Data Science Primer

Learning Objectives

After completing this lecture, you will be able to:-

- Identify the steps and roles in the data science workflow
- Identify common data science misconceptions
- Describe data sources and types.
- Recognize situations where more data samples or features are needed.
- Describe different data preprocessing methods.
- Identify challenges when working with data.

Data Science Workflow

Problem Statement	What problem are you trying to solve?
Data Collection	What data do you need to solve it?
Data Exploration & Preprocessing	How should you clean your data so your model can use it?
Modeling	Build a model to solve your problem?
Validation	Did I solve the problem?
Decision Making & Deployment	Communicate to stakeholders or put into production?

Problem Statement

What problem are you trying to solve?

- Data scientists first need to identify the problem to solve
- Knowledge of the business is needed to identify impactful opportunities
- Technical knowledge is needed to ask the right questions, and to know what is possible

Data Collection

What data do you need to solve your problem?

- The data required to solve the problem needs to be identified and collected
- Data and engineering skills are needed to collect and consolidate data from multiple sources

Data Exploration and Pre-processing

How should you clean your data so your model can use it?

- Data needs to be cleaned and processed so that it's in a usable format for modeling
- Exploration is required to identify important elements within the data and to identify any data quality issues
- Data, engineering, and statistics skills are needed to appropriately process the data and make inferences

Modelling

What model needs to be built to solve your problem?

- Several factors go into modeling such as complexity, required data, speed, and performance
- This step requires skills in engineering, modeling, and statistics

Validation

Did the problem get sufficiently solved?

- Validation is required to ensure the original problem was solved
- Model performance needs to be accurately measured
- Statistics and modeling skills, as well as domain knowledge, are needed to make sure the results align with the business problem

Decision Making And Deployment

Communicate to stakeholders or put into production

- A business decision needs to be made, or a product needs to be put into production, so the business can see value from the project
- This requires domain knowledge, as well as communication and storytelling skills
- Engineering skills are needed to integrate code into back-end software systems

Data Science Skill Sets

Data science teams need a variety of skills to succeed

Data Science Team Skills

Data science teams need a variety of skills to succeed

Communicate with business leaders ⟷ Integrate code into software systems

Communication/ Storytelling	**Modeling**	**Data Munging**
Domain Expertise	**Statistics**	**Software Engineering**

Communication, Storytelling, Domain Expertise

Understand the business needs and communicate how to address them

- Domain expertise to understand the process and business problem to help their business
- Persuade decision makers to support their idea
- Communicate complicated concepts clearly and tell stories

Modeling and Statistics

Use data to make predictions via models, and using statistics to assess the validity of those predictions

- Work with a variety of modeling techniques, from regression to DL
- Use statistics to assess the performance of one model vs. another
- Design experiments and perform A/B testing

Data Munging/Software Engineering

Transform messy data into clean, usable data, as well as building software systems to deploy their models

- Raw data can be messy and unstructured
- Data must be manipulated and stored in databases before it can be used
- Models need to be deployed

Data Science Team Roles

Roles have evolved that fit on different places on this spectrum

Communicate with business leaders ⟷ Integrate code into software systems

- Product Manager
- Machine Learning Engineer
- Data Engineer
- Business Analyst
- Statistician
- Research Scientist
- Software Engineer

Business Analysts

Business analysts interact with decision-makers

- Create reports and provide insights
- Create dashboards displaying key product KPIs
- Perform analysis to determine business impact of a new product/feature
- Excel, PowerBI and Tableau are examples of tools used

Product Managers

Product Managers get requirements from business

- Translate business ideas into product ideas
- Determine feasibility of solving business problems
- Consider impact of new product or model on key business metrics
- Prioritize projects and tasks
- Examples of tools used: Microsoft Project, Trello, and JIRA

Statisticians

Statisticians determine the validity of models

- Applies statistical concepts to determine amount of data required
- Explores outliers and trends
- Determines if results are statistically significant
- Examples of tools used: R, SAS, Python

Machine Learning Engineers

Machine learning engineers solve problems involving large amounts of high-dimensional data

- Apply machine learning techniques
- Focused on the engineering that makes models accurate and fast
- Examples of tools used: Python, R, and MATLAB

Research Scientists

Research scientists work on problems in bleeding-edge fields

- Work on the toughest problems in big data and machine learning
- Expert at a particular sub-discipline
- Understand how algorithms work under the hood
- Can be part of a separate research team, interfacing with data science team when necessary
- Examples of tools used:
  Python, MATLAB, R, C++, and Java

Data Engineers

Data engineers build data ingestion, storage, and infrastructure

- Databases experts
- Know the tradeoffs between speed, reliability, and size
- Automate data cleaning
- Build ETL (extract, transform, load) pipelines to make data available on a regular cadence
- Examples of tools used: Java, SQL, and noSQL

Software Engineers

Software engineers are responsible for optimizing code and deploying

- Get code into production
- Write tests to detect code breaking and bugs
- Ensure model code is maintainable
- Examples of tools used: Python, Ruby, C++ and Java

Skills and Roles

	Business Analyst	Product Manager	Statistician	Machine Learning Engineer	Research Scientist	Data Engineer	Software Engineer
Communication/Story	X	X					
Domain Expertise	X	X					
Modeling		X	X	X	X		
Statistics			X	X	X	X	
Data Munging				X	X	X	X
Software Engineering							X

Data Science Organizational Structure

There are multiple ways to organize data science teams.

- Centralized teams where are all data scientists report to the same head
- Distributed teams where individual data scientists work with a business team
- Teams embedded within functional business units

Centralized Teams

All the data scientists report to the same group head

- Usually within a technology or IT team
- Pros:
  - Standardization of skills and tools
  - Reduction of redundant roles
  - Closer collaboration amongst scientists
- Cons:
  - Further removed from the business units
  - Tendency to be more reactive to problems

Distributed Teams

Distributed teams are where individual data scientists sit within a particular team within a business unit

- Pros:
  - Closer access to the business, domain experts, and end users
  - More likely to come up with solutions to immediate business problems
- Cons:
  - Destandardization of tools and skills throughout the organization
  - Less communication and collaboration between data scientists

Additional Team Structures

There are multiple hybrid ways to organize data science teams

- Some examples include:
  - Full data science teams can be embedded within a business function
  - Data science centers of excellence can be created for the company

Misconception #1: Data Science "Unicorn"

Data Scientists who are experts in all areas are called "unicorns"

- Successful teams contain people with a diverse array of skillsets and backgrounds
- Some excel at communication, while others may excel at statistics
- Successful teams have experts in the three main areas: business, science, and engineering

Misconception #2: Research and Algorithms Focus

Data science teams cannot just focus on research and algorithms

- Effective teams have mechanisms to:
  - Identify problems
  - Communicate findings
  - Work with engineering to understand how to put their models into production

Misconception #3: Complex and Advanced Systems

The most complicated solution isn't always the best

- Teams tend to be more successful when they start simple and then move on to more complex modeling techniques
- Complex models may be more accurate, but are less interpretable, more likely to fail in unpredictable ways, and harder to maintain
- Starting simple also ensures that what the team is building aligns with business needs

Misconception #4: Industry Differences

The underlying modeling and data analysis techniques are largely transferable between industries

- Domain expertise is required to understand which data is relevant and which problems are most important to solve
- The techniques used to clean data, store it, and extract useful insights and modeling remain very similar

Misconception #5: Projects Begin Well-Defined

Data Science projects are often exploratory and experimental in nature

- It may not be clear how hard the problem is to solve until investing time exploring the data
- Product Managers must actively work with both the team and the business stakeholders to manage expectations

Misconception #6: Best Prediction Models are Best

There are more challenges involved when selecting a model than its predictive abilities

- Some models may be too slow or complicated to include in production
- Some models may not be interpretable, and would have a tough sell with decision makers

After Deploying a Model

Once a model is deployed, relevant teams must monitor and manage the model for it to be useful

- **Business Intelligence teams** should build reports/dashboards displaying model results
- **Business teams** (e.g. operations, merchandising) or customers should use model outputs
- **Data Science teams** must update model at appropriate cadence (for example, monthly)

Business Intelligence Team

The business intelligence team monitors how model predictions are changing over time

- Monitor model usage and output
- Develop key performance indicators and dashboards
- Visualizations should align with the needs of the functional teams (for example, operations) that will be using the model

Functional Business Team

Use dashboards to improve decision making
E.g:

- Marketing team could use churn model to decide when to reach out to groups of customers
- Operations team could use model to predict shipping/logistics times

Data Science Team

Monitor model to ensure continued validity and usefulness

- For example: changes in marketing may lead to different types of customers coming in the door, making the old model of customer behavior less accurate
- Depending on how fast the business conditions are changing, the data science team should update model with appropriate frequency
- Retraining models with new data

Data Collection and Preprocessing

Problem Statement	What problem are you trying to solve?
Data Collection	What data do you need to solve it?
Data Exploration & Preprocessing	How should you clean your data so your model can use it?
Modeling	Build a model to solve your problem?
Validation	Did I solve the problem?
Decision Making & Deployment	Communicate to stakeholders or put into production?

Data Collection

There are several things to consider when collecting data.

- Where does the data come from?
- What type of data is there?
- How much data and what attributes do I need?

Data Sources

Data is sourced from many different places, for example:

- Human generated
- Internet of Things (IoT) and machine generated
- Public website
- Legacy documents
- Multimedia

Data Sources: Human Generated Data

- Within the last two years 90% of the world's data was created (Domo research[1])
- Experts believe that about 70% of this data is coming from social media.
- Our current amount of data output is 2.5 quintillion bytes per day (2.5 exabytes per day).

[1] - https://www.iflscience.com/technology/how-much-data-does-the-world-generate-every-minute/

Data Sources: Human Generated Data

2.5 Exabytes Per Day?

Data Sources: Human Generated Data

There are various APIs to access this data.

- Facebook: Graph API
- Twitter: REST API or Streaming API
- Instagram: Graph API or Platform API
- LinkedIn: REST API
- Pinterest: REST API

Data Sources: Human Generated Data

Social media is not the only source of human generated data; every year humans generate a large amount of media and publications.

- 2.2 million books published every year
- 2 million blog posts published every day
- 269 billion emails published every year

Though not publicly available, this data contains valuable information for companies—e.g. powering services like Google Smart Reply.

Data Sources: IoT Data

IoT (Internet of Things) data consists of all web-enabled devices that collect, send and respond to data they required from their respective environments.

- By 2020 IoT will include a projected 200 billion smart-and-connected devices.
- The data produced is expected to double every two years to total 40 zettabytes (40 trillion gigabytes).
- Collected via sensors, cameras, and processors.

Data Sources: IoT Data – Consumer

Consumer IoT provides new pathways for user experience and interfaces.

E.g. connected cars, smart home devices, and wearables.

Data Sources: IoT Data – Industrial

Used to monitor and control industrial operations and tools.

- Surgery bots capable of ‘seeing’ during surgery via cameras.
- Numerous sensors and cameras installed on autonomous trucks.
- Jet engine sensors providing real-time feedback

Data Sources: IoT Data – Accessing?

By design, it is difficult and expensive to access data that organizations maintain and control.

- There are some known open-source datasets (E.g. https://old.datahub.io/dataset/knoesis-linked-sensor-data)
- Design your own with development platforms such as Raspberry PI or Netduino P1.

Data Sources: Public Website

Data that is publically available on the web will allow you access to multiple genres of data as needed.

- Encyclopedia (E.g. Wikipedia)
- Stock data (E.g. Quandl)
- Entertainment (E.g. IMDb)

WIKIPEDIA
The Free Encyclopedia

Data Sources: Public Website

How to access this data?

Webscraping:

- Use with caution: the website's stance on crawlers and webscraping is usually within the terms and conditions section of their site.
- Make sure your crawler follows the rules defined in a website's robots.txt file (don't exceed request rate limit)

APIs:

- Often easier than webscraping

Data Sources: Legacy Documents

Several industries have traditionally used paper forms to collect data, creating huge potential opportunities to shift to digital.

- Two major industries shifting to more digital approaches are insurance and medicine.
- Medical records have traditionally been paper-and-pencil.
- The push for digital promises to lead to better outcomesfor patients.

Data Sources: Multimedia

Data is present in more media types than ever before.

- Companies collect text, images, audio, and video.
- New database technologies have evolved to store this data. (MMDBs – multimedia databases)
- New ML techniques (e.g. Deep Learning) have evolved in part due to the necessity of analyzing this data.

Data Types

There are different data types.

- Numerical
- Discrete
- Categorical
- Ordinal
- Binary
- Date-time
- Text
- Image
- Audio

Data Types: Count Data

Integer valued data that comes from counting

- For example, number of cars in a parking lot
- The programming term for count data is **integer**
- In Python it is an **int**

Data Types: Continuous Numerical

Numerical value that can represent any quantity over a continuous range

- Can take take on decimal values (E.g. engine stroke = 3.40 in)
- Can be reduced to finer levels of precision (E.g. engine stroke = 3.3775 in)
- The programming term is **floating-point**
- In Python it is a **float**

Data Types: Categorical Data

Data that is restricted to a finite set of known categories.

- Also known as nominal data
- Raw categorical data can come in the form of different data types (E.g. text data (vehicle color: red) or numerical data (number of doors: 4))
- The programming term is **enumerated type**
- Not a type in Python, instead inherit from the class **Enum**

Data Types: Ordinal Data

Categorical data that is ordered

- The distance between categories is not known (E.g. car price values of low, medium, and high or customer reviews of poor, ok, and good)
- A common mistake is to simply convert ordinal data into integers (E.g. G, PG, and R to 1, 2, and 3)
- The inherent problem is that this assumes that the distance between the categories are known and the same

Data Types: Binary Data

Two mutually exclusive categories

- Binary data is very common, especially in supervised learning problems (E.g. true/false, heads/tails)
- The programming term for binary data is **Boolean**
- In Python* it is a **bool**

Data Types: Date-Time Data

Data that represents date and time information

- Date, time of day, and fractional seconds based on a 24-hour clock are combined
- It is possible to convert to primitive types
- Can be converted to categorical data, or convert to integer via Unix time stamp (E.g. to only Date, Month, Year, Time)
- Most languages have a built-in datatype for date-time data
- In Python the data type is **datetime**

Data Types: Text Data

Alphanumeric strings, a sequence of characters

- Typically human-readable
- Can be encoded into computer-readable formats such as ASCII or unicode
- Often is unstructured data
- The programming term for text data is **string**
- In Python it is a **str**

Data Types: Image Data

Still and video images are being generated at a rapid rate

- They are stored in various ways (E.g PNG and JPEG)
- Images can be very large and computationally intensive
- They can be multi-dimensional, as in 3D body scans
- Generated from many types of devices in a diverse set of fields (E.g. satellites, self driving cars, surveillance cameras, mobile devices)
- Medical imaging is an increasingly important area

Data Types: Audio

Audio data is used in consumer devices like Alexa/Google Home, as well as industrial settings like call centers

- Stored in various compressed and uncompressed ways (E.g. WAV and MP4)

Data Shape

Identify the number of samples and features needed to solve the problem

- The more features you have, the more samples are generally required
- If your features don’t contain enough information, then adding more samples wont help - additional features are required
- If you have too few samples, then adding additional features can cause overfitting
- Difficult problems usually require more features and additional samples

Data Shape: How Many Samples?

There are several ways to determine the number of sample needed

- A **learning curve** plots the performance score by number of samples
- Use this curve to determine if increasing the number of samples is helping your model
- **Statistical heuristic**: 10x as many samples as degrees of freedom
- Peruse similar studies to learn from other successes and failures

Data Shape: Under-vs-Over-fitting

A learning curve can also help diagnose whether a model is under-fit or overfit

- Plot the performance score by the model complexity
- More features means more complexity
- When train and test performance are similar - but low - the model is under-fit
- When test performance suffers, while train performance improves, the model is overfit

Data Shape: Under-vs-Over-fitting

There is a relationship between under-fit and over-fit models and the number of samples and features

- In general, the number of samples should be greater than your number of features
- If overfitting:
  - adding more features usually hurts
  - adding more samples usually helps
- If under-fitting:
  - adding samples generally doesn't help
  - strong explanatory generally variables helps

Data Shape: Increasing No. of Features

Increasing the number of features can help when the model is under-fitting

- Revisit the data pipeline and add features that had previously been removed
- Design new features - this is referred to as **feature engineering**
- **Polynomial features** are a common method - this builds new features by multiplying existing ones (E.g. x1*x2 or x2*x2)
- Images/Audio: add transformations (E.g. **delta features** - a measure of how color changes from pixel-to-pixel - can be added to an existing image)
- Image augmentation for computer vision
- Create new images by shifting, flipping, rotating, existing images

Data Shape: Feature Selection

Select a subset of features in order to **reduce** model complexity and **remove** redundant or weak features

- **Filter methods**: filter features via a statistical method (p-value)
- Feature is considered on a univariate basis
- **Regularization methods**: learn which features best contribute to accuracy while the model is being created
- **Wrapper methods**: selects subsets of features as a search problem, where different combinations are evaluated to capture interactions between features

Data Shape: Dimensionality Reduction

Dimensionality reduction is an unsupervised learning technique used to reduce the number of features.

- Reduce dimensions (compress data) without losing too much information (E.g. using the eigenvectors of the covariance matrix as the reduced dimensional space)

Datasets for AI

There is no standard type of datasets used for AI

- They can vary in shape and size
- They can be made up of many different data types
- They differ depending on the tasks

Datasets: Opensource Repositories

There are several repositories to get open source datasets

- **UCI Machine Learning Repository**: Includes many popular datasets used to analyze algorithms within the ML community
- **ImageNet**: Large database of images
- **KDD Cup**: An annual competition organized by the Association for Computing Machinery. Yearly competition datasets are archived
- **Kaggle**: A platform for data science competitions. Competition datasets and other datasets are available
- **Data.gov**: Open data from the US government
- And many more…

Datasets: ImageNet

ImageNet is a large image database that is popular in the AI community

- First presented at the 2009 Conference on Computer Vision and Pattern Recognition
- Currently has over 14 million images
- Hosts annual object detection and classification competition
  - Deep learning models have had breakthrough results that have led to the current focus on deep learning for modern AI
  - Many of the most popular deep learning models have come from this competition

Datasets: Image Datasets

Some other popular image datasets

- **MNIST**: A popular benchmark for many machine learning models
  - Images of handwritten digits that has served as a benchmarking basis
  - 70,000 28 x 28 pixel black and white images
- **Cifar-10**: A widely used dataset for computer vision research
  - 60,000 32 x 32 pixel color images
  - 10 different classes for classification. 6,000 images for each class
- **ILSVRC**: The annual ImageNet competition has several components each with their own datasets
  - Object localization: 1.2 million training images of 1,000 object classes
  - Object detection: 456,567 training images of 200 object classes

Datasets: Natural Language Datasets

Some popular language datasets

- **Common Crawl**: Crawls the web four time a year and makes its archives and datasets free for the public

  The archive consists of 145 TB of data from 1.81 billion webpages as of 2015

- **Stanford Question Answering Dataset**: A dataset of over 100,000 question-answer pairs
- **Project Gutenberg**: Offers over 56,000 free eBooks

Datasets: Other Datasets

There are many other datasets available for different tasks.

- **YouTube-8M Dataset**: Roughly 7 million video URLs containing 450,000 hours of video
- **MovieLens Dataset**: A dataset of roughly 20 million ratings on 27,000 movies by 138,000 users
- **OpenStreetMap**: A crowdsourcing project to create a free map of the world
  - Over 2 million users collect data using survey, GPS, aerial photographs, and more
- **1000 Genome Project**: Human genotype and variation data collected 2008-2015
  - 1,000 genomes with 84.4 million variants from 2,504 individuals

Data Preparation

Data preprocessing is needed because the quality and structure of data is often not immediately ready for analysis

- Data obtained via webscraping or APIs is usually unstructured and must be manipulated into numerical rows and columns for analysis
- This often involves low-level manipulation of data objects, turning strings into numeric data and vice versa
- There could be missing data values
- Different ML models require different data formatting

Data Preparation: Missing Values

Often certain parts of the data are missing.

- Replace missing values with a reasonable default
- Features with symmetric distribution: input the missing values with the mean of the feature
- If not symmetric or categorical, imputing the median or mode respectively is best
- If too many features are missing, it is best to drop the observation

Data Preparation

Some examples of ML models that require data to be formatted differently include:

- Models requiring features to have a similar scale
  - Option 1: Scale each feature so that the maximum is one and the minimum is zero
  - Option 2: Standardize each feature to have a mean of zero and a standard deviation of one
- Others transform the dependent or target so it is not extremely skewed (E.g. by using logarithms)

Data Labeling

We need labeled data for supervised learning tasks, but there is usually more unlabeled than labeled data

- Data can be hand-labeled by employees, but this can be expensive and time consuming
- Other options include Amazon Mechanical Turk and semi-supervised learning techniques

Data Labeling: Amazon Mechanical Turk

Amazon Mechanical Turk* is an online marketplace for tasks

- Organizations can post tasks on the website that "Turkers" can then choose to perform
- Major open source datasets have been complied using Amazon Mechanical Turk
  - Microsoft Common Objects in Context (COCO) dataset contains images with 5 short descriptions each - from "Turkers"

Data Labeling: Semi-supervised

Semi-supervised learning involves using both labeled and unlabeled data to solve tasks

- Build a classifier on the labeled data, use it to label some of the unlabelled data, and then proceed iteratively until all data has been labeled

Learned Divisions

Challenges

There are many challenges with the collection, structure, and usage of data that can lead to poor performance

- All these can lead to overly optimistic and misleading results or model failure
- Some examples include: biases and outliers in the data, inappropriate validation and testing, class imbalance in classification problems

Challenges: Biases in Data

Algorithms will reflect the data they are trained on

- When trained on biased data, they will reflect those biases (E.g. the meaning of the word “man” rather than “woman” may be associated more with “power”, after being trained on a large corpus of human-generated data)
- It’s important for the modeler to know how the data was collected to address any biases that might be present.
  - A lot of the time this cannot be corrected during the analysis stage. There are corrections for some special cases

Challenges: Data Leakage

Data leakage is any time the train/test split is violated.

- The model doesn't see test data until evaluation.
- Data leakage can lead to overestimates of how well the model will generalize to unseen examples
- There is no single solution to data leakage. The modeler should be careful when developing validation strategies and splitting the dataset
  - Data leakage is common when working with time series data and during preprocessing
  - For example, models that make predictions about the future based on the past should have train and test sets organized this way or using test data to help preprocess the training data

Challenges: Outliers

Outliers are data points far away from other data points

- They can skew model results by suggesting performance is much lower than it actually is
- The modeler should detect outliers and try to understand why they're present during the EDA stage
  - There are many outlier detection methods.
- There are many ways to combat outliers
  - Often outliers must be removed from the dataset, or adjusted to fit the pattern of remaining dataset
  - Use models and/or metrics that are less sensitive to outliers

Challenges: Imbalanced Datasets

Imbalanced classes can lead to difficulties with training and evaluating models

- For example, if 99% of labels belong to one class, then a model that always predicts the majority class will be 99% accurate by default
- Some modeling techniques to address this include **down-sampling** the larger class, or using scoring metrics other than accuracy to assess models

End of Lecture

Many thanks to Intel Software for providing a variety of resources for this lecture series