываемых значений
- 📌 aggfunc – функция, применяемая к values
- 📌 fill_value – значение по умолчанию
- 📌 margins – если True, то добавляется столбец All (Итого). По умолчанию: False

```
users.pivot_table(index='Geography',
                   aggfunc={'Age': ['mean'], 'EstimatedSalary': 'min'})
```

	Age	EstimatedSalary
	mean	min
Geography		
France	38.513864	90.07
Germany	39.770734	11.58
Spain	38.890997	417.41



Метод crosstab

Параметры crosstab():

- index - значения для группировки по строкам
- columns - значения для группировки по столбцам
- values - агрегируемый столбец (или столбцы)
- aggfunc - функция, которая будет применена к каждой группе значений values, сгруппированным по значениям index и columns. Значения этой функции и есть значения сводной таблицы
- margins - добавляет результирующий столбец/строку
- normalize: boolean, {'all', 'index', 'columns'} - нормировка всей таблицы (или только по строкам/столбцам).

```
pd.crosstab(index=users['Geography'],
             columns=users['Gender'],
             normalize='all')
```

Gender	F	M
Geography		
France	0.226045	0.275355
Germany	0.119324	0.131526
Spain	0.108922	0.138828

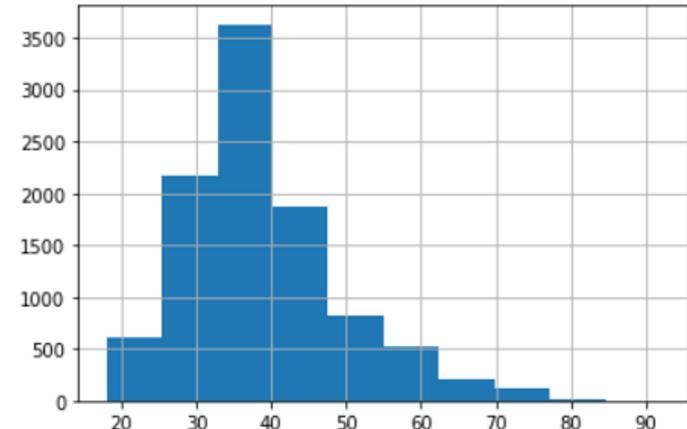


Встроенные визуализации

Встроенные визуализации

hist() - гистограмма. С помощью неё можем изучить распределение возраста наших клиентов

```
users['Age'].hist();
```





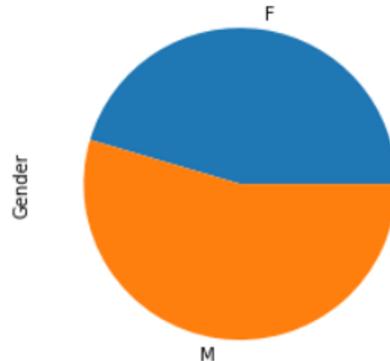
Встроенные визуализации

pie() - круговая диаграмма, с помощью
неё изучаем долю мужчин и долю
женщин среди наших клиентов

```
data = users.groupby('Gender').count()['Age']
data.name = 'Gender'
data
```

```
Gender
F    4542
M    5456
Name: Gender, dtype: int64
```

```
data.plot.pie(y='Gender');
```

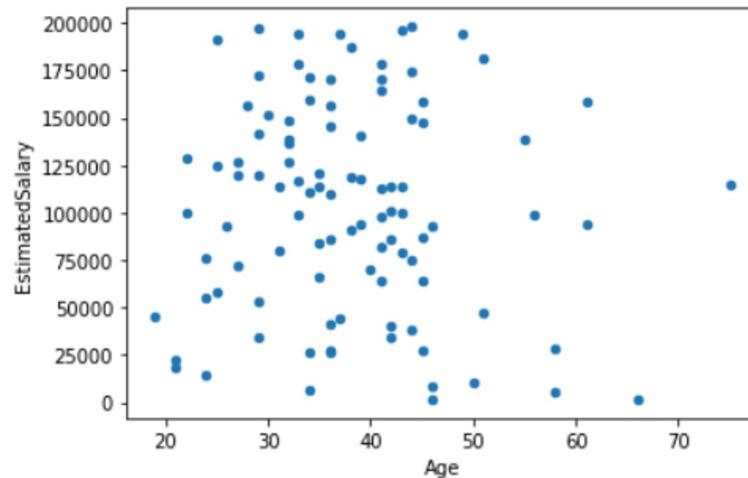




Встроенные визуализации

scatter() - точечный график, он показывает взаимное распределение признаков. Изучим, а есть ли зависимость между возрастом клиента и его заработной платой

```
users.iloc[:100].plot.scatter(x='Age', y='EstimatedSalary');
```





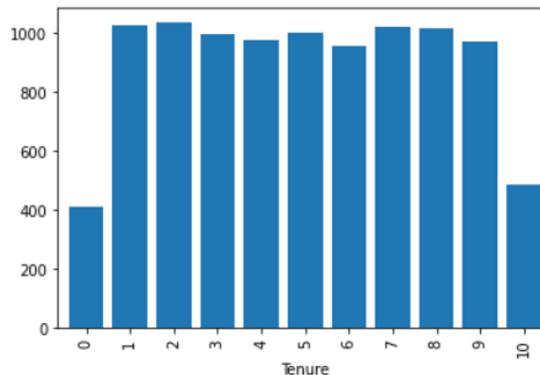
Встроенные визуализации

`bar()` - столбчатая диаграмма, показывает количество объектов в каждой категории. Посмотрим, сколько уже лет люди являются клиентами нашего банка

```
data = bank.groupby('Tenure').count()['Balance']
data.name = 'num_clients'
data
```

```
Tenure
0      411
1     1027
2     1036
3      994
4      978
5     1000
6      957
7     1020
8     1014
9      971
10     487
Name: num_clients, dtype: int64
```

```
data.plot.bar(width=0.8);
```





Итоги урока

- 📌 Научились создавать, изменять и удалять признаки
- 📌 Изучили группировки данных и объединение таблиц
- 📌 Познакомились со встроенными визуализациями





Что будет на следующем уроке

- 📌 Виды графиков
- 📌 Интерпретация графиков
- 📌 Визуальный анализ данных
- 📌 Анализ геоданных





Спасибо за урок!

