Pandas

PIERIAN DATA

Pandas

- Давайте вспомним этапы работ по машинному обучению...

Этапы работ по машинному обучению

Реальный мир

Сбор и хранение данных

Очистка и организация данных

Исследование данных

Отчёт

Визуали-зация

Коммуни-кация

Ответить на вопрос

Принятие решений

Ответы на ключевые вопросы

PIERIAN DATA

Этапы работ по машинному обучению

Реальный мир

Сбор и хранение данных

Очистка и организация данных

Исследование данных

Модели Машинного Обучения

Service

Dashboard

Application

Продукт на основе данных

Прогнозирование событий

Результаты анализа данных

PIERIAN DATA

Этапы работ по машинному обучению

Реальный мир

→ Сбор и хранение данных → Очистка и организация данных → Исследование данных

pandas

Pandas

- Pandas – это библиотека для анализа данных
- Очень мощная система работы с таблицами (датафреймами – DataFrame), построенная на основе NumPy
- Прекрасная документация:
  - https://pandas.pydata.org/docs/

Pandas

- Что можно делать в Pandas?
  - Инструменты для чтения и записи данных в различных форматах
  - Умная выборка данных с помощью индексов, логики, поднаборов данных и т.д.
  - Обработка отсутствующих данных (missing data)
  - Доработка и реструктуризация данных

Pandas – обзор раздела

- Объекты Series и DataFrames
- Фильтрация по условиям и полезные методы
- Отсутствующие данные
- Операции группировки данных - Group By
- Комбинации объектов DataFrame
- Методы для работы с текстом и временем
- Ввод и вывод данных

PIERIAN DATA

Давайте начнём!

PIERIAN DATA

Series

часть 1

PIERIAN DATA

Series

- Series – это структура данных в Pandas, которая хранит массив данных, а также именованный индекс
- Отличается от простого массива NumPy именно наличием именованного индекса
- Формальное определение: одномерный массив ndarray с метками по оси (axis labels)

Series

- В массивах NumPy есть числовой индекс

Index	Data
0	1776
1	1867
2	1821

Series

- В Pandas Series добавляется именованный индекс

Labeled Index	Data
USA	1776
CANADA	1867
MEXICO	1821

PIERIAN DATA

Series

- Числовая нумерация по-прежнему сохраняется

Numeric Index	Labeled Index	Data
0	USA	1776
1	CANADA	1867
2	MEXICO	1821

PIERIAN DATA

Series

- Давайте рассмотрим различные варианты создания объекта Pandas Object
- Также посмотрим ключевые свойства и операции
- Позже мы узнаем, как можно использовать общий индекс для объединения объектов Series для создания табличной структуры данных под названием DataFrame

Series

часть 2

PIERIAN DATA

DataFrames

часть 1

PIERIAN DATA

DataFrames

- DataFrame – это таблица из колонок и строк в Pandas, которую мы можем легко фильтровать м реструктурировать
- Формальное определение: набор объектов Pandas Series, имеющих один и тот же общий индекс

DataFrames

- Пример объекта Series

Labeled Index	Data
USA	1776
CANADA	1867
MEXICO	1821

DataFrames

- Пример объектов Series с общим индексом

Index	Year
USA	1776
CANADA	1867
MEXICO	1821

Index	Pop
USA	328
CANADA	38
MEXICO	126

Index	GDP
USA	20.5
CANADA	1.7
MEXICO	1.22

DataFrames

- Пример объектов Series с общим индексом

Index	Year
USA	1776
CANADA	1867
MEXICO	1821

Index	Pop
USA	328
CANADA	38
MEXICO	126

Index	GDP
USA	20.5
CANADA	1.7
MEXICO	1.22

DataFrames

- DataFrame

Index	Year	Pop	GDP
USA	1776	328	20.5
CANADA	1867	38	1.7
MEXICO	1821	126	1.22

DataFrames

- DataFrame – главный объект в Pandas, с которым мы будем работать. Он очень полезен!
- Давайте изучим основы работы с ним:
  - Создание DataFrame
  - Извлечение колонки или нескольких колонок
  - Извлечение строки или нескольких строк
  - Добавление нового колонки или новой строки

DataFrames

- Небольшое замечание – эта и несколько следующих лекций соответствуют одному блокноту 01-DataFrames.ipynb

DataFrames

часть 2

PIERIAN DATA

DataFrames

часть 3

PIERIAN DATA

DataFrames

часть 4

PIERIAN DATA

Фильтрация по условию (conditional filtering)

Фильтрация по условию

- Обычно в больших массивах удобнее фильтровать данные не по позиции элементов, а по некоторому условию
- “Фильтрация по условию” позволяет нам выбрать строки, накладывая условия на значения колонок
- Это приводит нас к необходимости обсудить вопрос организации данных...

PIERIAN DATA

Фильтрация по условию

- Колонки – это признаки (features)

Index	Year	Pop	GDP
USA	1776	328	20.5
CANADA	1867	38	1.7
MEXICO	1821	126	1.22

PIERIAN DATA

Фильтрация по условию

- Строки – это экземпляры данных

Index	Year	Pop	GDP
USA	1776	328	20.5
CANADA	1867	38	1.7
MEXICO	1821	126	1.22

PIERIAN DATA

Фильтрация по условию

- Такой формат будет нужен для машинного обучения!

Index	Year	Pop	GDP
USA	1776	328	20.5
CANADA	1867	38	1.7
MEXICO	1821	126	1.22

PIERIAN DATA

Фильтрация по условию

- Какие страны имеют население больше 50 (млн.)?

Index	Year	Pop	GDP
USA	1776	328	20.5
CANADA	1867	38	1.7
MEXICO	1821	126	1.22

PIERIAN DATA

Фильтрация по условию

- df[“Pop”]

Index	Year	Pop	GDP
USA	1776	328	20.5
CANADA	1867	38	1.7
MEXICO	1821	126	1.22

PIERIAN DATA

Фильтрация по условию

- df[“Pop”] > 50

Index	Year	Pop	GDP
USA	1776	328	20.5
CANADA	1867	38	1.7
MEXICO	1821	126	1.22

PIERIAN DATA

Фильтрация по условию

- df[“Pop”] > 50

Index	Year	Pop	GDP
USA	1776	True	20.5
CANADA	1867	False	1.7
MEXICO	1821	True	1.22

PIERIAN DATA

Фильтрация по условию

- df[df[“Pop”] > 50]

Index	Year	Pop	GDP
USA	1776	True	20.5
CANADA	1867	False	1.7
MEXICO	1821	True	1.22

PIERIAN DATA

Фильтрация по условию

- df[df[“Pop”] > 50]

Index	Year	Pop	GDP
USA	1776	True	20.5
MEXICO	1821	True	1.22

Фильтрация по условию

- Варианты фильтрации по условию
  - Фильтр по одному условию
  - Фильтр по нескольким условиям
  - Проверка нескольких возможных значений

Полезные методы

часть 1 – методы apply

PIERIAN DATA

Полезные методы

- Теперь мы знаем, как получать данные и фильтровать данные в объектах Series и DataFrame в Pandas
- Далее мы рассмотрим множество методов, доступных в Pandas
- Здесь будет несколько лекций, поскольку этих методов достаточно много

PIERIAN DATA

Полезные методы

- Для Вашего удобства, блокнот для этих лекций в начале содержит ссылки, с помощью которых можно перейти в нужный раздел блокнота

Полезные методы

- Хотя в Pandas есть много встроенных методов, мы также можем использовать метод .apply() для вызова любой функции для каждого элемента в объекте Series
- В качестве значений входных параметров можно использовать одну или несколько колонок, рассмотрим это на примерах!

Полезные методы

часть 2 - apply с несколькими колонками

PIERIAN DATA

Полезные методы

часть 3 – описание данных и сортировка

PIERIAN DATA

Отсутствующие данные

часть 1 – обзор

PIERIAN DATA

Отсутствующие данные (missing data)

- В реальном мире отдельные данные могут отсутствовать по многим причинам
- Однако многие методы машинного обучения и статистические методы не могут работать с отсутствующими данными, в этом случае нам нужно решить, что делать с такими данными

Отсутствующие данные (missing data)

- При чтении отсутствующих данных, Pandas будет отображать их как значения NaN
- Также есть специализированные неопределённые значения в pandas, например pd.NaT для обозначения того, что отсутствующее значение должно иметь тип TimeStamp.

Отсутствующие данные (missing data)

- Варианты обработки отсутствующих значений
  - Оставить их
  - Удалить их
  - Чем-то заменить их
- Обратите внимание – не существует подхода, который был бы правильным в 100% случаев, всё зависит от конкретной ситуации!

Отсутствующие данные (missing data)

- Вариант “Оставить их”
  - ПЛЮСЫ
    - Легче всего
    - Мы не вносим изменения в исходные данные
  - МИНУСЫ
    - Многие методы не поддерживают значения NaN
    - Часто можно аргументированно заменить данные

Отсутствующие данные (missing data)

- Вариант “Удалить их”
  - ПЛЮСЫ
    - Легко реализуем
    - Можно сделать на основе правил
  - МИНУСЫ
    - Возможна потеря данных или полезной информации
    - Модели меньше пригодны для будущих данных

Отсутствующие данные (missing data)

- Вариант “Удалить их”
  - Удаление строки
    - Имеет смысл, когда данных нет во многих колонках

	Year	Pop	GDP	Area
USA	1776	NAN	NAN	NAN
CANADA	1867	38	1.7	3.86
MEXICO	1821	126	1.22	0.76

Отсутствующие данные (missing data)

- Вариант “Удалить их”
  - Удаление строки
    - Всегда полезно посчитать процент таких строк

	Year	Pop	GDP	Area
USA	1776	NAN	NAN	NAN
CANADA	1867	38	1.7	3.86
MEXICO	1821	126	1.22	0.76

Отсутствующие данные (missing data)

- Вариант “Удалить их”
  - Удаление колонки
    - Полезно, когда значений нет почти во всех строках

	Year	Pop	GDP	Area
USA	1776	328	20.5	NAN
CANADA	1867	38	1.7	NAN
MEXICO	1821	126	1.22	0.76

Отсутствующие данные (missing data)

- Вариант “Чем-то заменить их”
  - ПЛЮСЫ
    - Потенциально мы сохраняем больше данных для обучения модели
  - МИНУСЫ
    - Сложнее всего сделать, и можно сделать по-разному
    - Возможно получение ложных выводов

Отсутствующие данные (missing data)

- Вариант “Чем-то заменить их”
  - Заменить каким-то одним значением
    - Хороший вариант, когда NaN был заменой какого-то значения по умолчанию

	Year	Pop	GDP	Carriers
USA	1776	328	20.5	11
CANADA	1867	38	1.7	NAN
MEXICO	1821	126	1.22	NAN

PIERIAN DATA

Отсутствующие данные (missing data)

- Вариант “Чем-то заменить их”
  - Заменить каким-то одним значением
    - Здесь можно использовать значение 0

	Year	Pop	GDP	Carriers
USA	1776	328	20.5	11
CANADA	1867	38	1.7	0
MEXICO	1821	126	1.22	0

PIERIAN DATA

Отсутствующие данные (missing data)

- Вариант “Чем-то заменить их”
  - Заменить средним или интерполированным значением
    - Намного сложнее, и требует некоторых предположений о данных

	Year	Pop	GDP	Perct
USA	1776	328	20.5	75%
CANADA	1867	38	1.7	NAN
MEXICO	1821	126	1.22	25%

PIERIAN DATA

Отсутствующие данные (missing data)

- Вариант “Чем-то заменить их”
  - Заменить средним или интерполированным значением
    - Намного сложнее, и требует некоторых предположений о данных

	Year	Pop	GDP	Perct
USA	1776	328	20.5	75%
CANADA	1867	38	1.7	50%
MEXICO	1821	126	1.22	25%

PIERIAN DATA

Отсутствующие данные (missing data)

- Давайте изучим синтаксис Pandas для работы с отсутствующими значениями
- Позже в этом курсе у нас будет более детальная дискуссия о том, как выбрать между вариантами оставить такие значения, удалить их или заменить их какими-то другими значениями.

Отсутствующие данные

часть 2 – Pandas

PIERIAN DATA

Группировка данных

часть 1

PIERIAN DATA

Группировка данных - GroupBy

- Операция groupby() позволяет изучать данные отдельно по категориям
- Давайте посмотрим, как это выглядит в Pandas…

Группировка данных - GroupBy

Category	Data Value
A	10
A	5
B	2
B	4
C	12
C	6

PIERIAN DATA

Группировка данных - GroupBy

Category	Data Value
A	10
A	5
B	2
B	4
C	12
C	6

Для groupby() мы должны выбрать категориальную колонку

Категориальные колонки принимают дискретные (не непрерывные) значения

Они могут быть и числовыми, например класс вагона в поезде или на корабле – 1-й класс, 2-й класс, 3-й класс.

PIERIAN DATA

Группировка данных - GroupBy

Category	Data Value
A	10
A	5
B	2
B	4
C	12
C	6

A	10
A	5

B	2
B	4

C	12
C	6

Агрегатная функция
.sum()

Category	Result
A	15
B	6
C	18

PIERIAN DATA

Группировка данных - GroupBy

Category	Data Value
A	10
A	5
B	2
B	4
C	12
C	6

A	10
A	5

B	2
B	4

C	12
C	6

Агрегатная функция
.mean()

Category	Result
A	7.5
B	3
C	9

PIERIAN DATA

Группировка данных - GroupBy

Category	Data Value
A	10
A	5
B	2
B	4
C	12
C	6

A	10
A	5

B	2
B	4

C	12
C	6

Агрегатная функция
.count()

Category	Result
A	2
B	2
C	2

PIERIAN DATA

Группировка данных - GroupBy

- Обратите внимание, что операция groupby() создаёт “ленивый” (“lazy”) объект groupby, который не выполняет всю работу сразу, а ждёт того момента, когда Вы обратитесь к этому объекту, и данные действительно понадобятся
- Давайте посмотрим, как это выглядит в Pandas…

Группировка данных

часть 2 - мультииндекс

PIERIAN DATA

Объединение датафреймов

Конкатенация

PIERIAN DATA

Объединение датафреймов

- Очень часто нужные Вам данные находятся в двух и более датафреймах. К счастью, их очень просто объединить в один датафрейм
- Самый простой случай – когда оба датафрейма имеют одинаковый формат. В этом случае достаточно выполнить конкатенацию с помощью метода pd.concat()

Объединение датафреймов

- Конкатенация – это просто “склеивание” двух датафреймов, по колонкам

	Year	Pop
USA	1776	328
CANADA	1867	38
MEXICO	1821	126

	GDP	Perct
USA	20.5	75%
CANADA	1.7	NAN
MEXICO	1.22	25%

Объединение датафреймов

- Конкатенация – это просто “склеивание” двух датафреймов, по колонкам

	Year	Pop	GDP	Perct
USA	1776	328	20.5	75%
CANADA	1867	38	1.7	NAN
MEXICO	1821	126	1.22	25%

Объединение датафреймов

- Конкатенация – это просто “склеивание” двух датафреймов, по строкам

	Year	Pop	GDP
USA	1776	328	20.5
CANADA	1867	38	1.7

	Year	Pop	GDP
MEXICO	1821	126	1.22
BRAZIL	1822	209	1.86

PIERIAN DATA

Объединение датафреймов

- Конкатенация – это просто “склеивание” двух датафреймов, по строкам

	Year	Pop	GDP
USA	1776	328	20.5
CANADA	1867	38	1.7
BRAZIL	1822	209	1.86
MEXICO	1821	126	1.22

Объединение датафреймов

- Pandas будет автоматически обрабатывать значения NaN
- Давайте посмотрим несколько примеров!

Объединение датафреймов

“inner” merge

PIERIAN DATA

Объединение датафреймов

- Зачастую датафреймы имеют разный формат, и поэтому их нельзя объединять с помощью конкатенации
- В этом случае применяется слияние – merge
- Это действие аналогично операции JOIN для соединения таблиц в SQL

Объединение датафреймов

- Метод .merge() принимает на вход параметр how
- Есть три основных способа:
  - Inner
  - Outer
  - Left или Right
- Вопрос в том, что делать с информацией, которая есть только в одном из датафреймов

Объединение датафреймов

- Метод .merge() принимает на вход параметр how
- Есть три основных способа объединения:
  - Inner
  - Outer
  - Left или Right
- Вопрос в том, что делать с информацией, которая есть только в одном из датафреймов

Объединение датафреймов

- Представим себе простой пример
- Наша компания проводит конференцию для людей в сфере киноиндустрии
- Некоторые люди зарегистрировались онлайн заранее, и они входят в систему в день конференции

Объединение датафреймов

- В итоге у нас есть две таблицы:

REGISTRATIONS	
reg_id	name
1	Andrew
2	Bob
3	Charlie
4	David

LOGINS	
log_id	name
1	Xavier
2	Andrew
3	Yolanda
4	Bob

Объединение датафреймов

- Для простоты предположим, что все имена уникальны. Например, есть только один Andrew.

REGISTRATIONS	
reg_id	name
1	Andrew
2	Bob
3	Charlie
4	David

LOGINS	
log_id	name
1	Xavier
2	Andrew
3	Yolanda
4	Bob

Объединение датафреймов

- Далее мы должны решить, по какой колонке объединять эти две таблицы (параметр on).

REGISTRATIONS	
reg_id	name
1	Andrew
2	Bob
3	Charlie
4	David

LOGINS	
log_id	name
1	Xavier
2	Andrew
3	Yolanda
4	Bob

Объединение датафреймов

- Колонка on должна существовать в обеих таблицах. В нашем случае это колонка name: on=“name”.

REGISTRATIONS	
reg_id	name
1	Andrew
2	Bob
3	Charlie
4	David

LOGINS	
log_id	name
1	Xavier
2	Andrew
3	Yolanda
4	Bob

Объединение датафреймов

- В случае how=“inner” мы берём строки, которые соответствуют друг другу в обеих таблицах.

REGISTRATIONS	
reg_id	name
1	Andrew
2	Bob
3	Charlie
4	David

LOGINS	
log_id	name
1	Xavier
2	Andrew
3	Yolanda
4	Bob

Объединение датафреймов

- В случае how="inner" мы берём строки, которые соответствуют друг другу в обеих таблицах.

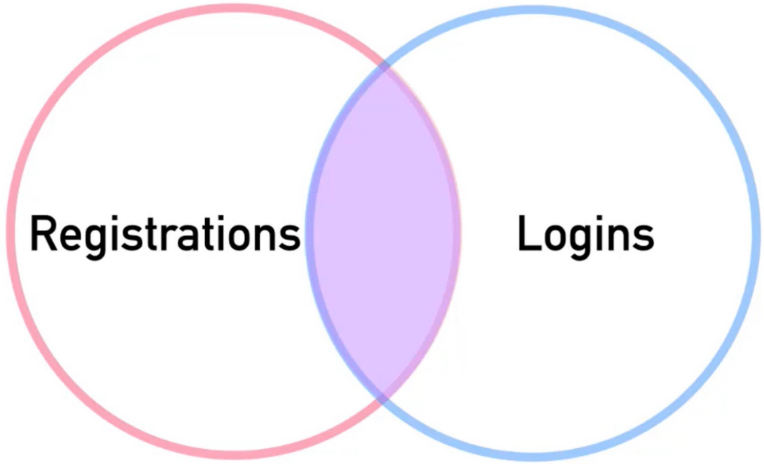
REGISTRATIONS	
reg_id	name
1	Andrew
2	Bob
3	Charlie
4	David

LOGINS	
log_id	name
1	Xavier
2	Andrew
3	Yolanda
4	Bob

Объединение датафреймов

Операции merge можно изобразить в виде пересечения множеств, используя окружности.

REGISTRATIONS	
reg_id	name
1	Andrew
2	Bob
3	Charlie
4	David



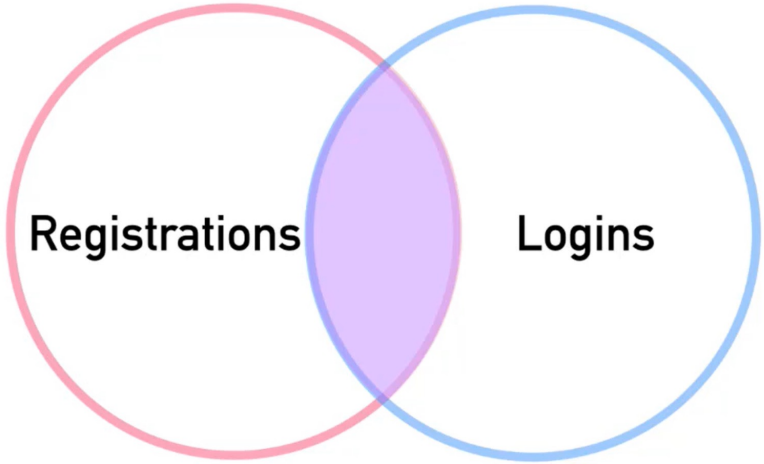
LOGINS	
log_id	name
1	Xavier
2	Andrew
3	Yolanda
4	Bob

PIERIAN DATA

Объединение датафреймов

pd.merge(registrations, logins, how='inner', on='name')

REGISTRATIONS	
reg_id	name
1	Andrew
2	Bob
3	Charlie
4	David



LOGINS	
log_id	name
1	Xavier
2	Andrew
3	Yolanda
4	Bob

PIERIAN DATA

Объединение датафреймов

pd.merge(registrations, logins, how='inner', on='name')

REGISTRATIONS	
reg_id	name
1	Andrew
2	Bob
3	Charlie
4	David

RESULTS		
reg_id	name	log_id
1	Andrew	2
2	Bob	4

LOGINS	
log_id	name
1	Xavier
2	Andrew
3	Yolanda
4	Bob

PIERIAN DATA

Объединение датафреймов

- Посмотрим, как это выглядит в Pandas!

Объединение датафреймов

“left” и “right” merge

PIERIAN DATA

Объединение датафреймов

- Теперь, когда мы изучили “inner” merge, давайте рассмотрим операции “left” merge и “right” merge
- Обратите внимание, что здесь будет важно, в каком порядке указаны таблицы

Объединение датафреймов

- Посмотрим, как операция how=“left” будет применяться к нашим двум таблицам:

REGISTRATIONS	
reg_id	name
1	Andrew
2	Bob
3	Charlie
4	David

LOGINS	
log_id	name
1	Xavier
2	Andrew
3	Yolanda
4	Bob

Объединение датафреймов

- Обратите внимание – таблица registrations слева, таблица logins справа:

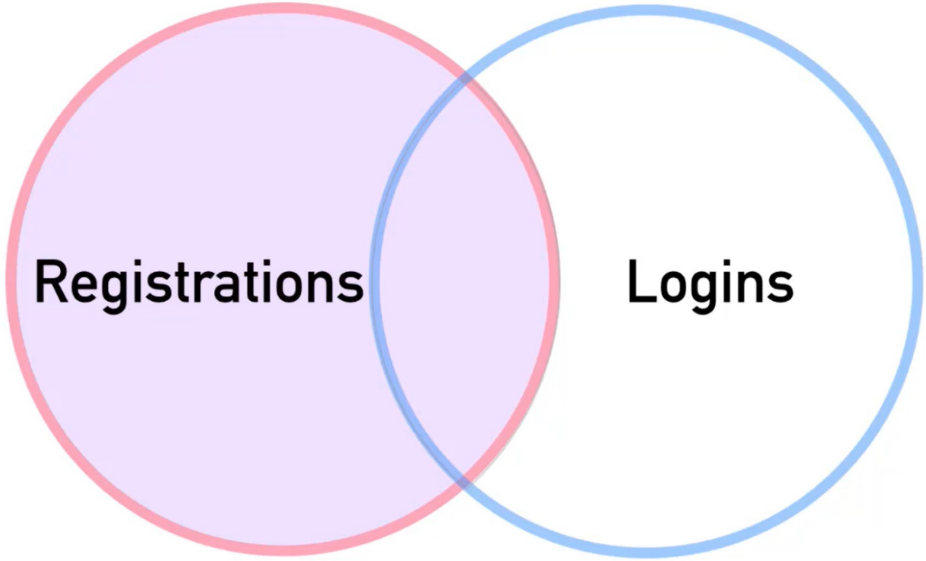
REGISTRATIONS	
reg_id	name
1	Andrew
2	Bob
3	Charlie
4	David

LOGINS	
log_id	name
1	Xavier
2	Andrew
3	Yolanda
4	Bob

Объединение датафреймов

pd.merge(registrations, logins, how="left", on="name")

REGISTRATIONS	
reg_id	name
1	Andrew
2	Bob
3	Charlie
4	David



LOGINS	
log_id	name
1	Xavier
2	Andrew
3	Yolanda
4	Bob

PIERIAN DATA

Объединение датафреймов

pd.merge(registrations, logins, how="left", on="name")

REGISTRATIONS	
reg_id	name
1	Andrew
2	Bob
3	Charlie
4	David

RESULTS		
reg_id	name	log_id
1	Andrew	2
2	Bob	4
3	Charlie	NaN
4	David	NaN

LOGINS	
log_id	name
1	Xavier
2	Andrew
3	Yolanda
4	Bob

PIERIAN DATA

Объединение датафреймов

pd.merge(registrations, logins, how="left", on="name")

REGISTRATIONS	
reg_id	name
1	Andrew
2	Bob
3	Charlie
4	David

RESULTS		
reg_id	name	log_id
1	Andrew	2
2	Bob	4
3	Charlie	NaN
4	David	NaN

LOGINS	
log_id	name
1	Xavier
2	Andrew
3	Yolanda
4	Bob

PIERIAN DATA

Объединение датафреймов

- Теперь посмотрим, что будет в случае “right” merge

Объединение датафреймов

pd.merge(registrations, logins, how="right", on="name")

REGISTRATIONS	
reg_id	name
1	Andrew
2	Bob
3	Charlie
4	David

RESULTS		
reg_id	name	log_id
1	Andrew	2
2	Bob	4
NaN	Xavier	1
NaN	Yolanda	3

LOGINS	
log_id	name
1	Xavier
2	Andrew
3	Yolanda
4	Bob

Объединение датафреймов

- Посмотрим, как это выглядит в Pandas!

Объединение датафреймов

“outer” merge

PIERIAN DATA

Объединение датафреймов

- Вариант how=“outer” позволяет нам включить в результат все строки, которые существуют хотя бы в одной из двух таблиц

Объединение датафреймов

- Вспомним – у нас есть Andrew и Bob в обеих таблицах:

REGISTRATIONS	
reg_id	name
1	Andrew
2	Bob
3	Charlie
4	David

LOGINS	
log_id	name
1	Xavier
2	Andrew
3	Yolanda
4	Bob

Объединение датафреймов

- И есть имена, которые находятся только в одной из двух таблиц:

REGISTRATIONS	
reg_id	name
1	Andrew
2	Bob
3	Charlie
4	David

LOGINS	
log_id	name
1	Xavier
2	Andrew
3	Yolanda
4	Bob

Объединение датафреймов

- С помощью how=“outer” мы возьмём все имена из обеих таблиц:

REGISTRATIONS	
reg_id	name
1	Andrew
2	Bob
3	Charlie
4	David

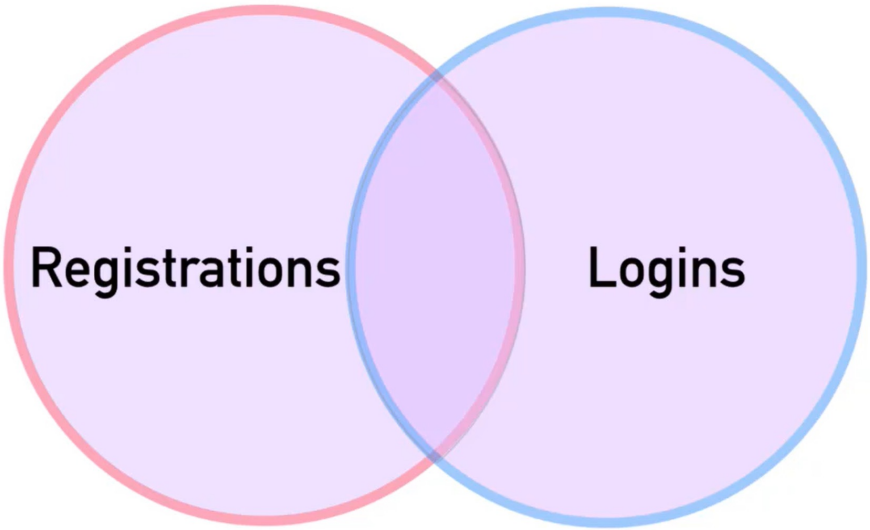
LOGINS	
log_id	name
1	Xavier
2	Andrew
3	Yolanda
4	Bob

PIERIAN DATA

Объединение датафреймов

pd.merge(registrations, logins, how="outer", on="name")

REGISTRATIONS	
reg_id	name
1	Andrew
2	Bob
3	Charlie
4	David



LOGINS	
log_id	name
1	Xavier
2	Andrew
3	Yolanda
4	Bob

PIERIAN DATA

Объединение датафреймов

pd.merge(registrations, logins, how="outer", on="name")

REGISTRATIONS	
reg_id	name
1	Andrew
2	Bob
3	Charlie
4	David

RESULTS		
reg_id	name	log_id
1	Andrew	2
2	Bob	4
3	Charlie	NaN
4	David	NaN
NaN	Xavier	1
NaN	Yolanda	3

LOGINS	
log_id	name
1	Xavier
2	Andrew
3	Yolanda
4	Bob

PIERIAN DATA

Объединение датафреймов

- Посмотрим, как это выглядит в Pandas!

Методы для текста

PIERIAN DATA

Методы для текста

- Очень часто текстовые данные нужно очистить или преобразовать
- Хотя можно с помощью .apply() применить любую функцию, в Pandas есть много встроенных методов и функций для работы с текстом
- Давайте научимся их использовать!

Методы для времени

PIERIAN DATA

Методы для времени

- В базовом Python есть объект datetime для хранения информации о дате и времени
- Pandas позволяет легко извлекать информацию из объектов datetime, чтобы использовать их для построения признаков

Методы для времени

- Например, у нас могут быть данные о продажах, с метками Timestamp для каждой транзакции
- С помощью Pandas мы можем взять отдельные компоненты из Timestamp, например:
  - День недели
  - Рабочий день или Выходной
  - Время до полудня или после полудня – AM или PM

Ввод и вывод данных

CSV-файлы

PIERIAN DATA

Ввод и вывод данных

- Pandas может читать данные из широкого спектра источников, и имеет отличную онлайн-документацию
- В этой и последующих лекциях мы рассмотрим наиболее часто встречающиеся варианты чтения данных

PIERIAN DATA

Ввод и вывод данных

- Внимание!
  - Вы должны точно знать название файла и папку, в которой он находится
  - Для некоторых данных может потребоваться пароль (например, пароль к базе данных SQL)

Ввод и вывод данных

- Видео-лекции:
  - CSV-файлы
  - HTML-таблицы
  - Excel-файлы
  - Базы данных SQL

Ввод и вывод данных

HTML-таблицы

PIERIAN DATA

Ввод и вывод данных

- Важные замечания:
  - Не все таблицы на веб-сайте могут быть доступны в виде HTML-таблиц
  - Некоторые веб-сайты могут блокировать Ваши попытки извлечь HTML с их сайта с помощью Pandas
  - Может быть более эффективным использовать API

Ввод и вывод данных

- В качестве примера давайте считаем все таблицы из некоторой статьи Википедии, и затем очистим эти данные и преобразуем их в датафрейм
- Вывод данных в формате HTML можно применять для отображения табличных данных на веб-странице

Ввод и вывод данных

Excel-файлы

PIERIAN DATA

Ввод и вывод данных

- Pandas может читать и записывать в Excel-файлы
- Важное замечание:
  - Pandas может читать и записывать только обычные данные, он не сможет прочитать макросы, визуализации и формулы в ячейках

Ввод и вывод данных

- Pandas работает с Excel-файлом как со словарём, в котором ключи словаря – это названия листов, а значения словаря – это данные листов в виде датафреймов
- Чтобы работать с Excel-файлами в Pandas, нужно установить дополнительные библиотеки
- Давайте посмотрим, как это работает!

Ввод и вывод данных

SQL

PIERIAN DATA

Ввод и вывод данных

- Pandas может читать и записывать данные в базу данных SQL, используя драйвер и библиотеку sqlalchemy в Python
- Как это работает?

Ввод и вывод данных

- Шаг 1:
  - Выяснить, к какой базе данных Вы планируете подключаться. Например:
    - PostgreSQL
    - MySQL
    - MS SQL Server

Ввод и вывод данных

- Шаг 2:
  - Установить соответствующую библиотеку Python с драйвером (можно найти с помощью поиска в Google) :
    - PostgreSQL – psycopg2
    - MySQL - pymysql
    - MS SQL Server - pyodbc

Ввод и вывод данных

- Шаг 3:
  - Подключиться к базе данных с помощью библиотеки sqlalchemy:
    - https://docs.sqlalchemy.org/en/13/dialects/index.html

Ввод и вывод данных

- Шаг 4:
  - Используя подключение sqlalchemy к базе данных, применить метод Pandas read_sql для отправки SQL-запросов
  - Pandas может считывать таблицы целиком и класть их в датафреймы. Или же возвращать результаты запросов:
    - SELECT * FROM table;

Ввод и вывод данных

- Замечание:
  - Вы сможете самостоятельно найти в онлайне драйверы для многих основных баз данных, например:
    - Поиск в Google: Oracle SQL + pandas

Ввод и вывод данных

- Мы будем работать с базой данных SQLite, поскольку она поставляется вместе с Python, и позволяет легко создать временную базу данных в оперативной памяти

Сводные таблицы

PIERIAN DATA

Сводные таблицы – Pivot tables

- Сводные таблицы позволяют Вам поменять структуру данных, преобразуя ячейки данных с помощью нового индекса
- Это проще показать на визуальном примере...

Сводные таблицы – Pivot tables

- Датафрейм с повторяющимися значениями можно реорганизовать:

df

	foo	bar	baz	zoo
0	one	A	1	x
1	one	B	2	y
2	one	C	3	z
3	two	A	4	q
4	two	B	5	w
5	two	C	6	t

```
df.pivot(index='foo',
         columns='bar',
         values='baz')
```

bar	A	B	C
foo			
one	1	2	3
two	4	5	6

PIERIAN DATA

Сводные таблицы – Pivot tables

- Мы решаем, какие колонки будут новым индексом, колонками и значениями

df

	foo	bar	baz	zoo
0	one	A	1	x
1	one	B	2	y
2	one	C	3	z
3	two	A	4	q
4	two	B	5	w
5	two	C	6	t

```
df.pivot(index='foo',
         columns='bar',
         values='baz')
```

bar	A	B	C
foo			
one	1	2	3
two	4	5	6

PIERIAN DATA

Сводные таблицы – Pivot tables

- Обратите внимание: в колонках index и columns есть повторяющиеся значения

df

	foo	bar	baz	zoo
0	one	A	1	x
1	one	B	2	y
2	one	C	3	z
3	two	A	4	q
4	two	B	5	w
5	two	C	6	t

```
df.pivot(index='foo',
         columns='bar',
         values='baz')
```

bar	A	B	C
foo			
one	1	2	3
two	4	5	6

PIERIAN DATA

Сводные таблицы – Pivot tables

- Мы ничего не добавляли, просто те же данные структурировали по-другому

df

	foo	bar	baz	zoo
0	one	A	1	x
1	one	B	2	y
2	one	C	3	z
3	two	A	4	q
4	two	B	5	w
5	two	C	6	t

```
df.pivot(index='foo',
         columns='bar',
         values='baz')
```

bar	A	B	C
foo			
one	1	2	3
two	4	5	6

PIERIAN DATA

Сводные таблицы – Pivot tables

- Замечание! Далеко не всегда нужно применять сводные таблицы. Все наборы данных, которые мы будем использовать в этом курсе, не нуждаются в преобразовании с помощью сводных таблиц.

Сводные таблицы – Pivot tables

- Прежде чем применять операцию pivot(), нужно ответить самому себе на следующие вопросы:
  - На какой вопрос мы хотим найти ответ?
  - Как должен выглядеть датафрейм для ответа на этот вопрос? Нужно ли выполнять pivot()?
  - Как должна выглядеть результирующая сводная таблица?

Сводные таблицы – Pivot tables

- Ещё в Pandas есть метод pivot_table, который позволяет применить дополнительную агрегатную функцию.
- Это также можно сделать с помощью метода .groupby()
- Давайте изучим методы .pivot() и .pivot_table() в Pandas!

Упражнения по Pandas
Обзор

PIERIAN DATA

Упражнения по Pandas

- Давайте проверим Ваши новые навыки Pandas!
- Имейте ввиду:
  - Многие задачи можно решить с помощью одной-двух строк кода в Pandas
  - Задачи можно решать разными способами
  - Не запускайте ячейки непосредственно над ожидаемыми результатами

PIERIAN DATA

Упражнения по Pandas
РЕШЕНИЯ

PIERIAN DATA