

Ders01 - Introduction:

➤ Big Data

- 1 Bit = Binary Digit
- 8 Bits = 1 Byte
- 1000 Bytes = 1 Kilobyte
- 1000 Kilobytes = 1 Megabyte
- 1000 Megabytes = 1 Gigabyte
- 1000 Gigabytes = 1 Terabyte
- 1000 Terabytes = 1 Petabyte
- 1000 Petabytes = 1 Exabyte
- 1000 Exabytes = 1 Zettabyte
- 1000 Zettabytes = 1 Yottabyte
- 1000 Yottabytes = 1 Brontobyte
- 1000 Brontobytes = 1 Geopbyte

✓ Discover Relation:

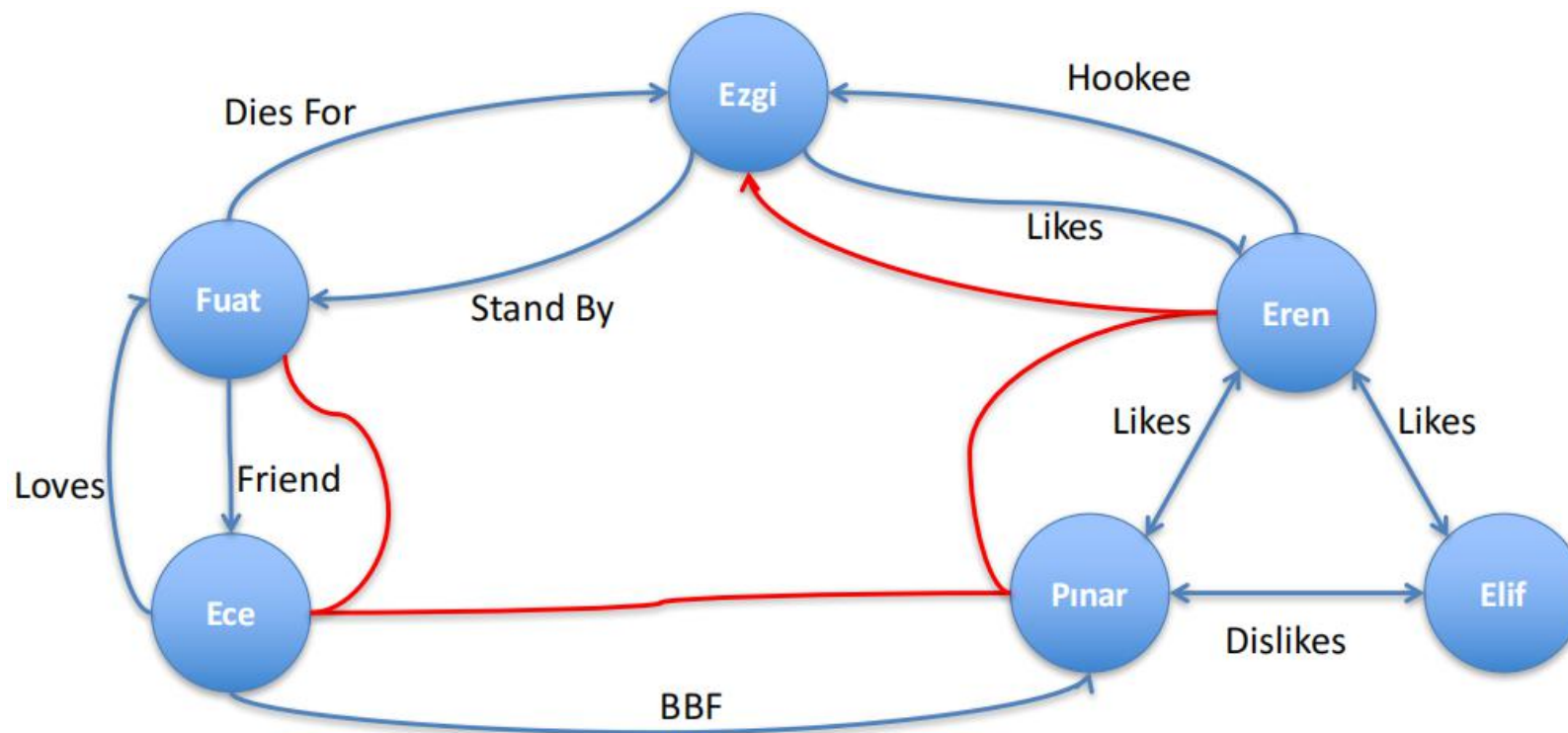
Your data (TB/PB)

Your friend's data (TB/PB)

The World (EB)

How to Sell More Video Games?

- Retail Industry
 - Busiest shopping time of the year and one video game is flying off the shelves.
 - In the past, stores that can react to this demand had the advantage.
 - Today, stores that can predict the trends and prepare for future demands had the advantage.
 - Combine data from web browsing patterns, movie releases, social media, enterprise data etc.



Big Data in Health Care

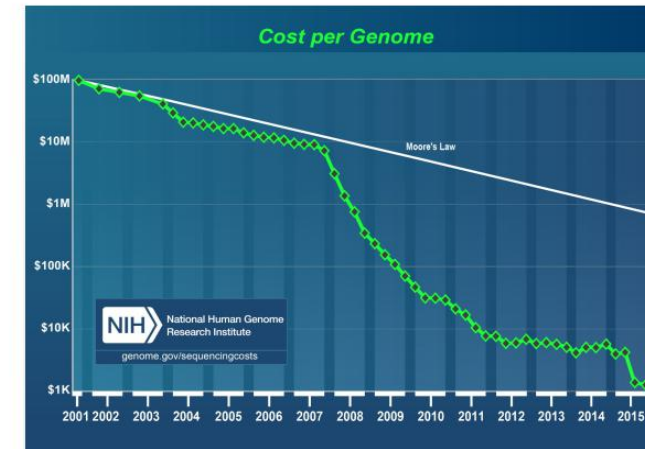
- Faster and cheaper technology and data storage
- Widespread sensing devices
- An increase in “born” digital data
- Greater availability of data via repositories
- Data sharing mandates

Why is Big Data Processing Difficult?

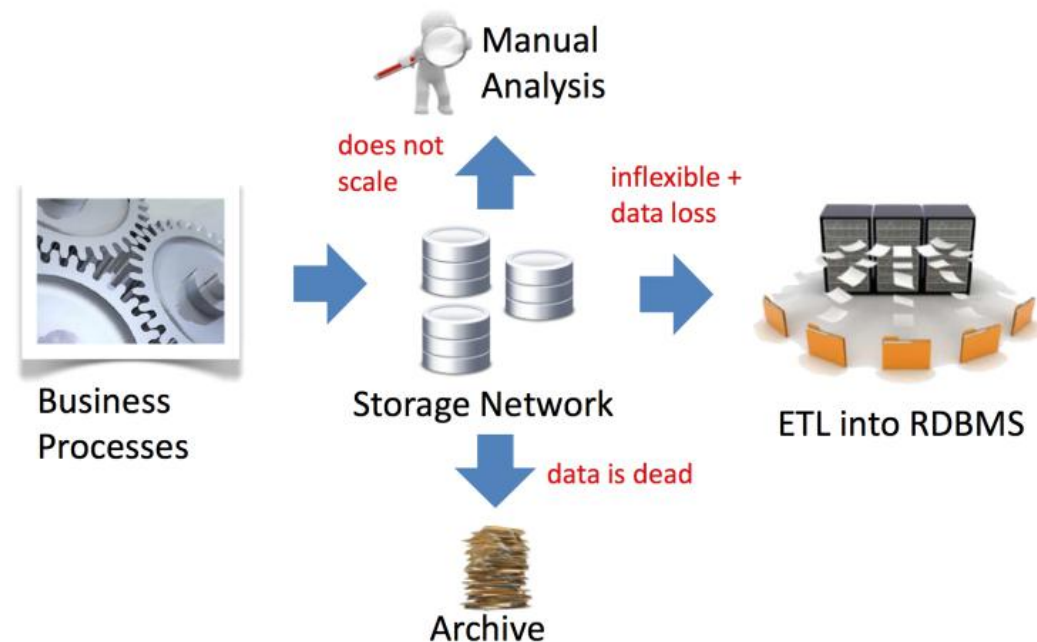
- You need more data than your data warehouse.
 - you need more data than you have
 - logs, Twitter feeds, blogs, customer surveys, ...
- You need to ask the right questions.
 - data alone is silent
- You need technology and organization that help you concentrate on asking the right questions.

Faster and Cheaper Technology and Data Storage

The cost to sequence a whole human genome sequence has fallen from +\$100 million to less than \$1,000 over the past 15 years.



Limitations of the State of the Art



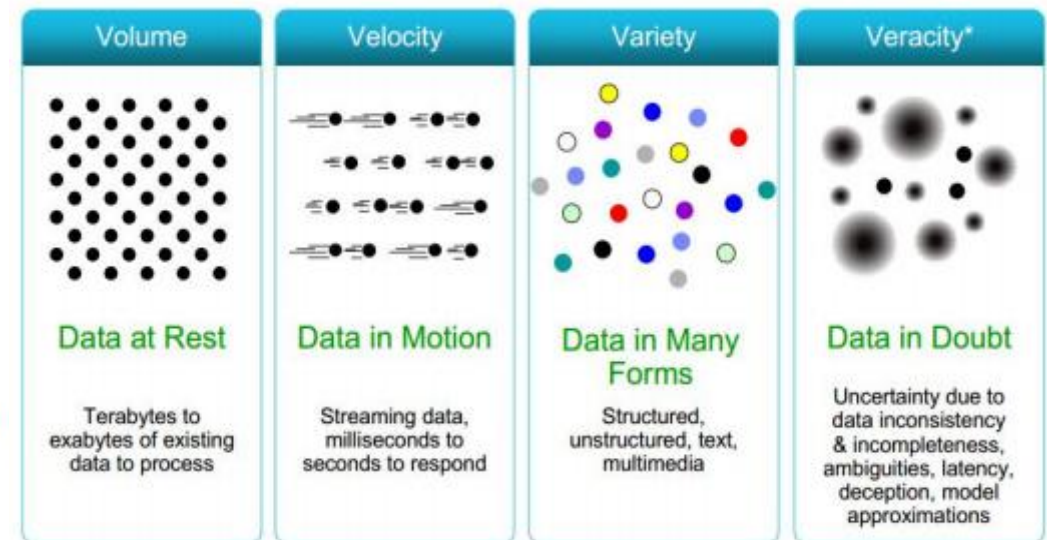
Is bigger = smarter?

- Yes!
 - tolerate errors
 - discover the long tail and corner cases
 - machine learning works much better
- But!
 - more data, more error (e.g., semantic heterogeneity)
 - with enough data you can prove anything
 - still need humans to ask right questions

Four V's of Big Data

- Volume
 - Big data is always large in volume. It actually doesn't have to be a certain number of *petabytes* to qualify.
- Velocity (or speed)
 - Refers to how fast the data is coming in, but also to how fast you need to be able to analyze and utilize it.
- Variety (diversity)
 - Points to the number of sources or incoming vectors leading to your databases.
 - That might be embedded sensor data, phone conversations, documents, video uploads or feeds, social media, and much more.
- Veracity (data in doubt)
 - If you can't trust the data itself, the source of the data, or the processes you are using to identify which data points are important, you have a veracity problem.

Four V's of Big Data



More V's

- Variability
 - Inconsistency in data, inconsistent speed at which big data is loaded into your database
- Validity
 - Similar to veracity, validity refers to how accurate and correct the data is for its intended use
- Vulnerability
 - Big data brings new security question
- Volatility
 - You cannot store data indefinitely anymore
- Visualization
 - Challenge to visualize, need different ways to represent data
- Value
 - The other Vs are meaningless if you don't derive business value from the data

Types of Data We Have

- Relational Data (Tables/Transaction/Legacy Data)
- Text Data (Web)
- Semi-structured Data (XML)
- Graph Data
- Social Network, Semantic Web (RDF), ...
- Streaming Data
- ...

Combined Effort

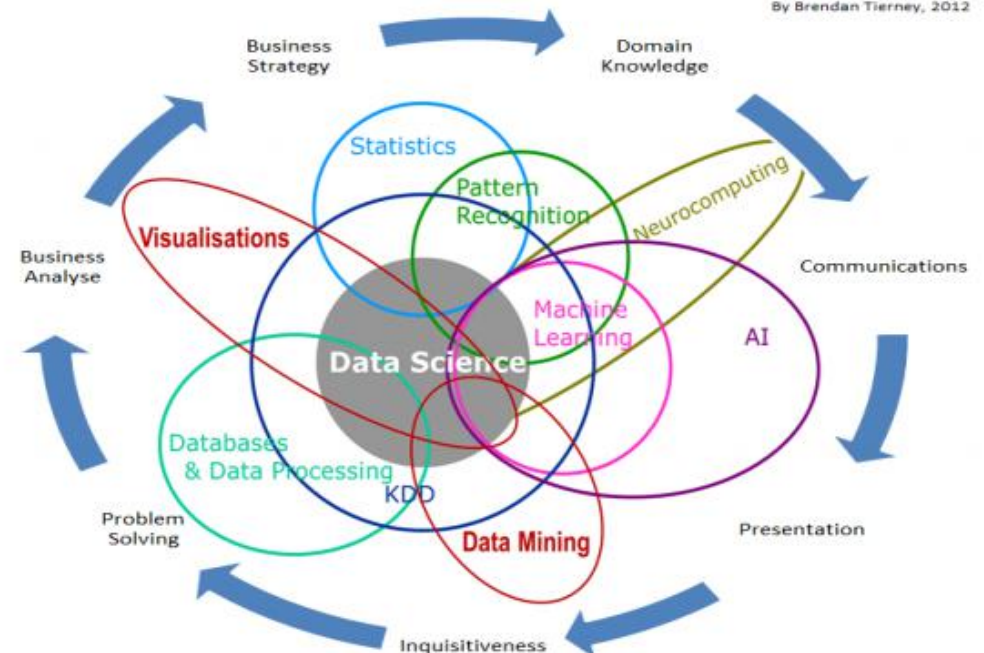
- Theories and techniques from many fields and disciplines are used to investigate and analyze a large amount of data to help decision makers in many industries such as science, engineering, economics, politics, finance, and education
 - Computer Science
 - Pattern recognition, visualization, data warehousing, High performance computing, Databases, AI
 - Mathematics
 - Mathematical Modeling
 - Statistics
 - Statistical and Stochastic modeling, Probability.

What is Data Science?

- An area that manages, manipulates, extracts, and interprets knowledge from tremendous amount of data
- Data science (DS) is a multidisciplinary field of study with goal to address the challenges in big data
- Data science principles apply to all data – big and small

Data Science Is Multidisciplinary

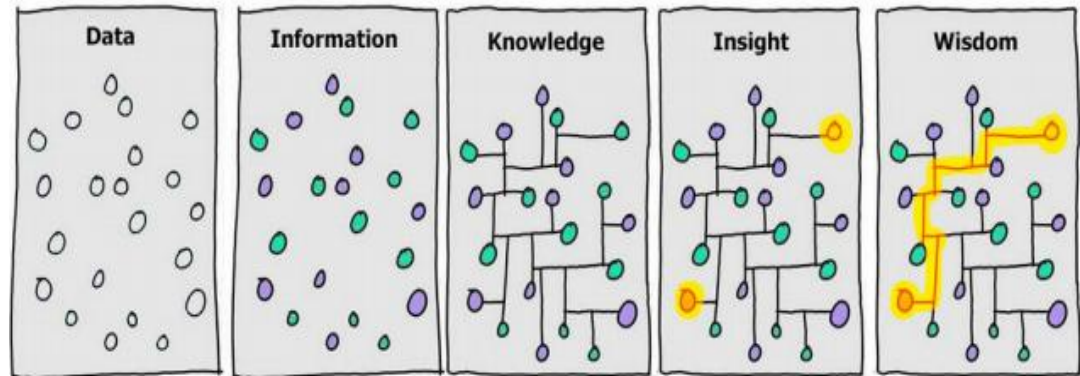
By Brendan Tierney, 2012



Who is Data Scientist?

- “A person employed to analyse and interpret complex digital data, such as the usage statistics of a website, especially in order to assist a business in its decision-making.”
- “A *data scientist* is a rare hybrid, a computer *scientist* with the programming abilities to build software to scrape, combine, and manage *data* from a variety of sources and a statistician who knows how to derive insights from the information within.”

Information Continuum



Cartoon by [David Somerville](#), based on a two pane version by [Hugh McLeod](#)

Ders02 - Data Science Methodology:

What is a methodology anyway?

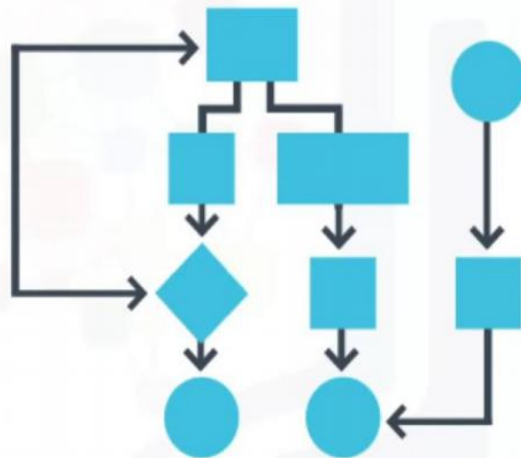
A methodology is a defined way of....

meth·od·ol·o·gy

noun

noun: **methodology**; plural noun: **methodologies**

- 1 a system of methods used in a particular area of study or activity. “a methodology for investigating the concept of focal points”



Methodology by John Rollins based on CRISP-DM



John Rollins

Data Scientist, IBM Analytics, IBM

John B. Rollins, Ph.D., P.E., is a Data Scientist, IBM Analytics, IBM. Prior to joining IBM Netezza, he was an engineering consultant, professor and researcher. He has authored many patents, papers and books. He holds doctoral degrees in economics and petroleum engineering and is a registered professional engineer in Texas.

In a nutshell...

The **Data Science Methodology** aims to answer the following 10 questions in this prescribed sequence:

From problem to approach:

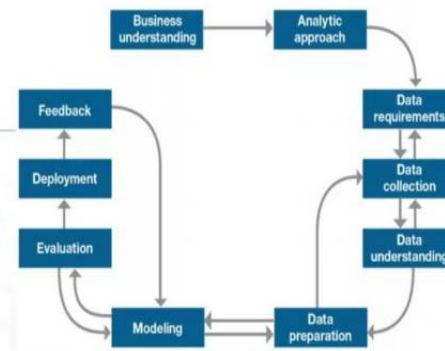
1. What is the problem that you are trying to solve?
2. How can you use data to answer the question?

Working with the data:

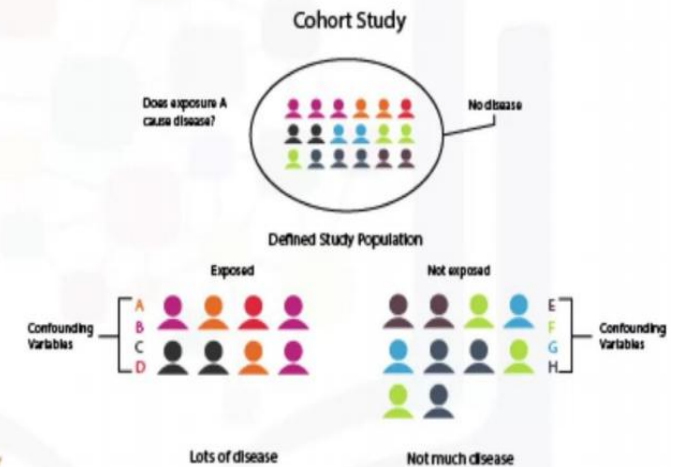
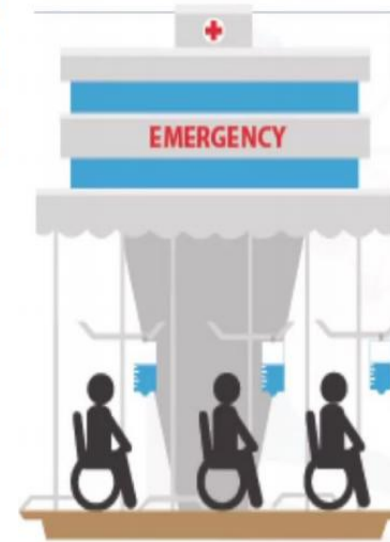
3. What data do you need to answer the question?
4. Where is the data coming from (identify all sources) and how will you get it?
5. Is the data that you collected representative of the problem to be solved?
6. What additional work is required to manipulate and work with the data?

Deriving the answer:

7. In what way can the data be visualized to get to the answer that is required?
8. Does the model used really answer the initial question or does it need to be adjusted?
9. Can you put the model into practice?
10. Can you get constructive feedback into answering the question?

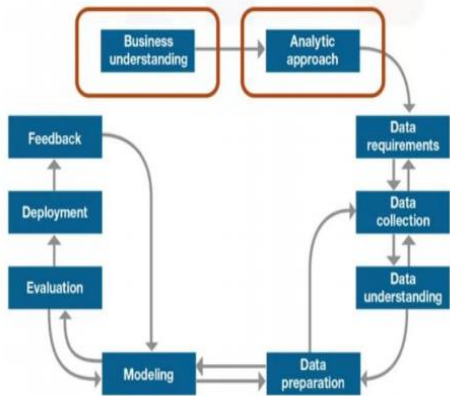


About the Case Study



Business Understanding:

From Understanding to Approach



Business understanding

- What is the problem that you are trying to solve?



Analytic approach

- How can you use data to answer the question?

Establishing a clearly defined question starts with understanding the goal of the person asking the question.

☐ False.

☒ True.

Correct

Correct.



Case Study – What are the goals & objectives?



Define the **GOALS**

- To provide quality care without increasing cost:

Define the **OBJECTIVES**

- To review the process to identify inefficiencies

Case Study – Identifying the business requirements



1. Predict CHF readmission outcome (Y or N) for each patient
2. Predict the readmission risk for each patient
3. Understand explicitly what combination of events led to the predicted outcome for each patient
4. Easy to understand and apply to new patients to predict their readmission risk

What are the types of questions?



Will machine learning be utilized?

If the question is to determine probabilities of an action

- Use a Predictive model

If the question is to show relationships

- Use a descriptive model

If the question requires a yes/no answer

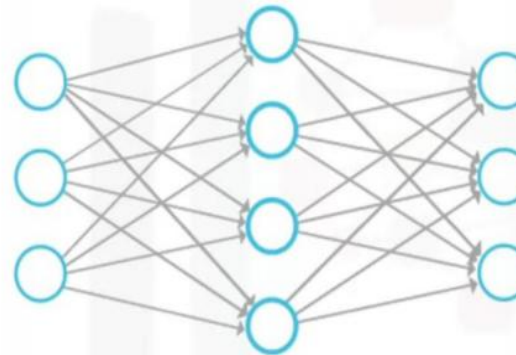
- Use a classification model

Analytic approach

- How can you use data to answer the question?



- The correct approach depends on business requirements for the model



Machine Learning

- Learning without being explicitly programmed
- Identifies relationships and trends in data that might otherwise not be accessible or identified
- Uses clustering association approaches



Case Study – Decision tree classification selected!

Although the analytics approach is the second stage of the data science methodology, it is still independent of the business understanding stage.

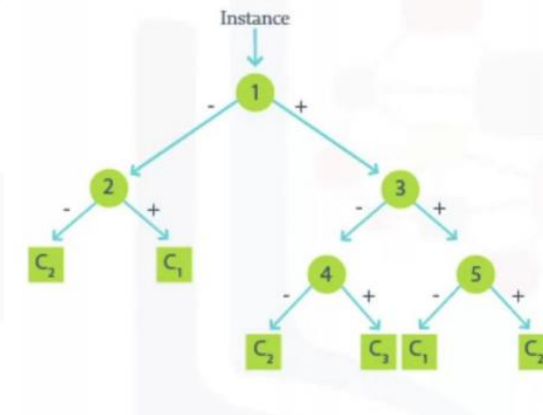
☒ False.



Correct

Correct.

☐ True.



Predictive model

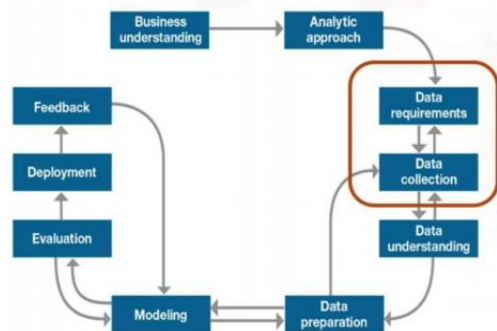
- To predict an outcome

Decision tree classification

- Categorical outcome
- Explicit “decision path” showing conditions leading to high risk
- Likelihood of classified outcome
- Easy to understand and apply

➤ Data Requirements:

From Requirements to Collection



Data Requirements

- What are data requirements?

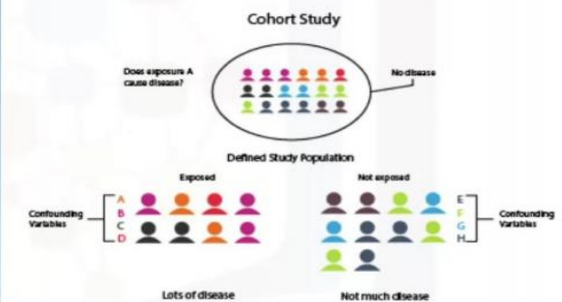


Data Collection

- What occurs during data collection?

Case Study – Selecting the cohort

- Define and select cohort
 - In-patient within health insurance provider’s service area
 - Primary diagnosis of CHF in one year
 - Continuous enrollment for at least 6 months prior to primary CHF admission
 - Disqualifying conditions



Congestive heart failure patients with other significant medical conditions were included in the study in order to increase the sample size of the patients included in the study.

☐ True

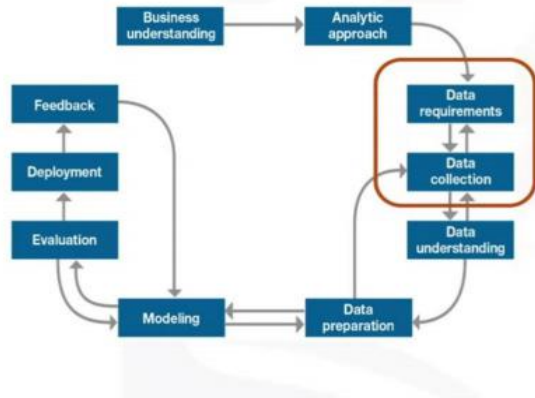


☒ False

Correct

Correct. Congestive heart failure patients with other significant medical conditions were actually excluded from the study.

From Requirements to Collection



Data Requirements

- What are data requirements?



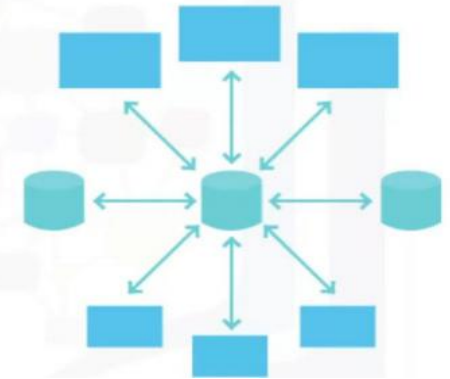
Data Collection

- What occurs during data collection?

Case Study – Gathering available data

- Available data sources

- Corporate data warehouse (single source of medical & claims, eligibility, provider and member information)
- In-patient record system
- Claim payment system
- Disease management program information



When collecting data, it is alright to defer decisions about unavailable data, and attempt to acquire it at a later stage.

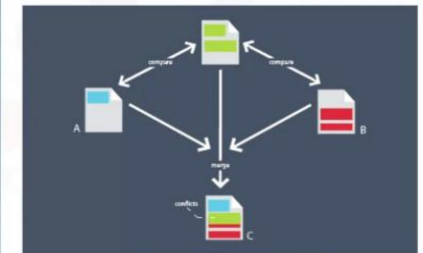
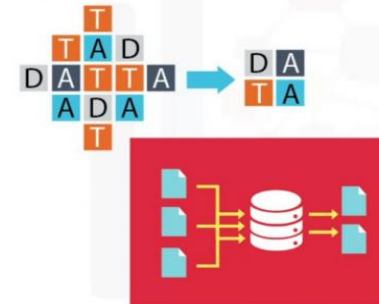
☒ True.

Correct
Correct.

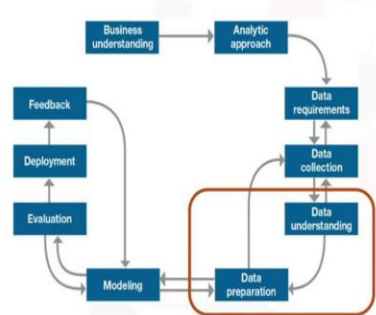
☐ False.

Case Study – Merging data

- Eliminate redundant data

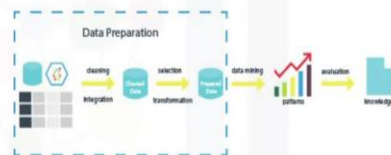


From Understanding to Preparation



Data understanding

- What does it mean to “prepare” or “clean” data?



Data preparation

- What are ways in which data is prepared?

Case Study – Understanding the data

- Descriptive statistics

- Univariate statistics
- Pairwise correlations
- Histogram

$$f(a) + \sum_{k=1}^n \frac{1}{k!} \left. \frac{d^k}{dt^k} \right|_{t=0} f(u(t)) + \int_0^1 \frac{(1-t)^n}{n!} \frac{d^{n+1}}{dt^{n+1}} f(u(t)) dt.$$

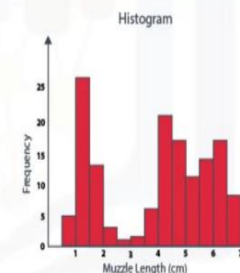
$F_{X,Y}(x, y)$ satisfies

$$F_{X,Y}(x, y) = F_X(x)F_Y(y),$$

or equivalently, their joint density $f_{X,Y}(x, y)$ satisfies

$$f_{X,Y}(x, y) = f_X(x)f_Y(y).$$

Histograms are a good way to understand how values or a variable are distributed, and what sorts of data preparation may be needed to make the variable more useful in a model.



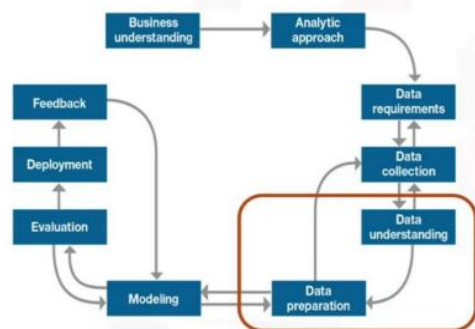
The Data Understanding stage encompasses sorting the data.

☐ True.

☒ False.

Correct
Correct.

From Understanding to Preparation



Data understanding

- What does it mean to "prepare" or "clean" data?



Data preparation

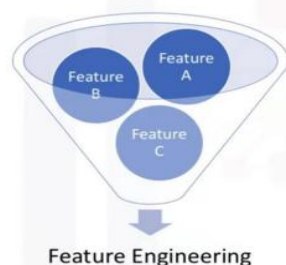
- What are ways in which data is prepared?

Examples of data cleansing

	A	B	C	D	E
1	Name	Date	Age	Location	Country
2	John Doe	2012 02 20	32	ON	CAN
3	May Lag	2013 02 33	2	ON	CA
4	Henry Oon	30-Sep-12	35	Ontario	CANADA
5	Kelly, Tom	2015 02 20	65	ON	CA
6	John Kell	2016 02 20	AB		CA
7	Henry Oon	30-Sep-12	35	Ontario	CANADA
8					

☐ Invalid Values
☐ Missing Data
☐ Remove Duplicates
☐ Formatting

Using domain knowledge



Feature engineering is the process of using domain knowledge of the data to create features that make the machine learning algorithms work.

Feature engineering is critical when machine learning tools are being applied to analyze the data.

The Data Preparation stage is the least time-consuming phase of a data science project, typically taking between 5 to 10 percent of the overall project time.

☐ True.

☒ False.

Correct

Correct.

Case Study – Aggregating records

Transactional records

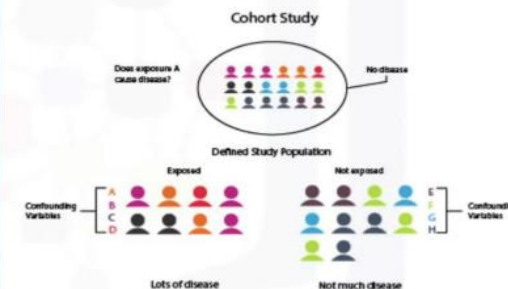
- Claims: professional provider, facility, pharmaceutical
- Inpatient & outpatient records: diagnoses, procedures, prescriptions, etc.
- Possibly thousands per patient, depending on clinical history



Case Study – Aggregating to patient level

Aggregate to patient level

- Roll up to 1 record per patient
- Create new columns representing the transaction
 - Outpatients visits/ Inpatient episodes: frequency, recency, diagnoses/length of stay, procedures, prescriptions
 - Comorbidities with CHF



Case Study – More or less data needed?

Literature review of important factors for CHF readmission

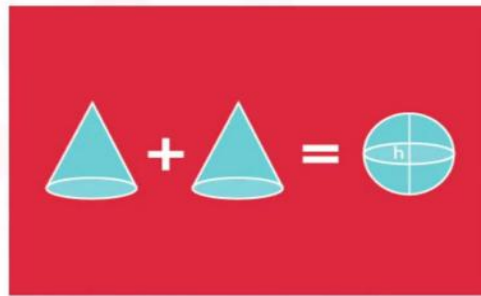
MORE LIKELY TO BE READMITTED

- Medicare / Medicaid insurance holders
- Comorbid conditions including:
 - Ischemic heart disease
 - Idiopathic Cardiomyopathy
 - Prior Cardiac surgery
 - Peripheral vascular disease
 - Diabetes mellitus
 - Anemia

LESS LIKELY TO BE READMITTED

- Patients treated at rural hospitals
- Patients discharged to skilled nursing facilities
- Patients receiving echocardiograms or cardiac catheterization

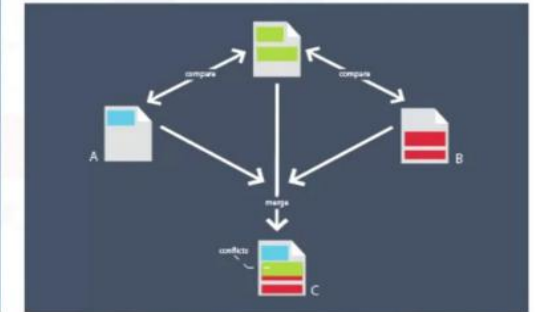
- Loop back to data collection stage and add additional data, if needed



Case Study – Completing the data set

Merge all data into one table

- One record per patient
- List of variables used in modeling
- Target: CHF readmission with 30 days (Yes/No), following discharge from CHF hospitalization



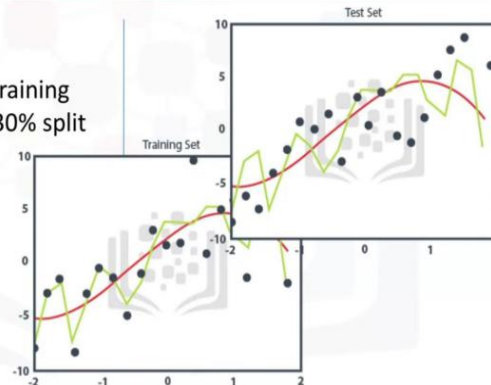
Case Study – Using training sets

Cohort: 2,343 patients

Randomly divided into training and testing sets: 70% / 30% split

Training: 1,640 patients

Testing: 703 patients



In the case study, the target variable was congestive heart failure (CHF) with 45 days following discharge from CHF hospitalization.

☐ False.

Correct
Correct.

☐ True.



A training set is used to build a predictive model.

☐ False.

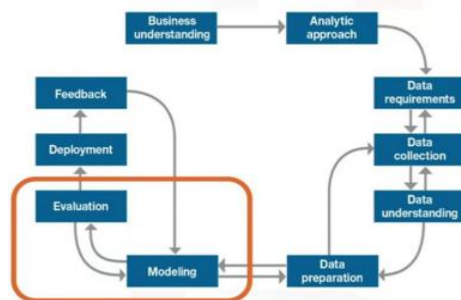
☒ True.

Correct
Correct.



Modeling & Evaluation:

From Modeling to Evaluation



Modeling

- In what way can the data be visualized to get to the answer that is required?



Evaluation

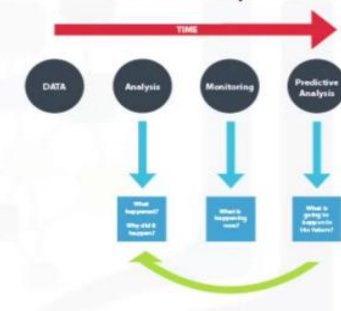
- Does the model used really answer the initial question or does it need to be adjusted?

Data Modeling – Using Predictive or Descriptive?

Descriptive Analytics



Predictive Analytics



Case Study – Analyzing the 1st model

Initial decision tree classification model

- Low accuracy on “Yes” outcome

Model	Relative Cost Y:N	Overall Accuracy (% correct Y & N)	Sensitivity (Y accuracy)	Specificity (N accuracy)
→ 1	1:1	85%	45%	97%
2	9:1	49%	97%	35%
3	4:1	81%	68%	85%

Case Study – Analyzing the 3rd model

Third model

- Better balance on “Yes” and “No” accuracy

Model	Relative Cost Y:N	Overall Accuracy (% correct Y & N)	Sensitivity (Y accuracy)	Specificity (N accuracy)
→ 1	1:1	85%	45%	97%
→ 2	9:1	49%	97%	35%
→ 3	4:1	81%	68%	85%



Case Study – Analyzing the 2nd model

Second model

- High accuracy on “Yes” but poor on “No”

Model	Relative Cost Y:N	Overall Accuracy (% correct Y & N)	Sensitivity (Y accuracy)	Specificity (N accuracy)
→ 1	1:1	85%	45%	97%
→ 2	9:1	49%	97%	35%
3	4:1	81%	68%	85%



In the case study, the best performing model was the second model, with a relative cost of 4:1 and an overall accuracy of 85%.

☐ True.

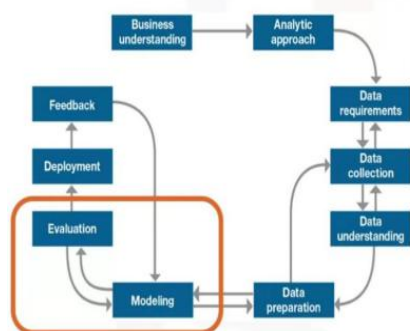
☒ False.

Correct

Correct.



From Modeling to Evaluation



Modeling

- In what way can the data be visualized to get to the answer that is required?



Evaluation

- Does the model used really answer the initial question or does it need to be adjusted?

Case Study – Misclassification costs

Misclassification cost tuning

- Tune the relative misclassification costs
- Balance true-positive rate and false-positive rate for best model

Model	Relative Cost Y:N	True Positive Rate (Sensitivity)	Specificity (accuracy on N)	False Positive Rate (1 – Specificity)
1	1:1	0.45	0.97	0.03
2	1.5:1	0.60	0.92	0.08
3	4:1	0.68	0.85	0.15
4	9:1	0.97	0.35	0.65

Case Study – True-Positives vs False-Positives

Misclassification cost tuning

- Tune the relative misclassification costs
- Balance true-positive rate and false-positive rate for best model

Model	Relative Cost Y:N	True Positive Rate (Sensitivity)	Specificity (accuracy on N)	False Positive Rate (1 – Specificity)
1	1:1	0.45	0.97	0.03
2	1.5:1	0.60	0.92	0.08
3	4:1	0.68	0.85	0.15
4	9:1	0.97	0.35	0.65

Case Study – How to determine the optimal model?

Misclassification cost tuning

- Tune the relative misclassification costs
- Balance true-positive rate and false-positive rate for best model

Model	Relative Cost Y:N	True Positive Rate (Sensitivity)	Specificity (accuracy on N)	False Positive Rate (1 – Specificity)
1	1:1	0.45	0.97	0.03
2	1.5:1	0.60	0.92	0.08
3	4:1	0.68	0.85	0.15
4	9:1	0.97	0.35	0.65

Model evaluation can have two main phases: a diagnostic measures phase and statistical significance testing.

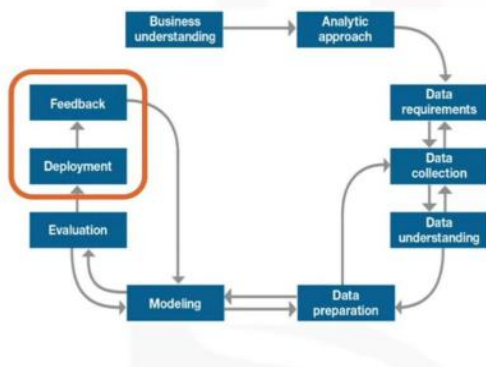
☐ False.

☒ True.

Correct

Correct.

From Deployment to Feedback



Case Study – Understand the results

Assimilate knowledge for business

- Practical understanding of the meaning of model results
- Implications of model results for designing intervention actions



Case Study – Gathering application requirements

Application requirements

- Automated, near-real-time risk assessments of CHF inpatients
- Easy to use
- Automated data preparation and scoring
- Up-to-date risk assessment to help clinicians target high-risk patients



Example 1 – Solution deployment

Hospitalization risk for juvenile diabetes patients



Before the model is evaluated and the data scientist is confident it will work, it is deployed and put to the ultimate test.

☐ False.

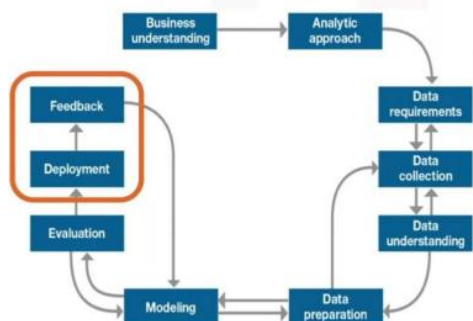
Correct

Correct.

☐ True.



Feedback – Problem solved? Question answered?



From Deployment to Feedback



Once the model is evaluated and the data scientist is confident it will work, it is deployed and put to the ultimate test

- Actual real-time use in the field

Case Study – Assessing model performance

Define review process

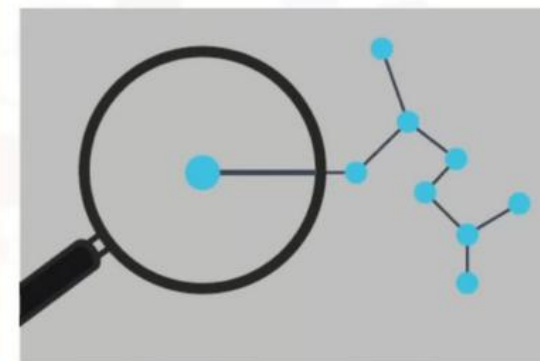
- To measure results of applying the risk model to the CHF patient population
- Track patients who received intervention
 - Actual readmission outcomes
- Measure effectiveness of intervention
 - Compare readmission rates before & after model implementation



Case Study – Refinement

Refine model

- Initial review after the first year of implementation
- Based on feedback data and knowledge gained
- Participation in intervention program
- Possibly incorporate detailed pharmaceutical data originally deferred
- Other possible refinements as yet unknown



The data science methodology is highly iterative, ensuring the refinement at each stage in the game.




☐ False.

☒ True.

Correct

Correct.

Python Libraries to Analyse Data

- Pandas 
 - Provides data structures and operations for data (e.g. tables and time series) manipulation and analysis.
- Numpy 
 - Provides means to work with multidimensional arrays.
- Matplotlib 
 - A plotting library used to create high-quality graphs, charts, and figures.

Create DataFrames using Dictionaries

```
import pandas as pd
data = { 'name': ['Fuat', 'Aykut', 'Erkut'],
        'midterm': [60, 85, 100],
        'final': [69, 90, 100],
        'attendance': [6, 10, 10]
}
df_bbm101 = pd.DataFrame(data)

print(df_bbm101.head()) # Prints top 5 rows
```

	name	midterm	final	attendance
0	Fuat	60	69	7
1	Aykut	85	90	10
2	Erkut	100	100	10

Pandas



- A library that contains high-performance, easy-to-use data structures and data analysis tools.
- Some important aspects of Pandas
 - A fast and efficient DataFrame object for data manipulation with integrated indexing.
 - Tools for reading and writing data in different formats, e.g. csv, Excel, SQL Database.
 - Slicing, indexing, subsetting, merging and joining of huge datasets.
- Typically imported as **import pandas as pd** in Python programs

Same Thing, in Another Way

```
names = ['Fuat', 'Aykut', 'Erkut']
midterms = [60, 85, 100]
finals = [69, 90, 100]
attendances = [6, 10, 10]

list_labels = ['name', 'midterm', 'final', 'attendance']
list_cols = [names, midterms, finals, attendances]

zipped = list(zip(list_labels, list_cols))

print(zipped) # [('name', ['Fuat', 'Aykut', 'Erkut']),
               # ('midterm', [60, 85, 100]),
               # ('final', [69, 90, 100]),
               # ('attendance', [6, 10, 10])]

data = dict(zipped)

df_bbm101 = pd.DataFrame(data)
```


Broadcasting

```
df_bbm101['total'] = 0
# Adds new column to df and
# broadcasts 0 to entire column

print(df_bbm101.head())
```

	name	midterm	final	attendance	total
0	Fuat	60	69	6	0
1	Aykut	85	90	10	0
2	Erkut	100	100	10	0

Subsetting/Slicing Data

```
print(df_bbm101[['name', 'grade']])

print(df_bbm101.iloc[:, [0, 5]])

print(df_bbm101.iloc[:, [True, False, False, False,
                          False, True]])

# They all return the same thing
# name and grade columns of the df
# Same principle can be applied to rows as well
```

	name	grade
0	Fuat	D
1	Aykut	B
2	Erkut	A

Compute Columns

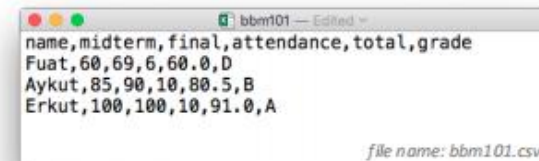
```
df_bbm101['total'] = df_bbm101['midterm']*0.3 + \
                    df_bbm101['final']*0.6 + \
                    df_bbm101['attendance']*0.1

df_bbm101.loc[(df_bbm101['total'] >= 60) &
              (df_bbm101['total'] < 70), 'grade'] = 'D'
...
# Code to compute Bs and Cs comes here
df_bbm101.loc[df_bbm101['total'] >= 90, 'grade'] = 'A'

print(df_bbm101.head())
```

	name	midterm	final	attendance	total	grade
0	Fuat	60	69	6	60.0	D
1	Aykut	85	90	10	80.5	B
2	Erkut	100	100	10	91.0	A

DataFrames from CSV Files



name	midterm	final	attendance	total	grade
Fuat	60	69	6	60.0	D
Aykut	85	90	10	80.5	B
Erkut	100	100	10	91.0	A

```
df_bbm101 = pd.read_csv('bbm101.csv')
print(df_bbm101.head())
```

	name	midterm	final	attendance	total	grade
0	Fuat	60	69	6	60.0	D
1	Aykut	85	90	10	80.5	B
2	Erkut	100	100	10	91.0	A

Indexing DataFrames

```
df_bbm101 = pd.read_csv('bbm101.csv', index_col = 'name')
print(df_bbm101.head())
```

name	midterm	final	attendance	total	grade
Fuat	60	69	6	60.0	D
Aykut	85	90	10	80.5	B
Erkut	100	100	10	91.0	A

```
print(df_bbm101.loc['Fuat'])
```

midterm	60
final	69
attendance	6
total	60
grade	D

Name: Fuat, dtype: object

```
print(df_bbm101.
      loc[['Aykut', 'Erkut']])
```

name	midterm	final	attendance	total	grade
Aykut	85	90	10	80.5	B
Erkut	100	100	10	91.0	A

11

Creating Numpy Arrays

```
import numpy as np
```

```
a = np.array([1,2,3])      # Create a rank 1 array
print(type(a))             # <class 'numpy.ndarray'>
print(a.shape)             # (3,)
print(a)                   # [1 2 3]
print(a[0], a[1], a[2])    # 1 2 3
```

```
b = np.array([[1,2,3],[4,5,6]]) # Create a rank 2 array
print(b.shape)                 # (2, 3)
print(b)                       # [[1 2 3]
                                #  [4 5 6]]
print(b[0, 0], b[0, 1], b[1, 0]) # 1 2 4
```

Numpy



- A library for the Python programming language, adding support for large **multi-dimensional arrays and matrices**,
 - along with a large collection of high-level mathematical functions to operate on these arrays.
- A numpy array is a grid of values, **all of the same type**, and is indexed by a tuple of nonnegative integers.
- The number of dimensions is the **rank** of the array.
- The **shape** of an array is a tuple of integers giving the size of the array along each dimension.
- Typically imported as **import numpy as np** in Python programs

Miscellaneous Ways to Create Arrays

```
a = np.zeros((2,2))      # Create an array of all zeros
print(a)                 # [[ 0.  0.]
                        #  [ 0.  0.]]

b = np.ones((1,2))       # Create an array of all ones
print(b)                 # [[ 1.  1.]]

c = np.full((2,2), 7)    # Create a constant array
print(c)                 # [[ 7.  7.]
                        #  [ 7.  7.]]

d = np.eye(2)            # Create a 2x2 identity matrix
print(d)                 # [[ 1.  0.]
                        #  [ 0.  1.]]

e = np.random.random((2,2)) # Create an array filled with
                             # random values
print(e)                 # Might print
                        # [[ 0.91940167  0.08143941]
                        #  [ 0.68744134  0.87236687]]
```


Slicing

- Similar to slicing Python lists.
- Since arrays may be multidimensional, you must specify a slice for each dimension of the array.
- Slices are views (not copies) of the original data.

Slicing Examples

```
a = np.array([[1, 2, 3, 4],      # Create a rank 2 array
              [5, 6, 7, 8],      # with shape (3, 4)
              [9, 10, 11, 12]])

print(a)                        # [[ 1  2  3  4]
                                #  [ 5  6  7  8]
                                #  [ 9 10 11 12]]

b = a[:2, 1:3]
print(b)                        # [[ 2  3]
                                #  [ 6  7]]

print(a[1, :])                  # [5 6 7 8]

print(a[:, :-2])                # [[ 1  2]
                                #  [ 5  6]
                                #  [ 9 10]]
```

Integer Indexing Examples

```
a = np.array([1, 2, 3, 4, 5, 6])
print(a)                        # [1 2 3 4 5 6]
print(a[[1, 3, 5]])            # [2 4 6]

a = np.array([[1, 2], [3, 4], [5, 6]])
print(a)                        # [[ 1  2]
                                #  [ 3  4]
                                #  [ 5  6]]

# The returned array will have shape (3,)
print(a[[0, 1, 2], [0, 1, 0]])  # [1 4 5]
print(np.array([a[0, 0], a[1, 1], a[2, 0]])) # [1 4 5]

# The same element from the source array can be reused
print(a[[0, 0], [1, 1]])        # [2 2]
print(np.array([a[0, 1], a[0, 1]])) # [2 2]
```

Boolean (or, Mask) Indexing Examples

```
a = np.array([1, 2, 3, 4, 5, 6])

bool_idx = (a > 2)
# Find the elements of a that are bigger than 2;
# this returns a numpy array of Booleans of the same
# shape as a, where each slot of bool_idx tells
# whether that element of a is > 2.

print(bool_idx)                 # [False False  True
                                #                               True  True  True]

# We use boolean array indexing to construct a rank 1 array
# consisting of the elements of a corresponding to the True
# values of bool_idx
print(a[bool_idx])              # [3 4 5 6]

# We can do all of the above in a single concise statement:
print(a[a > 2])                 # [3 4 5 6]
```

Boolean (or, Mask) Indexing

- Boolean array indexing lets you pick out arbitrary elements of an array.
- Frequently used to select the elements of an array that satisfy some condition.
 - Thus, called the mask indexing.

Array Math

- Basic mathematical functions operate elementwise on arrays.

```
x = np.array([[1, 2], [3, 4]])
y = np.array([[5, 6], [7, 8]])
```

```
# Elementwise sum
print(x + y)
print(np.add(x, y))
# [[ 6  8]
#  [10 12]]

# Elementwise product
print(x * y)
print(np.multiply(x, y))
# [[ 5 12]
#  [21 32]]
```

Same principle holds for
"np.divide, /" and "np.subtract, -"

Array Math (Cont'd)

```
x = np.array([[1, 2], [3, 4]])
y = np.array([[5, 6], [7, 8]])
```

```
v = np.array([9, 10])
w = np.array([11, 12])
```

```
# Inner product of vectors;
# both produce 219
print(v.dot(w))
print(np.dot(v, w))
```

```
# Matrix / vector product;
# both produce the rank 1
# array [29 67]
print(x.dot(v))
print(np.dot(x, v))
```

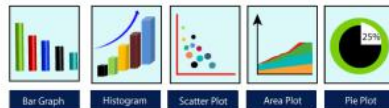
```
# Matrix / matrix product;
# both produce a rank 2 array
# [[19 22]
#  [43 50]]
print(x.dot(y))
print(np.dot(x, y))

# Transpose of x
# [[1 3]
#  [2 4]]
print(x.T)
```

Matplotlib



- Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments.
- Typically imported as **import matplotlib.pyplot as plt** in Python programs.
- Pyplot is a module of Matplotlib which provides simple functions to add plot elements like lines, images, text, etc.
- There are many plot types. Some of are more frequently used.



Make a Simple Plot

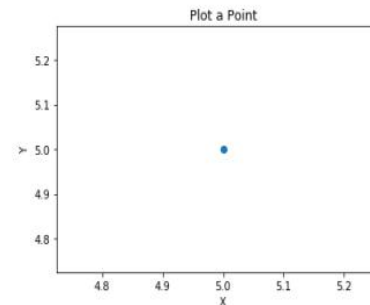
```
import matplotlib.pyplot as plt
```

```
plt.plot(5, 5, 'o')
```

```
plt.title("Plot a Point")
```

```
plt.xlabel("X")
plt.ylabel("Y")
```

```
plt.show()
```



Plot a Simple Line

```
import matplotlib.pyplot as plt
```

```
year = ['2016', '2017', '2018', '2019', '2020']
lowest_rank = [21358, 20816, 17555, 11743, 7500]
```

```
plt.plot(year, lowest_rank)
```

```
plt.title("HU-BBM Progress")
plt.xlabel('Year')
plt.ylabel('Lowest Rank')

plt.show()
```

