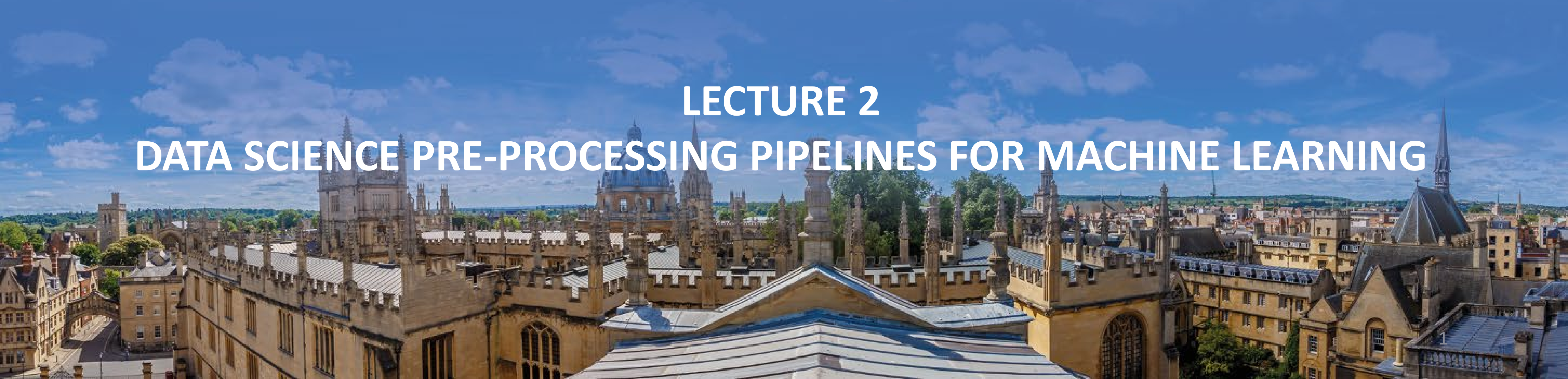
Weekly Oxford Worldwide

DEPARTMENT FOR
CONTINUING
EDUCATION

UNIVERSITY OF
OXFORD

PYTHON PROGRAMMING FOR DATA SCIENCE – PART 2
MASSIMILIANO IZZO & NICHOLAS DAY



LECTURE 2
DATA SCIENCE PRE-PROCESSING PIPELINES FOR MACHINE LEARNING

Announcements

https://gitlab.com/data-science-course/pp4ds-pt2-tt2022

Last week

- Introduction to Machine Learning
- Unsupervised/Supervised Learning
- Batch/Online Learning
- Instance-based/Model-based learning
- Underfitting and overfitting
- Training, validation and test set

End-to-End Machine Learning Project

Aurélien Géron, ***Hands-On Machine Learning***

1. Look at the big picture.
2. Get the data.
3. Discover and visualize the data to gain insights.
4. Prepare the data for Machine Learning algorithms.
5. Select a model and train it.
6. Fine-tune your model.
7. Present your solution.
8. Launch, monitor, and maintain your system.

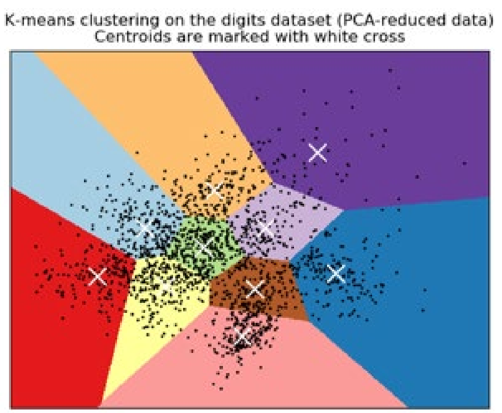
End-to-End Machine Learning Project

Aurélien Géron*, Hands-On Machine Learning*

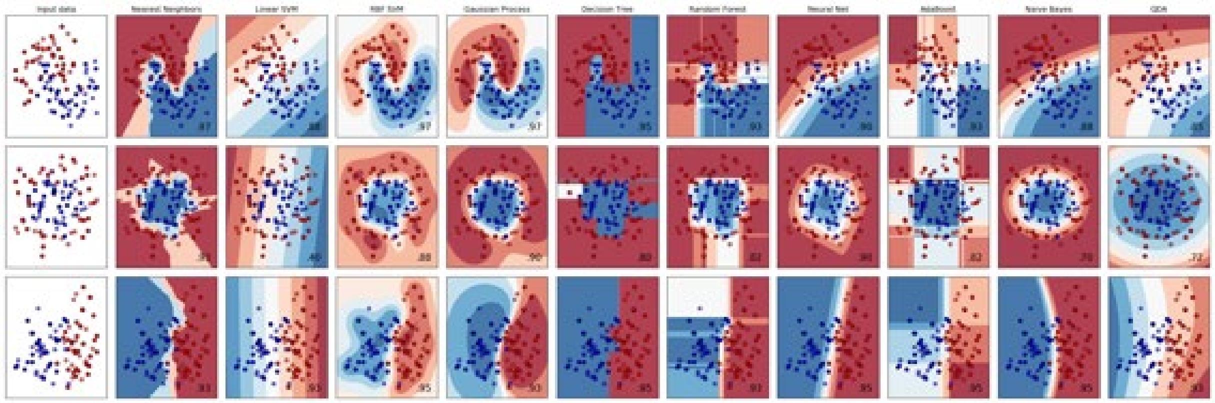
1. Look at the big picture.
2. Get the data.
3. Discover and visualize the data to gain insights.
4. Prepare the data for Machine Learning algorithms.
5. Select a model and train it.
6. Fine-tune your model.
7. Present your solution.
8. Launch, monitor, and maintain your system.

scikit-learn

- From today we will start using the scikit-learn library algorithms to perform some pre-processing steps, together with Pandas and NumPy
- scikit-learn has been designed to tie in with the set of numeric and scientific packages centered around the NumPy and SciPy libraries.
- scikit-learn offers a clean and simple API: https://arxiv.org/pdf/1309.0238.pdf



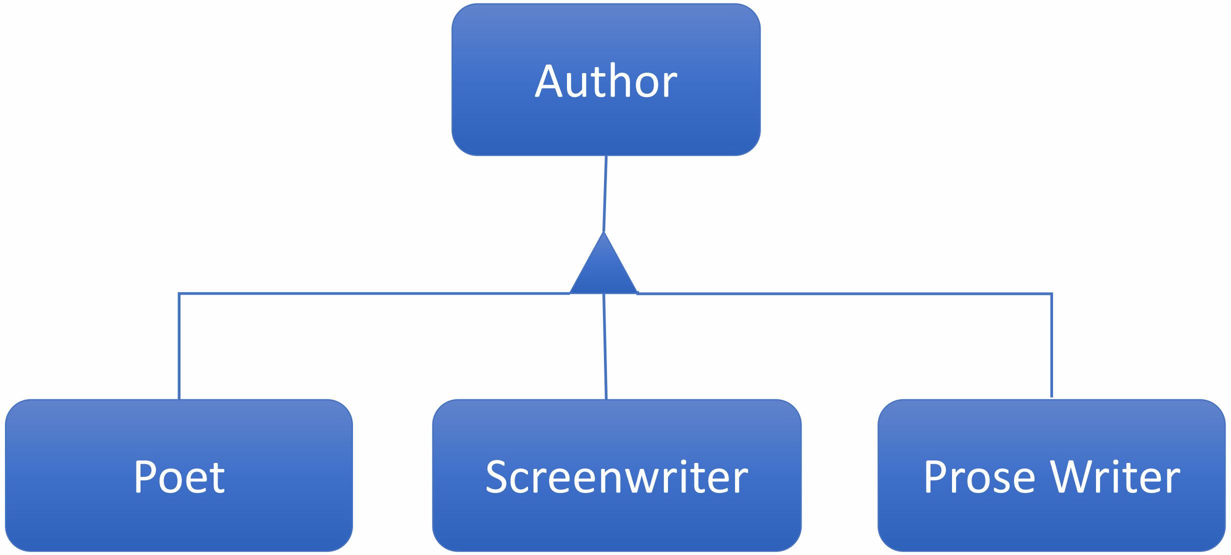
K-means clustering on the digits dataset (PCA-reduced data)
Centroids are marked with white cross



Object and classes

- Objects are a high level programming construct. They contain:
  - one or more variables (“attributes”)
  - the set of operat that work on these variable (“methods”)
- Every object belongs to a class => an object is an instance of a class
- (Public) methods constitute the interface (API) of the class
- Python is an object-oriented language
  - Almost everything (i.e. any variable) is an object and belongs to a certain class
  - Attributes and methods can be accessed with the dot notation:
    - `object_name.attribute_name`
    - `object_name.method_name()`

Inheritance



- Inheritance is the capability of one class to derive or inherit the properties from some another class. The benefits of inheritance are:
    - It represents real-world relationships well.
    - It provides **reusability** of a code. We don’t have to write the same code again and again.
    - It is transitive in nature, which means that if class B inherits from another class A, then all the subclasses of B would automatically inherit from class A.

Scikit-learn API

- All objects within scikit-learn share a uniform common basic API consisting of three complementary interfaces:
  - **Estimators**, the base object (implement a **.fit()** method to learn from data)
  - **Predictors** (for supervised and some unsupervised learning,implement a **.predict()** method)
  - **Transformers** (for filtering or modifying the data, implement a **.transform()** method)
- You can create your own estimators, predictors, and tranformers following the scikit-learn interface

Framing the Problem

- We want to be able to predict the price of houses in Kings County, Washington, US
    - is it supervised, unsupervised, or Reinforcement Learning?
    - is it a classification task, a regression task, or something else?
    - should you use batch learning or online learning techniques?

Importing the Data

- Create a workspace (with enough storage space).
- Get the data.
- Convert the data to a format you can easily manipulate (without changing the data itself).
- Ensure sensitive information is deleted or protected (e.g., anonymized).
- Check the size and type of data (time series, sample, geographical, etc.).
- Sample a test set, put it aside, and never look at it (no data snooping!).

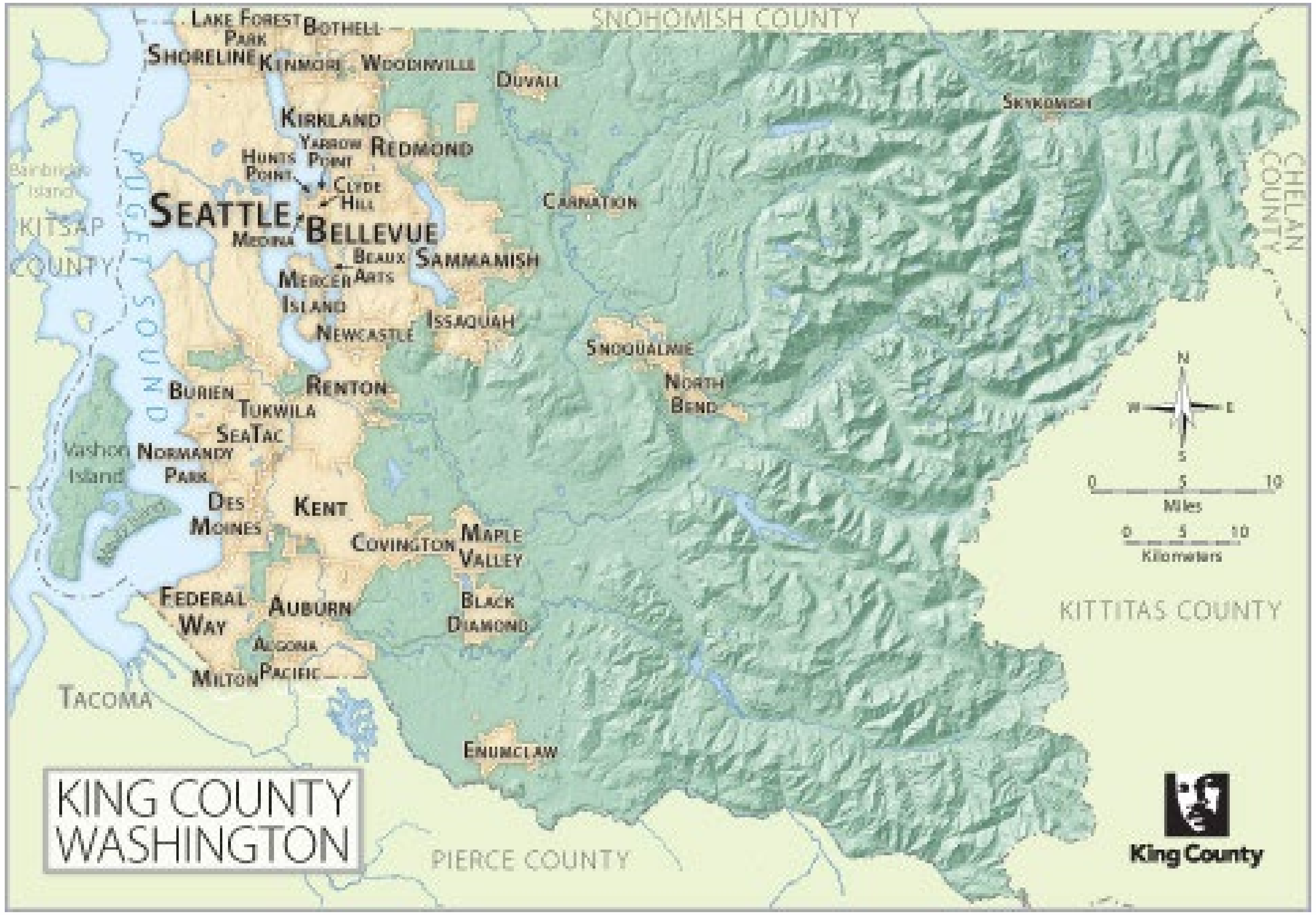
Inspecting the data to gain insights.

- Study each attribute and its characteristics:
  - Name
  - Type (categorical, int/float, bounded/unbounded, text, structured, etc.)
  - % of missing values
  - Noisiness and type of noise (stochastic, outliers, rounding errors, etc.)
  - Usefulness for the task
  - Type of distribution (Gaussian, uniform, logarithmic, etc.)
- For supervised learning tasks, identify the target attribute(s).

Inspecting the data to gain insights.

- Visualize the data.
- Study the correlations between attributes.
- Study how you would solve the problem manually.
- Identify the promising transformations you may want to apply.
- Identify extra data that would be useful

Visualising the data



Data Preparation (I)

- Data cleaning
  - Fix or remove outliers (optional).
  - Fill in missing values (e.g., with zero, mean, median…) or drop their rows (or columns).
- Feature selection (optional):
  - Discard the attributes that provide no useful information for the task.

Data preparation (I)

- Feature engineering, where appropriate:
  - Discretize continuous features.
  - Decompose features (e.g., categorical, date/time, etc.).
  - Add promising transformations of features
  - Aggregate features into promising new features.
- Feature scaling:
  - Standardize or normalize features.
- Dimensionality Reduction (optional)
  - Principal Component Analysis (PCA)

Pre-processing Pipeline

- A sequence of data processing components is called a data *pipeline*. Pipelines are very common in Machine Learning systems, since there is a lot of data to manipulate and many data transformations to apply.
- The goal of this lesson will be to complete a data pre-processing pipeline for our KC dataset