Microsoft

Introduction to Data Science

July 2019
Microsoft Reactor | Redmond, WA

#ReactorRedmond

Microsoft

Justin Garrett

Principal Program Manager,
Microsoft Cloud + AI

@justgar

Microsoft

Sarah Guthals, PhD

Senior Program Manager,
Microsoft Cloud + AI

@sarahguthals

Today’s workshop is a Microsoft Reactor event.

REACTOR

developer.microsoft.com/reactor/

@MSFTReactor on Twitter

Data Science

- Why is it important?
- Why should you care?
- How to begin?

What to expect…

Python

Jupyter Notebook

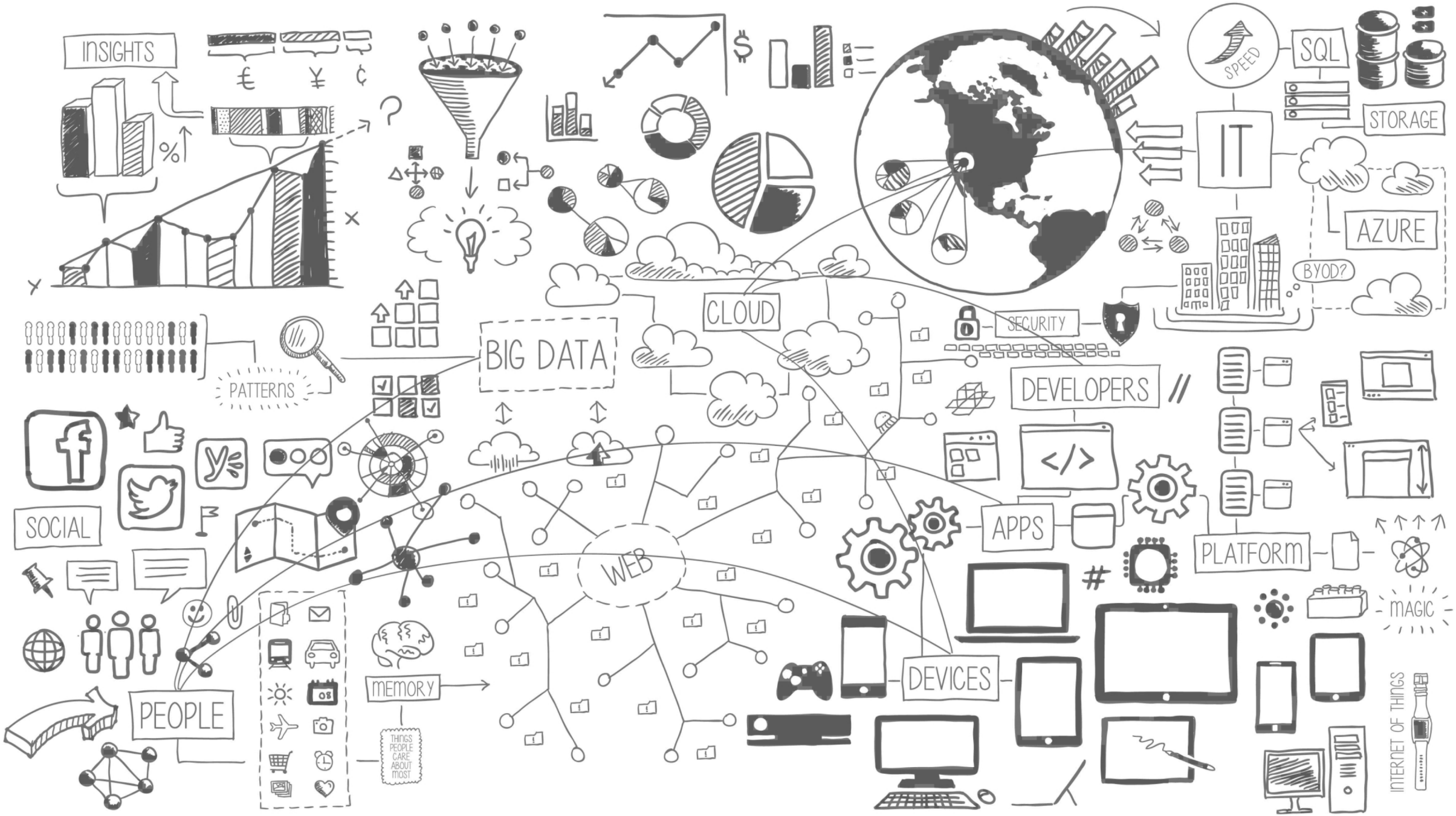
Microsoft Azure

Workshop Agenda | Day 1

09:30 am	Event Registration
10:00 am	Introduction to Data Science
10:15 am	Introduction to Python Basics
11:00 am	5 Minute Break
11:05 am	Python Basics Continued
12:00 pm	Lunch break
1:00 pm	NumPy: Your Local Data Friend
2:00 pm	Pandas are more than bears: How to import, clean, and store data
3:00 pm	10 Minute Break
3:10 pm	Data Science 101: Getting your Data Ready
4:00 pm	Event End

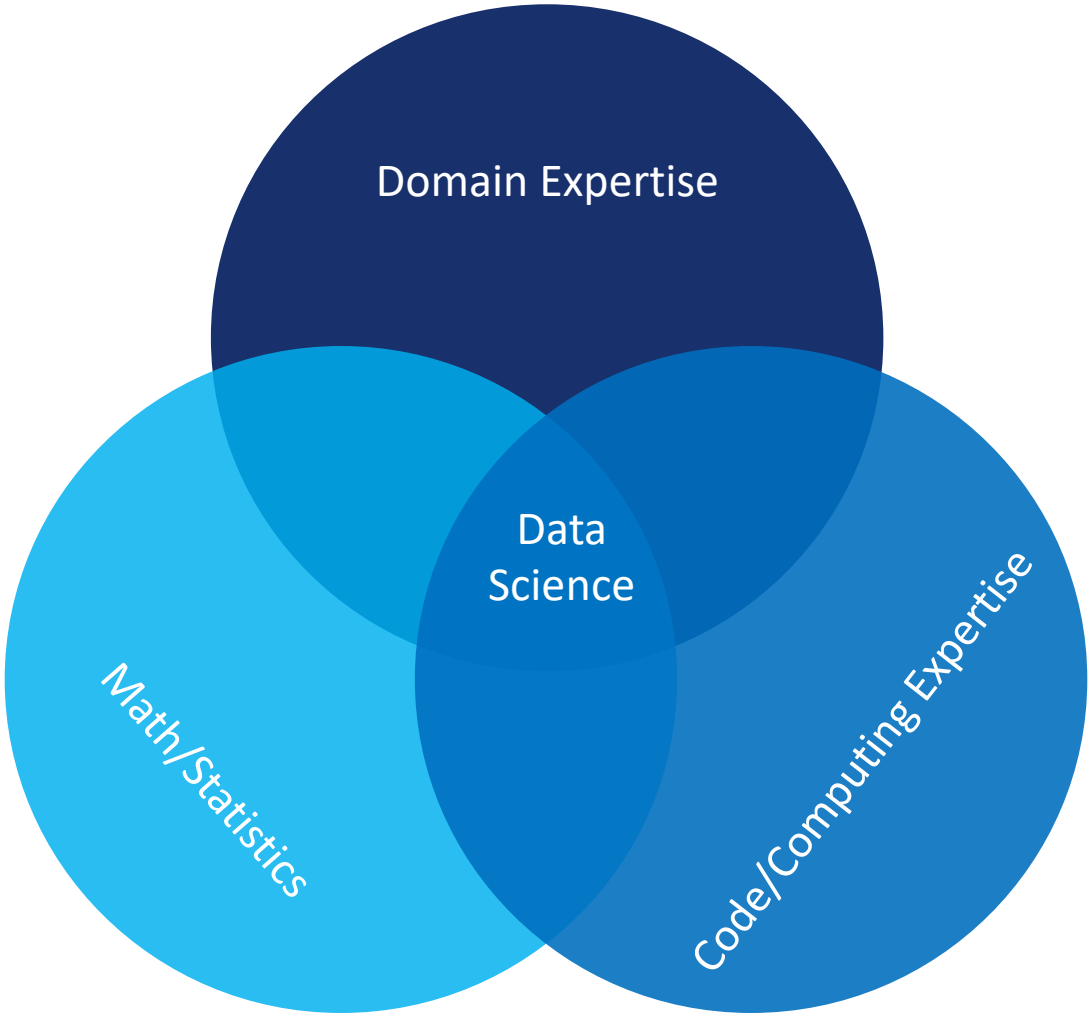
Workshop Agenda | Day 2

10:00 am	Introduction to Machine Learning Models and Linear Regression
11:00 am	Using the Cloud for Machine Learning with Azure Cognitive Services
12:00 pm	Lunch Break
1:00 pm	Machine Learning Capstone Project
2:00 pm	15 Minutes Break
2:15 pm	Capstone
4:00 pm	Event End


INSIGHTS
SPEED
SQL
STORAGE
IT
AZURE
BYOD?
CLOUD
SECURITY
BIG DATA
PATTERNS
DEVELOPERS
APPS
PLATFORM
MAGIC
SOCIAL
WEB
DEVICES
PEOPLE
MEMORY
THINGS PEOPLE CARE ABOUT MOST
INTERNET OF THINGS

Data Science: Using historical data to predict the future.

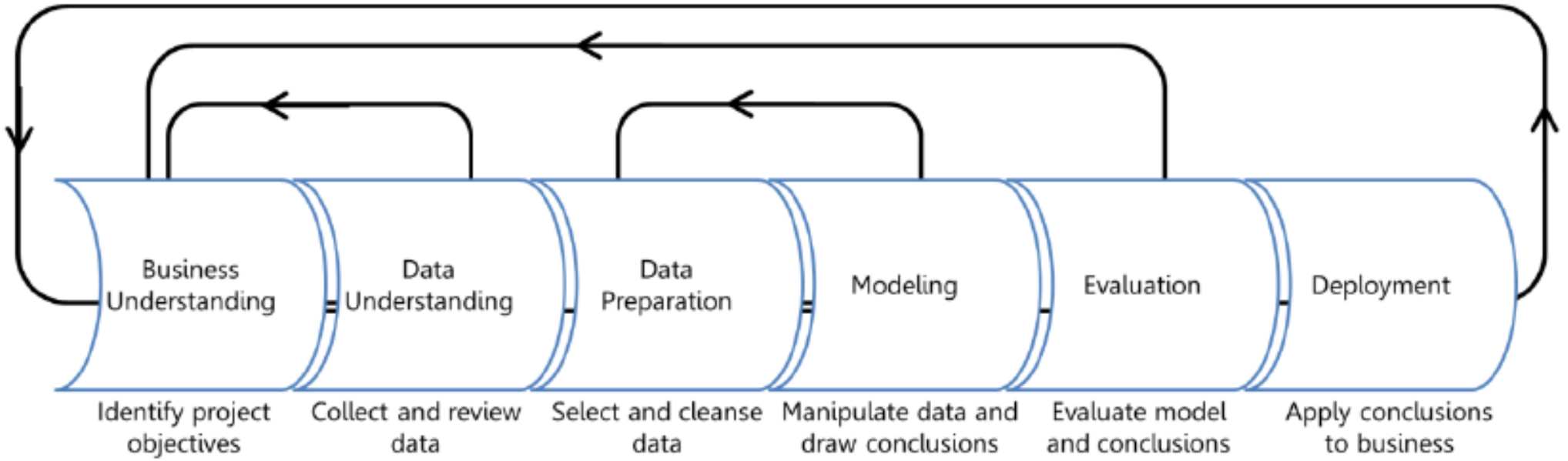
Data



Action

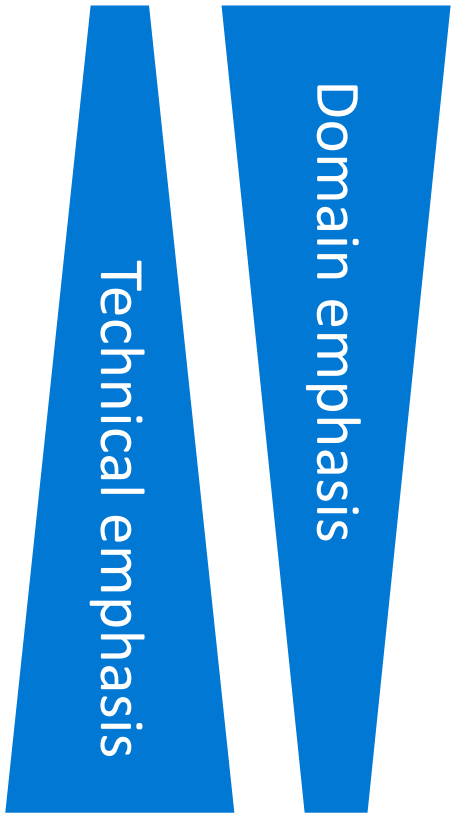
Data Science Process

Extracting value from large amounts of data **{and making human sense of it}** is the primary challenge of data science.



- from *Introduction to Data Science* on Microsoft Learn: https://docs.microsoft.com/learn/modules/intro-to-data-science-in-azure/2-data-science-process
- Other helpful beginners' series: https://docs.microsoft.com/en-us/azure/machine-learning/studio/data-science-for-beginners-the-5-questions-data-science-answers

Specialized roles in Data Science



Data Analyst

Data Engineer

Data Scientist

Data Architect

Developer

Data Scientist

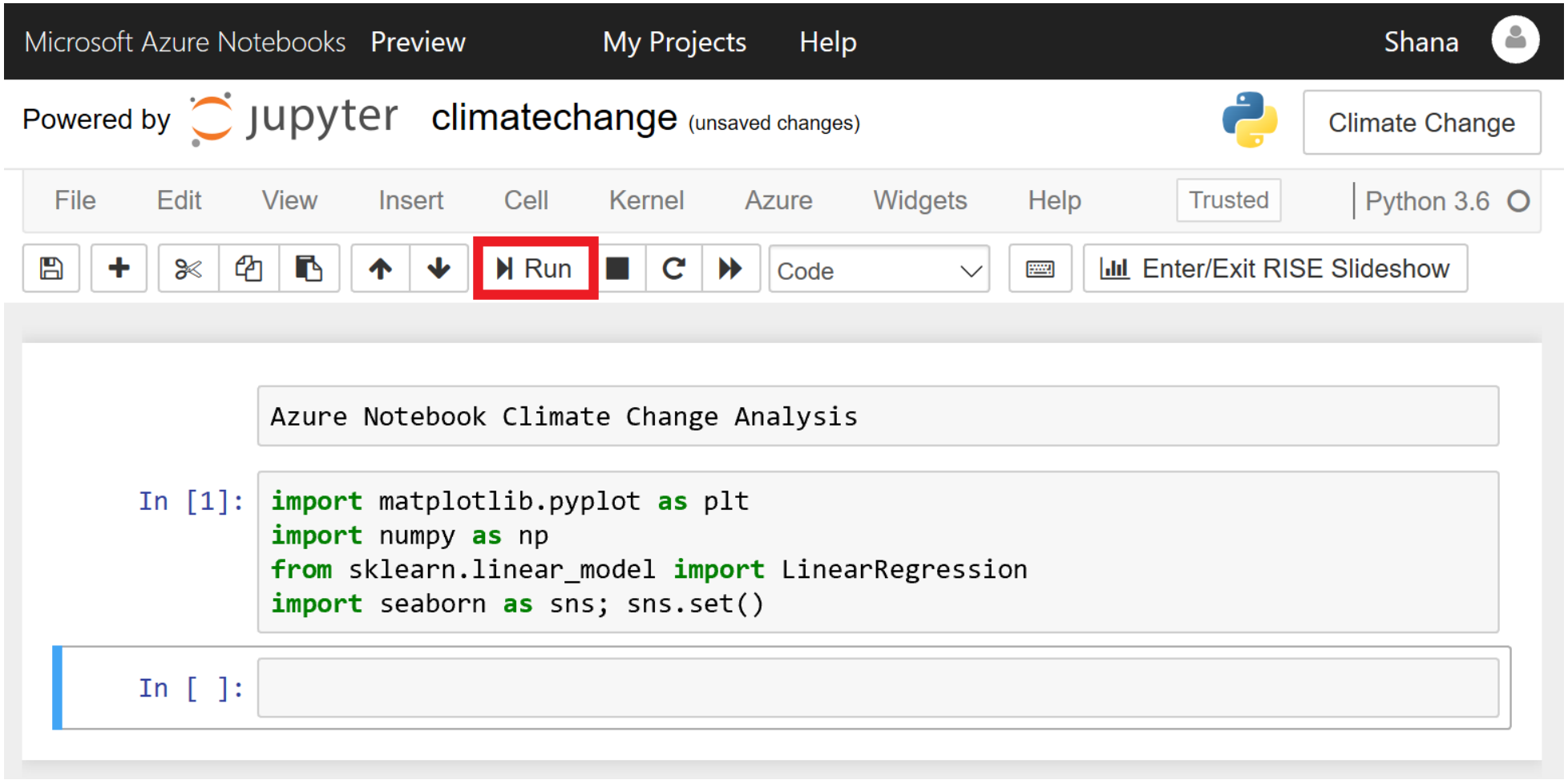
#1 in 2018 and 2019

Median base salary (USA):

$130,000

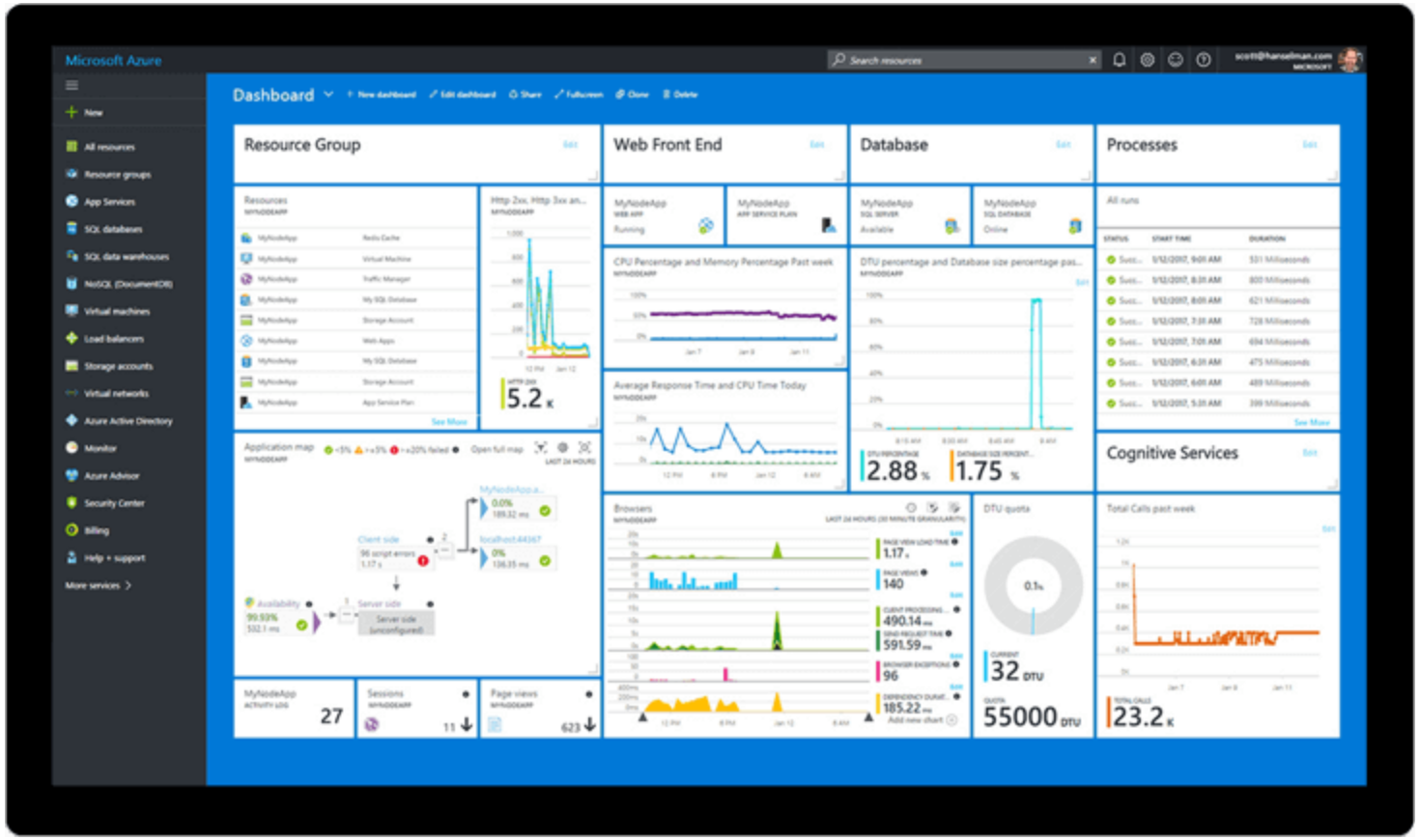
https://blog.linkedin.com/2019/january/10/linkedins-most-promising-jobs-of-2019

How to navigate Azure Notebooks



https://docs.microsoft.com/en-us/learn/modules/analyze-climate-data-with-azure-notebooks/1-create-an-azure-notebook

Free Azure for your workshop capstone and beyond



azure.microsoft.com/free/

azure.microsoft.com/free/students/

Sections

- Sections 1+2: **Python Overview**
- Section 3: **NumPy**
- Section 4: **Pandas**
- Section 5: **Data cleaning and manipulation**
- Section 6: **Introduction to machine learning**
- Section 7: **Cloud-based ML**
- Section 8: **Capstone project:** Significant Volcanic Eruptions database

Microsoft

Python basics (part 1)

Section 1

Section 1 overview

What you will know by the end of section 1

- Arithmetic and numeric types in Python
- Strings, strings, and more strings
- Other data types
  - Lists
  - Tuples
  - Dictionaries
- Membership testing

Assumptions for section 1

- None! Come as you are.

Section 1 overview

Why Python for data science?

- Easy to learn
- Flexible
- Powerful libraries

Leading social network, video streaming, and search engine companies built some of their core technologies using Python.

They also use Python for data science.

Adjusting list levels

- Python numeric operators: `+ - * / // ** %`

- Variables:

```
length = 15
width = 3 * 5
length * width
```

```
225
```

- Expressions:

```
1 < 2 or 1 > 2
```

```
True
```

Strings

- String literals:

```
'"Isn\'t," she said.'
```

```
'"Isn\'t," she said.'
```

- Concatenating strings:

```
3 * 'un' + 'ium'
```

```
'unununium'
```

Strings (continued)

- String indices:

```
word = 'Python'
word[0]  # Character in position 0.
```

```
'P'
```

- Slicing strings:

```
word[0:2]  # Characters from position 0 (included) to 2 (excluded).
```

```
'Py'
```

Other data types

- Lists:

```
squares = [1, 4, 9, 16, 25]
```

  - List-object methods: append(), extend(), index(), count(), remove(), pop(), insert(), reverse(), sort().

- Tuples:

```
t = (1, 2, 3)
```

- Dictionaries:

```
capitals = {'France': ('Paris', 2140526)}
capitals['Nigeria'] = ('Lagos', 6048430)
capitals
```

```
{'France': ('Paris', 2140526), 'Nigeria': ('Lagos', 6048430)}
```

Membership testing

- ‘in’:

```
tup = ('a', 'b', 'c')
'b' in tup
```

```
True
```

- ‘not in’:

```
lis = ['a', 'b', 'c']
'a' not in lis
```

```
False
```

Microsoft

Python basics (part 2)

Section 2

Section 2 overview

What you will know by the end of section 2

- Control flow in Python
  - If statements
  - For loops
  - While loops
- Functions
- List comprehensions
- Classes and objects
- Importing modules

Assumptions for section 2

- Familiarity with Python data types (section 1)

Control flow in Python: if statements

Perform repeated tasks in a program:

```
y = 1
if y % 2 == 0:
    print('Even')
elif y == 1:
    print('One')
else:
    print('Odd')
```

```
One
```

Control flow in Python: for loops

Perform repeated tasks in a program:

```
for n in range(2, 10):
    for x in range(2, n):
        if n % x == 0:
            print(n, 'equals', x, '*', n//x)
            break
    else:
        print(n, 'is a prime number')
```

```
2 is a prime number
3 is a prime number
4 equals 2 * 2
5 is a prime number
6 equals 2 * 3
7 is a prime number
8 equals 2 * 4
9 equals 3 * 3
```

Control flow in Python: while loops

Combines the logic of an if statement with the iteration of a for loop:

```
# In the Fibonacci series, the sum of two elements defines the next.
a, b = 0, 1

while b < 100:
    print(b, end=', ')
    a, b = b, a+b
```

```
1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89,
```

Functions

Create more complex logic that you can reuse in a program:

```
def fib(n):
    """Print a Fibonacci series up to n."""
    a, b = 0, 1
    while a < n:
        print(a, end=', ')
        a, b = b, a+b
```

```
fib(2000)
```

```
0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597,
```

List comprehensions

Programmatically create lists:

```
numbers = [x for x in range(1,11)]
numbers
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

```
odd_squares = [x*x for x in range(1,11) if x % 2 != 0]
odd_squares
```

```
[1, 9, 25, 49, 81]
```

Classes and instance objects (continued)

Create classes:

```
class BankAccount:
    """Where does your money go?"""

    account_count = 0

    # Above is an example of a class variable
    # Below are the class methods


    def __init__(self, balance=0):
        self.balance = balance
        BankAccount.account_count += 1

    def deposit(self, amount):
        self.balance += amount
        self.display_balance()

    def withdrawal(self, amount):
        self.balance -= amount
        self.display_balance()

    def display_balance(self):
        print('New balance: ${:.2f}'.format(self.balance))
```

```
my_account = BankAccount(50)
my_account.balance
```

```
50
```

```
my_account.deposit(100)
```

```
New balance: $150.00
```

```
my_account.withdrawal(125)
```

```
New balance: $25.00
```

Importing modules

```
import math
math.factorial(5)
```

```
120
```

```
from math import factorial
factorial(5)
```

```
120
```

Microsoft

Intro to NumPy (part 1)

Section 3

Section 3 (part 1) overview

What you will know by the end of section 3 (part 1)

- Built-in help
- NumPy arrays
  - Creating
  - Attributes
  - Indexing
  - Reshaping
  - Splitting and joining
  - Fancy indexing
  - Sorting

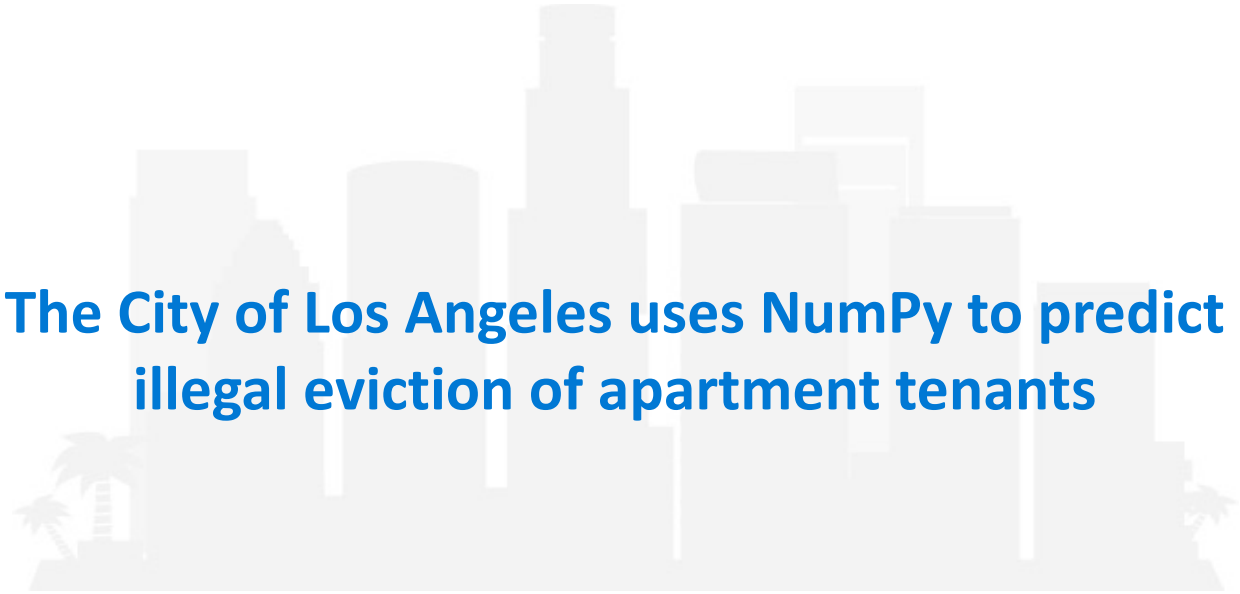
Assumptions for section 3

- Basic Python (sections 1 and 2)

Section 3 overview

Why NumPy?

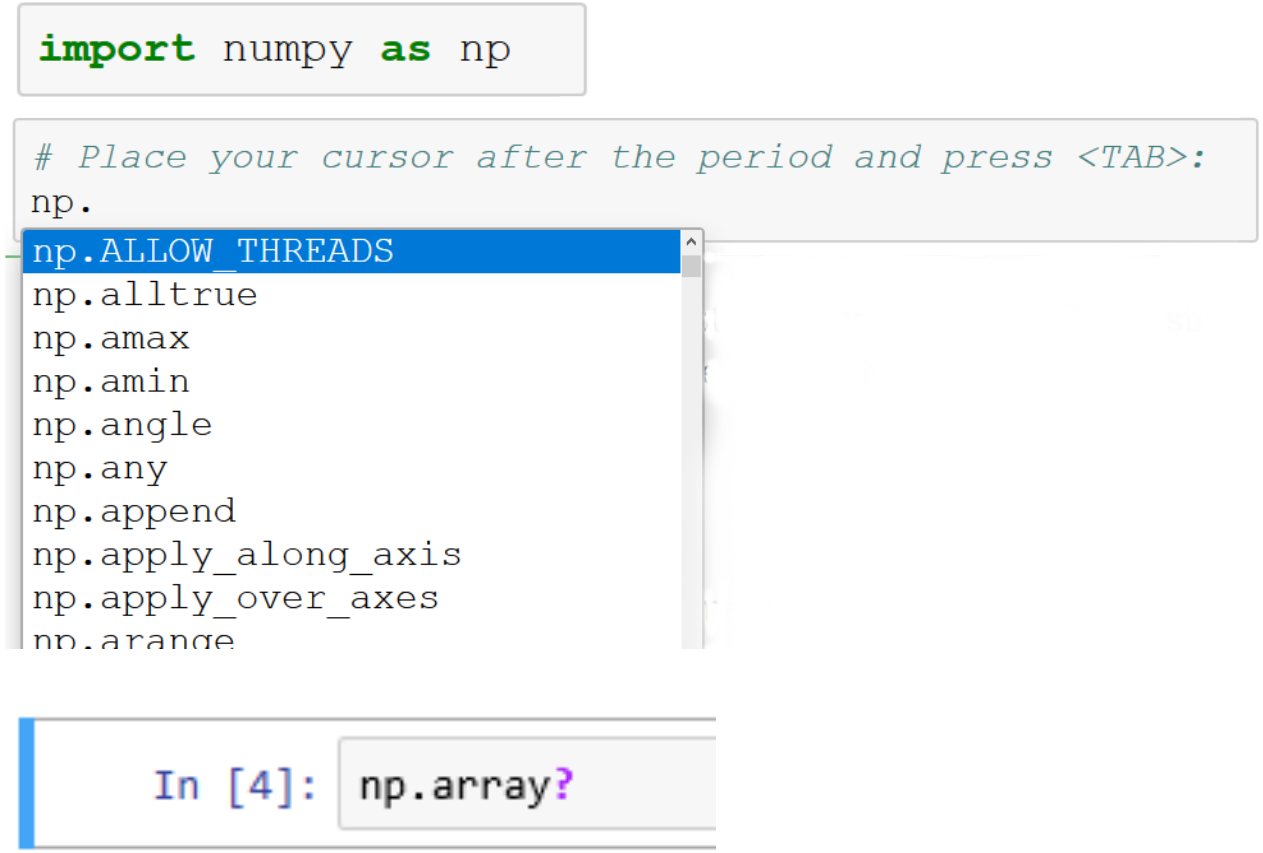
- Faster, pre-compiled functions
- Standard numeric library for data science using Python
- Foundation for other popular libraries (such as pandas)



The City of Los Angeles uses NumPy to predict illegal eviction of apartment tenants

IPython and built-in help

- Use the tab key to explore package contents.
- Use “?” for help info related to any method.



NumPy arrays

- Lists in Python:

```
myList2 = [True, "2", 3.0, 4]
[type(item) for item in myList2]
```

```
[bool, str, float, int]
```

- NumPy data types: floating point, integer, Boolean, string, general Python object

- Fixed-type arrays in Python:

```
# Create an integer array:
np.array([1, 4, 2, 5, 3])
```

```
array([1, 4, 2, 5, 3])
```

Working with NumPy arrays: array attributes

```
a3 = np.random.randint(10, size=(3, 4, 5))  # Three-dimensional array
```

```
# Change the values in this code snippet to look at the attributes for a1, a2, and a3:
print("a3 ndim: ", a3.ndim)
print("a3 shape:", a3.shape)
print("a3 size: ", a3.size)
```

```
a3 ndim:  3
a3 shape: (3, 4, 5)
a3 size:  60
```

Working with NumPy arrays: indexing

```
np.random.seed(0)
a1 = np.random.randint(10, size=6)
a1
```

```
array([5, 0, 3, 3, 7, 9])
```

```
a1[0]
```

```
5
```

```
a1[4]
```

```
7
```

```
a1[-1]
```

```
9
```

Working with NumPy arrays: slicing arrays

```
a = np.arange(10)
a
```

```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

```
a[4:7]  # middle sub-array
```

```
array([4, 5, 6])
```

Working with NumPy arrays: reshaping

```
grid = np.arange(1, 10).reshape((3, 3))
print(grid)
```

```
[[1 2 3]
 [4 5 6]
 [7 8 9]]
```

Working with NumPy arrays: joining and splitting

```
a = np.array([1, 2, 3])
b = np.array([3, 2, 1])
np.concatenate([a, b])
```

```
array([1, 2, 3, 3, 2, 1])
```

```
a = [1, 2, 3, 99, 99, 3, 2, 1]
a1, a2, a3 = np.split(a, [3, 5])
print(a1, a2, a3)
```

```
[1 2 3] [99 99] [3 2 1]
```

Working with NumPy arrays: fancy indexing

```
rand = np.random.RandomState(42)

arr = rand.randint(100, size=10)
print(arr)
```

```
[51 92 14 71 60 20 82 86 74 74]
```

```
ind = [3, 7, 4]
arr[ind]
```

```
array([71, 86, 60])
```

Working with NumPy arrays: sorting

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

```
array([1, 2, 3, 4, 5])
```

```
a.sort()
print(a)
```

```
[1 2 3 4 5]
```

NumPy universal functions (ufuncs)

- Operate on NumPy arrays
- Compiled in C
- Much faster than traditional Python-based operations
- Noticeable performance gain working with large datasets

Microsoft

Intro to NumPy (part 2)

Section 3

Section 3 (part 2) overview

What you will know by the end of section 3 (part 2)

- Aggregations
- Multidimensional aggregations
- Broadcasting
- Boolean masking
- Comparison functions

Assumptions for section 3 (part 2)

- Introductory NumPy (section 3, part 1)

Aggregations

- Summing the values in arrays:

```
myList = np.random.random(100)
sum(myList)
```

```
52.12818058833704
```

- Minimum and maximum values:

```
print(large_array.min(), large_array.max(), large_array.sum())
```

```
1.4057692298008462e-06 0.9999994392723005 500202.5348847683
```

Multidimensional aggregations

```
md = np.random.random((3, 4))
print(md)
```

```
[[0.50063048 0.07383653 0.49018646 0.72521956]
 [0.84926562 0.10226215 0.99559424 0.59250301]
 [0.53509    0.88518089 0.25518136 0.13130483]]
```

```
md.sum()
```

```
6.136255127264715 4
```

```
md.min(axis=0)
```

```
array([0.50063048, 0.07383653, 0.25518136, 0.13130483])
```

Broadcasting

Apply ufuncs to arrays of different sizes:

- “Pad” dimensions in arrays to match
- Stretch one-dimensional arrays

```
horizontal_array = np.arange(3)
vertical_array = np.arange(3)[:, np.newaxis]

print(horizontal_array)
print(vertical_array)
```

```
[0 1 2]
[[0]
 [1]
 [2]]
```

```
horizontal_array + vertical_array
```

```
array([[0, 1, 2],
       [1, 2, 3],
       [2, 3, 4]])
```

Comparison operators as ufuncs

```
rand = np.random.RandomState(0)
two_dim_array = rand.randint(10, size=(3, 4))
two_dim_array
```

```
array([[5, 0, 3, 3],
       [7, 9, 3, 5],
       [2, 4, 7, 6]])
```

```
two_dim_array < 6
```

```
array([[ True,  True,  True,  True],
       [False, False,  True,  True],
       [ True,  True, False, False]])
```

Boolean arrays as masks

```
two_dim_array
```

```
array([[5, 0, 3, 3],
       [7, 9, 3, 5],
       [2, 4, 7, 6]])
```

```
two_dim_array < 5
```

```
array([[False,  True,  True,  True],
       [False, False,  True, False],
       [ True,  True, False, False]])
```

```
two_dim_array[two_dim_array < 5]
```

```
array([0, 3, 3, 3, 2, 4])
```

Application to the capstone project

NumPy ufuncs will form the basis for your examination of the data in the capstone project.

```
volcano_deaths = pd.read_csv('data/noaa_volerup.csv')['DEATHS'].values
np.sum(volcano_deaths > 1000)
```

```
38
```

```
volcano_explosion = pd.read_csv('data/noaa_volerup.csv')['VEI'].values
np.nanmean(volcano_explosion)
```

```
2.868501529051987
```

```
np.nanmedian(volcano_explosion)
```

```
3.0
```

Microsoft

Intro to pandas

Section 4

Section 4 overview

What you will know by the end of section 4

- Data structures in pandas
  - Series
  - DataFrames
- Manipulating data
  - In Series
  - In DataFrames
- Data operations
  - Index preservation
  - Index alignment
  - Operations between Series and DataFrames

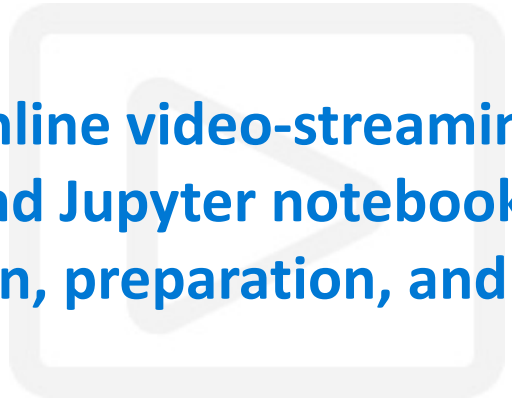
Assumptions for section 4

- Basic Python and NumPy (sections 1–3)

Section 4 overview

Why pandas?

- Standard Python library for handling and manipulating general data
- Intuitive tabular data structure (DataFrame)
- Fast data calculation and transformation using NumPy ufuncs
- Close integration with visualization libraries to easily explore data



The leading online video-streaming service uses pandas and Jupyter notebooks for data exploration, preparation, and validation

Fundamental pandas data structures: Series

```
series_example = pd.Series([-0.5, 0.75, 1.0, -2])
series_example
```

```
0   -0.50
1    0.75
2    1.00
3   -2.00
dtype: float64
```

```
series_example.values
```

```
array([-0.5 ,  0.75,  1.  , -2.  ])
```

```
series_example.index
```

```
RangeIndex(start=0, stop=4, step=1)
```

Pandas data structures: DataFrames

Creating a DataFrame from a dictionary:

```
population_dict = {'France': 65429495,
                   'Germany': 82408706,
                   'Russia': 143910127,
                   'Japan': 126922333}
population = pd.Series(population_dict)
population
```

```
France      65429495
Germany     82408706
Japan      126922333
Russia     143910127
dtype: int64
```

Pandas data structures: DataFrames

Creating a DataFrame from two series:

```
area = pd.Series({'Albania': 28748,
                  'France': 643801,
                  'Germany': 357386,
                  'Japan': 377972,
                  'Russia': 17125200})
population = pd.Series ({'Albania': 2937590,
                         'France': 65429495,
                         'Germany': 82408706,
                         'Russia': 143910127,
                         'Japan': 126922333})
countries = pd.DataFrame({'Area': area, 'Population': population})
countries
```

	Area	Population
Albania	28748	2937590
France	643801	65429495
Germany	357386	82408706
Japan	377972	126922333
Russia	17125200	143910127

Manipulating data in pandas: index objects

```
series_example = pd.Series([-0.5, 0.75, 1.0, -2], index=['a', 'b', 'c', 'd'])
ind = series_example.index
ind
```

```
Index(['a', 'b', 'c', 'd'], dtype='object')
```

```
ind[1]
```

```
'b'
```

Manipulating data: data selection in Series

```
series_example2 = pd.Series([-0.5, 0.75, 1.0, -2], index=['a', 'b', 'c', 'd'])
series_example2
```

```
a   -0.50
b    0.75
c    1.00
d   -2.00
dtype: float64
```

```
series_example2['b']
```

```
0.75
```

```
series_example2[0:2]
```

```
a   -0.50
b    0.75
dtype: float64
```

Manipulating data: data selection in DataFrames

```
countries.iloc[:3, :2]
```

	Area	Population
Albania	28748	2937590
France	643801	65429495
Germany	357386	82408706

```
countries.loc[:'Germany', :'Population']
```

	Area	Population
Albania	28748	2937590
France	643801	65429495
Germany	357386	82408706

```
countries['Area']
```

```
Albania        28748
France        643801
Germany       357386
Japan         377972
Russia      17125200
Name: Area, dtype: int64
```

Operating on data in pandas: index alignment

```
series1 = pd.Series([2, 4, 6], index=[0, 1, 2])
series1
```

```
0    2
1    4
2    6
dtype: int64
```

```
series2 = pd.Series([3, 5, 7], index=[1, 2, 3])
series2
```

```
1    3
2    5
3    7
dtype: int64
```

```
series1 + series2
```

```
0     NaN
1     7.0
2    11.0
3     NaN
dtype: float64
```

```
series1.add(series2, fill_value=0)
```

```
0     2.0
1     7.0
2    11.0
3     7.0
dtype: float64
```

Application to the capstone project

DataFrames will be your primary tools for working with the National Oceanic and Atmospheric Administration (NOAA) Significant Volcanic Eruptions database in the capstone project.

```
noaa_volcano_df = pd.read_csv('data/noaa_volerup.csv')
noaa_volcano_df.head()
```

	Year	Month	Day	TSU	EQ	Name	Location	Country	Latitude	Longitude	...	TOTAL_DEATHS	TOTAL_DEATHS_DESCRIPTION	TOTAL_MISSING
0	-4360	NaN	NaN	NaN	NaN	Macauley Island	Kermadec Is	New Zealand	-30.200	-178.470	...	NaN	NaN	NaN
1	-4350	NaN	NaN	NaN	NaN	Kikai	Ryukyu Is	Japan	30.780	130.280	...	NaN	3.0	NaN
2	-4050	NaN	NaN	NaN	NaN	Masaya	Nicaragua	Nicaragua	11.984	-86.161	...	NaN	NaN	NaN
3	-4000	NaN	NaN	NaN	NaN	Pago	New Britain-SW Pac	Papua New Guinea	-5.580	150.520	...	NaN	1.0	NaN
4	-3580	NaN	NaN	NaN	NaN	Taal	Luzon-Philippines	Philippines	14.002	120.993	...	NaN	NaN	NaN

5 rows × 36 columns

Microsoft

Manipulating and cleaning data

Section 5

Section 5 overview

What you will know by the end of section 5

- Exploring information in DataFrames
- Working with missing data values
  - Identifying
  - Removing
  - Filling
- Combining datasets
- Exploratory statistics and visualizations
- Example: Boston Housing dataset

Assumptions for section 5

- Basic familiarity with pandas (section 4)

Section 5 overview

Why practice "data munging?"

- Both science…and art
- Essential to high-quality insights / outcomes

Exploring DataFrame information

DataFrame.info

```
iris_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 4 columns):
sepal length (cm)    150 non-null float64
sepal width (cm)     150 non-null float64
petal length (cm)    150 non-null float64
petal width (cm)     150 non-null float64
dtypes: float64(4)
memory usage: 4.8 KB
```

DataFrame.head

```
iris_df.head()
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

DataFrame.tail

```
iris_df.tail()
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
145	6.7	3.0	5.2	2.3
146	6.3	2.5	5.0	1.9
147	6.5	3.0	5.2	2.0
148	6.2	3.4	5.4	2.3
149	5.9	3.0	5.1	1.8

Dealing with missing data

- Python null value: None
- NumPy/pandas null value: NaN

```
example = pd.DataFrame([0, np.nan, '', None])
```

`example`

	0
0	0
1	NaN
2	
3	None

`example.isnull()`

	0
0	False
1	True
2	False
3	True

`example.dropna()`

	0
0	0
2	

`example.fillna(0)`

	0
0	0
1	0
2	
3	0

Removing duplicate data

- Identifying duplicates
- Dropping duplicates

```
example
```

	letters	numbers
0	A	1
1	B	2
2	A	1
3	B	3
4	B	3

```
example.duplicated()
```

```
0    False
1    False
2     True
3    False
4     True
dtype: bool
```

```
example.drop_duplicates()
```

	letters	numbers
0	A	1
1	B	2
3	B	3

Combining datasets

- Categories of joins:
  - *one-to-one*, *many-to-one*, and *many-to-many*

```
df3 = pd.merge(df1, df2)
```

- Concatenation in NumPy:

```
np.concatenate([x, y, z])
```

- Concatenation in pandas:

```
pd.concat([ser1, ser2])
```

- Concatenation with joins:

```
pd.concat([df10, df11], join='inner')
```

Exploratory statistics and visualization

```
df = pd.read_csv('/Data/housing_dataset.csv')
df.head()
```

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	LSTAT	MEDV
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	15.3	4.98	24.0
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2.0	242.0	17.8	9.14	21.6
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2.0	242.0	17.8	4.03	34.7
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3.0	222.0	18.7	2.94	33.4
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3.0	222.0	18.7	5.33	36.2

```
df.describe()
```

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS
count	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000
mean	3.593761	11.363636	11.136779	0.069170	0.554695	6.284634	68.574901	3.795043
std	8.596783	23.322453	6.860353	0.253994	0.115878	0.702617	28.148861	2.105710
min	0.006320	0.000000	0.460000	0.000000	0.385000	3.561000	2.900000	1.129600
25%	0.082045	0.000000	5.190000	0.000000	0.449000	5.885500	45.025000	2.100175
50%	0.256510	0.000000	9.690000	0.000000	0.538000	6.208500	77.500000	3.207450
75%	3.647423	12.500000	18.100000	0.000000	0.624000	6.623500	94.075000	5.188425
max	88.976200	100.000000	27.740000	1.000000	0.871000	8.780000	100.000000	12.126500

```
df.shape
```

```
(506, 13)
```

```
df['MEDV'].mean()
```

```
22.532806324110698
```

Application to the Capstone Project

Tools from this section will be important in exploring the NOAA Significant Volcanic Eruptions Database for the capstone project.

```
noaa_volcano_df.shape
```

```
(827, 36)
```

```
noaa_volcano_df.describe()
```

	Year	Month	Day	Latitude	Longitude	Elevation	VEI
count	827.000000	700.000000	639.000000	827.000000	827.000000	827.000000	654.000000
mean	1722.093108	6.372857	15.348983	15.412455	54.241282	1989.828295	2.868502
std	725.273336	3.345119	9.040416	25.836023	99.888478	1226.020912	1.313548
min	-4360.000000	1.000000	1.000000	-62.970000	-178.470000	-642.000000	0.000000
25%	1785.500000	4.000000	7.000000	-6.745000	-19.700000	1117.000000	2.000000
50%	1918.000000	7.000000	15.000000	13.257000	110.442000	1718.000000	3.000000
75%	1982.000000	9.000000	23.000000	36.404000	130.770000	2690.000000	3.750000
max	2019.000000	12.000000	31.000000	65.730000	177.180000	5967.000000	7.000000

8 rows × 27 columns

```
noaa_volcano_df.isnull().sum()
```

```
Year                              0
Month                           127
Day                             188
TSU                             683
EQ                              770
Name                              0
Location                          0
Country                           0
Latitude                          0
Longitude                         0
Elevation                         0
Type                              0
Status                            0
Time                              0
VEI                             173
Agent                           367
DEATHS                          399
DEATHS_DESCRIPTION              272
MISSING                         819
MISSING_DESCRIPTION             816
INJURIES                        736
INJURIES_DESCRIPTION            712
DAMAGE_MILLIONS_DOLLARS         813
DAMAGE_DESCRIPTION              606
HOUSES_DESTROYED                794
HOUSES_DESTROYED_DESCRIPTION    719
TOTAL_DEATHS                    381
TOTAL_DEATHS_DESCRIPTION        250
TOTAL_MISSING                   819
TOTAL_MISSING_DESCRIPTION       815
TOTAL_INJURIES                  734
TOTAL_INJURIES_DESCRIPTION      704
TOTAL_DAMAGE_MILLIONS_DOLLARS   810
TOTAL_DAMAGE_DESCRIPTION        593
TOTAL_HOUSES_DESTROYED          789
TOTAL_HOUSES_DESTROYED_DESCRIPTION  700
dtype: int64
```

Microsoft

Machine learning

Section 6

Section 6 overview

What you will know by the end of section 6

- Machine learning (ML) overview
- Prediction
  - Linear regression
  - Example: U.S. Housing dataset
- Classification:
  - Logistic regression
    - Example: *Titanic* dataset
  - Decision tree
    - Example: *Titanic* dataset revisited

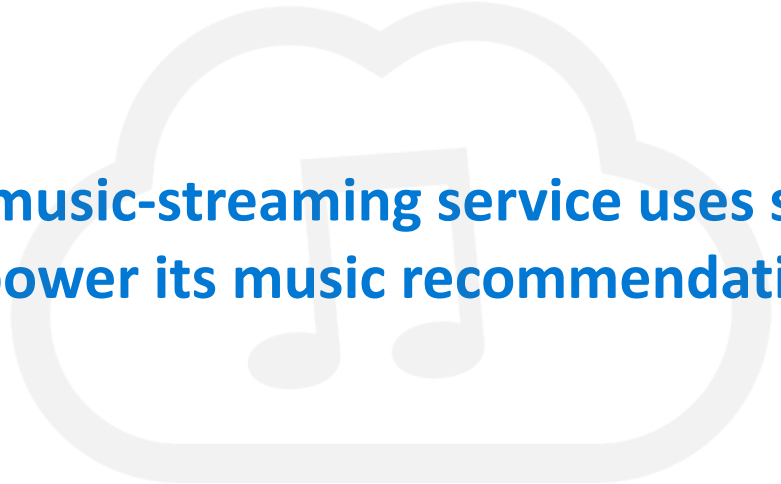
Assumptions for section 6

- Basic familiarity with Python and pandas for data (sections 1, 2, and 5)

Section 6 overview

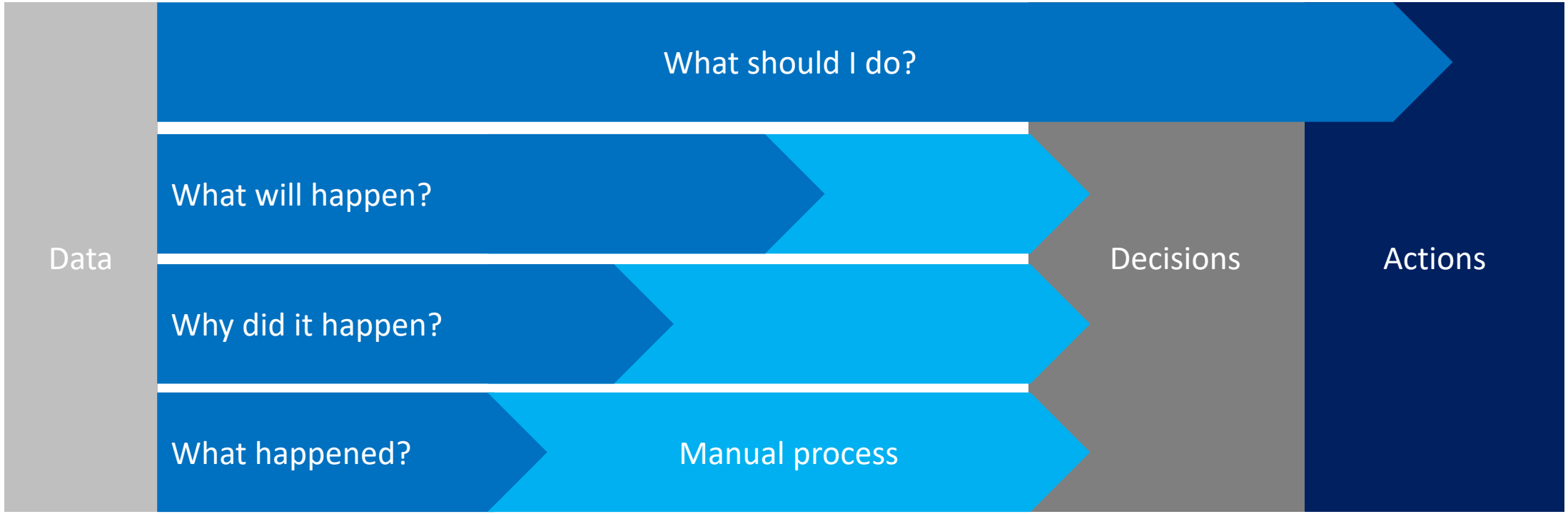
Why do machine learning?

- Where the “magic” of data science happens—it’s fun!
- Automates data extrapolation and classification
- Produces remarkable results in science, medicine, and business



A leading music-streaming service uses scikit-learn to power its music recommendations

Predictive & Preemptive Analytics



”Machine learning is the science of getting computers to act without being explicitly programmed, but instead letting them learn a few tricks on their own.”

-- Dr. Danko NIkolic

Demystifying machine learning

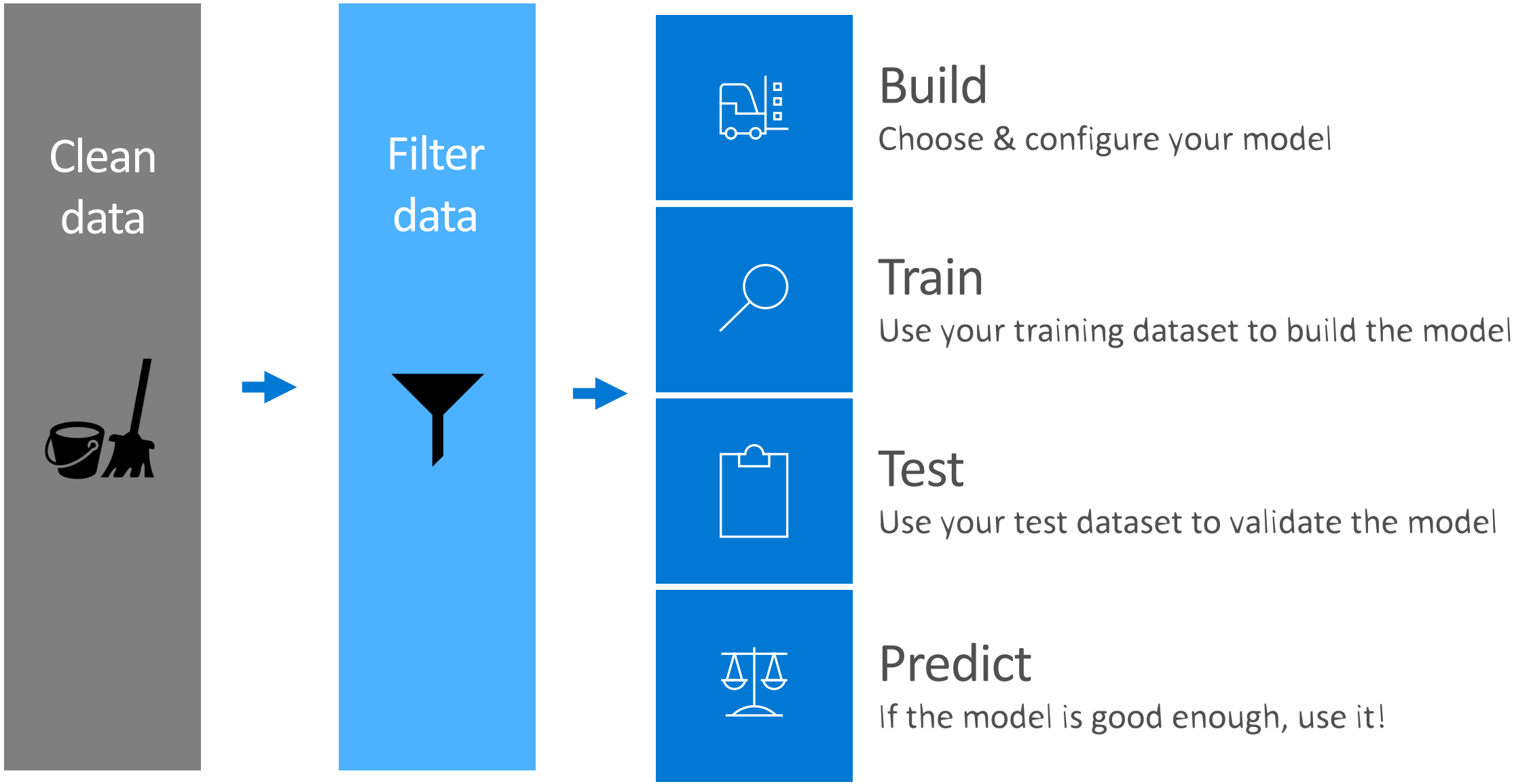
Age	Income	Education	Gender	Housing
61	$65,000	Moderate	F	Own
42	$72,000	High	F	Rent
18	$25,000	Moderate	M	Other
22	$36,000	Low	M	Rent
31	$52,000	High	M	?

Training data

What we know
"features", "attributes"

Thing to predict
"class", "label"

How to create a predictive ML model



Types of machine learning

Supervised Learning

Ground truth observations to:

- Learn from a set of training data
- Teach the algorithm to learn the mapping

Unsupervised Learning

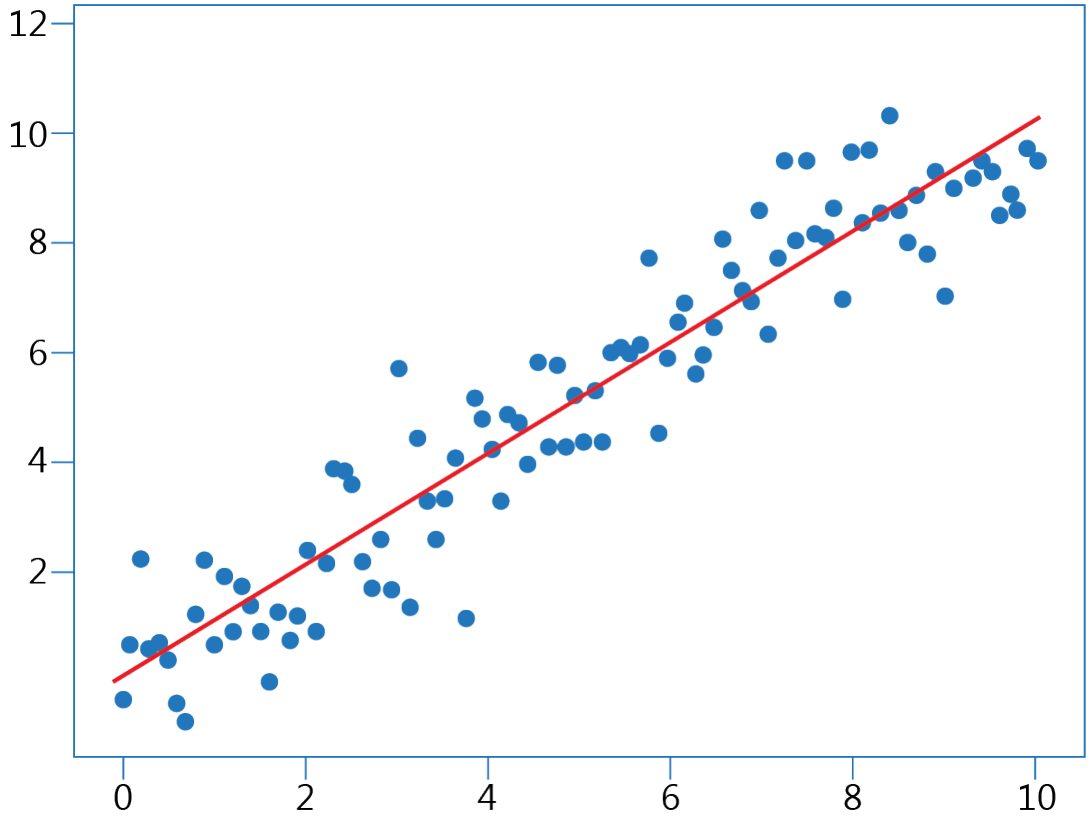
Learning from unlabeled observations

- Discover latent patterns and structure
- And end in of itself:
  - Exploratory data analysis
  - Discovery of hidden patterns

Prediction: linear regression

"Draw a line":

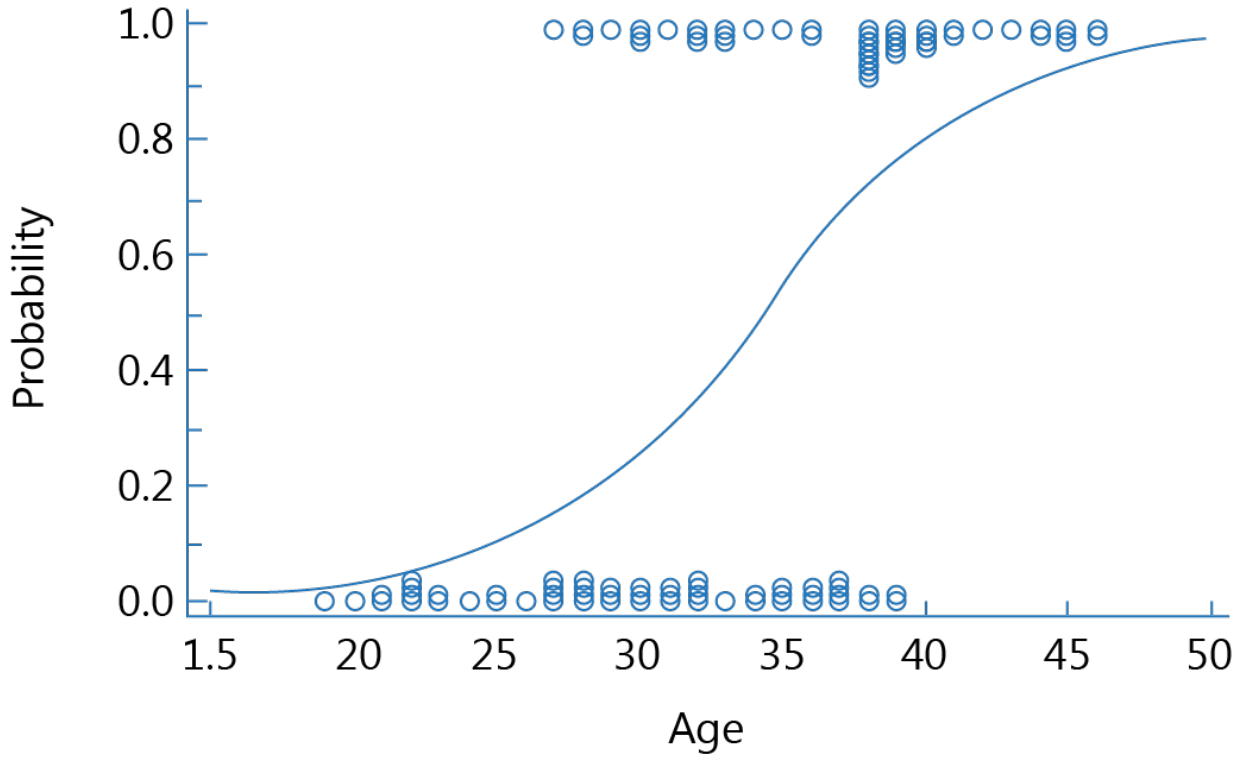
- Extrapolate trends from existing data points
- Example: U.S. Housing dataset



Classification: logistic regression

Binary classification:

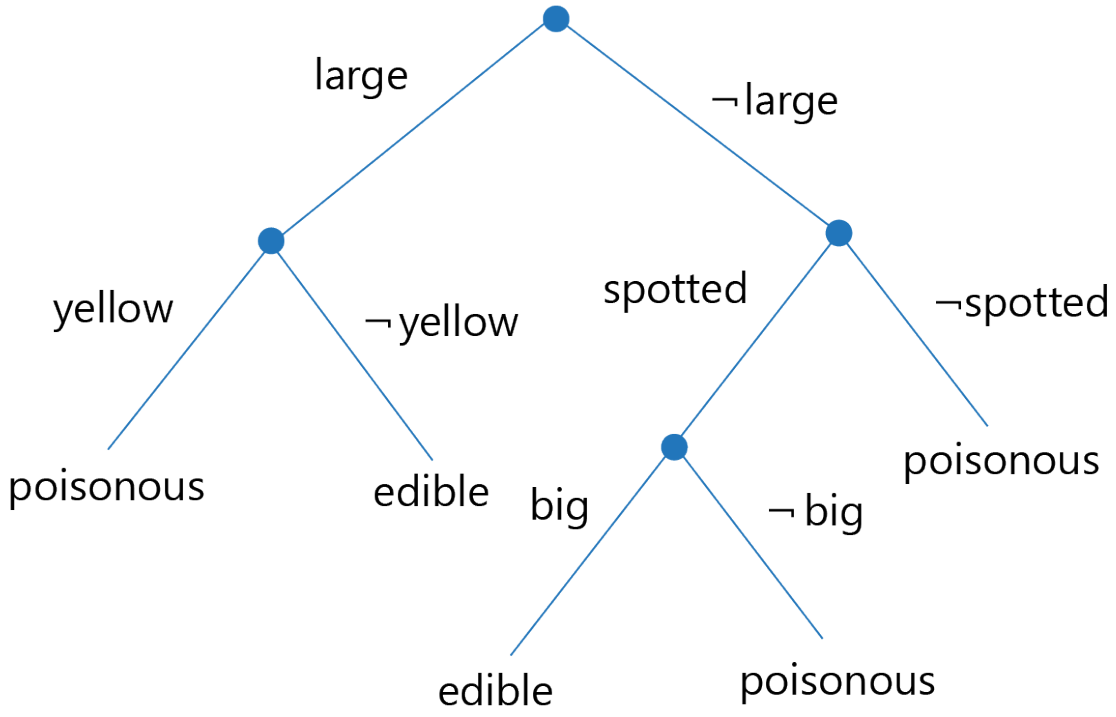
- Fit an S-curve between 0 and 1
- 0 is the false condition
- 1 is true
- Example: *Titanic* dataset



Classification: decision tree

Binary classification:

- True/false questions at each node
- Human-readable model



Application to the capstone project

NOAA Significant Volcanic Eruptions database contains data suitable for ML classification or prediction:

Potential classification

Type
Caldera
Caldera
Caldera
Caldera
Stratovolcano
Stratovolcano
Complex volcano
Complex volcano
Stratovolcano
Stratovolcano
Stratovolcano
Caldera

Potential prediction

VEI
6
7
6
6
6
6
5
6
6
6
6
6

Microsoft

Cloud-based machine learning

Section 7

Section 7 overview

What you will know by the end of section 7

- Microsoft AI Platform overview
- Creating an ML web service with Azure Machine Learning Studio
- Explore high-level finished AI with Azure Cognitive Services
  - Computer Vision API
  - Text Analytics API

Assumptions for section 7

- Basic familiarity with Python and ML (sections 1, 2, and 6)

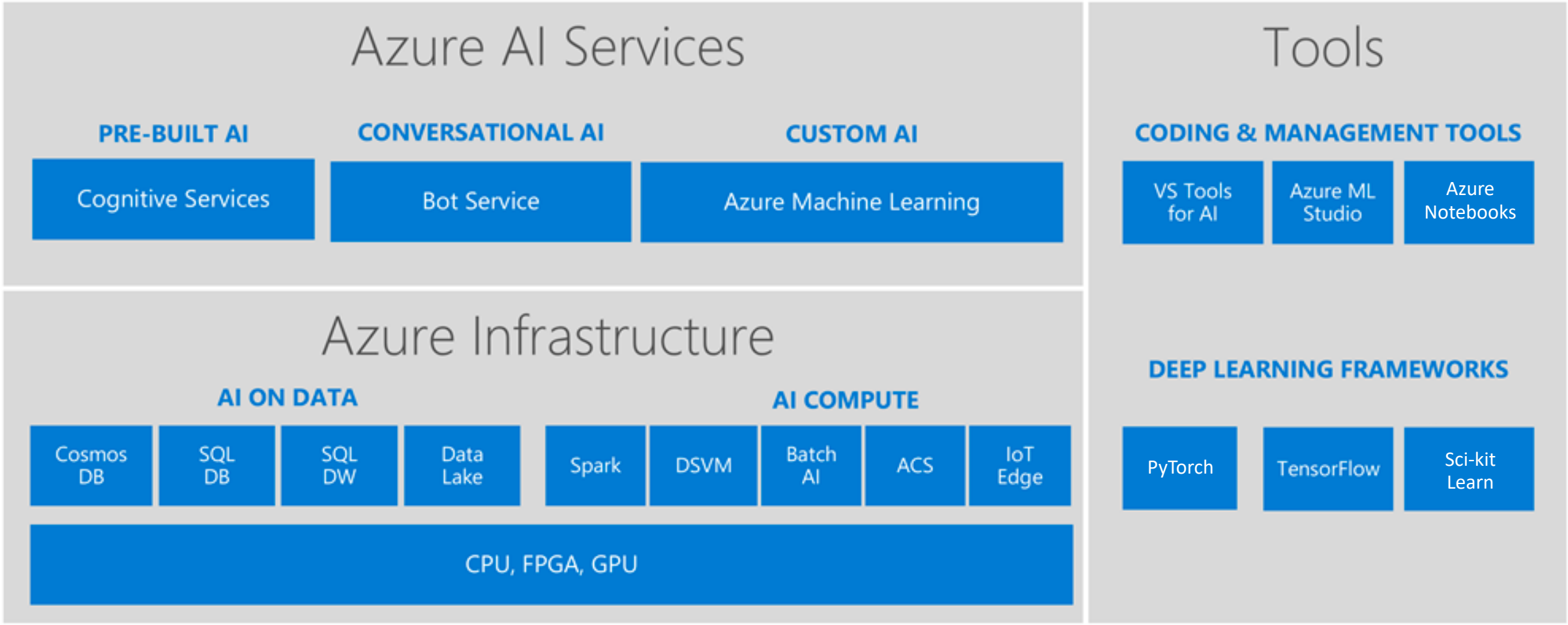
Section 7 overview

Why use cloud platforms for ML?

- Quickly and affordably access sophisticated algorithms
- Deploy trained and tuned models for repeated use online and on edge devices

Cincinnati Children's Hospital uses Azure Cognitive Services Language Understanding for natural-language interaction

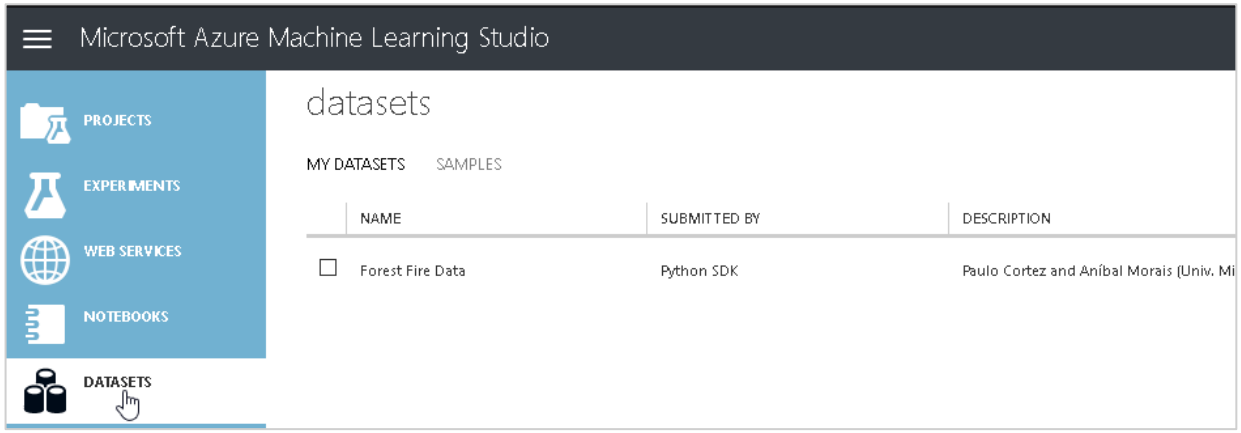
Microsoft AI Platform



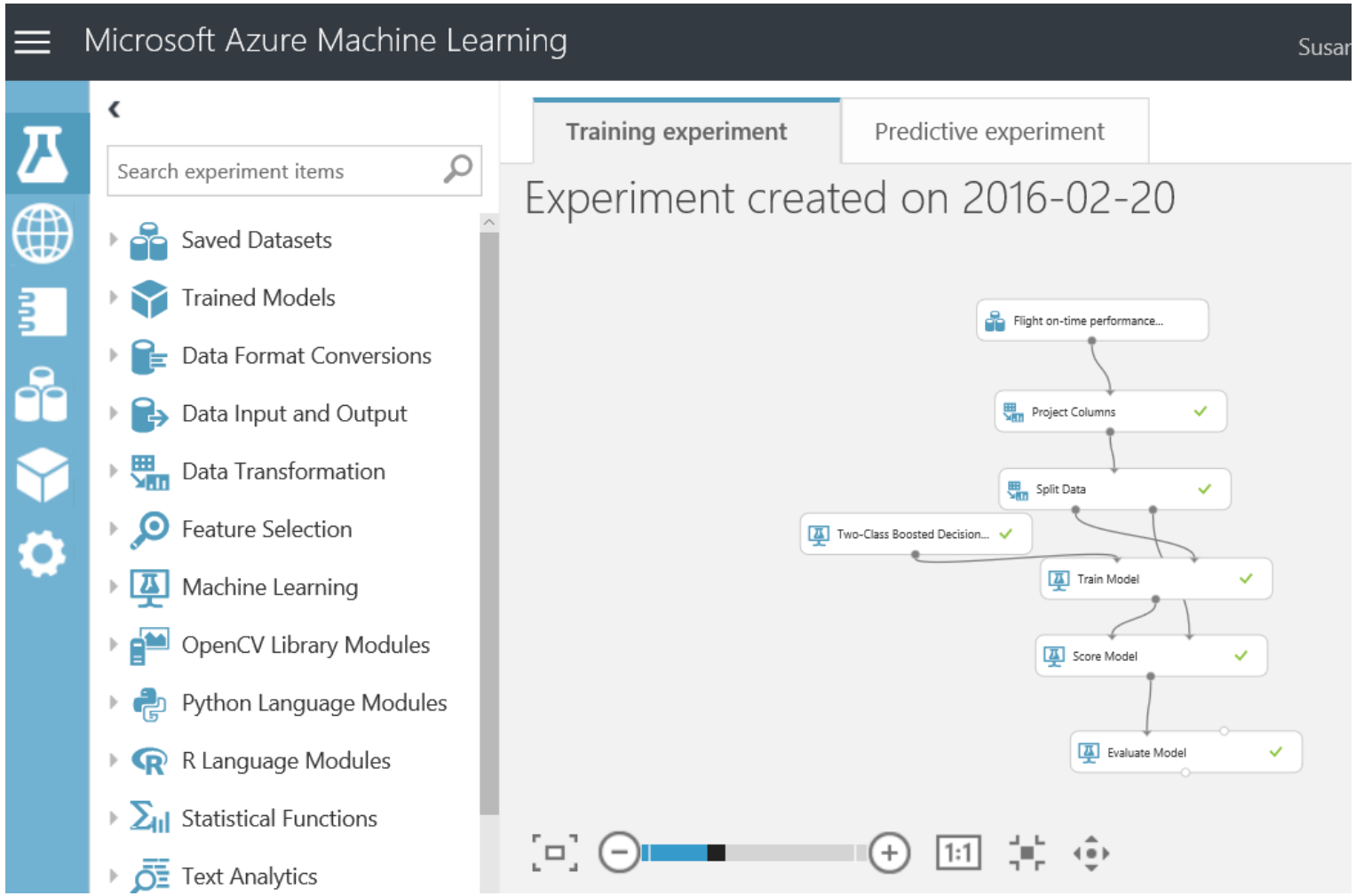
https://docs.microsoft.com/learn/modules/choose-data-science-option-in-azure/2-ml-options-on-azure

Azure Machine Learning Studio

1. Create and connect to an Azure Machine Learning Studio workspace
2. Explore source data (Portugal forest fire data)
3. Transfer your data to Azure Machine Learning Studio
4. Create your model
5. Deploy your model as a web service
6. Consume the web service

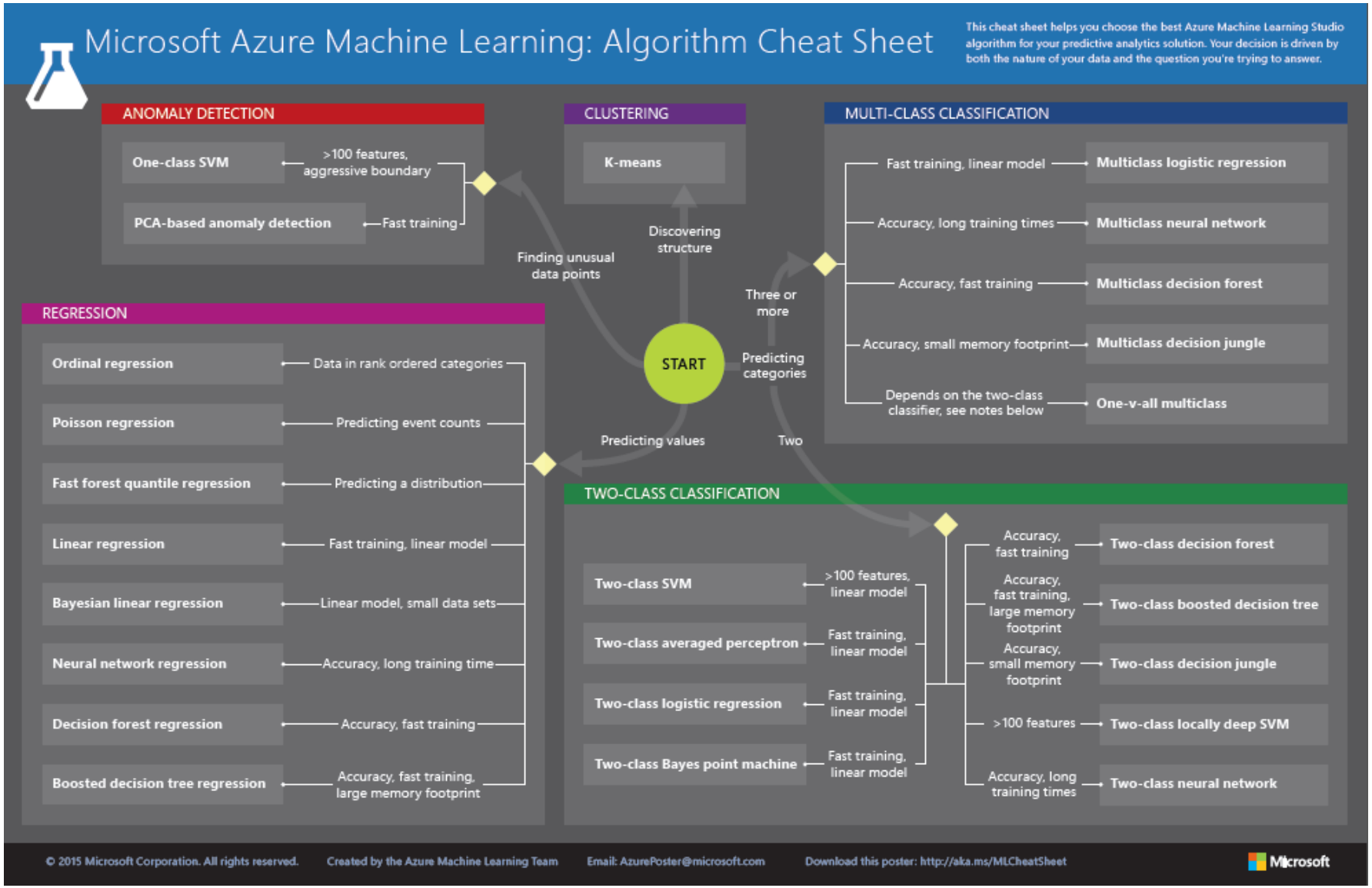


ML Studio is ideal for rapid experimentation…



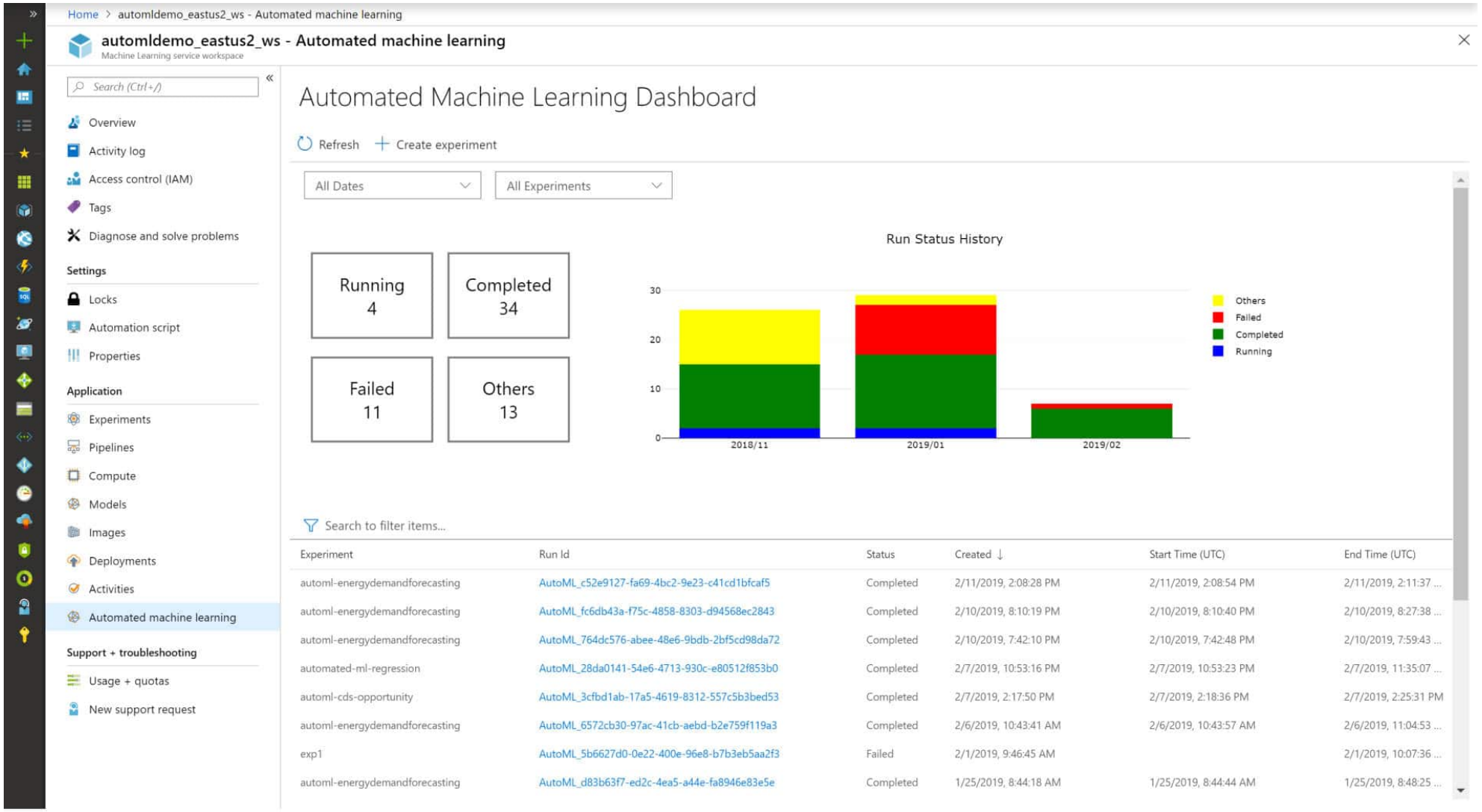
Which algorithm to use depends on the type of prediction you want to make…

Algorithm type	Type of prediction
Classification	Predict a category, e.g. what income range you fall into, which hockey team you cheer, or which political party you prefer
Regression	Predict a value, e.g. someone's income or the price of gas
Anomaly detection	Find anomalies in the data, e.g. credit-card fraud detection

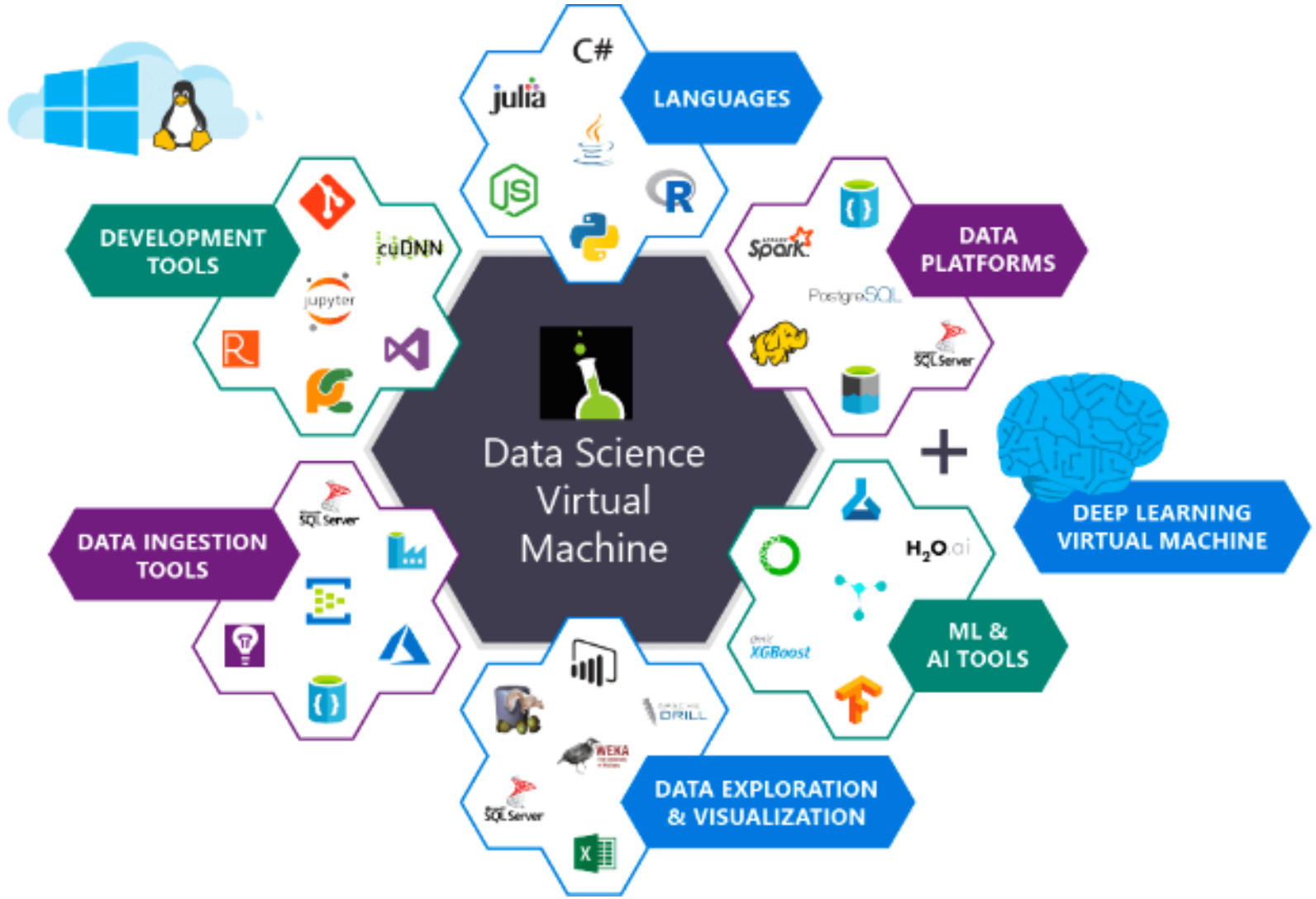


http://aka.ms/MLCheatSheet

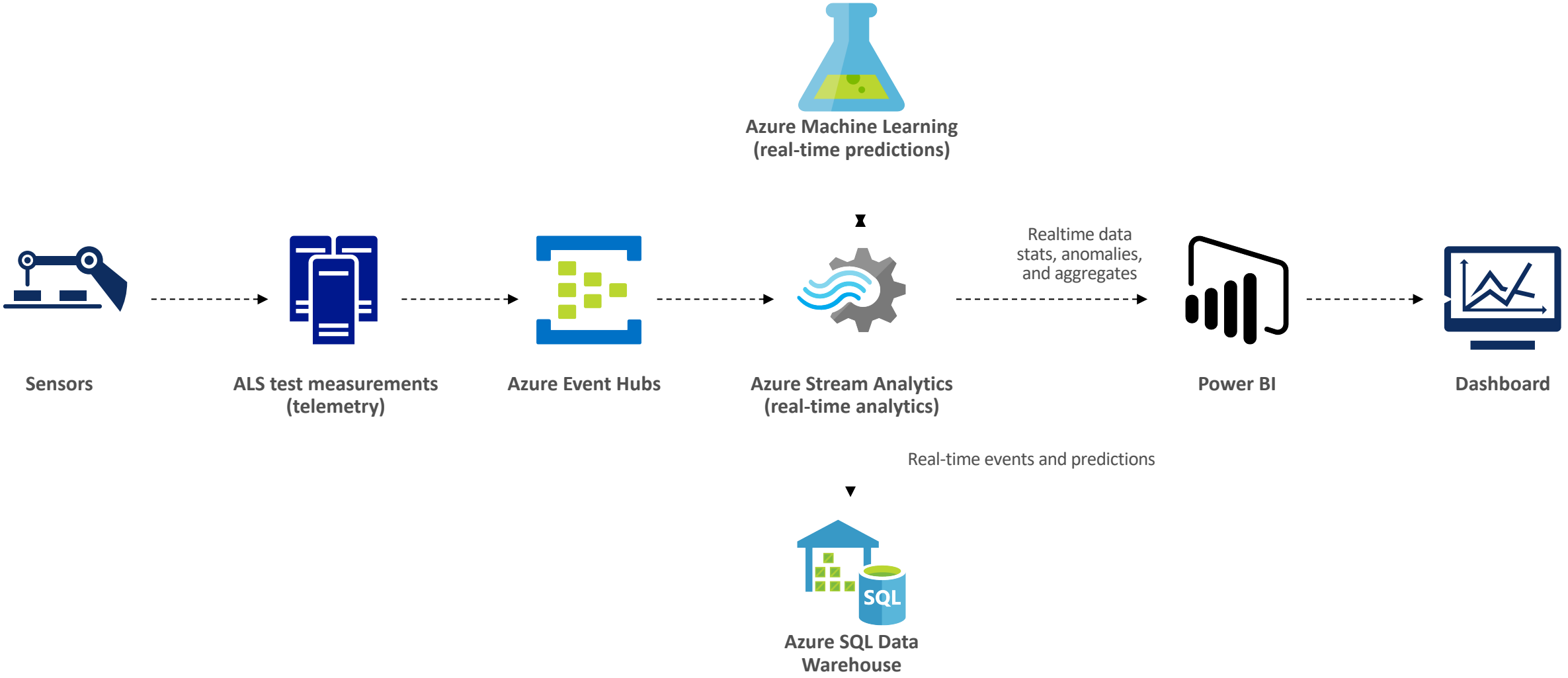
Other Tools: Azure Machine Learning Services



Other Tools: Data Science Virtual Machine



Other tools: E2E Solution Architecture



Azure Cognitive Services are ready-to-go, pre-trained AI models

Decision

Build apps that surface recommendations for informed and efficient decision-making

Vision

From faces to feelings, allow your apps to understand images and videos

Speech

Hear and speak to your users by filtering noise, identifying speakers, and understanding intent

Language

Process text and learn how to recognize what users want

Knowledge

Tap into rich knowledge amassed from the web, academia, or your own data

Search

Access billions of webpages, images, videos, and news articles with the power of Bing APIs

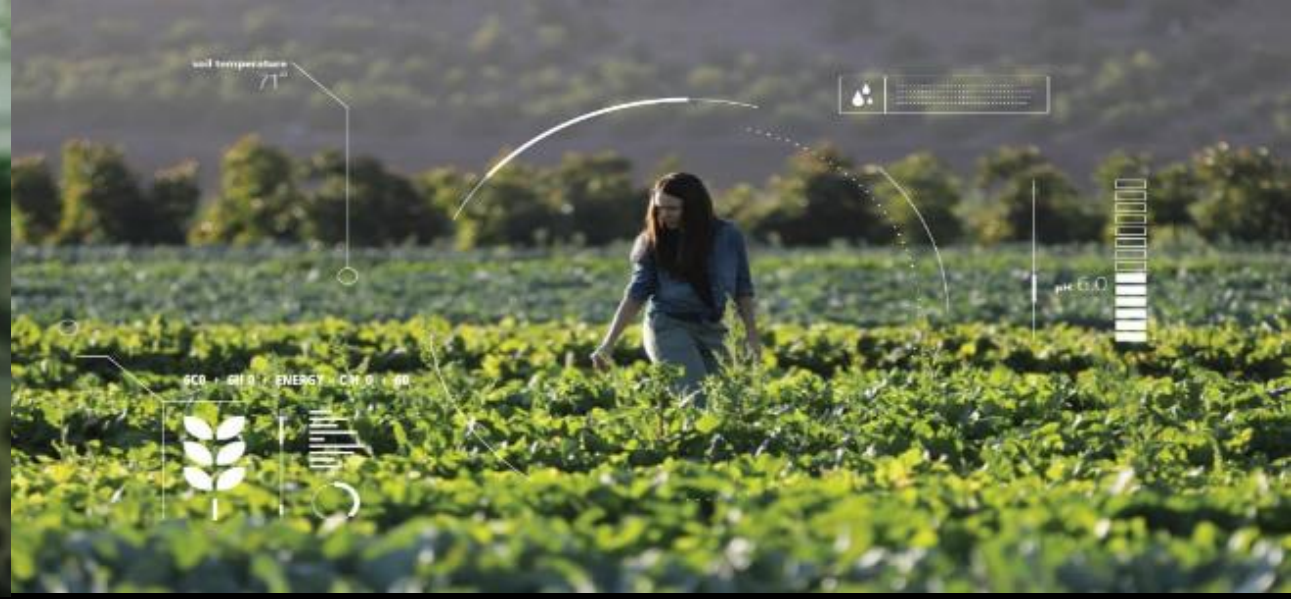
Microsoft

Microsoft’s aim:

Making AI available to everyone

We are pursuing AI to empower every person and every institution ... so that they can go on to solve the most pressing problems of our society and our economy.

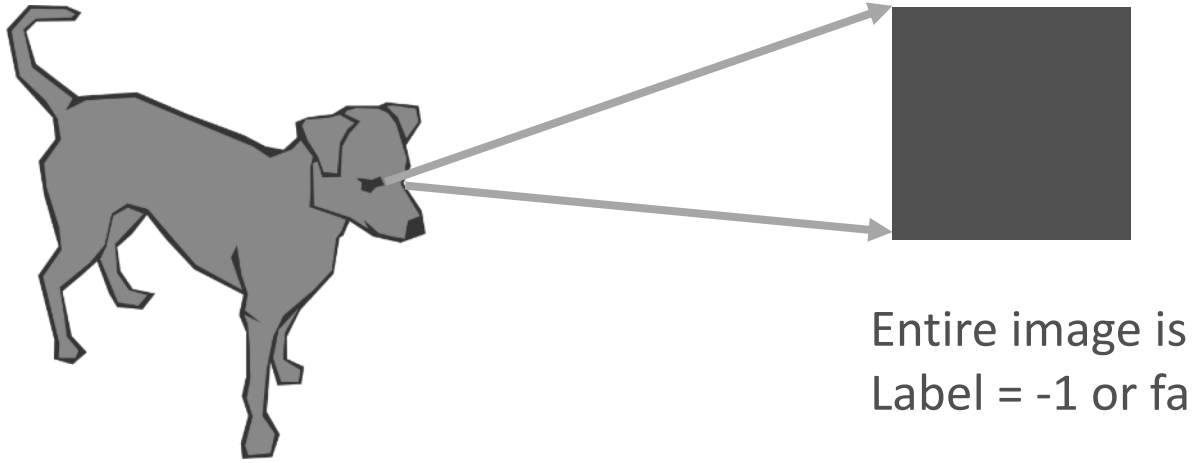
– Satya Nadella, CEO, Microsoft



Microsoft AI for Good

Image Analysis: Feature Vectorization

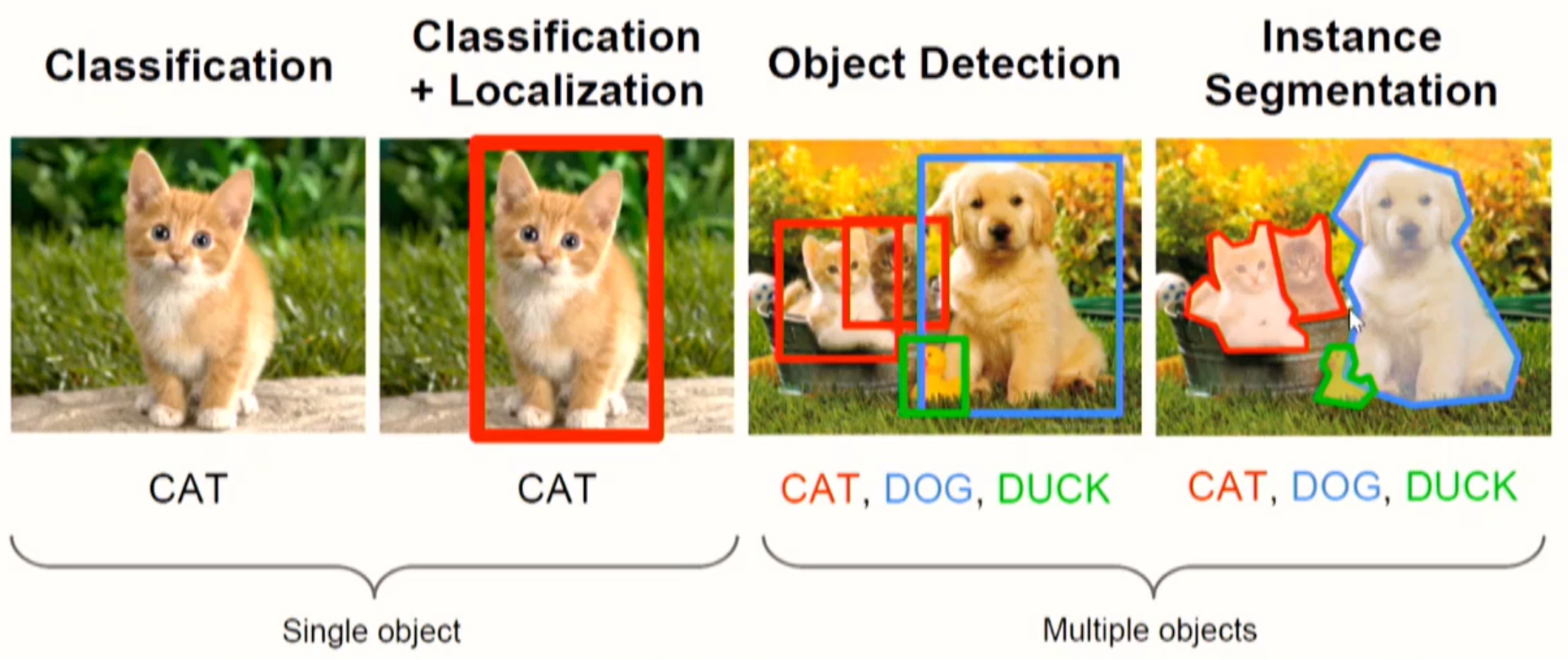
Each observation is represented by a set of numbers.



Each pixel has a RGB value like [1.0, 1.1, 0.8]

Entire image is represented as [1.0, 1.1, 0.8, ……]
Label = -1 or false (i.e., this image is not a chair)

Image Analysis: Identification



Wild Me + Microsoft

Image Analysis: Identify species

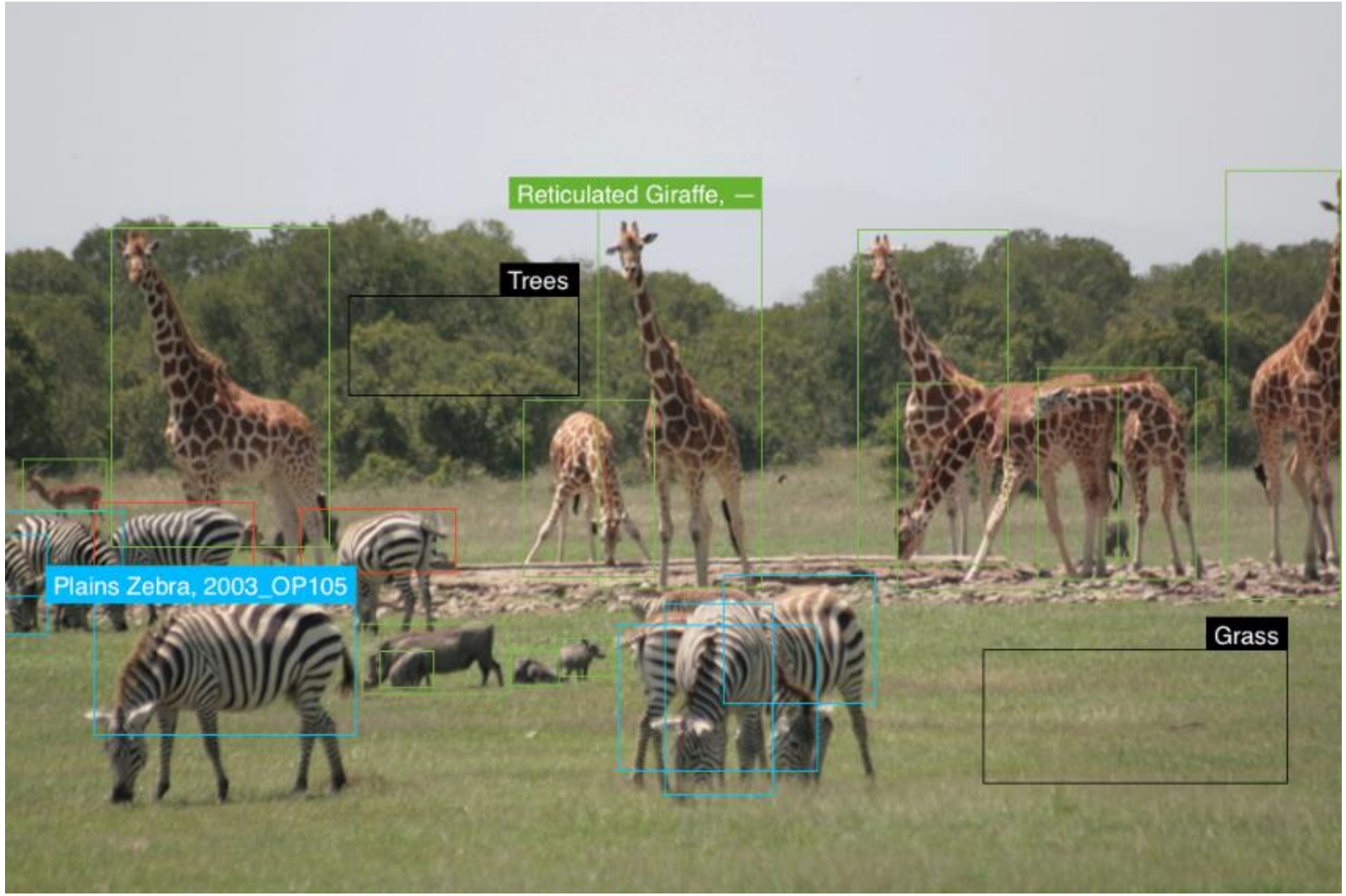
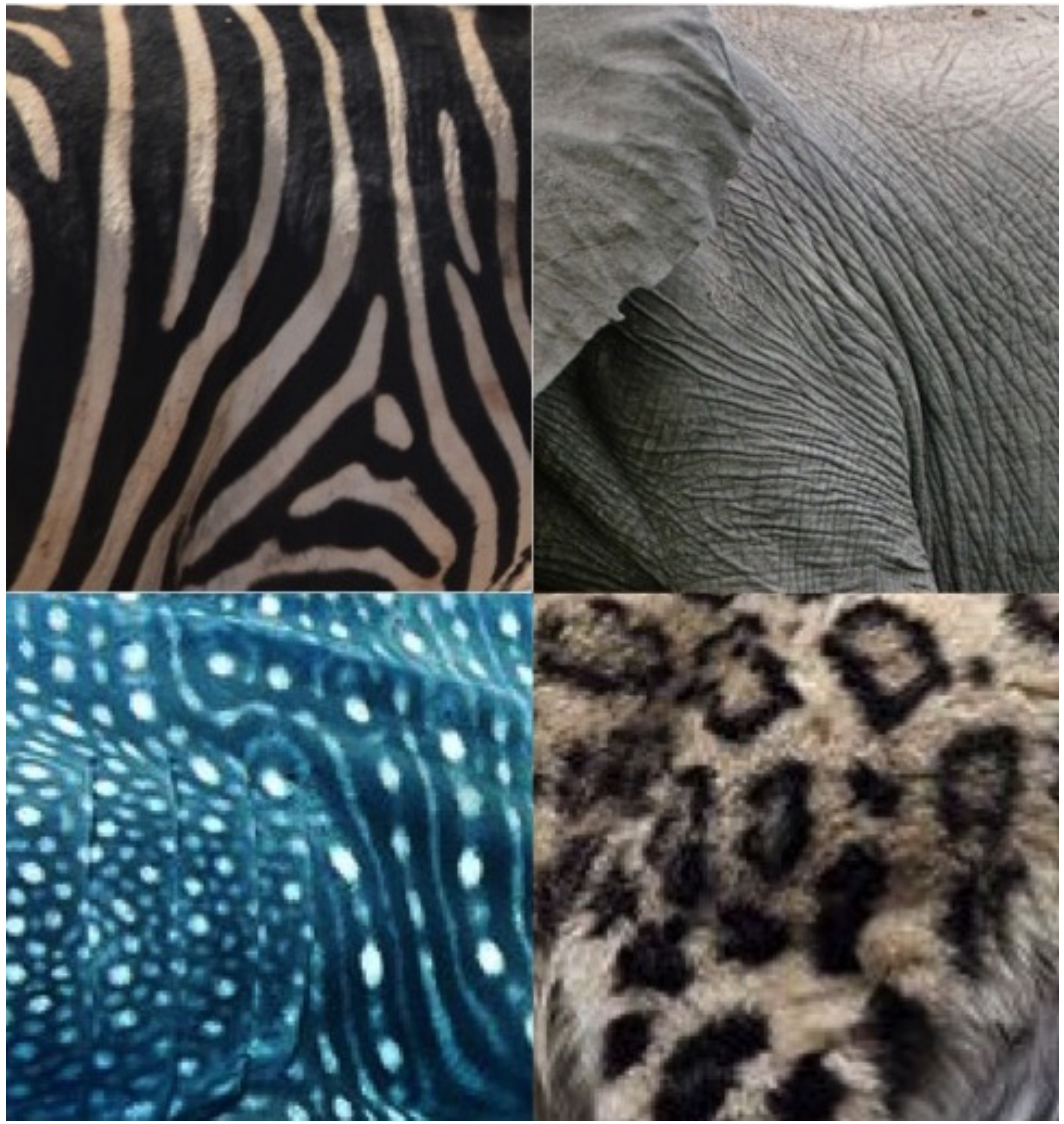
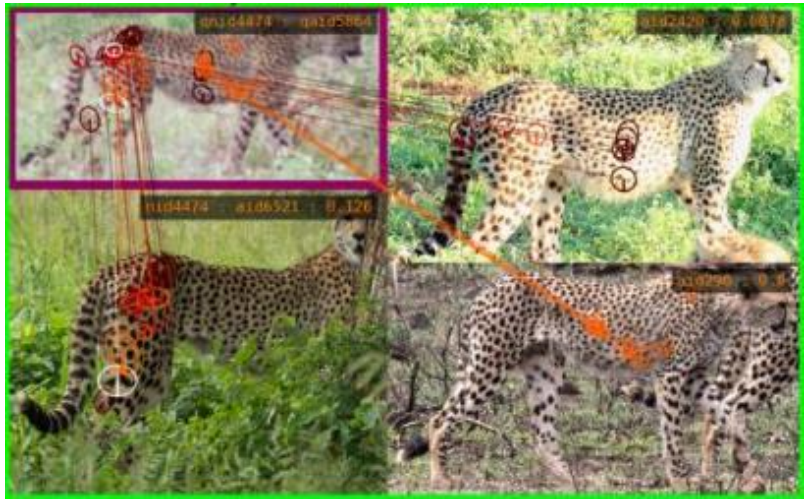
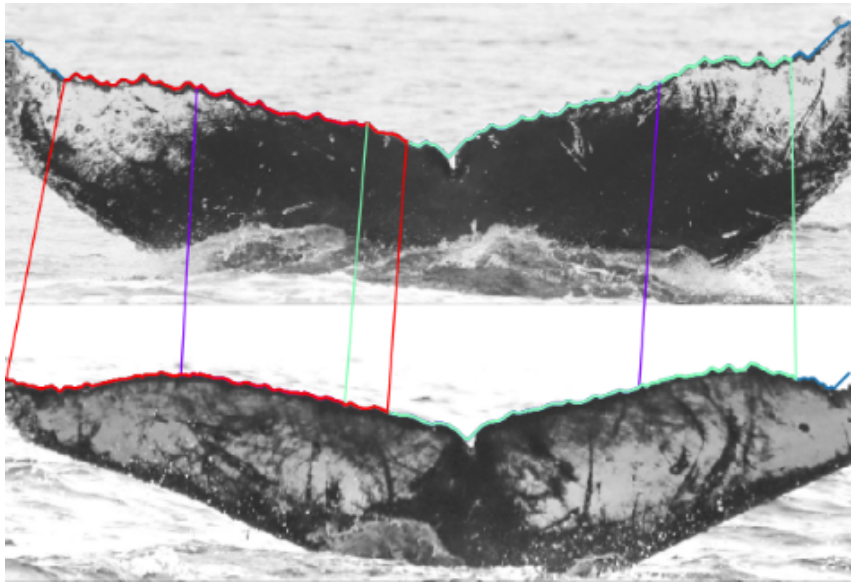


Image Analysis: Identify unique individuals







Detecting Coral Reefs with Scuba Diving Imagery

<input>

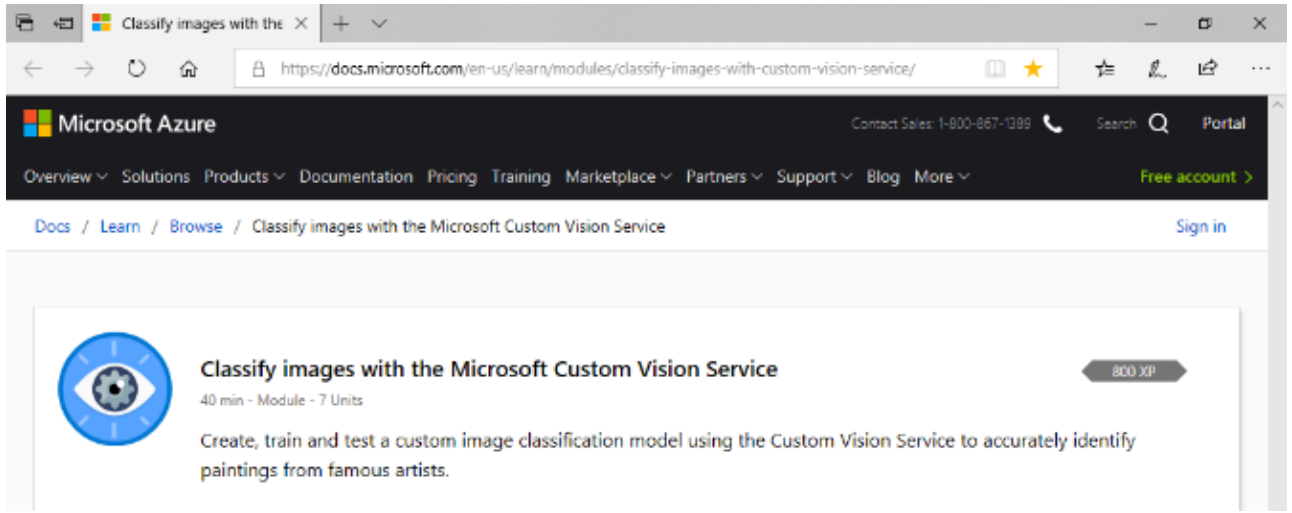
<call Cog Services>

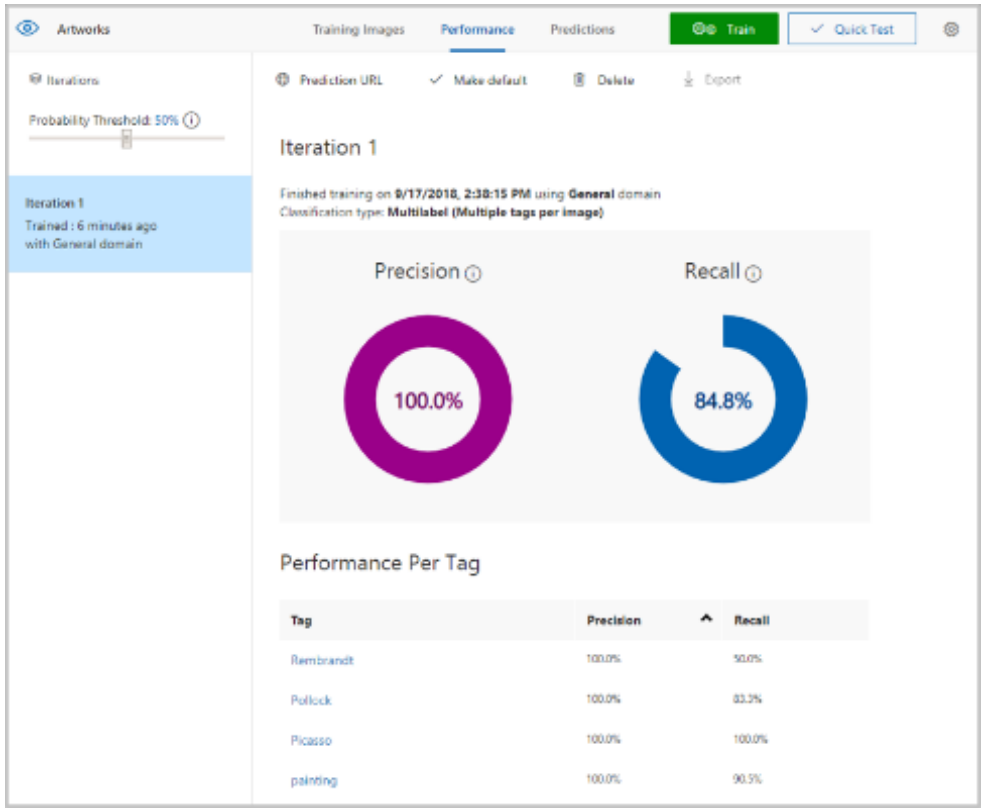
```
In [8]: body = "{'url':'https://timeincsecure-a.akamaihd.net/rtmp_uds/293884104/201703/2681/293884104_5360456295001_5360434467001-vs.jpg?pubId=293884104&videoId=5360434467001'}"

try:
    # Execute the REST API call and get the response.
    conn = http.client.HTTPSConnection('westus2.api.cognitive.microsoft.com')
    conn.request("POST", "/vision/v1.0/analyze?%s" % params, body, headers)
    response = conn.getresponse()
    data = response.read()

    # 'data' contains the JSON data. The following formats the JSON data for display.
    parsed = json.loads(data.decode())
    print ("Response:")
    print (json.dumps(parsed, sort_keys=True, indent=2))
    conn.close()
```

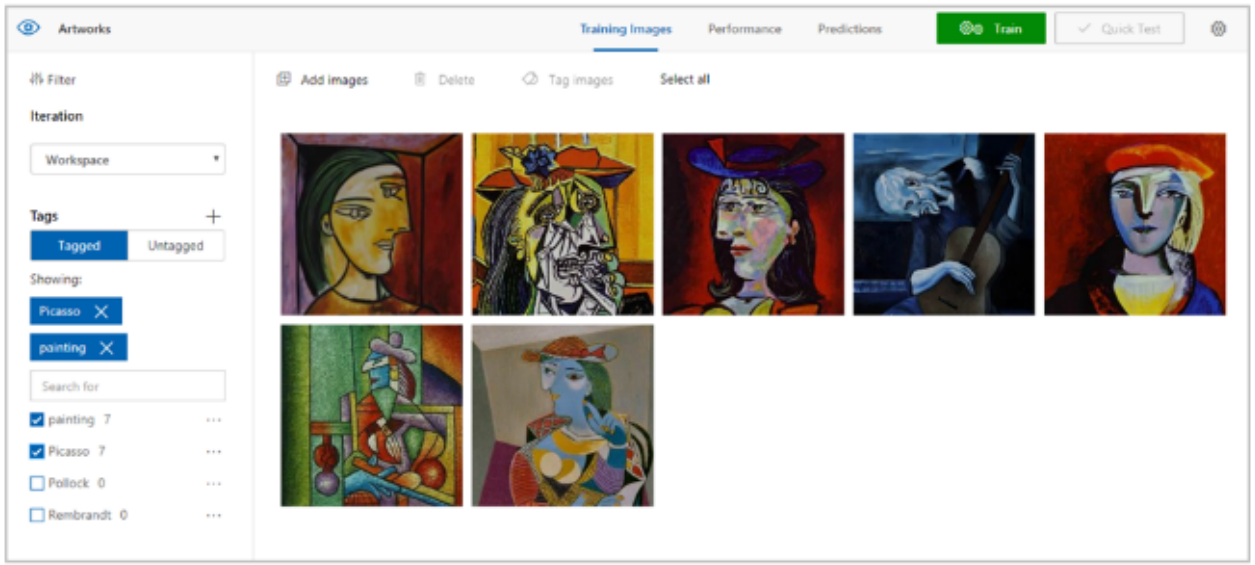
<output.JSON>


```
Response:
{
  "color": {
    "accentColor": "048AC7",
    "dominantColorBackground": "Brown",
    "dominantColorForeground": "Brown",
    "dominantColors": [],
    "isBwImg": false
  },
  "description": {
    "captions": [
      {
        "confidence": 0.2669259815507952,
        "text": "a group of colorful underwater"
      }
    ],
"tags": [
  "nature",
  "covered",
  "colorful",
  "orange",
  "reef",
  "lot",
  "colored",
  "many",
  "sitting",
  "table",
  "surrounded",
  "painted",
  "colors",
  "fire",
  "old",
  "field",
  "water",
  "large",
  "group",
  "underwater",
  "blue",
  "display",
  "room",
  "hydrant",
  "umbrella",
  "people",
  "standing",
  "street",
  "white"
```

Classify images with Custom Vision

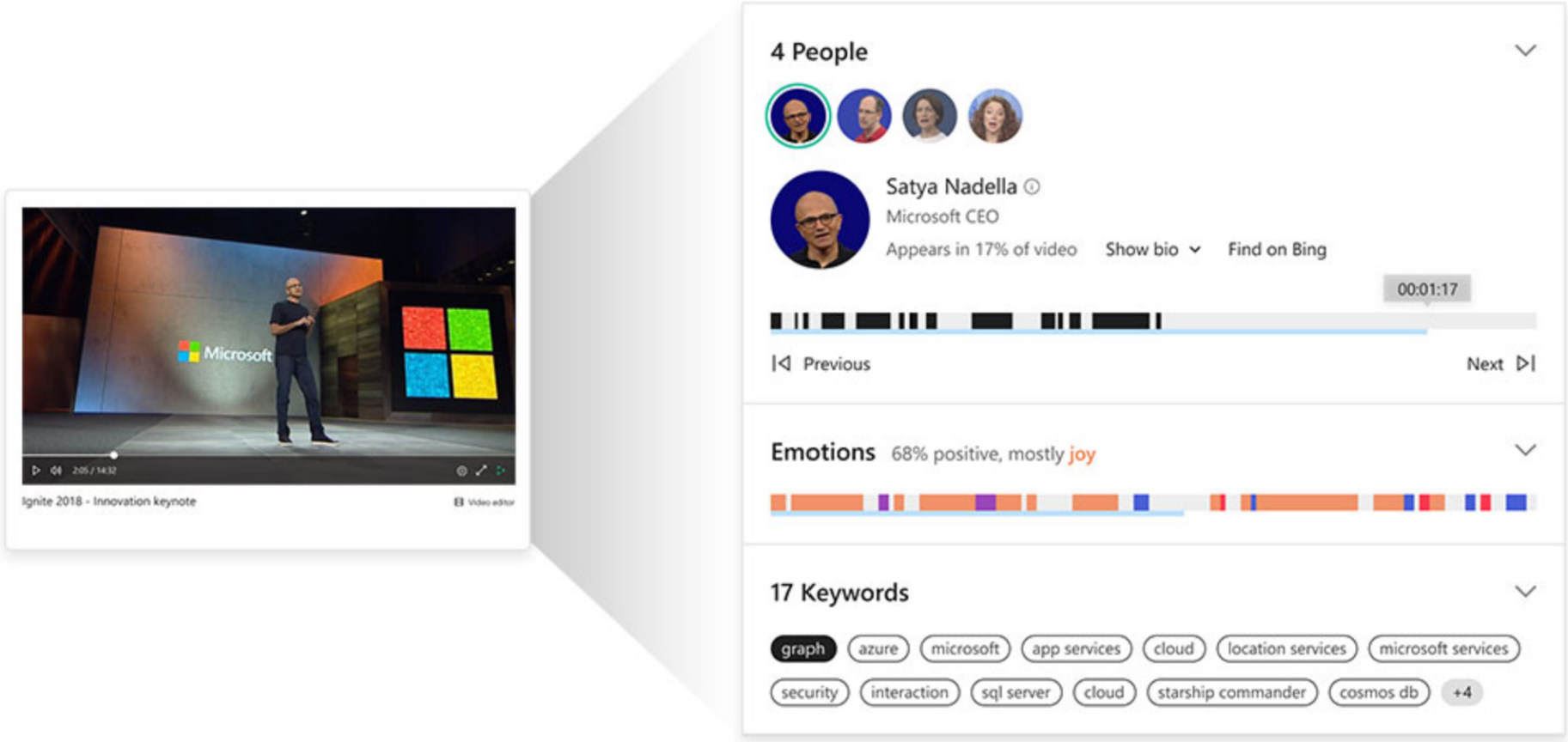






docs.microsoft.com/learn/modules/classify-images-with-custom-vision-service

Unlock video insights with VideoIndexer



videoindexer.ai

Labs today: Analyze landmarks with Computer Vision API

- Analyze images
- Recognize celebrities and landmarks
- Read text in images
- Read handwritten text from images

```
In [5]: domain = "landmarks"
url = "https://images.pexels.com/photos/726484/pexels-photo-726484.jpeg"
language = "en"
max_descriptions = 3

analysis = client.describe_image(url, max_descriptions, language)

for caption in analysis.captions:
    print(caption.text)
    print(caption.confidence)
```

```
a bridge over a body of water
0.6252800581978962
a bird standing on a bridge
0.31510386901709425
a bridge over some water
0.31410386901709425
```

Labs today: Extract sentiment with Text Analytics API

- Sentiment analysis
- Key phrases
- Language detection
- Named entity recognition

Text	Detected languages(scores)
This is a document written in English.	English(1.0)
Este es un document escrito en Español.	Spanish(1.0)
这是一个用中文写的文件	Chinese_Simplified(1.0)
Ez egy magyar nyelvű dokumentum.	Hungarian(1.0)
Dette er et dokument skrevet på dansk.	Norwegian(1.0)
これは日本語で書かれた文書です。	Japanese(1.0)

```
documents = {'documents' : [
  {'id': '1', 'language': 'en', 'text': 'I had a wonderful experience! The rooms were wonderful and the staff was helpful.'},
  {'id': '2', 'language': 'en', 'text': 'I had a terrible time at the hotel. The staff was rude and the food was awful.'},
  {'id': '3', 'language': 'es', 'text': 'Los caminos que llevan hasta Monte Rainier son espectaculares y hermosos.'},
  {'id': '4', 'language': 'es', 'text': 'La carretera estaba atascada. Había mucho tráfico el día de ayer.'}
]}
```

```
{'documents': [{'id': '1', 'score': 0.9708490371704102},
               {'id': '2', 'score': 0.0019068121910095215},
               {'id': '3', 'score': 0.7456425428390503},
               {'id': '4', 'score': 0.334433376789093}],
 'errors': []}
```

Microsoft

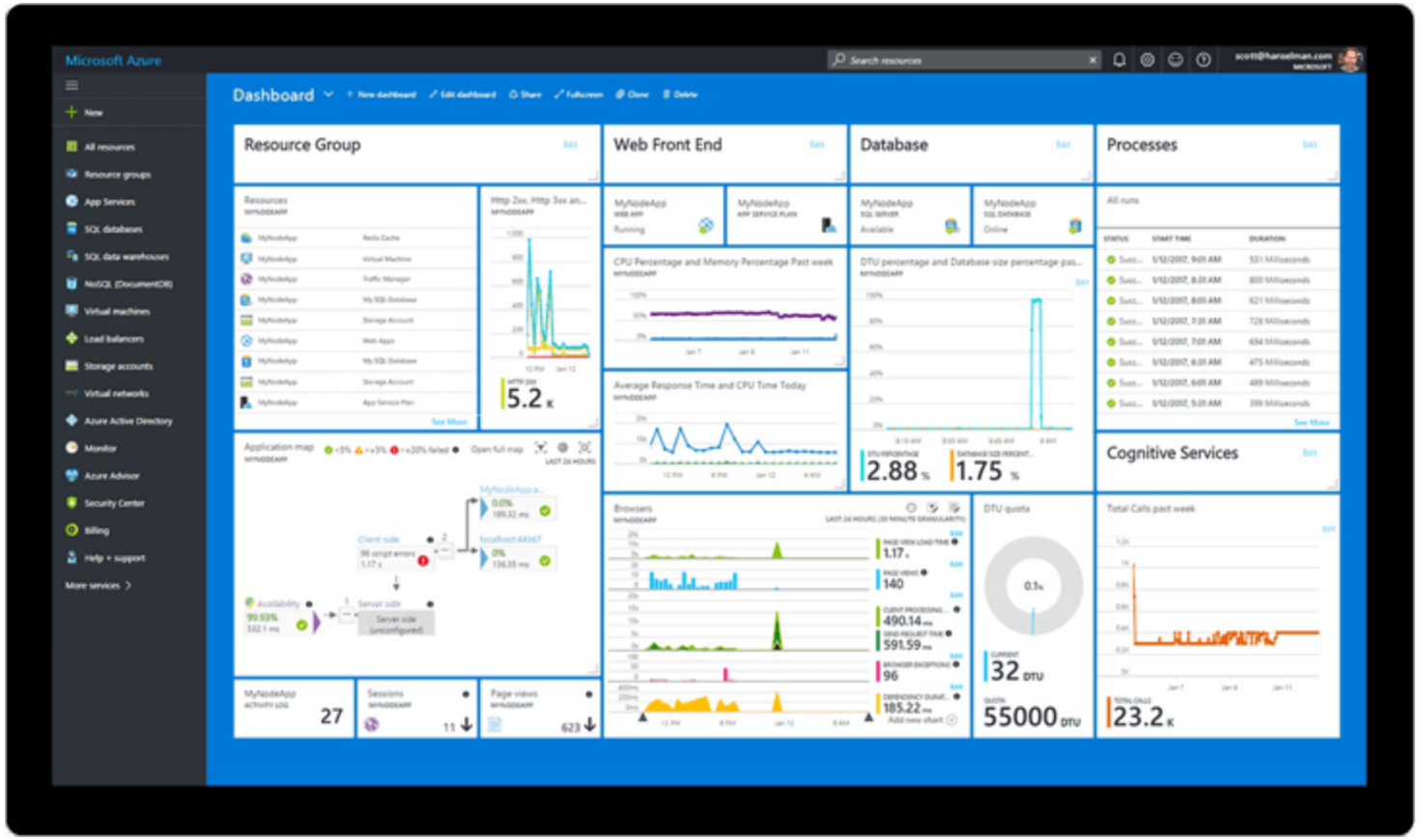
Capstone project

Section 8

Capstone project

- Dataset: NOAA Significant Volcanic Eruption database
- Objectives
  - Build a basic ML solution
- Tasks
  - Identify requirements for success
  - Identify possible risks in the data if this were a real-world scenario
  - Prepare the data
  - Select features (variables)
  - Split the data between training and testing
  - Choose algorithms

Free Azure for your workshop capstone and beyond



azure.microsoft.com/free/

azure.microsoft.com/free/students/

Share Your Capstone on GitHub

- Visit https://aka.ms/ReactorWorkshops
- Open the Data Science -> Track I content
- Modify the README
- Create PR (Pull Request)
- Add Sarah as a Reviewer

Further Learning

- Microsoft Learn
  - https://docs.microsoft.com/en-us/learn/paths/intro-to-ml-with-python/
  - https://docs.microsoft.com/en-us/learn/paths/build-ai-solutions-with-azure-ml-service/
  - https://docs.microsoft.com/en-us/learn/paths/get-started-with-azure-dsvm/
  - https://docs.microsoft.com/en-us/learn/paths/explore-data-science-tools-in-azure/
  - https://docs.microsoft.com/en-us/learn/paths/azure-fundamentals/

Further Learning

- Introduction to Data Science Track II
  - Joining Datasets
  - Principal Component Analysis (PCA)
  - Machine Learning Accuracy
  - K-Means Clustering
  - Naïve Bayes
  - Linear Regression

Data Science Certification

https://www.microsoft.com/en-us/learning/azure-data-scientist.aspx

Skills measured

Define and prepare the development environment

Prepare data for modeling

Perform feature engineering

Develop models

Define and prepare the development environment

Select development environment

- assess the deployment environment constraints
- analyze and recommend tools that meet system requirements
- select the development environment

Set up development environment

- create an Azure data science environment
- configure data science work environments

Quantify the business problem

- define technical success metrics
- quantify risks

Data Science Certification

https://www.microsoft.com/en-us/learning/azure-data-scientist.aspx

Skills measured

Define and prepare the development environment

Prepare data for modeling

Perform feature engineering

Develop models

Prepare data for modeling

Transform data into usable datasets

- develop data structures
- design a data sampling strategy
- design the data preparation flow

Perform Exploratory Data Analysis (EDA)

- review visual analytics data to discover patterns and determine next steps
- identify anomalies, outliers, and other data inconsistencies
- create descriptive statistics for a dataset

Cleanse and transform data

- resolve anomalies, outliers, and other data inconsistencies
- standardize data formats
- set the granularity for data

Data Science Certification

https://www.microsoft.com/en-us/learning/azure-data-scientist.aspx

Skills measured

Define and prepare the development environment

Prepare data for modeling

Perform feature engineering

Develop models

Perform feature engineering

Perform feature extraction

- perform feature extraction algorithms on numerical data
- perform feature extraction algorithms on non-numerical data
- scale features

Perform feature selection

- define the optimality criteria
- apply feature selection algorithms

Data Science Certification

https://www.microsoft.com/en-us/learning/azure-data-scientist.aspx

Skills measured

Define and prepare the development environment

Prepare data for modeling

Perform feature engineering

Develop models

Develop models

Select an algorithmic approach

- determine appropriate performance metrics
- implement appropriate algorithms
- consider data preparation steps that are specific to the selected algorithms

Split datasets

- determine ideal split based on the nature of the data
- determine number of splits
- determine relative size of splits
- ensure splits are balanced

Train the model

- select early stopping criteria
- tune hyper-parameters

Evaluate model performance

- score models against evaluation metrics
- implement cross-validation
- identify and address overfitting
- identify root cause of performance results

Identify data imbalances

- resample a dataset to impose balance
- adjust performance metric to resolve imbalances
- implement penalization

2-Minute (Anonymous) Survey

aka.ms/IntroToDataScienceSurvey