DataSci 400

lesson 1: intro to data science

Seth Mottaghinejad

today's agenda

- about me and my teaching style
- assignments, quizzes and milestones
- participation and grading
- grading expectations and supplimentary material
- coding environment
- overview of machine learning

about me

- I have been working in data science / analytics for over 10 years
- background in statistics, self-taught programmer
- worked across many industries
- I love teaching and include a lot of **hands-on** work
- on average, we spend about 40% on lecture and 60% on hands-on
- let's take frequent short breaks to fit online format

assignments, quizzes and milestones

- assignments are due by **11:59 PM each Sunday following lecture**
- I will make **no exceptions** about assignment due dates
- quizzes are taken during the lecture and **answered in chat window**
- assignments and milestones are graded by our grader
- milestones are similar to assignment but more project-oriented
- questions about the assignments should be posted on the **discussion board** on Canvas, where I or grader will answer them (or students if they wish)

grading expectations

- use discussion boards for questions on assignment and milestones
- if you have no questions to ask, help by answering them
- it's more important to be on time than perfect:
  - meet the requirements with working code
  - write good comments for your code
  - write good explanations showing your line of thinking

participation and grading

activity	what you need to do	grade
participation	be active in **discussion boards**	16%
quizzes	answer **in chat window** during lecture	22%
assignments	submit by **11:59 PM every Sunday**	22%
milestone 1	due same time as assignment 5	10%
milestone 2	due same time as assignment 8	10%
milestone 3	due one week after assignment 10	20%

break time

coding environment

- install Anaconda (use **Python 3.7**), which installs everything we need
- we will be using browser-based Jupyter notebooks as our Python environment
- basics of Jupyter notebooks:
  - code cells and Markdown cells
  - running and re-running code cells - **order matters!**
  - magics are very useful shortcuts

break time

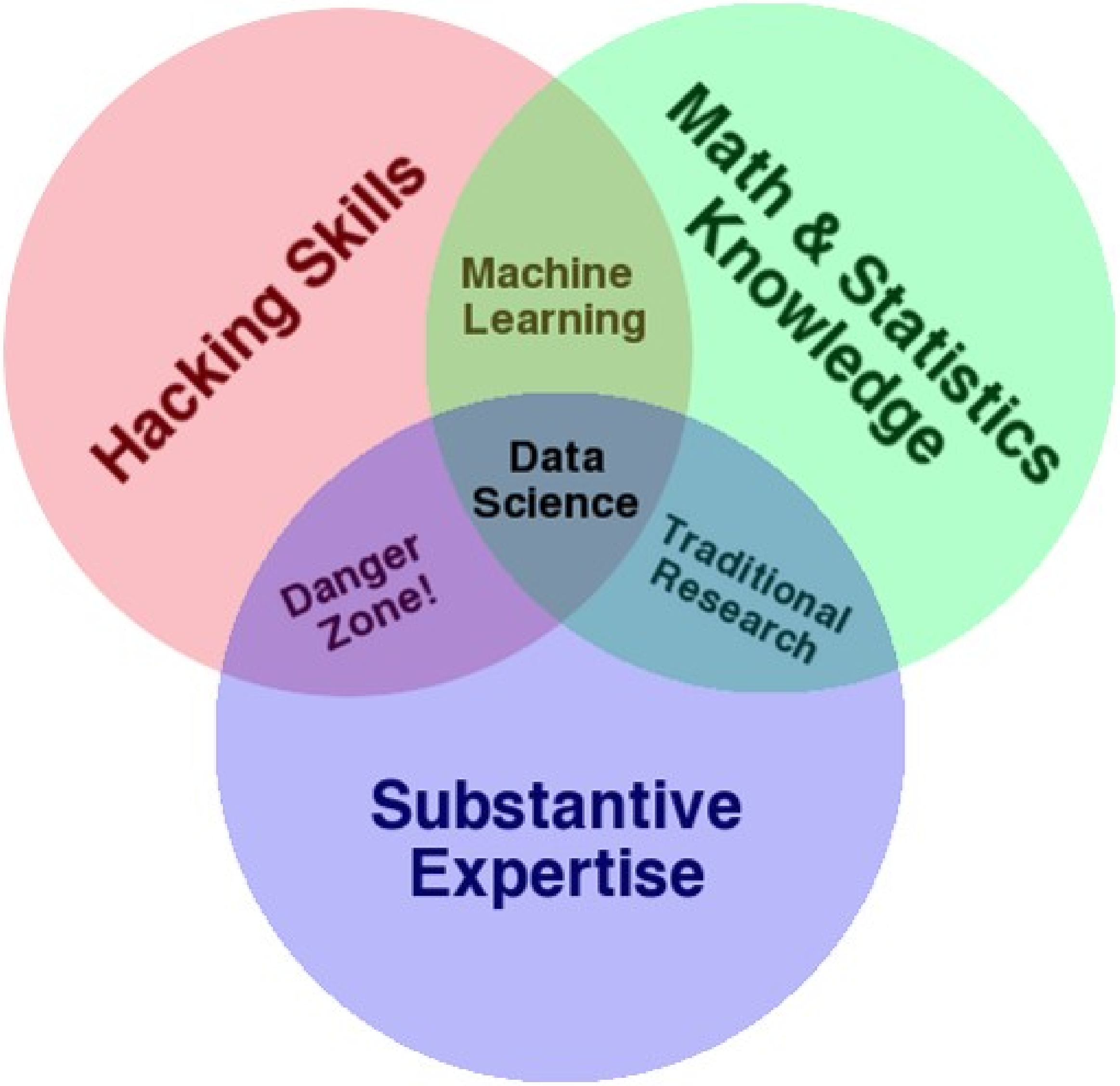
lab time

- what do you think data scientists do?
  - come up with a list of 5-8 activities that you think they do
  - rate those activities by how much of their time is spent on each
  - next to each activity, specify the skills required
    - programming and computer science
    - general business skills: communication, management, etc.
    - academic skills: math and science

three essential skills

- **programming skills**: SQL, Python, R, Scala, Java, ...
- **math & stats**: linear algebra, probability and statistics, data viz, ...
- **domain knowledge**: whatever field you're in

Source: [Drew Conway]



what data scientists do

- create a **data-driven culture** around the business
  - find answers to business questions driven by data and ML
- use **machine learning** to find a good-enough solution:
  - better than a non-data-driven solution
  - **more accurate** than the current data-driven solution
  - **faster / cheaper / simpler** than the current data-driven solution
- make the solution useful by **operationalizing** it
  - who's using it and what do they expect from it?

a data-driven culture

- **define**: what is the business need?
  - **quantitative or qualitative**?
- **measure**: how do we measure the business need?
  - what are the **success metrics**?
- **aquire**: does our data meet the business need?
  - what data do **we have**? and what data do **we need**?
- **explore**: explore the data
  - **quality checks** and **sanity checks**

machine learning

- an algorithm is a self-contained set of **rules or instructions** used to *solve problems*
- **machine learning** is the field of study that gives computers the ability to **learn models** from data (**learn** here means without being **explicitly** programmed)
- the *problems* ML algorithms try to solve are usually
  - prediction: **supervised learning** (by far the **most common**)
  - finding structure in data: **unsupervised learning**
  - ruling over humans: **reinforcement learning** (not covered here)

lab time

scientists use **mathematical models** or **statistical models** to describe the world, i.e. **equations with variables**

- but we all use **models** in our head, and call them **clichés**, **prejudices**, **maxims**: they are usually wrong, but also be useful
- can you express the following two **models** more rigirously (in terms of data and equations):
  - you are the average of your five closest friends
  - early bird gets the worm

supervised learning simplified example

let's look at credit rating use case

- **data we have**: age, income, credit score (300-800)
- **algorithm choice**: linear regression

  `a + b * age + c * income + some error = credit score`
- during **training**, the algorithm **learns** `a`, `b`, and `c`

  `5.8 + 3.9 * age + 1.98 * income + error = credit score`
- during **scoring**, we apply it to get predictions

  `5.8 + 3.9 * age + 1.98 * income = predicted credit score`

lots of ways of doing it

- should we use age **buckets** instead of age itself?
- should we have a special way of dealing with **extreme values**?
- should we obtain **additional data**, such as economic sectors?
- should we predict raw credit score or just buckets (`300-500`, `500-600`, `600-700` etc.)
- should we try a different algorithm?
  - a more **simple** one that is more explainable?
  - a more **complex** one that has higher accuracy?

lab time

return to the simplified credit rating example from earlier, and try to guess what **advantages and disadvantages** the following changes would have on the underlying **model**

- say we use age buckets (**categorical**) instead of age (**numeric**)
- say we use a more complex model with **higher accuracy** but **harder to explain** how it makes predictions
- say we obtain **external data** and use it in our model, such as information about your LinkedIn social network

break time

different choices result in different models

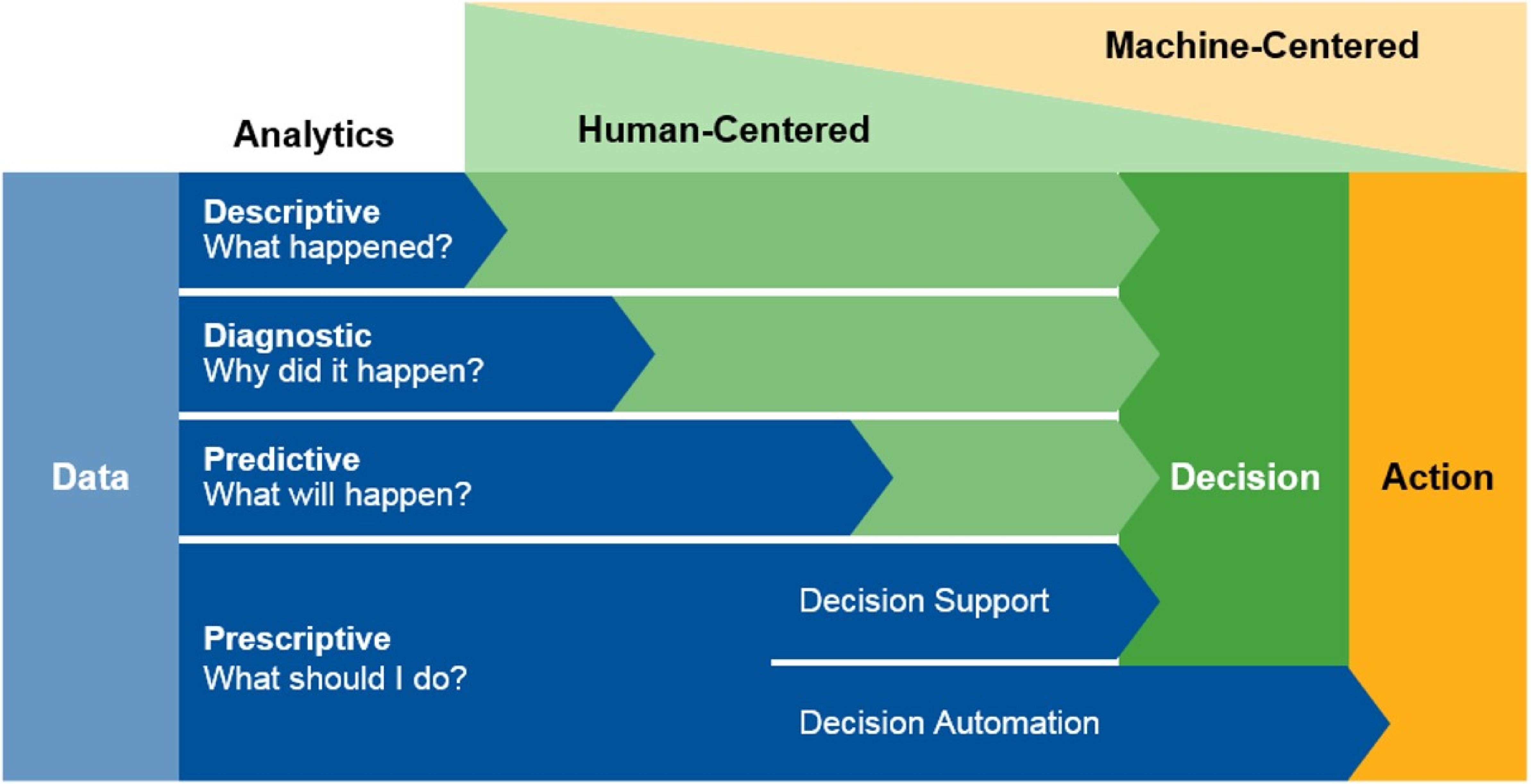
- every model makes a different assumption about the nature of the relationship between our variables
    - George Box: **"all models are wrong, but some are useful."**
- be aware of **trade-off between explainability and accuracy**
    - in the hard sciences, explainability is very important: Okam's razor: **"the simplest explanation is usually the correct one."**
    - in the soft sciences, accuracy is usually emphasized over explainability

machine learning: recap

- once we gather the data to meet the business requirements, we can begin training models
- modeling is **part art and part science**, and includes
  - **pre-processing**: set up **data pipeline** to clean and prepare data for machine learning
  - **feature engineering**: create new features from existing features in the data, which are more relevant to the problem at hand
  - **model selection**: explore and find the "right" model based on business assumptions, explainability vs accuracy, etc.

operationalization

- once we have a trained model, we need to make it useful to the **target audience**, not just the data scientist
- we need to think about this from the get-go, when collecting business requirements
  - **operationalization:** how to deploy model to production?
  - **model consumption:** who is using model and how?
  - **business validation**: how do we know if the model is working well in production?



Source: Gartner (October 2016)

the end