Data Analysis 2021 Spring





**Lecture 02:**

**Statistical Learning & Python Programming**

March 10 & 15, 2021

**Taesoo Kwon**

[tskwon@seoultech.ac.kr](mailto:tskwon@seoultech.ac.kr)

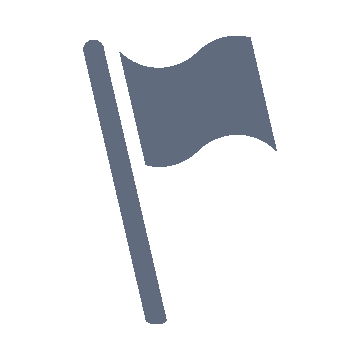
©copyright CIEL 2015 https://ciel.seoultech.ac.kr/

# Course Schedule (Tentative)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Week** | **Topics** | **Note** | **Date (W)** | **Date (M)** |
| 1 | Orientation, Statistical Learning (Ch2) | Online | 03/03 | 03/08 |
| **2** | Statistical Learning (Ch2), Python Programming | Online | 03/10 | 03/15 |
| 3 | Probability & Statistics | Online | 03/17 | 03/22 |
| 4 | Probability & Statistics | Online | 03/24 | 03/29 |
| 5 | Linear Regression (Ch3) | Online | 03/31 | 04/05 |
| 6 | Linear Regression (Ch3) | Online | 04/07 | 04/12 |
| 7 | Classification (Ch4) | Online | 04/14 | 04/19 |
| 8 | **Midterm exam** | **7pm or Class hours (W1-W7)** | **04/21or26** | **04/21or26** |
| 9 | Resampling Methods (Ch5) | Online | 04/28 | 05/03 |
| 10 | Linear Model Selection and Regularization (Ch6) | Online | 05/05 | 05/10 |
| 11 | Moving Beyond Linearity (Ch7) | Online | 05/12 | 05/17 |
| 12 | Tree-Based Methods (Ch8) | Online | 05/19 | 05/24 |
| 13 | Support Vector Machines (Ch9) | Online | 05/26 | 05/31 |
| 14 | Unsupervised Learning (Ch10) | Online | 06/02 | 06/07 |
| 15 | **Final exam** | **7pm or Class hours (W9-W14)** | **06/09or14** | **06/09or14** |

©copyright CIEL 2015 **2**

* Statistical Learning (Ch2-2)



**OUTLINES**

* + Assessing Model Accuracy
* Python Programming
* Summary & Next class

©copyright CIEL 2015



**Statistical Learning (Ch2-2)**



**: Assessing Model Accuracy**

* Statistical Learning
  + Assessing model accuracy
* Python Programming
* Summary & Next class

©copyright CIEL 2015 https://ciel.seoultech.ac.kr/

# Assessing Model Accuracy

* Suppose we fit a model performs.

𝑓𝑓̂(𝑥𝑥) to some training data Tr =

𝑁𝑁, and we wish to see how well it

* + We could compute the average squared prediction error over Tr :

𝑥𝑥𝑖𝑖 , 𝑦𝑦𝑖𝑖

1



* This may be biased toward more overfit models.
  + Instead we should, if possible, compute it using fresh test data Te =

𝑀𝑀 1

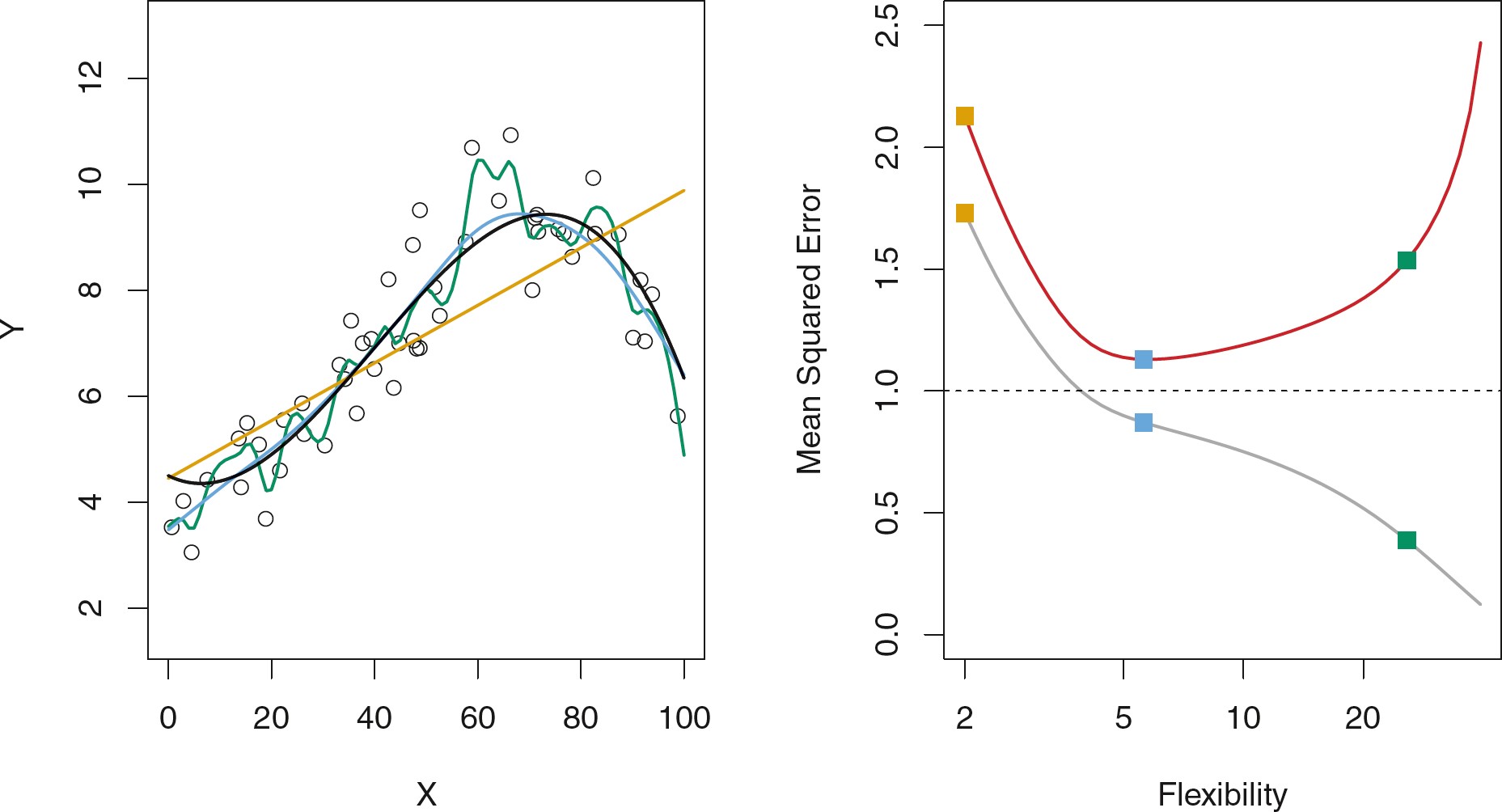
𝑥𝑥𝑖𝑖 , 𝑦𝑦𝑖𝑖

:



©copyright CIEL 2015 **5**

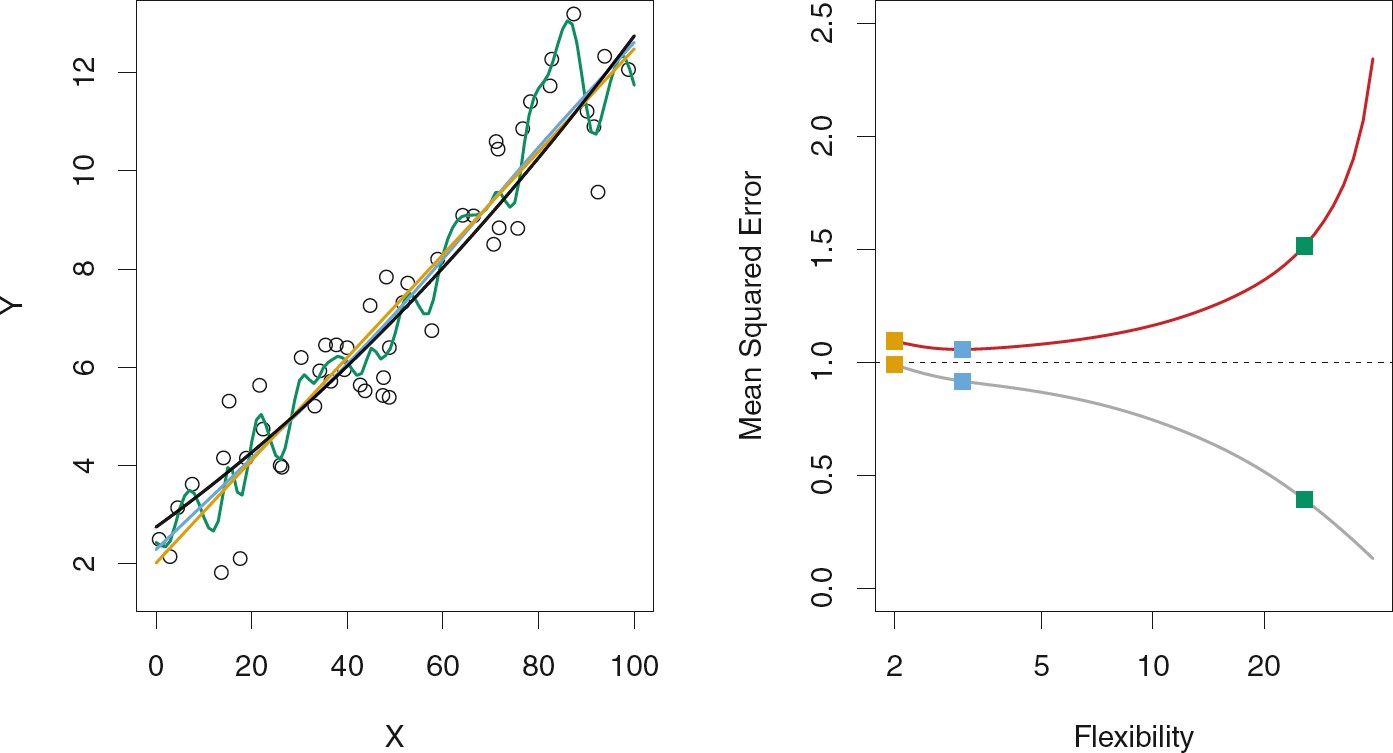
# Assessing Model Accuracy [cont.]



* Black curve is truth. Red curve on right is MSETe, grey curve is MSETr. Orange, blue and green curves/squares correspond to fits of different flexibility.

©copyright CIEL 2015 **6**

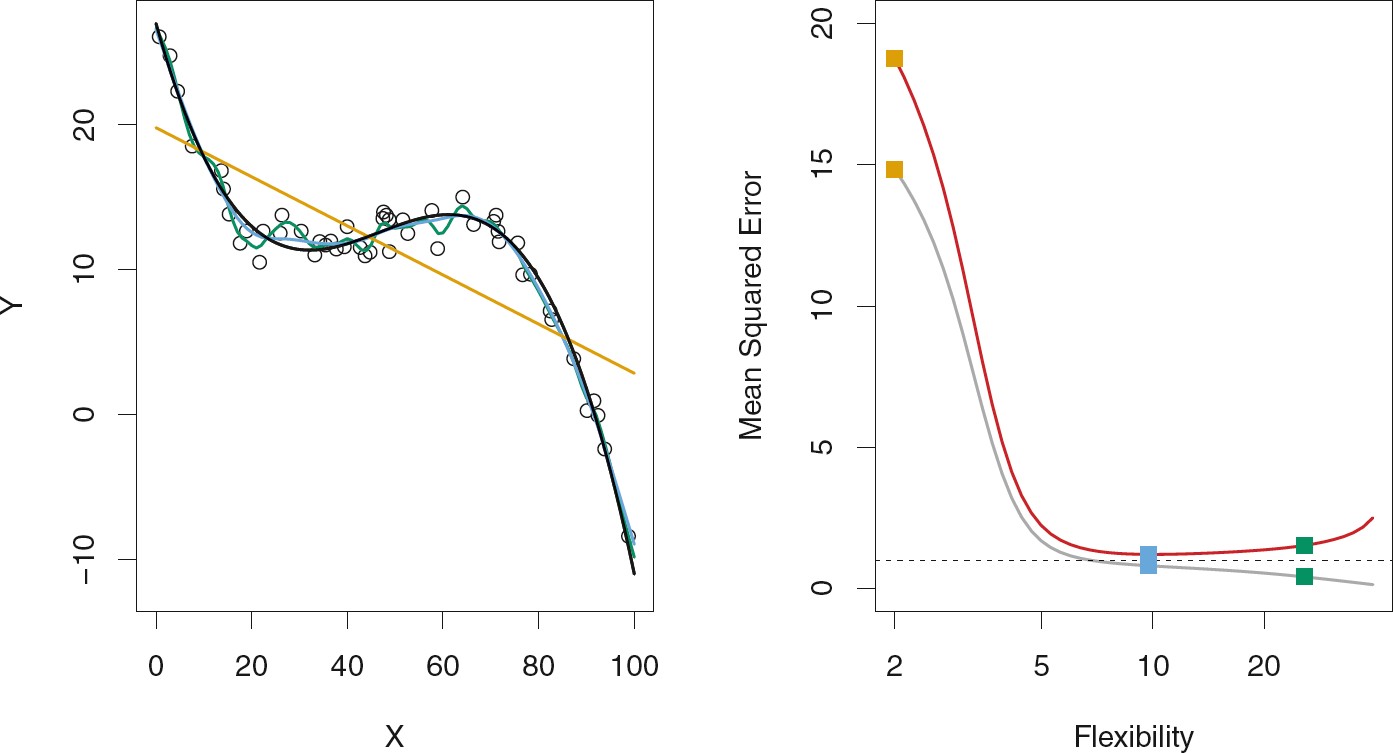
# Assessing Model Accuracy [cont.]



* Here the truth is smoother, so the smoother fit and linear model do really well.

©copyright CIEL 2015 **7**

# Assessing Model Accuracy [cont.]



* Here the truth is wiggly and the noise is low, so the more flexible fits do the best.

©copyright CIEL 2015 **8**

# Bias-Variance Trade-off

𝑥𝑥0, 𝑦𝑦0

* Suppose we have fit a model drawn from the population.

𝑋𝑋

𝑓𝑓̂(𝑥𝑥) to some training data Tr, and let

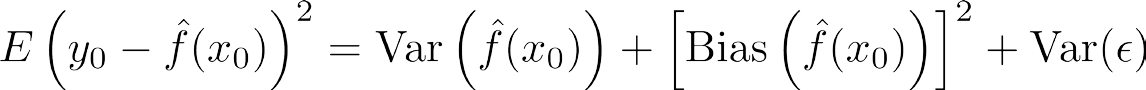
be a test observation

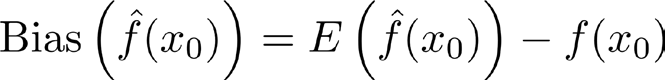
* If the true model is 𝑌𝑌 = 𝑓𝑓

𝑋𝑋

+ 𝜖𝜖 (with 𝑓𝑓

= 𝐸𝐸(𝑌𝑌|𝑋𝑋 = 𝑥𝑥)), then



* The expectation averages over the variability of 𝑦𝑦0 as well as the variability in Tr. Note that
* Typically as the flexibility of

𝑓𝑓̂

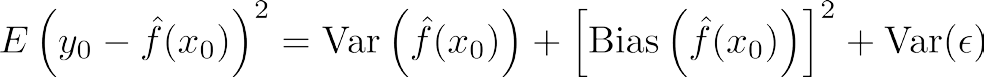
increases, its variance increases, and its bias decreases. So

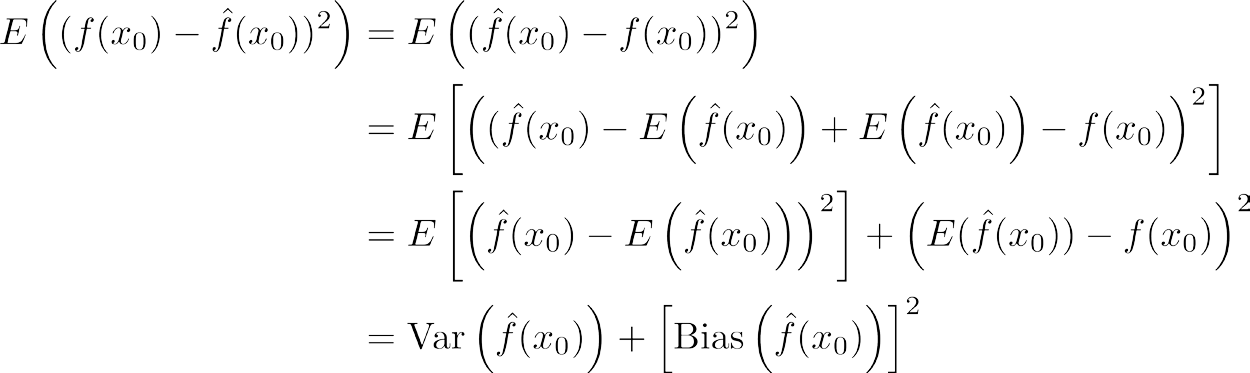
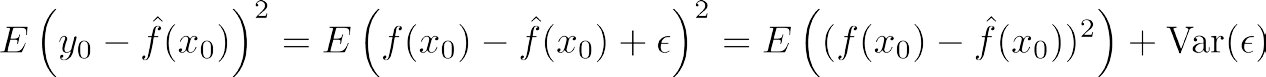
choosing the flexibility based on average test error amounts to a bias-variance trade-off

.

©copyright CIEL 2015 **9**

# Proof

* Proof of 



©copyright CIEL 2015 **10**

# [Review] Regression Function *f* (*X*)

* Is also defined for vector *X* ; e.g.
* Is the ideal or optimal predictor of *Y* with regard to

mean-squared prediction error: *f* (*X*) = *E*(*Y* | *X* = *x*) is the function that minimizes *f* (*X*) = *E*[ ( *Y – g*(*X*) )2 | *X* = *x* ] over all functions *g* at all points *X* = *x*.

 = *Y – f* (*X*) is the irreducible error, i.e., even if we knew *f* (*X*), we would still make errors in prediction, since at each *X* = *x* there is typically a distribution of possible *Y* values.

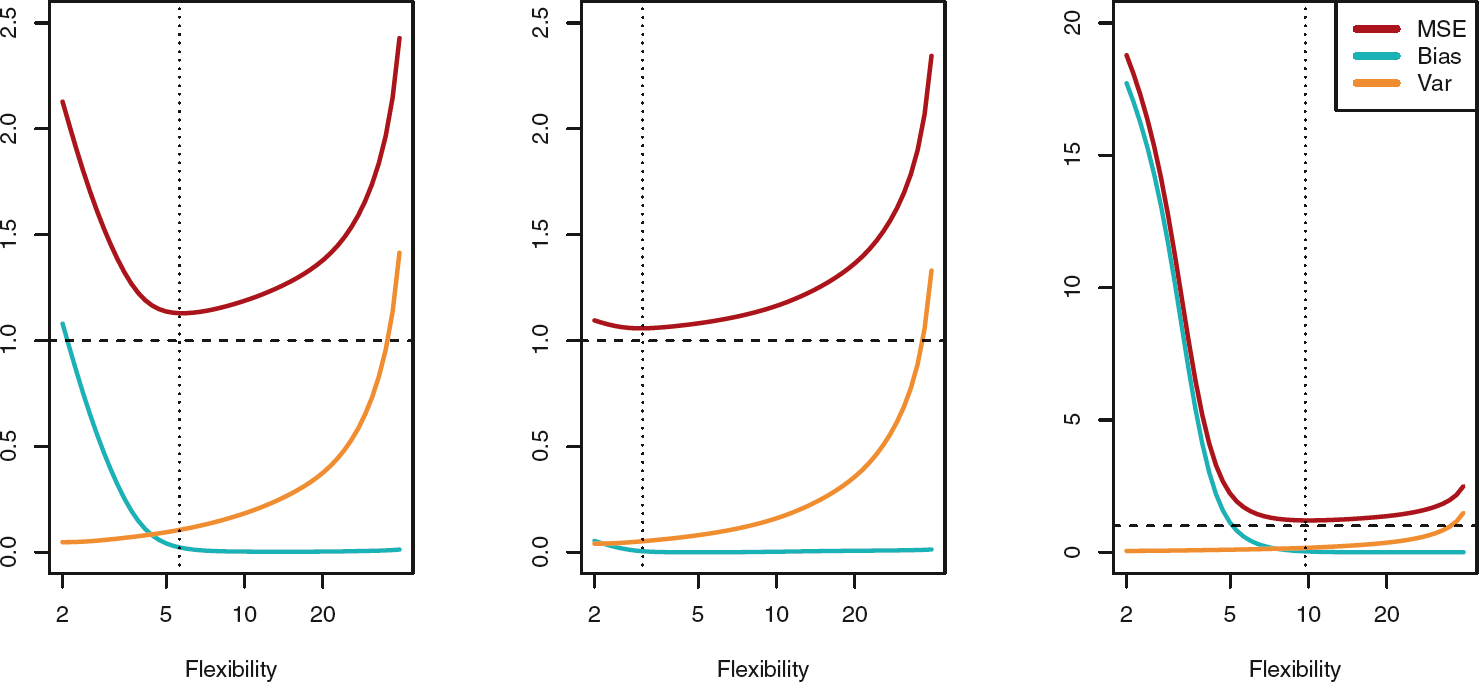
* For any estimate  of , we have



Reducible Irreducible

©copyright CIEL 2015 **11**

# Bias-Variance Trade-off for the Three Examples



©copyright CIEL 2015 **12**

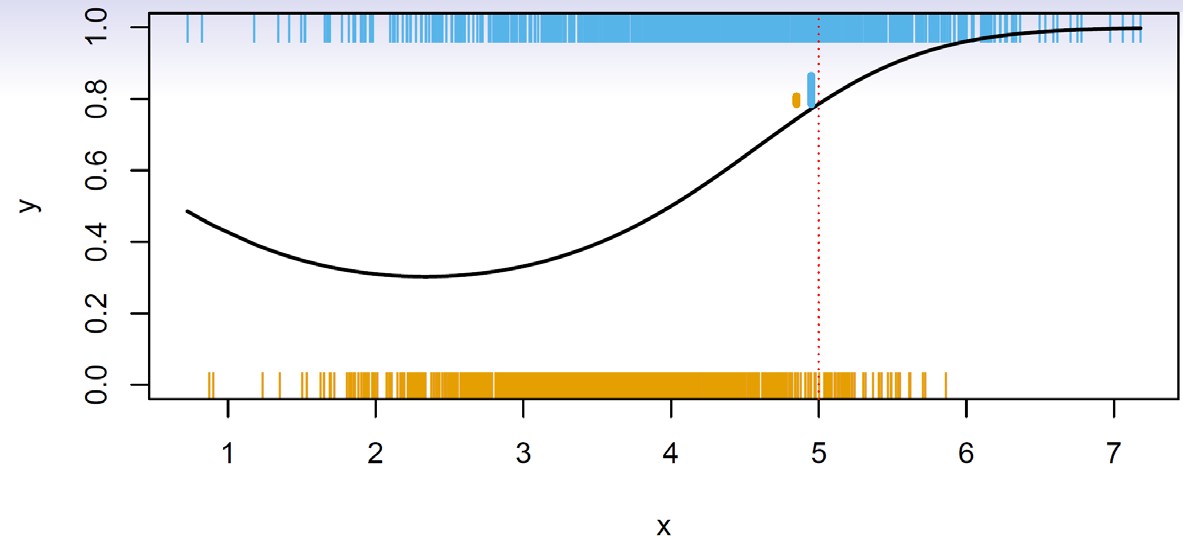
# Classification Problems

* Here the response variable *Y* is qualitative, e.g. email is one of *C*=(spam , ham) (ham=good email), digit class is one of *C* = {0, 1,…, 9}.
* Our goals are to:
  + Build a classifier *C*(*X*) that assigns a class label from *C* to a future unlabeled observation *X*.
  + Assess the uncertainty in each classification
  + Understand the roles of the different predictors among



©copyright CIEL 2015 **13**

# Classification Problems [cont.]



Occurrence of a one

Probability of the ones

Exact probability of the ones

Occurrence of a zero

* Is there an ideal *C*(*X*)? Suppose the *K* elements in *C* are numbered 1,2,…,*K*. Let



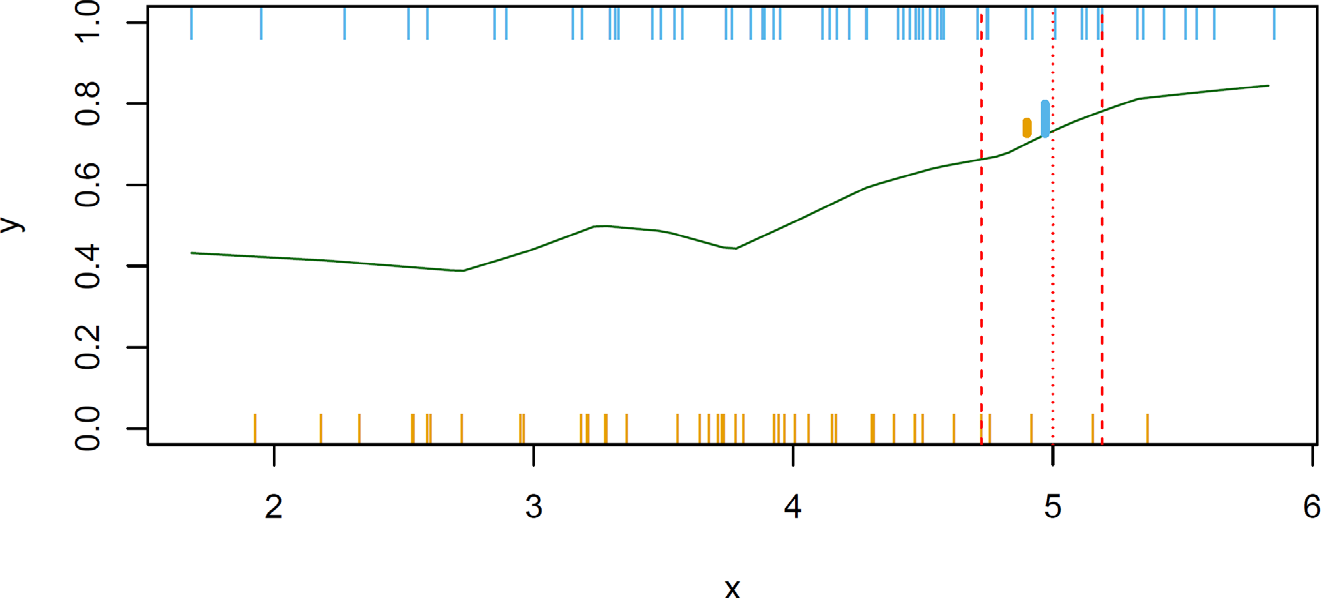
* These are the conditional class probabilities at *x*; e.g. see little barplot at x = 5. Then the Bayes optimal classifier at *x* is



©copyright CIEL 2015 **14**

# Classification Problems [cont.]

* + Nearest-neighbor averaging can be used as before.



100 points

Estimated probability of the ones

No zero at 5, so using 10% neighborhood

* + Also breaks down as dimension grows. However, the impact on

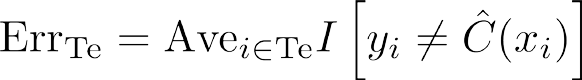
= 1,,2,…,*K*

𝐶𝐶̂(𝑥𝑥) is less than

𝑝𝑝̂𝑘𝑘 (𝑥𝑥), *k*

©copyright CIEL 2015 **15**

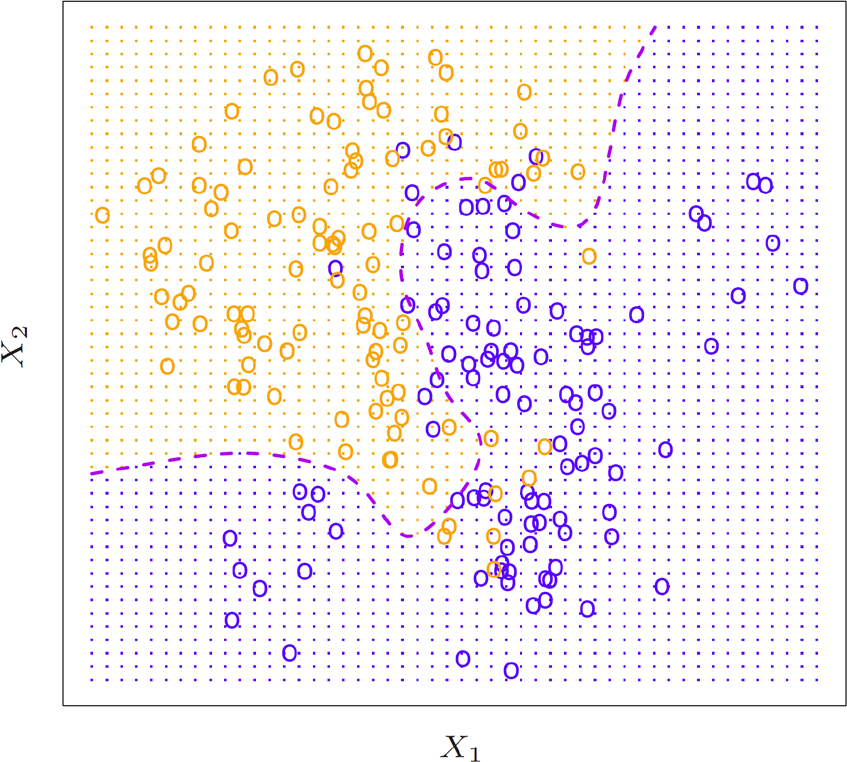
# Classification: Some Details

* Typically we measure the performance of 𝐶𝐶̂(𝑥𝑥) using the misclassification error rate:
* The Bayes classifier (using the true *pk*(*x*)) has smallest error (in the population).
* Support-vector machines build structured models for *C*(*X*).
* We will also build structured models for representing the *pk*(*x*).
  + E.g., Logistic regression, generalized additive models.

©copyright CIEL 2015 **16**

# Example: *K*-Nearest Neighbors in Two Dimensions

* Ideal, all of the true probabilities known



Region for orange class

Bayes decision boundary:

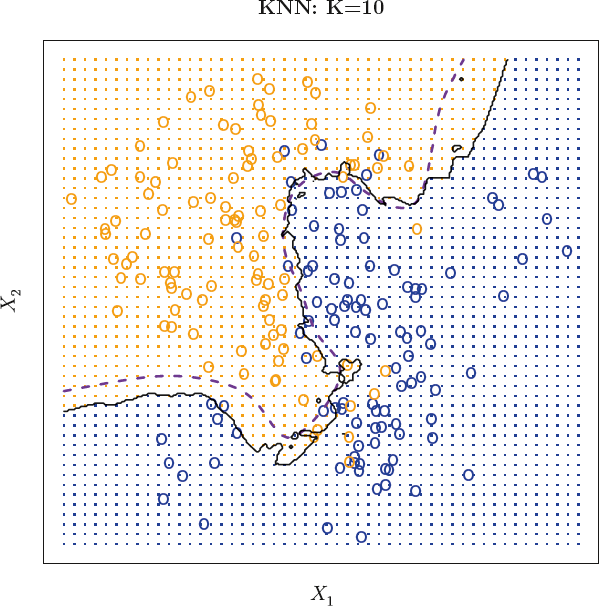
Here, contour with equal probabilities

Region for blue class

©copyright CIEL 2015 **17**

# KNN (*K*=10)

* Using neighborhood



Difference between true and estimated boundaries

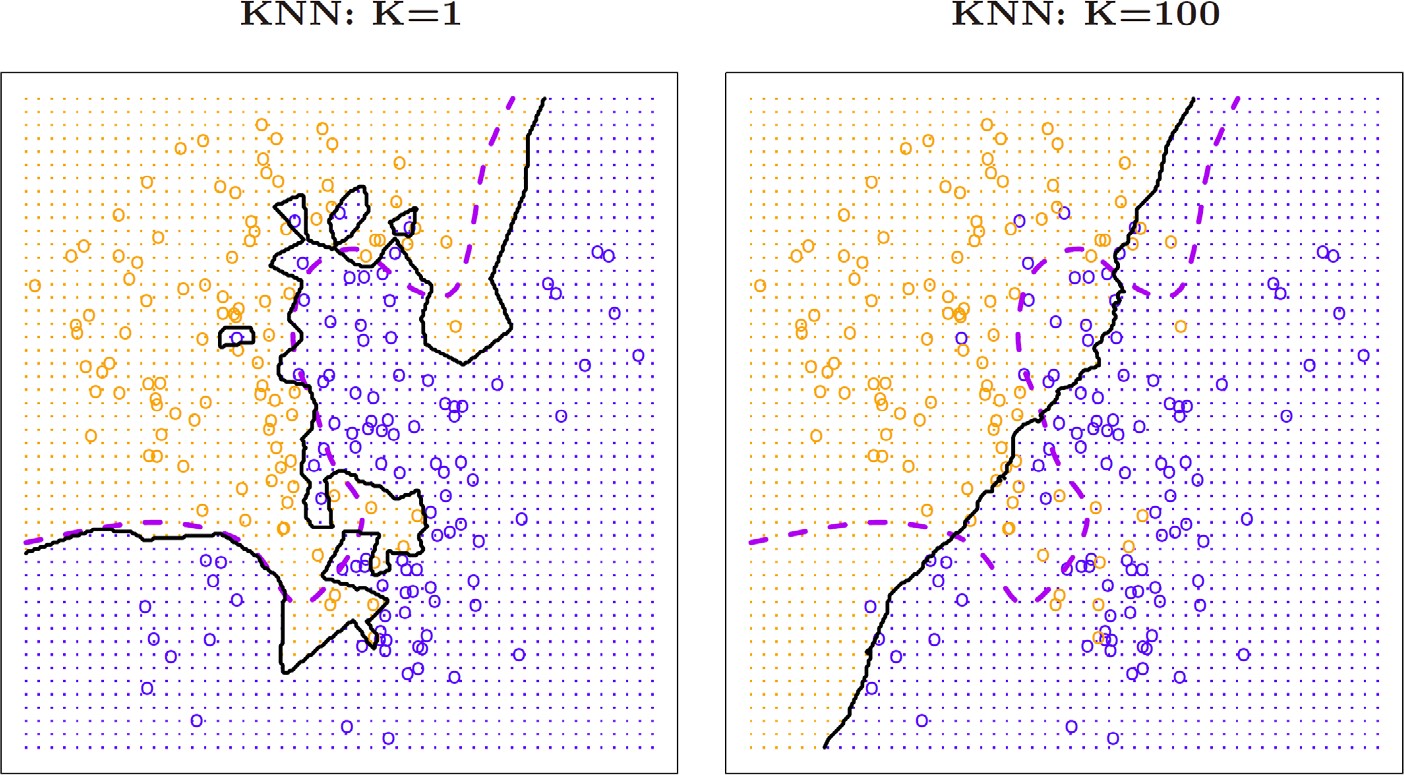
Black curve: Estimated decision boundary

* + In this example, the 10 closest points to the target point
  + Then, estimating the probability of a center point
* K=10: Good choice

©copyright CIEL 2015 **18**

# Comparison of KNN Decision Boundaries

Piecewise linear decision boundary

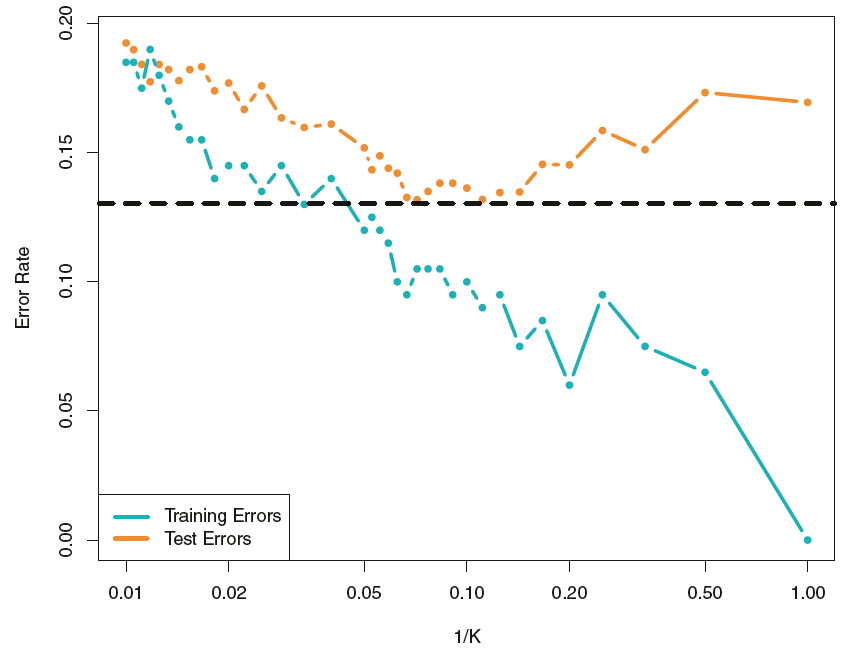


* K=1
  + Nearest neighbor classifier
  + Depending on the closest point
  + Popular choice
* K=100
  + Almost like a linear boundary
  + Less interesting
  + Not picking up the nuances of the decision boundary

©copyright CIEL 2015 **19**

# KNN Training Error

* In KNN,



Bayes error

* + Choice of K is a tuning parameter
* Using validation set, we can decide the value of K
* K=1 (nearest neighbor classification)
  + For about one third of classification problems

**Less flexibility**

**Higher flexibility**

©copyright CIEL 2015 **20**

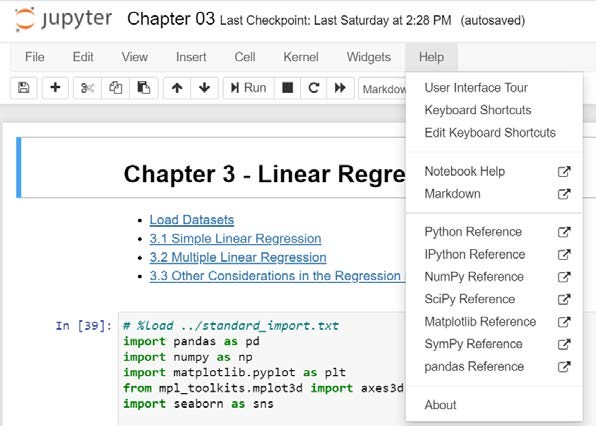
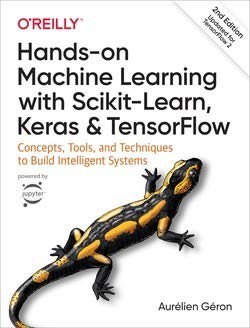
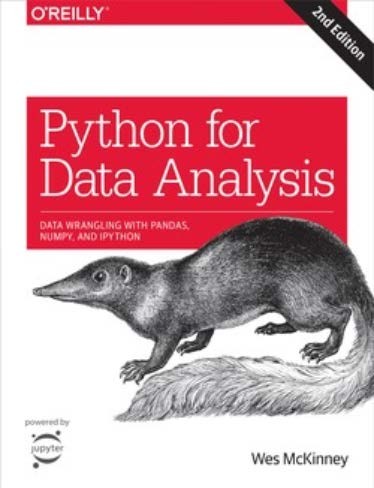


**Python Programming**

* + - Statistical Learning
      * Assessing model accuracy
    - Python Programming
    - Summary & Next class

©copyright CIEL 2015 https://ciel.seoultech.ac.kr/

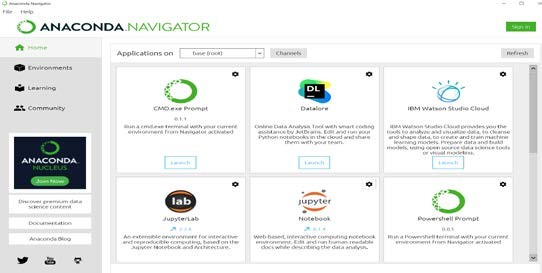
# [Review] Python Programming for SL

* Jupyter notebook (within Anaconda)
  + Weekly assignments: submitting your ipynb files
* Python built-in data structures and syntax
  + List, Tuple, Dictionary, Set
  + Functions, File input/output, Loops, Branches etc.
* NumPy: arrays and vectorized computation
* Pandas: panel data or python data analysis
* Data visualization
  + Matplotlib, seaborn
* Scikit-learn
* Others
  + Statsmodels, Mpl\_toolkits.mplot3d

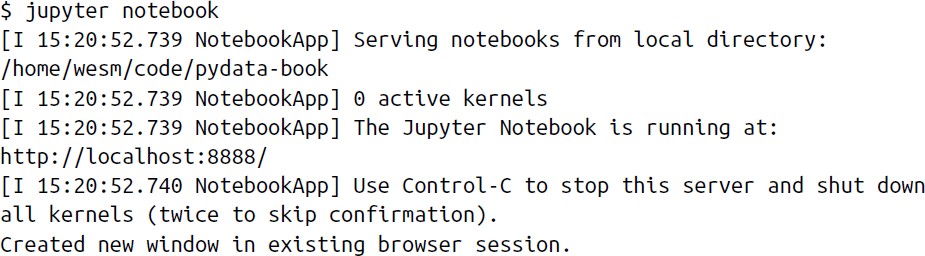
©copyright CIEL 2015 **22**

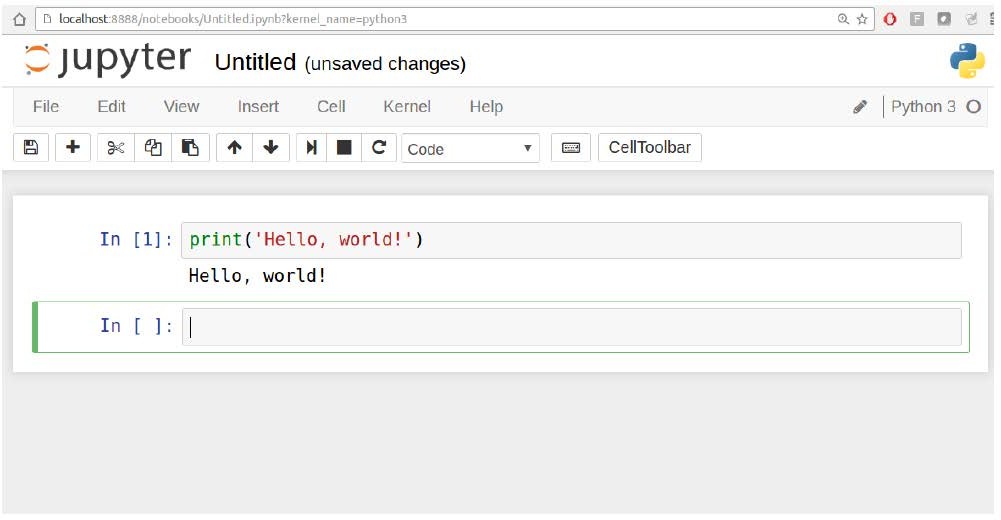
# Jupyter Notebook (\*.ipynb)

* Interacting with kernels: Python’s Jupyter kernel uses Ipython (Interactive Python)
* To start up Jupyter
  + Prompt  Anaconda



navigator





©copyright CIEL 2015 **23**

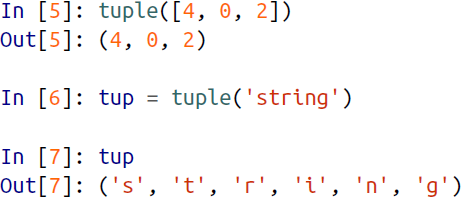
# Python Libraries for Data Analysis

|  |  |
| --- | --- |
| **Libraries** | **Description** |
| NumPy  (Numeric Python) | * A fast and efficient multidimensional array object ndarray * Functions for performing element-wise computations with arrays or mathematical operations between arrays * Tools for reading and writing array-based datasets to disk * Linear algebra operations, Fourier transform, and random number generation |
| pandas  (panel data or Python data analysis) | * High-level data structures and functions designed to make working with structured or tabular data fast, easy, and expressive * DataFrame: a tabular, column-oriented data structure with both row and column labels * Series: a one-dimensional labeled array object |
| matplotlib | * Python library for producing plots and other two-dimensional data visualizations |
| SciPy | * A collection of packages addressing a number of different standard problem domains in scientific computing * E.g., scipy.integrate, scipy.stats |
| Scikit-learn | * Premier general purpose machine learning toolkit for Python programmers |
| statsmodels | * Statistical analysis package that contains algorithms for classical (primarily frequentist) statistics and econometrics, compared with scikit-learn, e.g., Regression ANOVA |
| seaborn | * Providing several built-in plot themes or styles that use matplotlib’s configuration system internally |

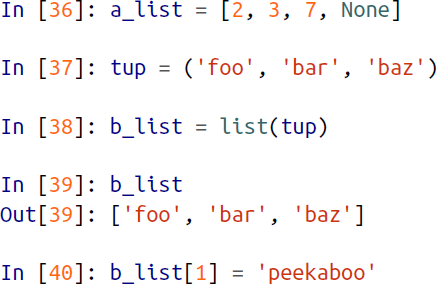
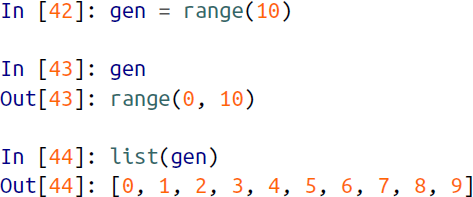
©copyright CIEL 2015 **24**

# Built-in Data Structures in Python

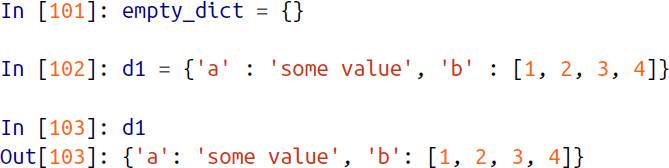
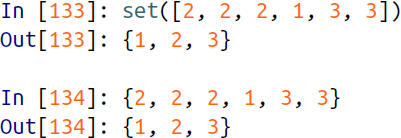
* Tuple: a fixed-length, immutable sequence of Python objects



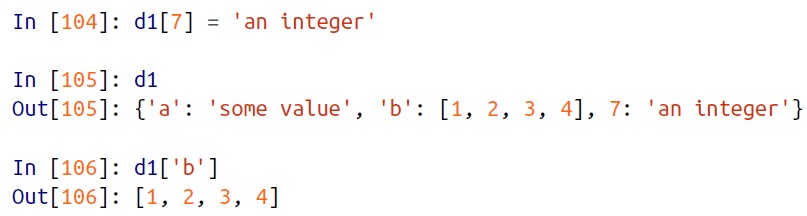
* dict: a flexibly sized collection of key-value pairs, where key and value are Python objects
* List: variable-length and their contents can be modified in-place

* set: an unordered collection of unique elements

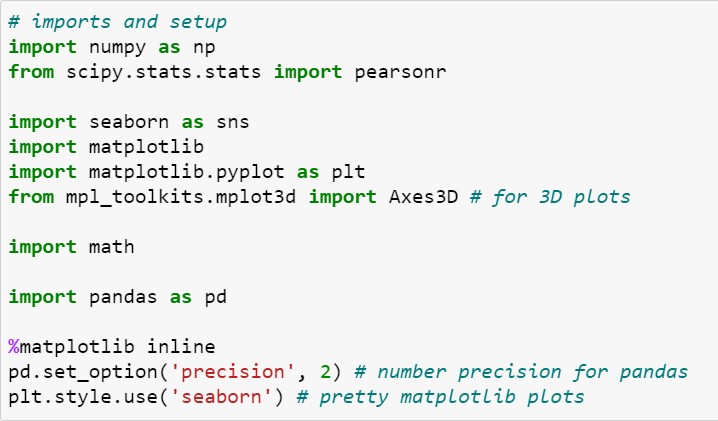
 

©copyright CIEL 2015 **25**



# 2.3.1. Basic Commands

* + Using Python Libraries
    - Import the libraries that are often used for data analysis



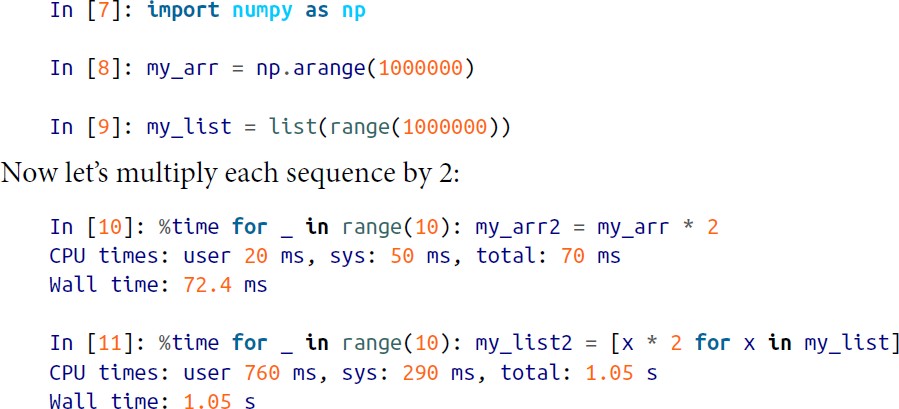
Correlation coefficient

Drawing 2D plots

©copyright CIEL 2015 **26**

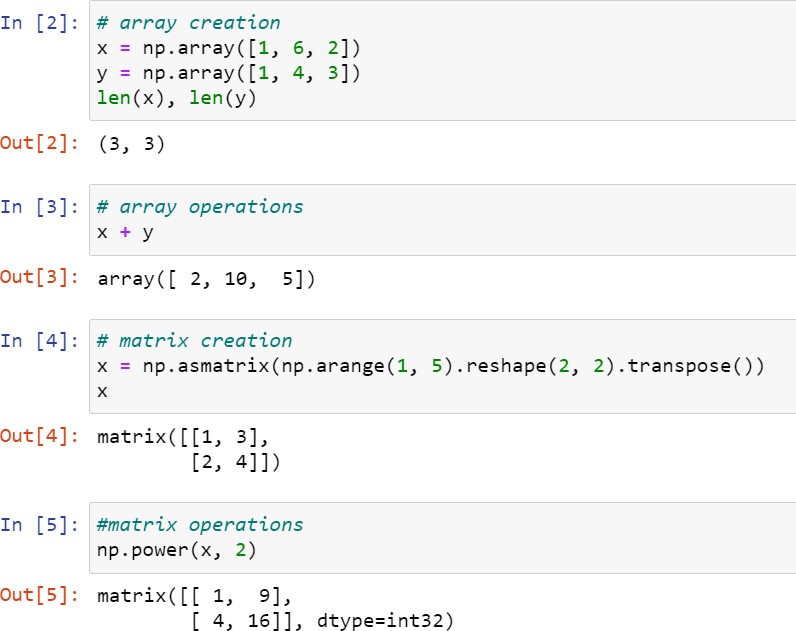
# NumPy

* + Very efficient on large arrays of data
    - NumPy internally stores data in a contiguous block of memory
    - Complex computations on entire arrays without the need for Python for loops

x 10 to 100

©copyright CIEL 2015 **27**

# 2.3.1. Basic Commands

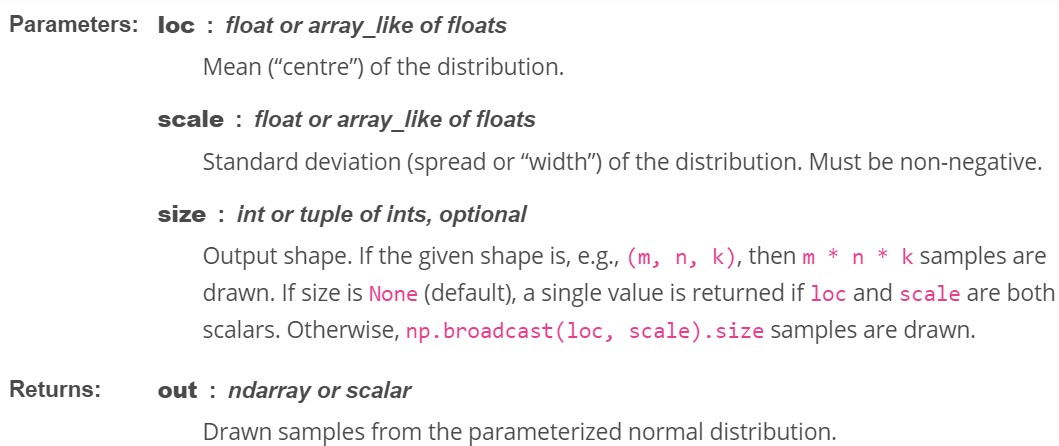


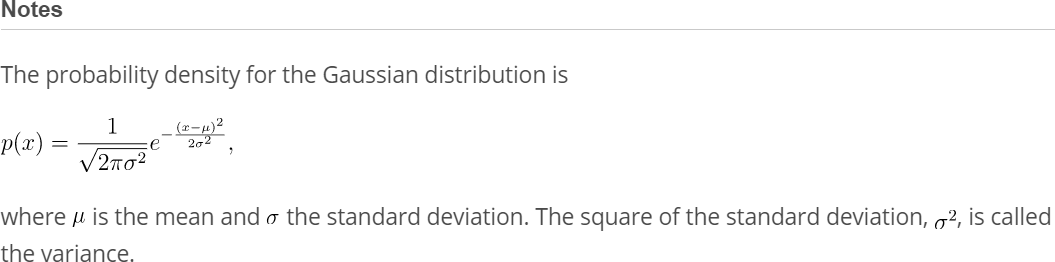
Unlike matrix, asmatrix does not make a copy if the input is already a matrix or an ndarray. Equivalent to matrix(data, copy=False).

©copyright CIEL 2015 **28**

# Numpy

* + random.normal(loc=0.0, scale=1.0, size=None)
    - Draw random samples from a normal (Gaussian) distribution.





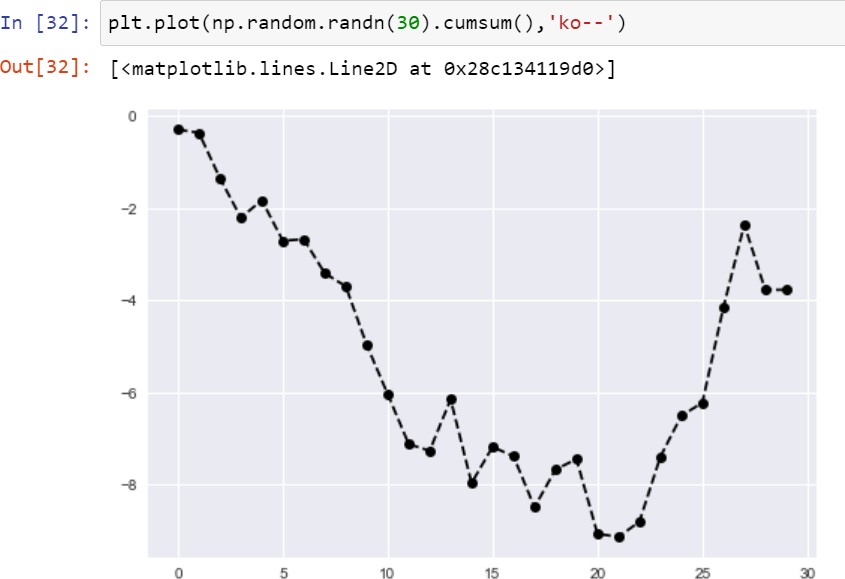
©copyright CIEL 2015 **29**

# 2.3.1. Basic Commands



©copyright CIEL 2015 **30**

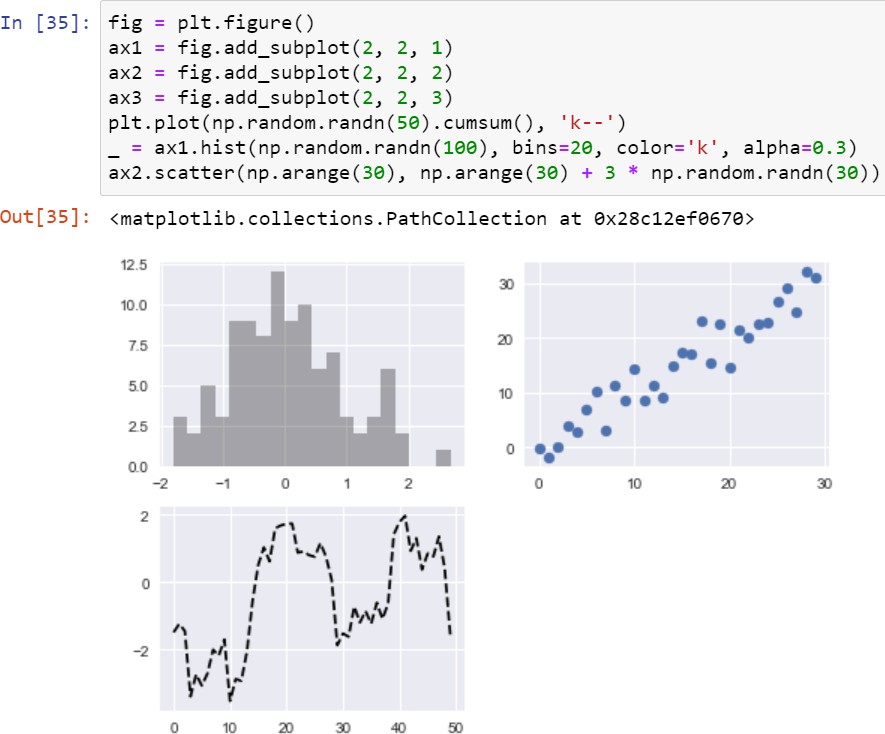
# Matplotlib

* + Basic plot

©copyright CIEL 2015 **31**

# 2.3.2 Graphics

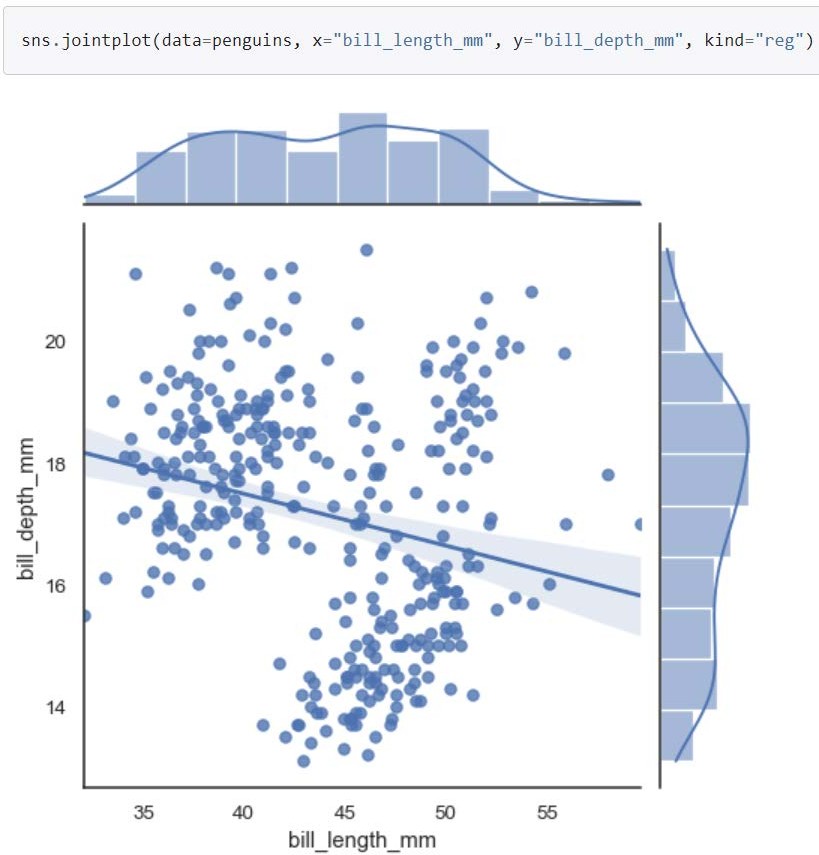
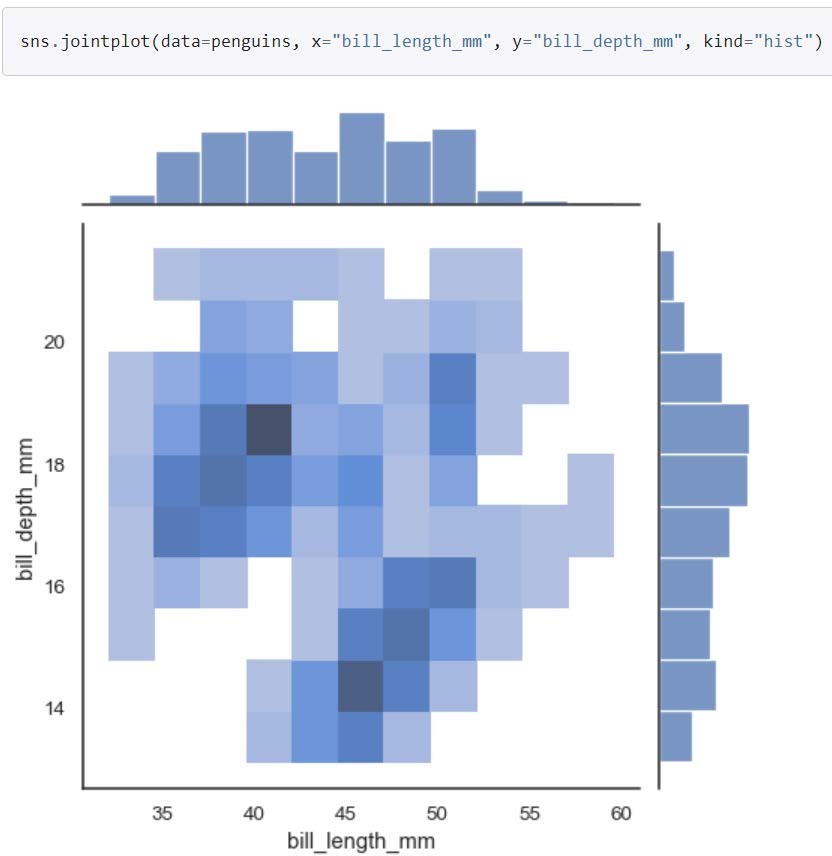
* + Figures and subplots

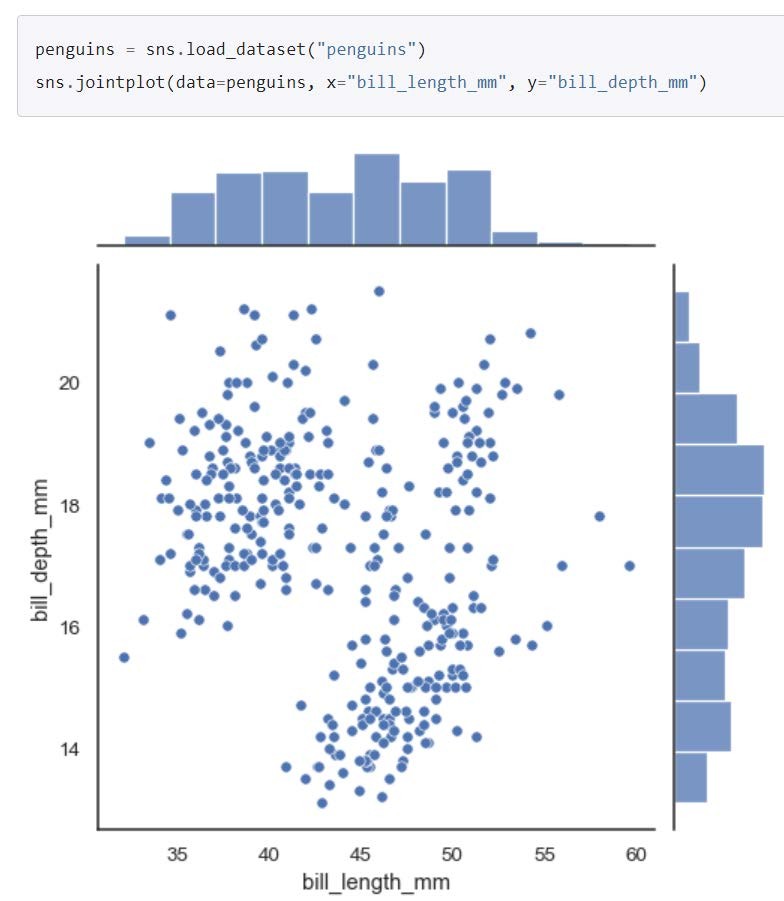


transparency

©copyright CIEL 2015 **32**

# Seaborn Plot

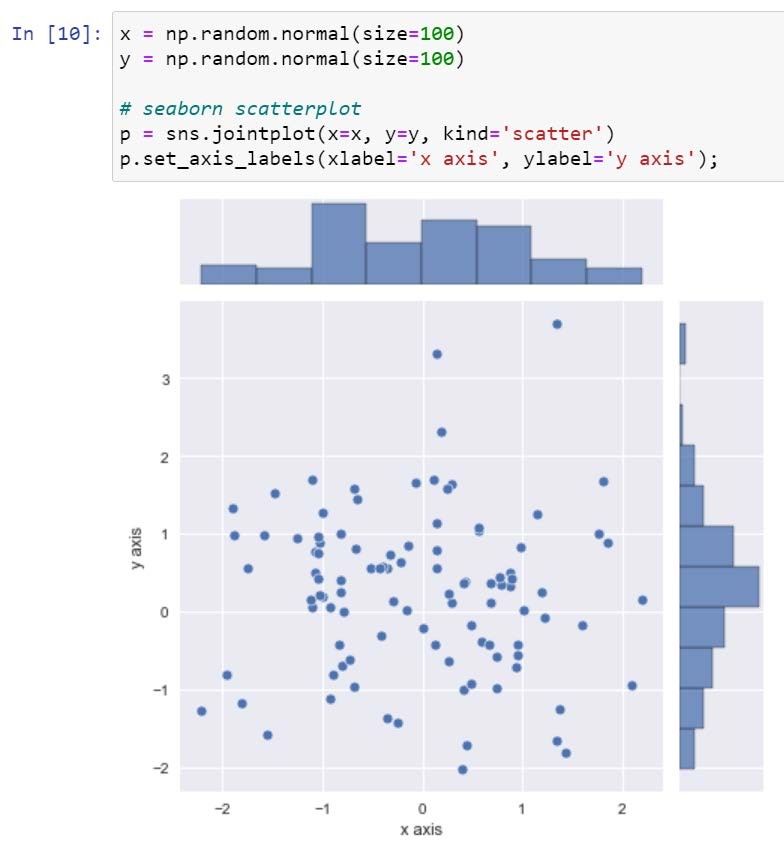
* + seaborn.jointplot
    - Draw a plot of two variables with bivariate and univariate graphs
    - kind{ “scatter” | “kde” | “hist” | “hex” | “reg” | “resid” }



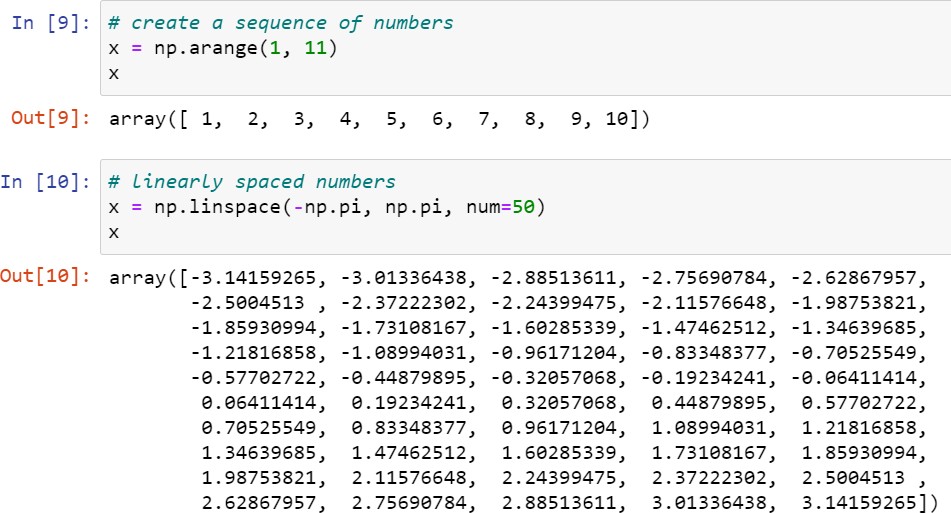
©copyright CIEL 2015 **33**

# [P] 2.3.2 Graphics

©copyright CIEL 2015 **34**

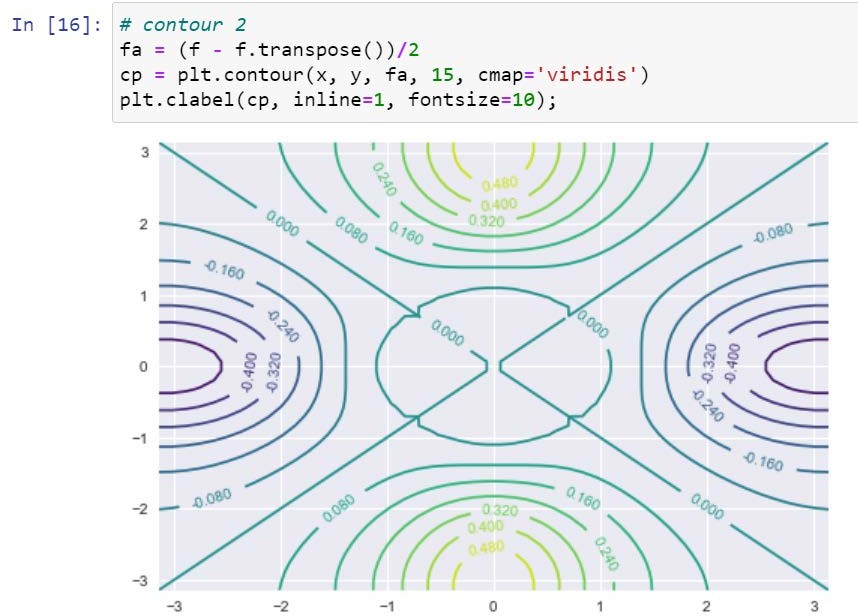
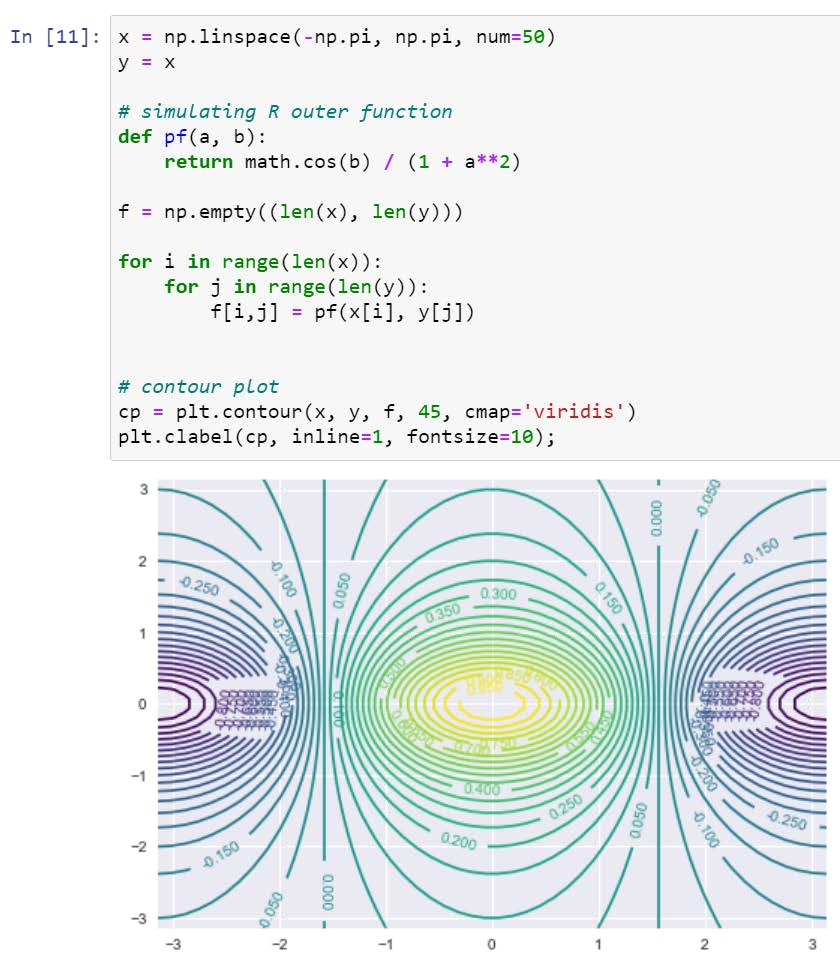


# [P] 2.3.2 Graphics



©copyright CIEL 2015 **35**

# [P] 2.3.2 Graphics

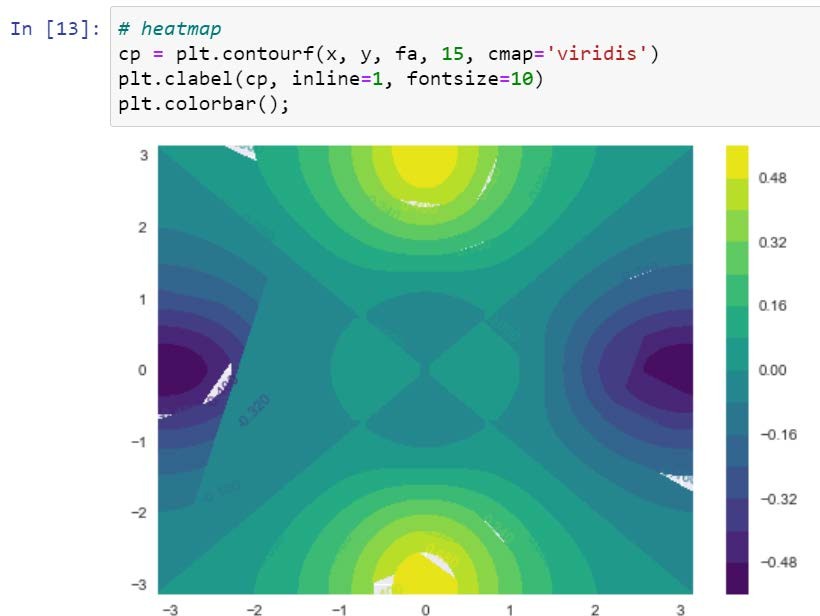


Number of lines

If true, remove segment of contour beneath label

©copyright CIEL 2015 **36**

# 2.3.2 Graphics

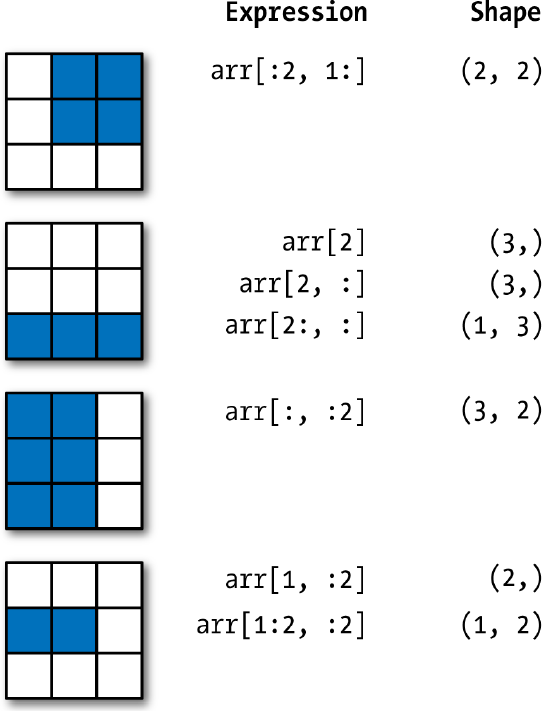
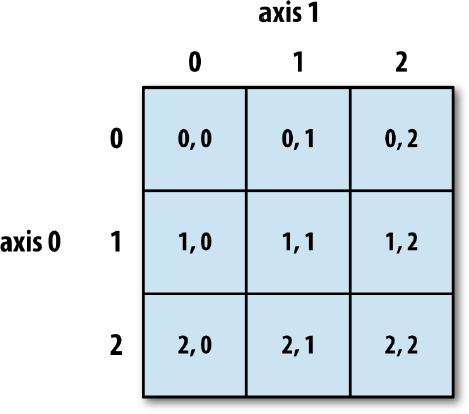
* + Heatmap
    - Colors depending on z value
    - E.g., temperature in weather forecasts
    - matplotlib.pyplot.contourf

o Filled countours

©copyright CIEL 2015 **37**

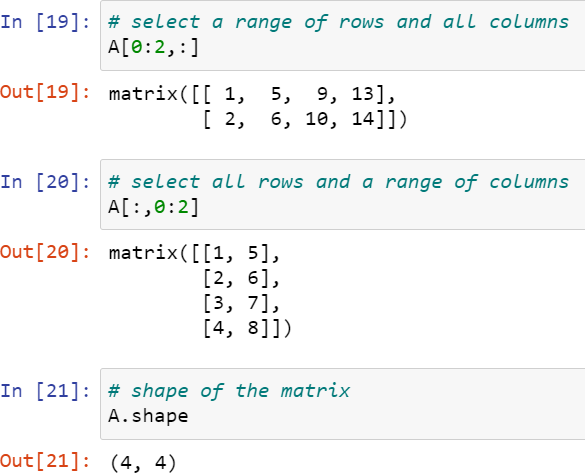
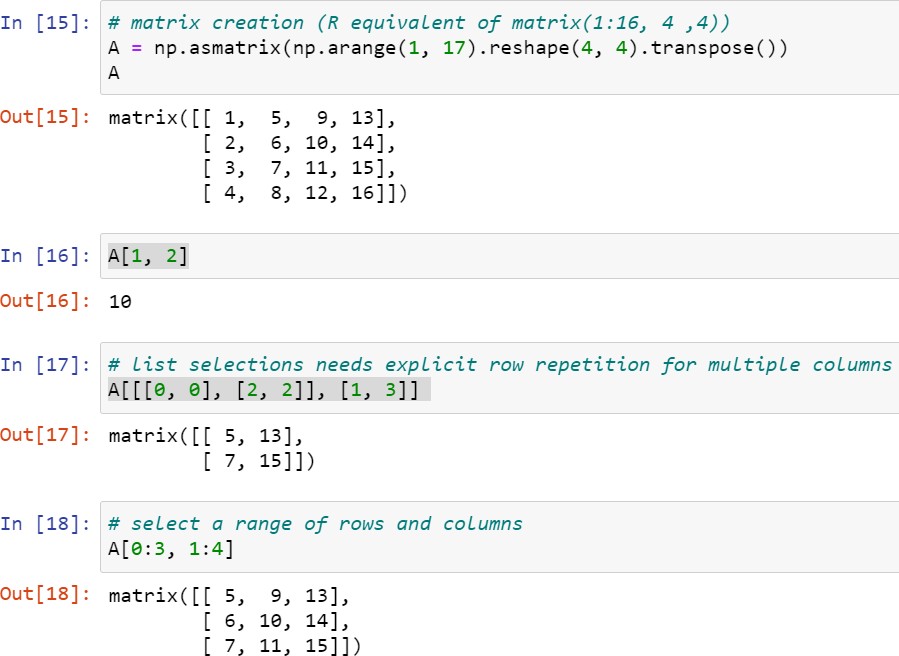
# Indexing in NumPy Array

* + Indexing elements in a NumPy array  Two-dimensional array slicing



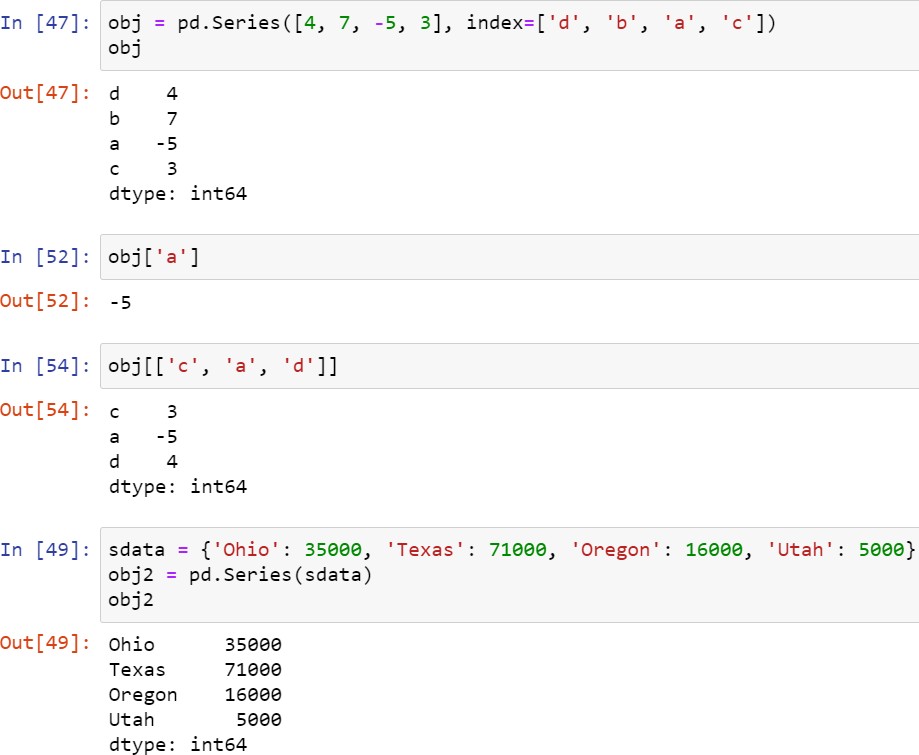
©copyright CIEL 2015 **38**

# 2.3.3 Indexing Data



©copyright CIEL 2015 **39**

# Pandas

* + Series
    - One-dimensional array-like object containing a sequence of values (of similar types to NumPy types) and an associated array of data labels, called its index

©copyright CIEL 2015 **40**

# Pandas

* + DataFrame
    - A rectangular table of data and contains an ordered collection of columns, each of which can be a different value type (numeric, string, boolean, etc.)

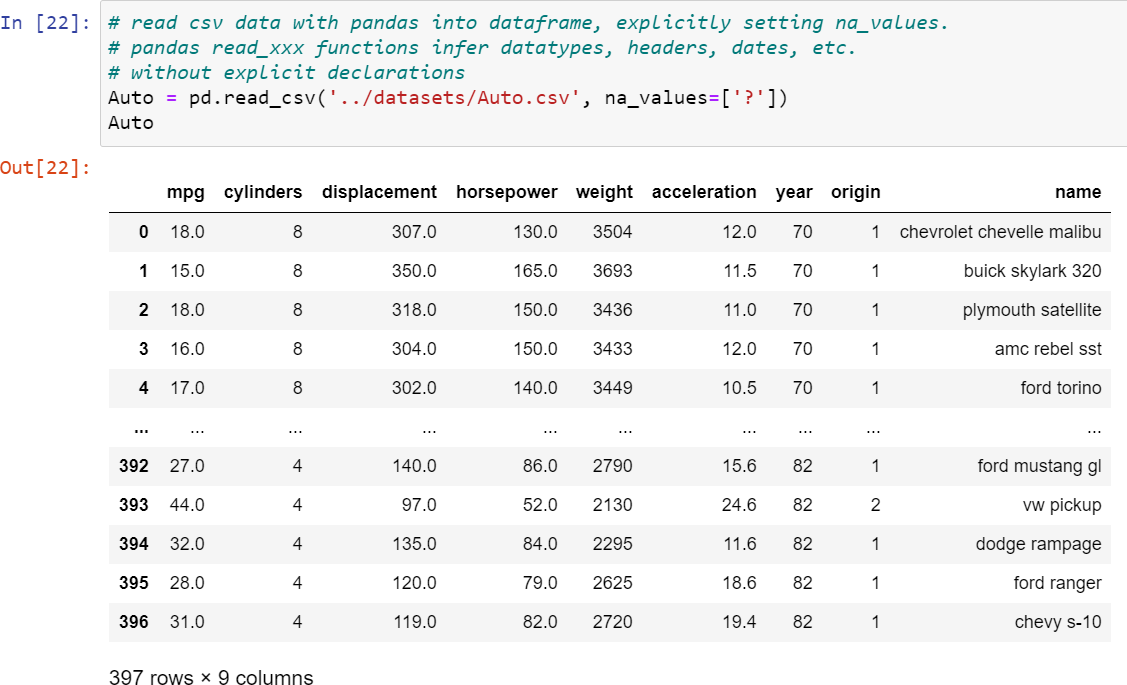
©copyright CIEL 2015 **41**

# Pandas

* + Categorical data
    - A column in a table may contain repeated instances of a smaller set of distinct values
    - The representation as integers is called the categorical or dictionary- encoded representation
    - The categorical representation can yield significant performance improvements when you are doing analytics

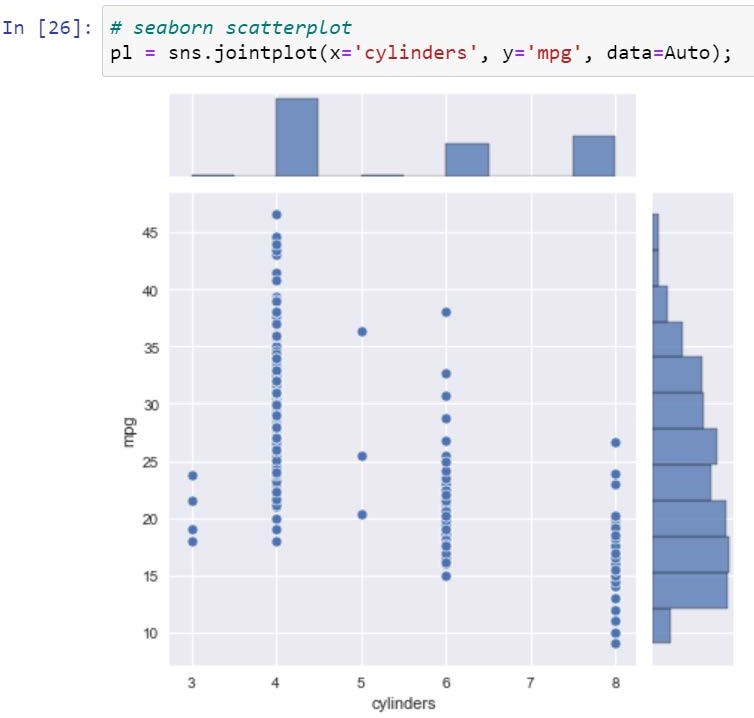
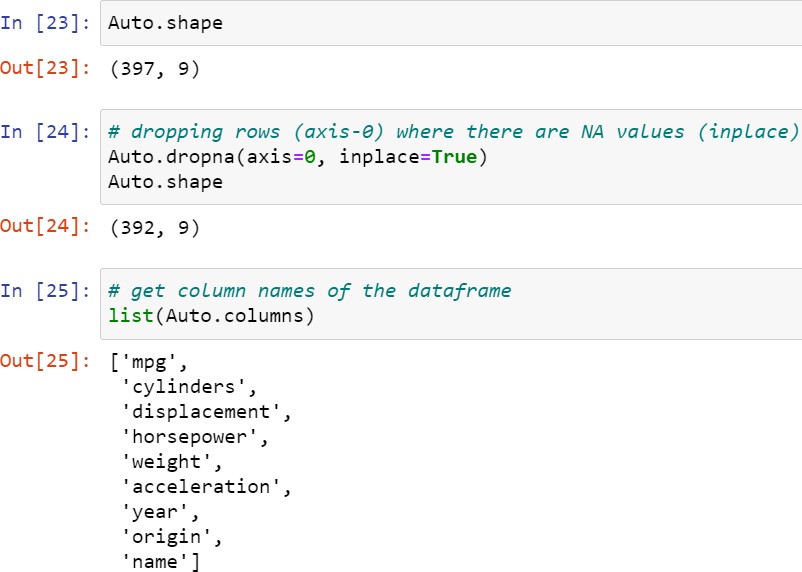
©copyright CIEL 2015 **42**

# [P] 2.3.4 Loading Data



©copyright CIEL 2015 **43**

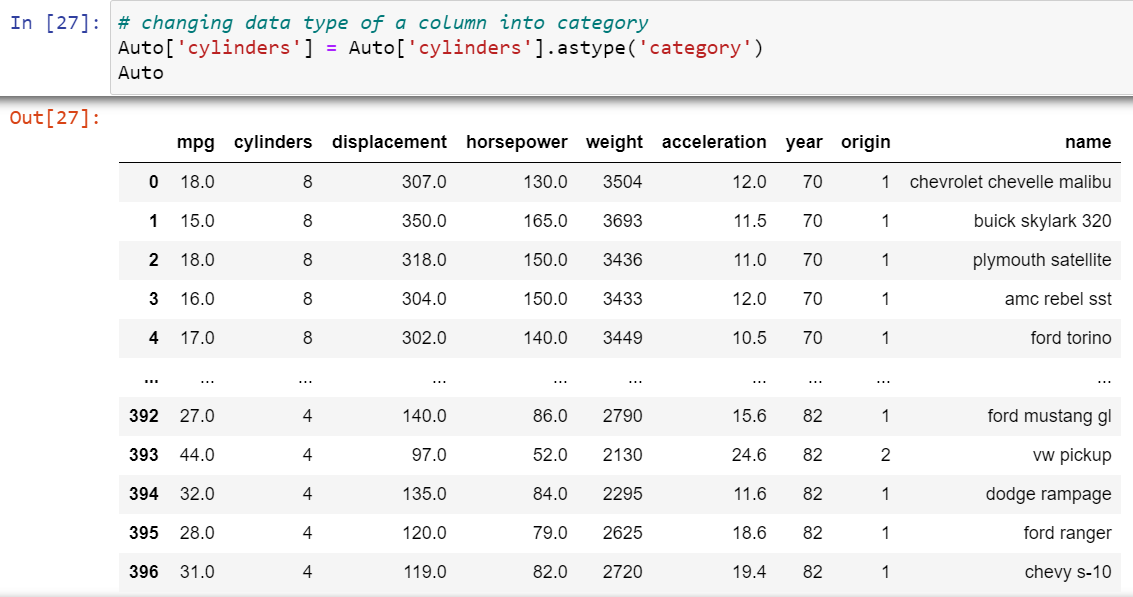
# [P] 2.3.4 Loading Data



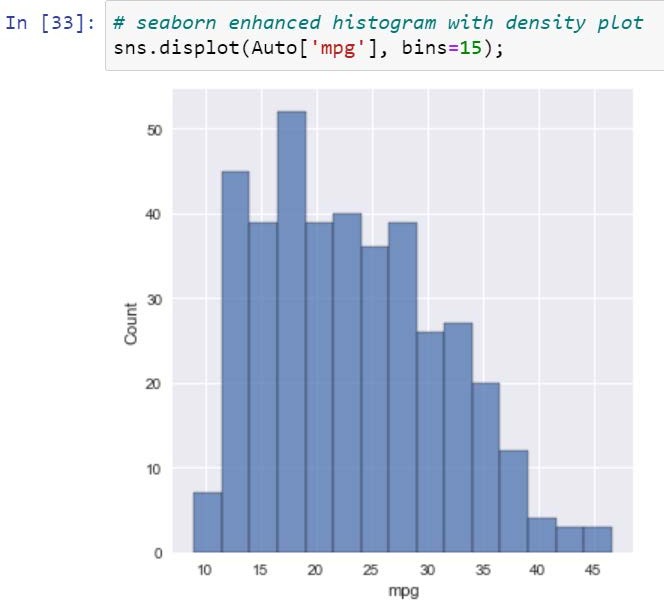
©copyright CIEL 2015 **44**

# [P] 2.3.4 Loading Data

©copyright CIEL 2015 **45**

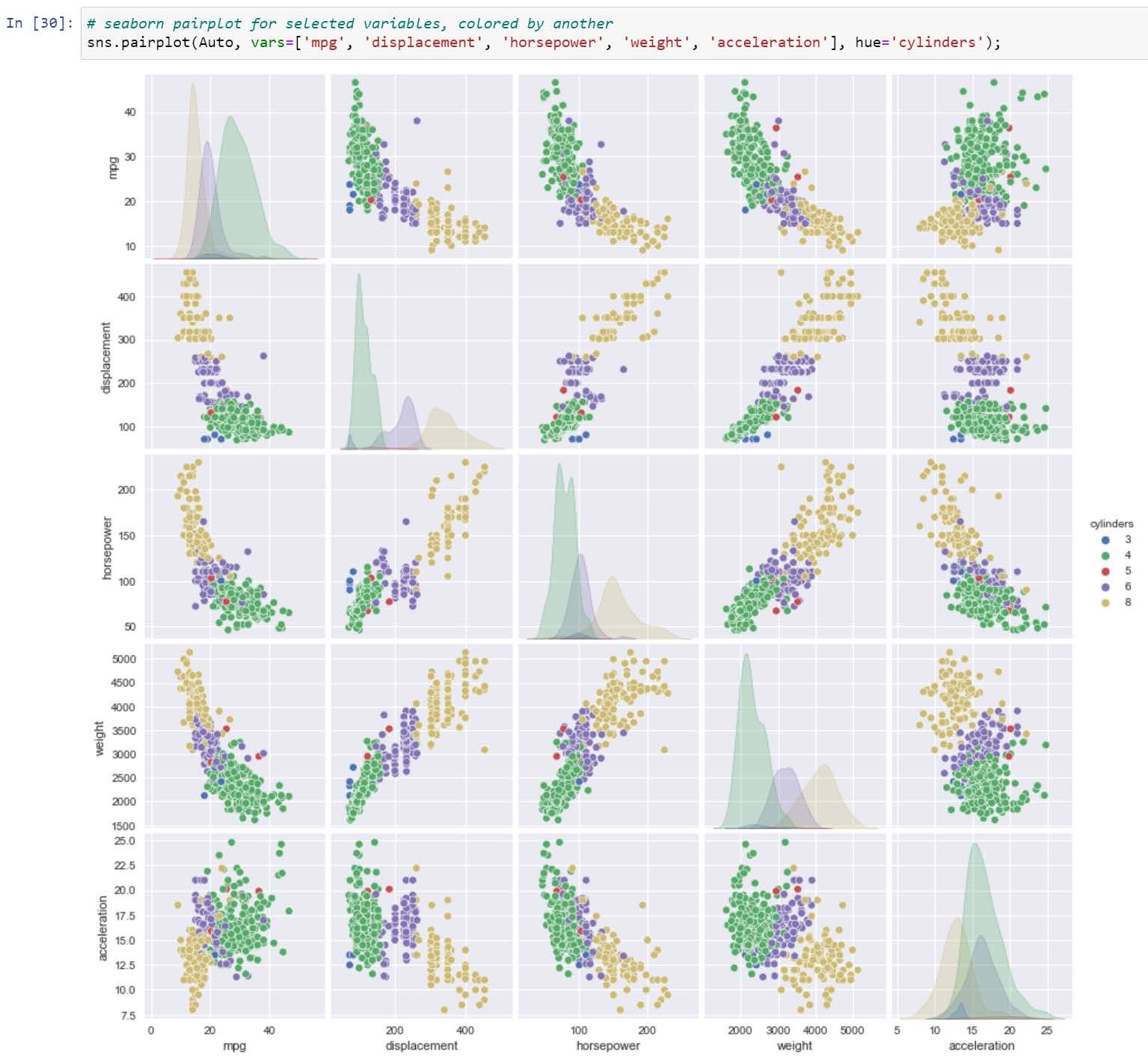


# [P] 2.3.4 Loading Data



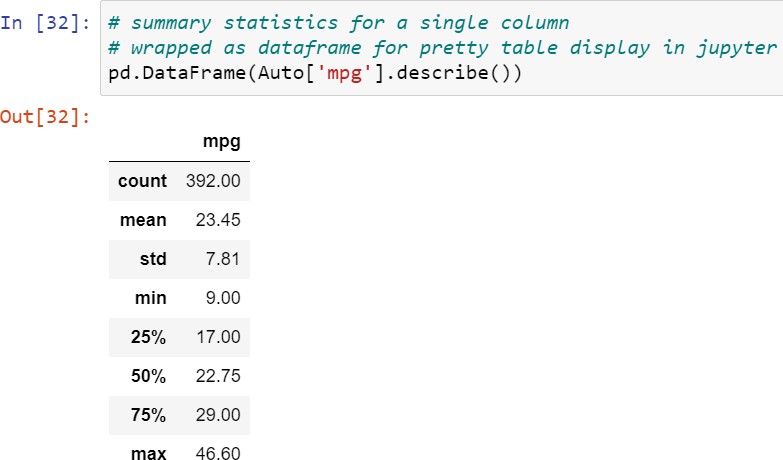
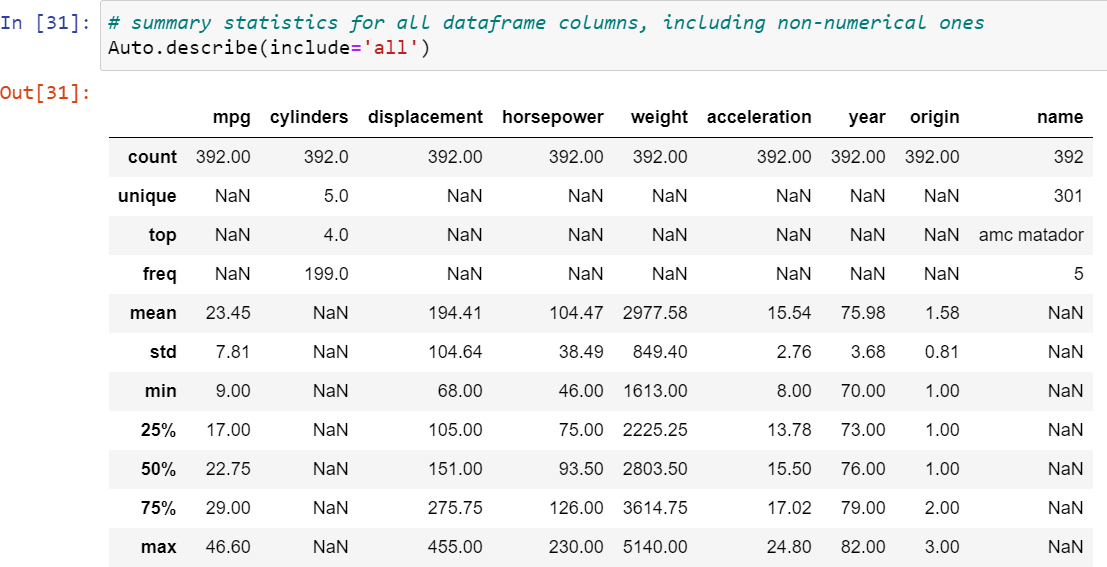
©copyright CIEL 2015 **46**

# [P] 2.3.4 Loading Data



©copyright CIEL 2015 **47**

# 2.3.4 Loading Data



©copyright CIEL 2015 **48**



**Summary & Next Class**

* + Statistical Learning
    - Assessing model accuracy
  + Python Programming
  + Summary & Next class

©copyright CIEL 2015 https://ciel.seoultech.ac.kr/

# Summary

* SL: assessing model accuracy (Ch2-2)
  + Bias-variance trade-off: as flexibility increases, variance increases while bias decreases
  + Classification problem: Bayes optimal classifier
  + KNN: tuning parameter K
  + KNN for K=1: nearest-neighbor classifier
* Python programming: using Jupyter notebook
  + Basic commands including NumPy
  + Graphics using matplotlib, seaborn
  + Indexing data: e.g., indexing elements in a NumPy array
  + Loading data using DataFrame (pandas)

©copyright CIEL 2015 **50**

# Assignments

* eClass > Assignments
  + Upload 2 or 3 files (do not compress them)
* Python practices in today’s lecture
  + Upload a single ipynb file
  + Referring to the lecture slides marked with [P]
  + File name: “StudentID” + “\_AssignmentNo w/ 2 digits” + “\_1.ipynb”, e.g., **20211234\_02\_1.ipynb**
* Textbook exercise problems for today’s lecture
  + Conceptual
    - Solving the given problems, then upload your own solution (only docx/hwp formats, not pdf/jpg/bmp etc.)
    - Only include your answers (not need to describe problems)
    - File name: “StudentID” + “\_AssignmentNo w/ 2 digits” + “\_2.ipynb”, e.g., **20211234\_02\_2.docs**
  + Applied
    - Implement your Python code for the given problems, then upload another single ipynb file
    - File name: “StudentID” + “\_AssignmentNo w/ 2 digits” + “\_1.ipynb”, e.g., **20211234\_02\_3.ipynb**
* If not complying with the above policies, some penalty on assignment scores may be given.

©copyright CIEL 2015 **51**

# Course Schedule (Tentative)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Week** | **Topics** | **Note** | **Date (W)** | **Date (M)** |
| 1 | Orientation, Statistical Learning (Ch2) | Online | 03/03 | 03/08 |
| 2 | Statistical Learning (Ch2), Python Programming | Online | 03/10 | 03/15 |
| **3** | Probability & Statistics | Online | 03/17 | 03/22 |
| 4 | Probability & Statistics | Online | 03/24 | 03/29 |
| 5 | Linear Regression (Ch3) | Online | 03/31 | 04/05 |
| 6 | Linear Regression (Ch3) | Online | 04/07 | 04/12 |
| 7 | Classification (Ch4) | Online | 04/14 | 04/19 |
| 8 | **Midterm exam** | **7pm or Class hours (W1-W7)** | **04/21or26** | **04/21or26** |
| 9 | Resampling Methods (Ch5) | Online | 04/28 | 05/03 |
| 10 | Linear Model Selection and Regularization (Ch6) | Online | 05/05 | 05/10 |
| 11 | Moving Beyond Linearity (Ch7) | Online | 05/12 | 05/17 |
| 12 | Tree-Based Methods (Ch8) | Online | 05/19 | 05/24 |
| 13 | Support Vector Machines (Ch9) | Online | 05/26 | 05/31 |
| 14 | Unsupervised Learning (Ch10) | Online | 06/02 | 06/07 |
| 15 | **Final exam** | **7pm or Class hours (W9-W14)** | **06/09or14** | **06/09or14** |

©copyright CIEL 2015 **52**