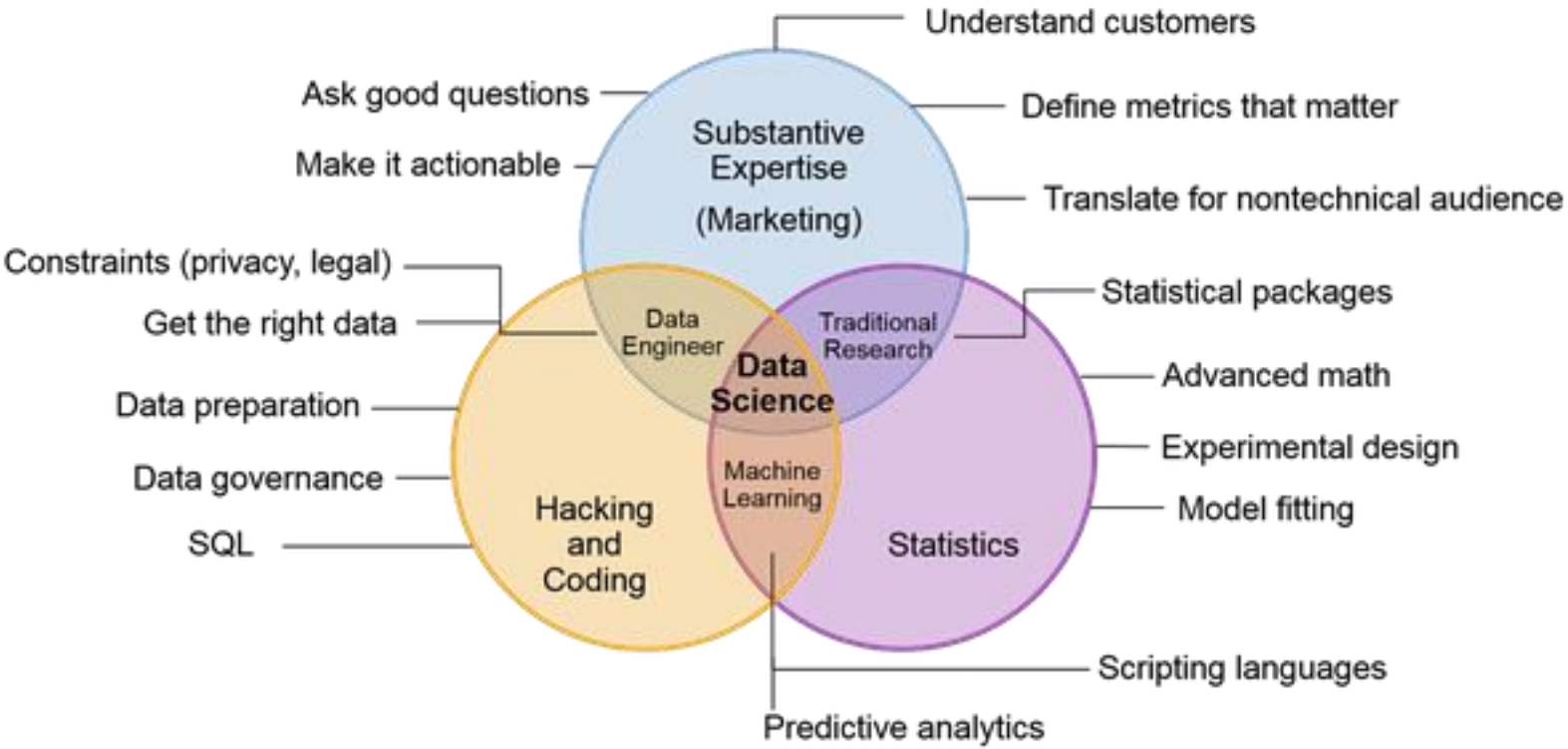
Getting Started with Data Science Using Python

These slides are based on:
https://www.slideshare.net/MSDEVMTL/data-science-presentation-94481624
By Andrei Lyskov

What is Data Science?



What is Data Science?

Typical job duties for data scientists

There's not a definitive job description when it comes to a data scientist role. But here are a few things you'll likely be doing:

- Collecting large amounts of unruly data and transforming it into a more usable format.
- Solving business-related problems using data-driven techniques.
- Working with a variety of programming languages, including SAS, R and Python.
- Having a solid grasp of statistics, including statistical tests and distributions.
- Staying on top of analytical techniques such as machine learning, deep learning and text analytics.
- Communicating and collaborating with both IT and business.
- Looking for order and patterns in data, as well as spotting trends that can help a business's bottom line.

Related to Data Science

Information Retrieval:

- *The key goal of an IR system is to **retrieve** all the items that are relevant to a user query, while retrieving as few nonrelevant items as possible*
- Query-document similarity. What features are relevant? How to query?

Machine learning

- ***Algorithms** that learn patterns in the data*
- Under the hood of most data science approaches

Data mining

- *Data Mining is about **finding the trends** in a data set (mostly for human post-processing).*
- Data Science encapsulate data mining, the distinction between DM and ML is a bit fuzzy (mostly in how are the found patterns utilized)

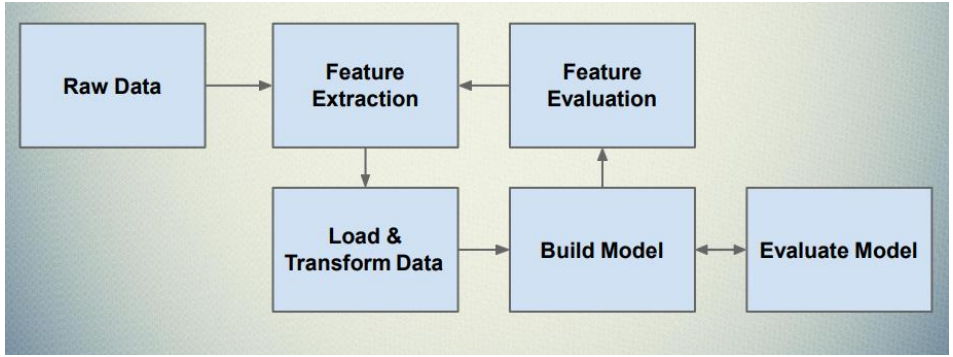
Data Science Steps

- Gather data

- Pre-processing
- Exploration phase
- Model building
- Model validation

Scope of the course

- Model deployment



Data Science Steps

- **Data Gathering (beyond scope of this course) [numpy, Pandas]**

  Not too much problem at the classes, but extremely important in reality
  (What data do I actually want? Can I have it? How to obtain it?)

  3rd party data, no unique identifier, missing data, approximate joins, updates…

  Wikidata, DBPedia, Linked Open Data

- **Data Preparation / pre-processing (this is where the magic is☺) [scikit-learn (numpy, Pandas)]**

  Data cleaning, inputs for missing values, features normalization, outliers detection

  Features augumentation / selection / construction / reduction

  - Transformers (fit -> transform)
  - https://scikit-learn.org/stable/modules/impute.html (deal with missing values)
  - https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.Normalizer.html (features/objects => axis=1/0)
  - https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html (too many correlated features)
  - https://scikit-learn.org/stable/modules/feature_extraction.html (transform feature space, e.g., numeric -> Binary)
  - https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.PolynomialFeatures.html (feature combinations)
  - https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.quantile_transform.html (transform to cum. distribution)

Data Science Steps

- **Exploration (iterates with pre-proc.) [pandas/scikit-learn/matplotlib/Seaborn…]**
  Get some insights on the data, understand its structure, create initial hypothesis
  Clustering algorithms, histograms, plots, correlation…
  https://scikit-learn.org/stable/modules/clustering.html -> Agglomerative clustering
  https://seaborn.pydata.org/generated/seaborn.heatmap.html
  https://seaborn.pydata.org/generated/seaborn.pairplot.html

- **Model Building [scikit-learn, TensorFlow, Keras,…]**
  Random Forests, SVM's, Rule-based, Deep Learning, K-Nearest Neighbours…
  (semi)-supervised / reinforced (dynamic model selection) / representation learning…
  Estimators (fit -> predict)
  https://docs.scipy.org/doc/scipy-0.15.1/reference/generated/scipy.spatial.distance.cdist.html (distance calculations <- embeddings)
  https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html (tree-based model)
  https://scikit-learn.org/stable/modules/neighbors.html (K-nearest neighbors, ML basics, curse of dimensionality)
  https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html (Linear regression , ML basics, high bias)

Data Science Steps

- **Model Validation [scikit-learn, matplotlib, Seaborn…]**
  Prediction accuracy, ranking correctness, Precision vs. Recall, Temporal complexity!!
  https://scikit-learn.org/stable/modules/model_evaluation.html
  Types of error (false positive vs. false negative – what is more important?)
  Results visualization (PCA, Self-organizing maps https://github.com/JustGlowing/minisom)
  Results vs. Model hyperparameters -> Seaborn heatmaps

- **Model Deployment**
  Finally you'll deploy your model into the wild. As you gather more data and feedback on how its doing you'll be able to tweak and improve it as time goes on.

So, what to start with:

- how to work with data?
- where to write code?

NumPy is 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.

Pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series.

Open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text
http://jupyter.org/

Data Science Specifics

You mainly code to get some answers

- **Proof-of-concept solutions**

  Perhaps 50%-90% of your ideas never make it into production

  Do not over-think the implementation

  - Should this be written as a lambda function? Who cares?
  - But beware of runtime complexity (you want your answers today)

- **Often, one-time running code**

  Some questions do not re-appear

  Often, two individuals wants to ask slightly different set of questions

  - Which is OK by default if you can say why
  - Certain level of „freedom“ with the assignments

*Get idea -> code -> evaluate -> **discard**, **save for later, extend** or plug into production*

Data Science Specifics

You mainly code to get some answers

- **Optimize for :**
  coding time +
  expected runtime +
  extendability (* Prob(code will ever be extended))