https://ttpoll.com/p/911058

COSC 325: Introduction to Machine Learning

Dr. Hector Santos-Villalobos

Dr. Santos

THE UNIVERSITY OF
TENNESSEE
KNOXVILLE

Lecture 04: SciKit-Learn and Nearest Neighbor Classifier

THE UNIVERSITY OF TENNESSEE KNOXVILLE

https://ttpoll.com/p/911058

Class Announcements

Homework:

Homework #1 due this Sunday

- Check instructions
- Check previous posts

Homework #2 published, due 09/08

Lectures:

We may still need to finish Scikit-learn material. I will record any remaining content.

Quizzes:

Quiz #2 published

Course Project:

Give feedback by 09/10

Propose new datasets/use cases by 09/05

Friday, the last day to propose a team

https://ttpoll.com/p/911058

Today's Topics

Wrap up Numpy

Matplotlib and Pandas

Scikit-learn

https://ttpoll.com/p/911058

Last Lecture

- Notation
  - Sample: vector feature x, $(m, 1)$
    - List of numbers
    - All modalities map to a list of numbers
  - Set of samples: matrix X, (n, m)
  - Rows -> samples (n)
  - Columns -> features $(m \text{ or } n_x)$
  - Targets -> One per sample. Vector $(n, 1)$
- Capacity
  - Ability of the model to fit the data
  - Increases with # of parameters
- Numpy Python library

https://ttpoll.com/p/911058

Let’s go back to Numpy

https://ttpoll.com/p/911058

Pop Quiz

1 | MULTIPLE CHOICE

POINTS: 1 | Edit

In Python, perform the following steps:

1. Create three Numpy arrays ***a*** = [1, 2, 3], ***b*** = [4, 5, 6], and ***c*** = [3, 2, 1] with shape (1, 3).
2. Add 5 to ***a***, divide ***b*** by 4, and multiply ***c*** by 3.
3. Concatenate ***a*** and ***c*** on the zero axis and store the result on variable ***d.***
4. Finally, store the dot product between ***d*** and ***c*** on variable ***f.*** (HINT: Do you need to reshape ***c*** before computing the dot product?)

What are the contents and shape of **f**?

A. f=[120], f.shape=(1,1)

B. f=[126], f.shape=(1)

C. f=[[120][126]], f.shape=(2,1)

D. f=[[120,126]], f.shape=(1,2)

https://ttpoll.com/p/911058

Pop Quiz

1 | MULTIPLE CHOICE

POINTS: 1 | Edit

In Python, perform the following steps:

1. Create three Numpy arrays ***a*** = [1, 2, 3], ***b*** = [4, 5, 6], and ***c*** = [3, 2, 1] with shape (1, 3).
2. Add 5 to ***a***, divide ***b*** by 4, and multiply ***c*** by 3.
3. Concatenate ***a*** and ***c*** on the zero axis and store the result on variable ***d.***
4. Finally, store the dot product between ***d*** and ***c*** on variable ***f***. (HINT: Do you need to reshape ***c*** before computing the dot product?)

What are the contents and shape of **f**?

A. f=[120], f.shape=(1,1)

B. f=[126], f.shape=(1)

C. f=[[120][126]], f.shape=(2,1)

D. f=[[120,126]], f.shape=(1,2)

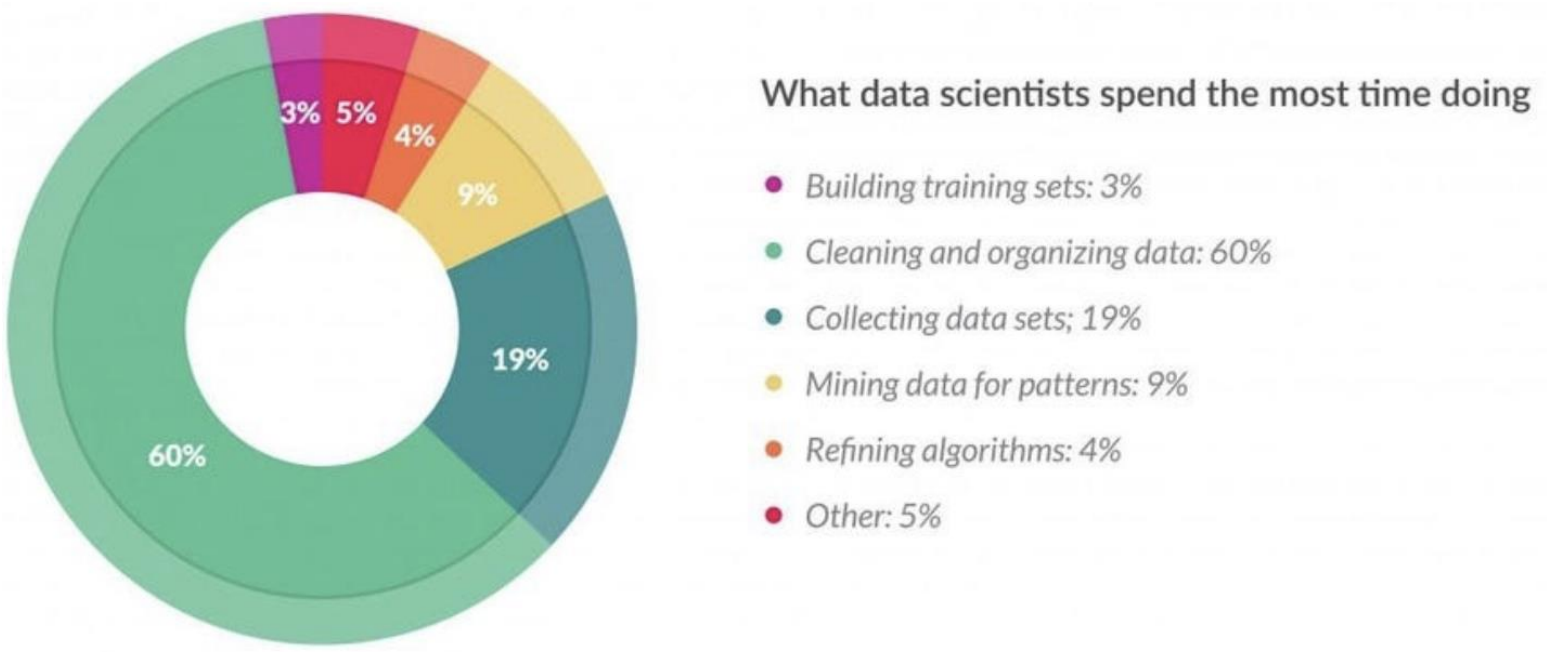
```python
import numpy as np

a = np.array([1, 2, 3])
print(f"Original shape of a is {a.shape}")
a = a.reshape((1,3)) + 5
print(f"Shape of a is {a.shape}")
b = np.array([[4, 8, 12]]) / 4
print(f"Shape of b is {b.shape}")
c = np.array([[3, 2, 1]]) * 3
print(f"Shape of c is {c.shape}")

d = np.concatenate((a, c), axis=0)
print(f"Shape of d is {d.shape}")

# take the transpose of c and compute the dot product between c and d
f = np.dot(d, c.T)
print(f"f value is {f}")
print(f"f shape is {f.shape}")
```

```
Original shape of a is (3,)
Shape of a is (1, 3)
Shape of b is (1, 3)
Shape of c is (1, 3)
Shape of d is (2, 3)
f value is [[120]
 [126]]
f shape is (2, 1)
```

https://ttpoll.com/p/911058

Data

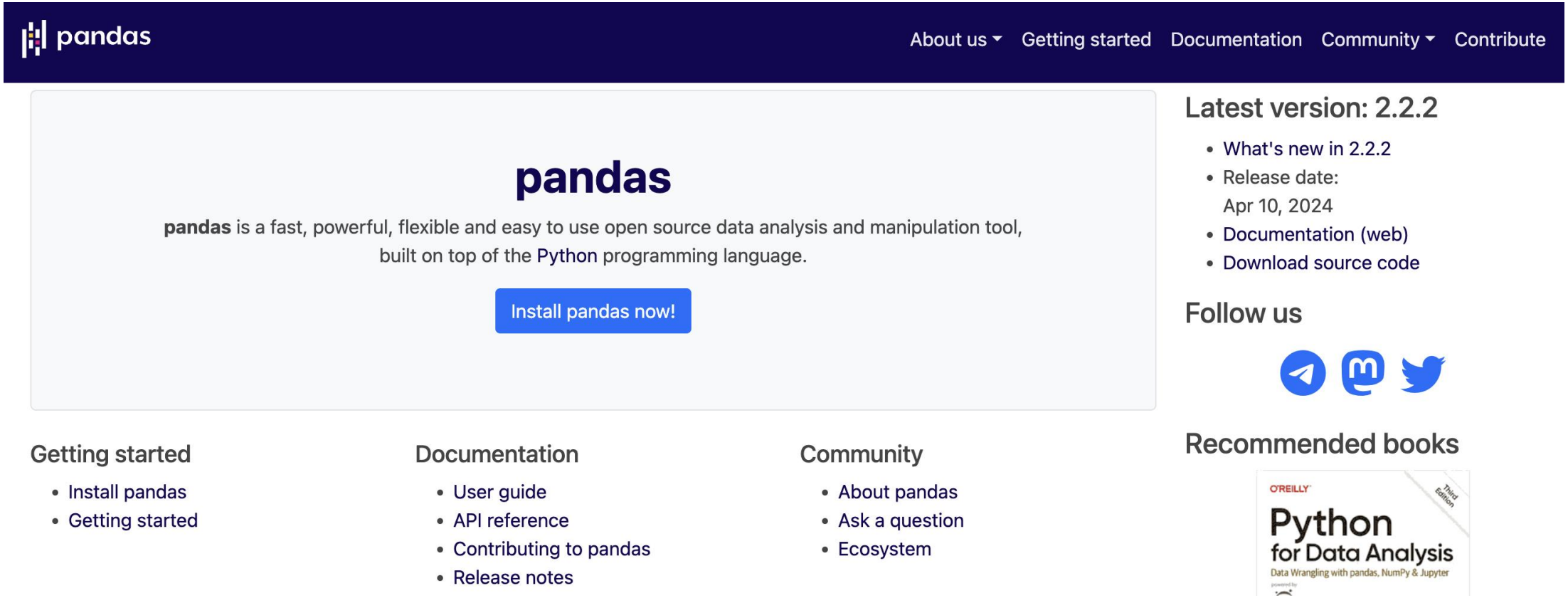
- Data is fundamental to what we do in machine learning
- We need data to learn from, but data isn't always handed to us in the best format
- We likely don't know much about the data a priori



Slide credit: Dr. Schumman

https://ttpoll.com/p/911058

Pandas



https://pandas.pydata.org/

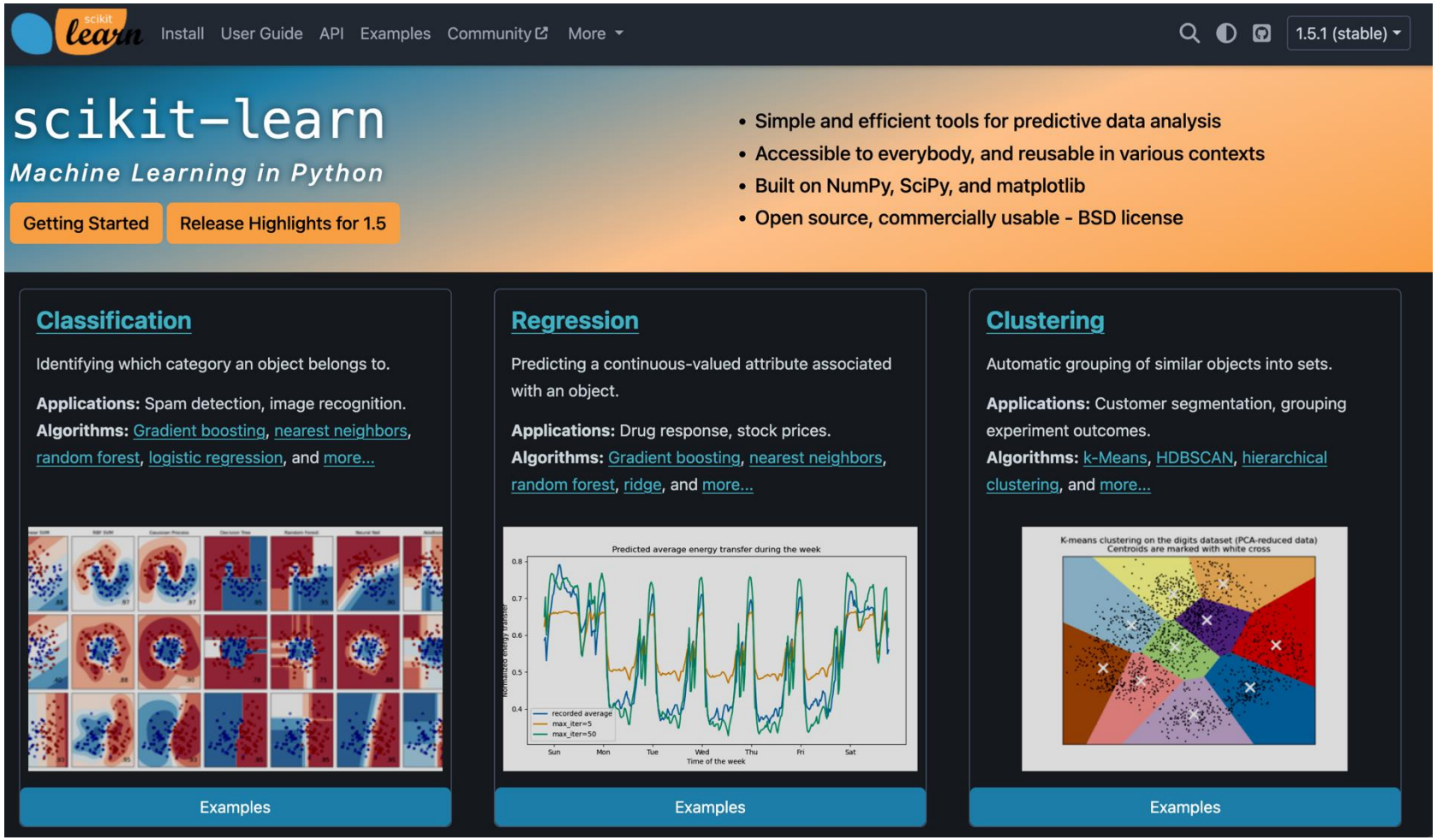
Pandas

- **Open-source** data analysis and manipulation tool
- A fast and efficient **DataFrame** object for data manipulation with integrated indexing
- Tools for **reading and writing data** between in-memory data structures and different formats
- Intelligent **data alignment** and integrated handling of **missing data**: gain automatic label-based alignment in computations and easily manipulate messy data into an orderly form;
- Flexible **reshaping** and pivoting of data sets;
- Intelligent label-based **slicing**, **fancy indexing**, and **subsetting** of large data sets;
- Columns can be inserted and deleted from data structures for **size mutability**;
- Aggregating or transforming data with a powerful **group by** engine allowing split-apply-combine operations on data sets;
- High-performance **merging and joining** of data sets;
- Highly **optimized for performance**, with critical code paths written in Cython or C.

Slide credit: Dr. Schumman

https://ttpoll.com/p/911058

Scikit-learn



https://scikit-learn.org/

Coefficient of Determination

In statistics, the **coefficient of determination**, denoted R^2 or r^2 and pronounced "R squared", is the proportion of the variation in the dependent variable that is predictable from the independent variable(s). --Wikipedia

Range from 0-1.

Mean target.

$$SS_{residual} = \sum_i \left(y^{(i)} - \hat{y}^{(i)}\right)^2 \qquad SS_{total} = \sum_i \left(y^{(i)} - \bar{y}\right)^2 \qquad R^2 = 1 - \frac{SS_{residual}}{SS_{total}}$$

https://ttpoll.com/p/911058

Overview K-Nearest Neighbor (KNN)

- Intuition: Similar features x tend to have similar targets y.
- Training: Store all samples and labels as reference
- Inference:
  - Compute the distance from input sample to all training samples (e.g., Euclidian distance)
  - Identify the closest k training samples
  - Assign the most common training label

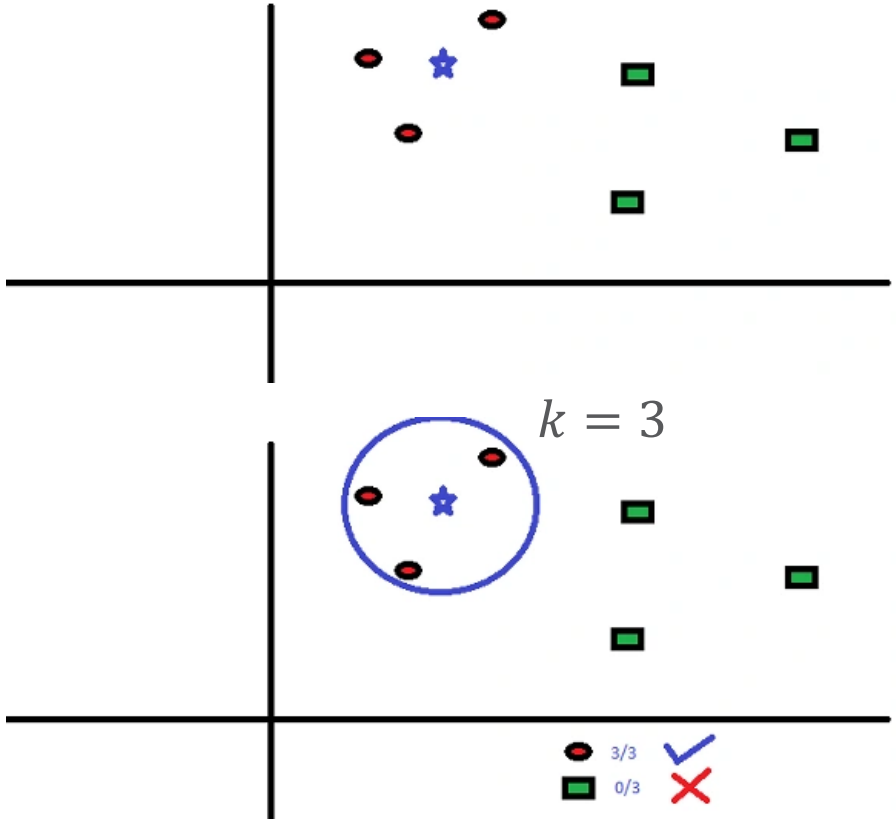


Image source: https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/

https://ttpoll.com/p/911058

Pandas and Scikit-Learn Notebook

https://ttpoll.com/p/911058

Next Lectures

- Linear Regression
- Gradient Descent