W4995 Applied Machine Learning

Introduction

01/18/17

Andreas Müller

Hey and welcome to my course on Applied Machine Learning. As you can see we have a pretty full class so make sure you don't hog too much space.

I'm Andreas Mueller, I'm a lecturer at the DSI and I spend most of my time working on scikit-learn development. Also, I decided to go on a first name basis with people in the course, so you can call me Andy.

This is the first time I'm teaching this class and also the first time I'm teaching at Columbia, so bear with me. If you have any feedback about the class, feel free to send me an email or drop by my office hours. Also feel free to interrupt me with questions. I'm not sure if that'll work with so big a class but I'm giving it a go.

The goal of this class is to provide you with the hands-on knowledge you need to be successful in applying machine learning in the real world. It complements the Machine Learning course, but doesn't rely on it.

For those of you who have taken a machine learning class, or who are taking it now, some things might be a bit redundant, but I promise you there'll be a lot of new stuff in this course.

Who's taking machine learning this semester, raise your hand? And who has taken it before? And who hasn't?

Scikit-learn Development

http://scikit-learn.org/dev/developers/contributing.html

A couple of you have approached me asking if I have any projects relating to scikit-learn.

Who here is interested in contributing to scikit-learn?

So answer is yes, there are many many projects. However, currently I don't have a lot of time to mentor you on this. This will probably change during the semester. You are very welcome to start contributing to scikit-learn, though, and I'm happy to answer questions.

There is a pretty extensive guide to contributing, the link is here.

Logistics

CAs: Akshay, Aarshay, Rohan, Sheallika

Office Hours

- Andreas Müller (lecturer) Wednesday 2pm-4pm, 410 Mudd
- Akshay Khatri (CA) Fridays 2pm-4pm, CS TA Room
- Aarshay Jain (CA) Mondays 2pm-4pm, CS TA Room
- Sheallika Singh (CA) Thursdays 3:20pm-5:20pm CS TA Room
- Rohan Pitre (CA) Thursday 5:20pm-7:20pm CS TA Room
  [This week on Friday same time!]

Before we get started, here's some logistics. There's four course assistance to this class, Akshay, Aarshay, Rohan and Sheallika, who are here to help you. They all have office hours in the CS TA Room.
You can find all the dates on the course website or piazza. You can also come to my office hours, just before the Wednesday lecture.
I'm in the Data Science space in the Mudd building.
And this week Rohan's office hours will be on Friday, not on Thursday as they'll be for the rest of the semester.

Oh and about enrollment: If you're not a DSI student, and you haven't heard from us about enrollment, you'll hear from us today. I'm not sure we have enough space for everybody, though.

Logistics

- Course website:
  https://amueller.github.io/applied_ml_spring_2017/

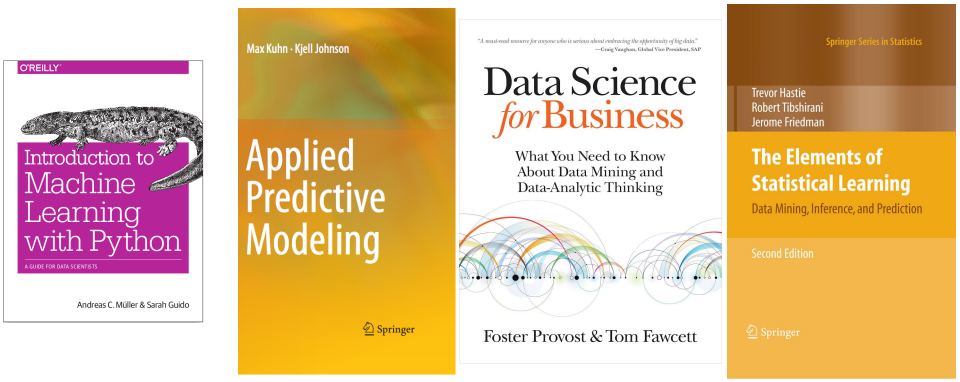
- Five homeworks, all programming
- Grade: 60% homeworks, 20% first exam, 20% second exam

If you haven't already, please check out the course website which has the syllabus and other information. I'll link to the slides there, too.

The most up-to-date material will always be on this website and the github repository, which is linked from there.

I won't produce all the details from the website, but here are some key points. There will be five homeworks, all will be programming, submitted via github. The homeworks will make up 60% of the grade, and there'll be two exams, each making up 20% of the grade.

Books

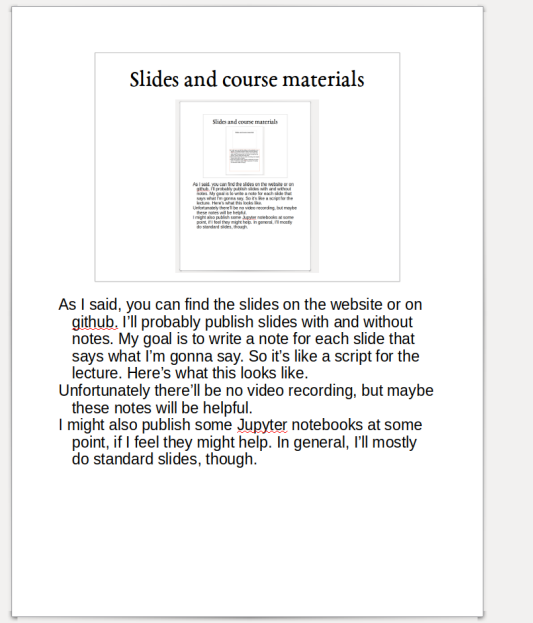


There are four books that I recommend looking into for this course. Definitely check out my book, Introduction to machine learning with Python. You can find the PDF on courseworks. My book should be a relatively easy read and it's quite short. The second one is Applied predictive modelling by Max Kuhn, which goes a bit deeper. This is about the level I want to go to in this course. You can get it for free at springer link, I posted a link in courseworks. These two are really the essential ones.

You can also check out "Data Science for Business" by one of my colleagues from NYU, Foster Provost and Tom Fawcett. There's a couple of those in the Business library. They emphasize on how to frame problems and measuring impact of data science.

Finally, Elements of statistical learning, also known as ESL or the stanford book, by Hastie Tibshibani and Friedman is a classic for a more theoretical view. You can get it for free on the authors website.

Slides and course materials



As I said, you can find the slides on the website or on github. I'll probably publish slides with and without notes. My goal is to write a note for each slide that says what I'm gonna say. So it's like a script for the lecture. Here's what this looks like.

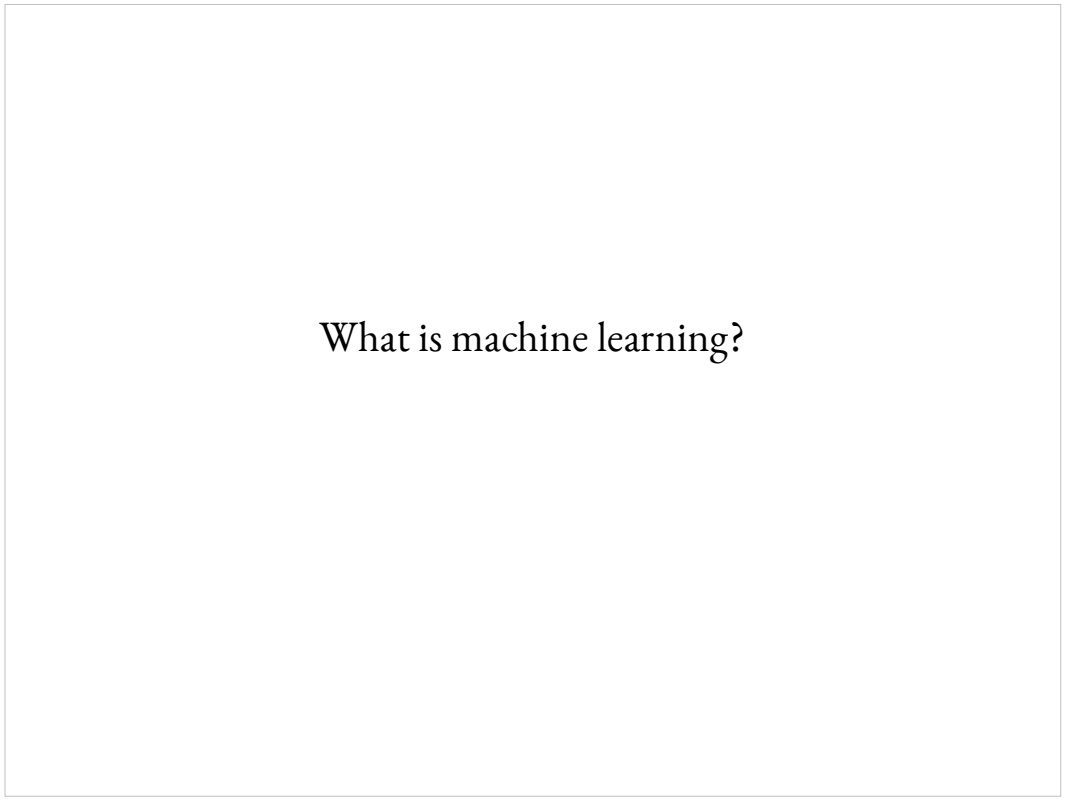
Unfortunately there'll be no video recording, but maybe these notes will be helpful.

I might also publish some Jupyter notebooks at some point, if I feel they might help. In general, I'll mostly do standard slides, though.

What and Why of Machine Learning

Ok, I think I spend enough time on logistics, let's finally get going. I first want to talk about what is machine learning, and why do we want it. As you're in this course, you're probably already somewhat convinced that it's useful, but I briefly want to give my own perspective.

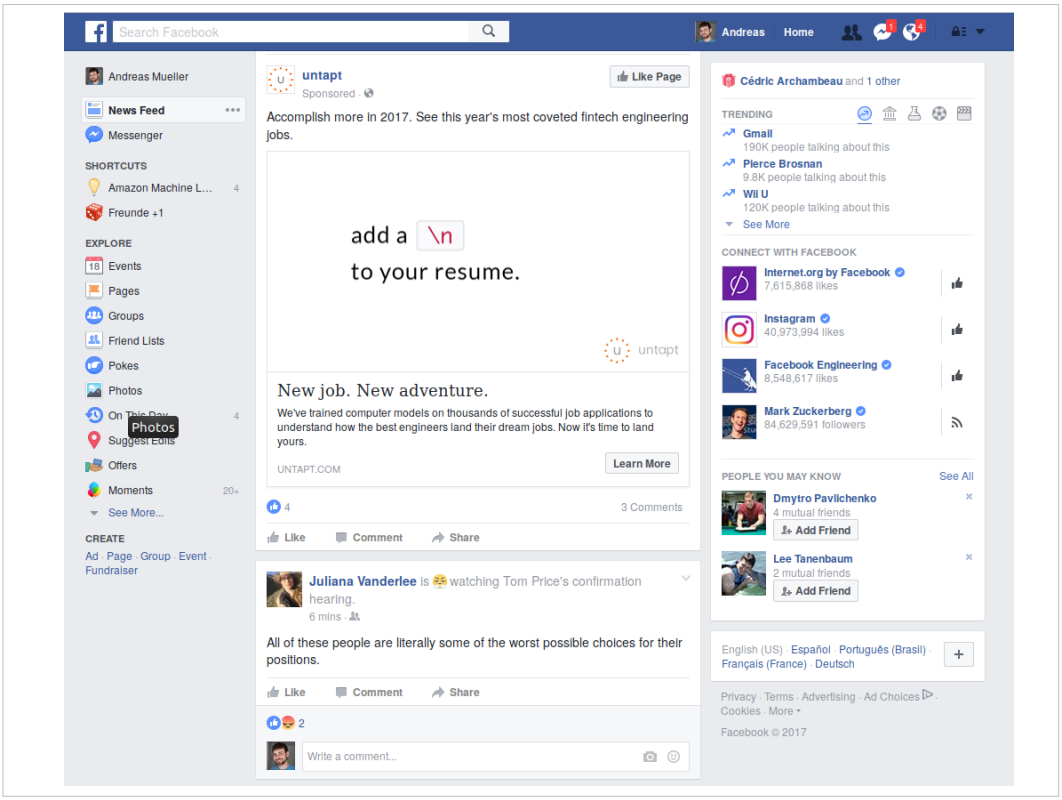
In general, today will not be very meaty and be more a loose collections of ideas and directions. The next class we will go down to the metal much more.



Machine learning is about extracting knowledge from data. It is closely related to statistics and optimization.
What distinguishes machine learning is that it is very focused on prediction.

We want to learn from a large dataset how to make decisions based on future observations.
You could say that the input to a machine learning program is the dataset, and the output is a program that can make decisions on future observations.

Machine learning is really widely used now, and I want to give you some examples that you probably all saw already today.



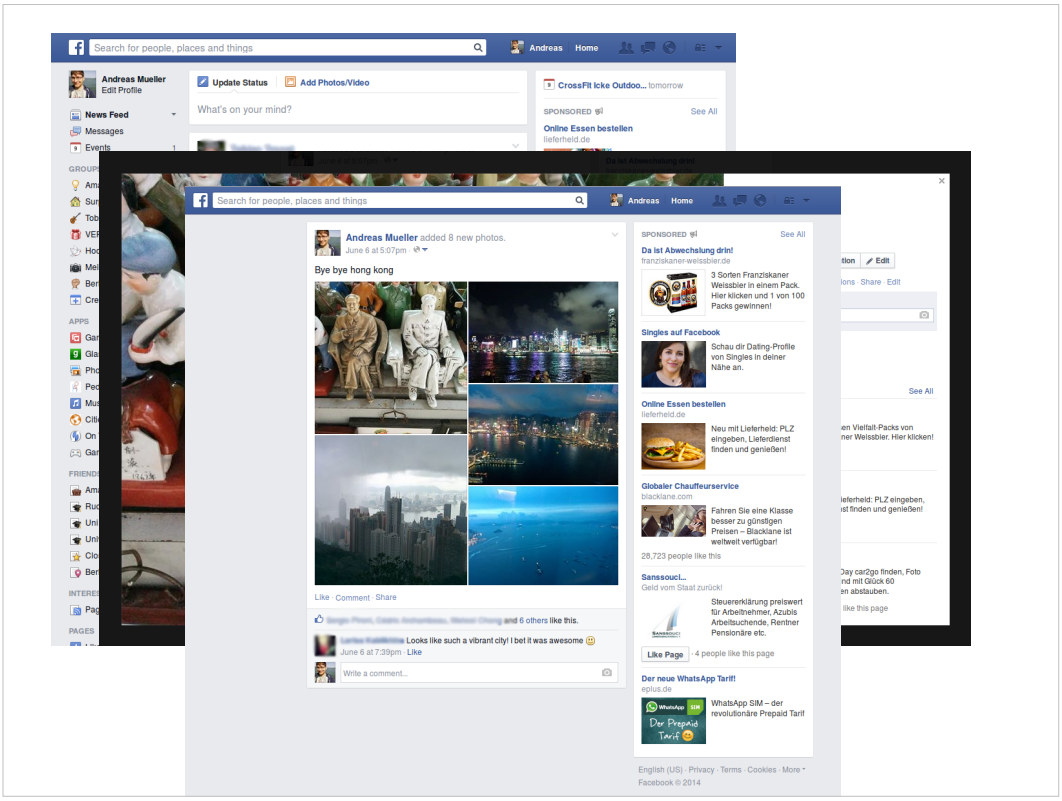
Here's the Facebook news-feed. There's so much machine learning here, it's crazy.

Can you point out some of them?

So the most space is a sponsored item. Facebook used ML to know who to show this too. It's clearly targetted at developers. Facebook also used ML to know how much to charge for showing it. Then there's a post below by a friend. Facebook ranked that top most interesting to me right now, again ML. Then on the right, you can see birthdays. It shows only one name, though there's two birthdays. Again ML to decide how many and whom to show. Below, trending topics, apps to connect and people I might now. All ML.

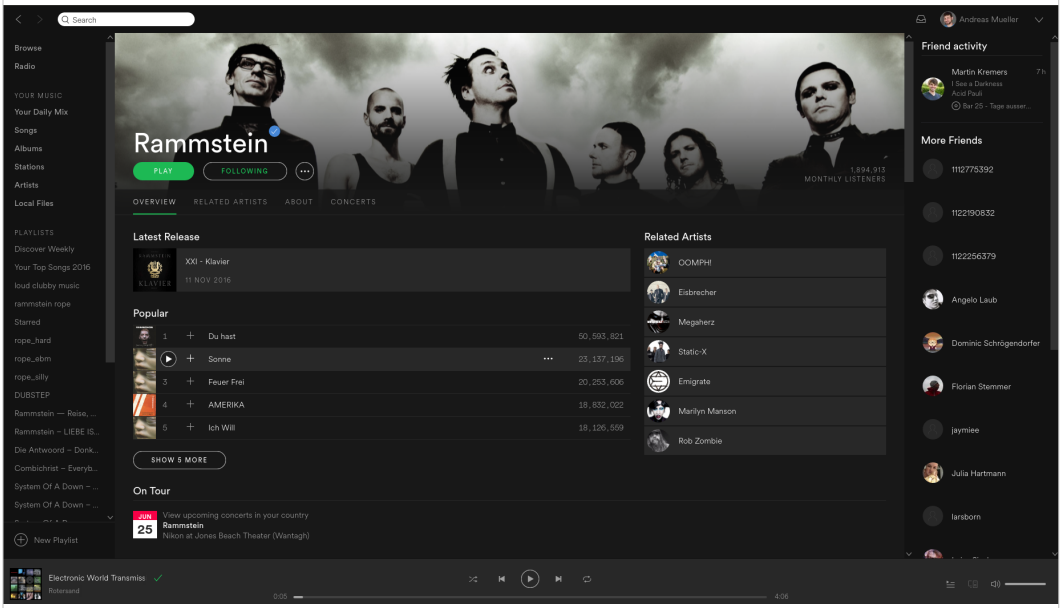
But wait, there's more. Here's me uploading a picture. This is an older screenshot, they probably change things since then. **Can you find more machine learning?**

It does find the faces on my friend Mao here, but doesn't recognize him. Also, adds to the right. Probably based on the image content and comments on the photo. Who doesn't want a burger with their communism?



And then after you post an album, facebook will select some most interesting pictures for you, and give them different kinds of space, to create a mosaic.

But that's just facebook. Let's see what else I've got open.



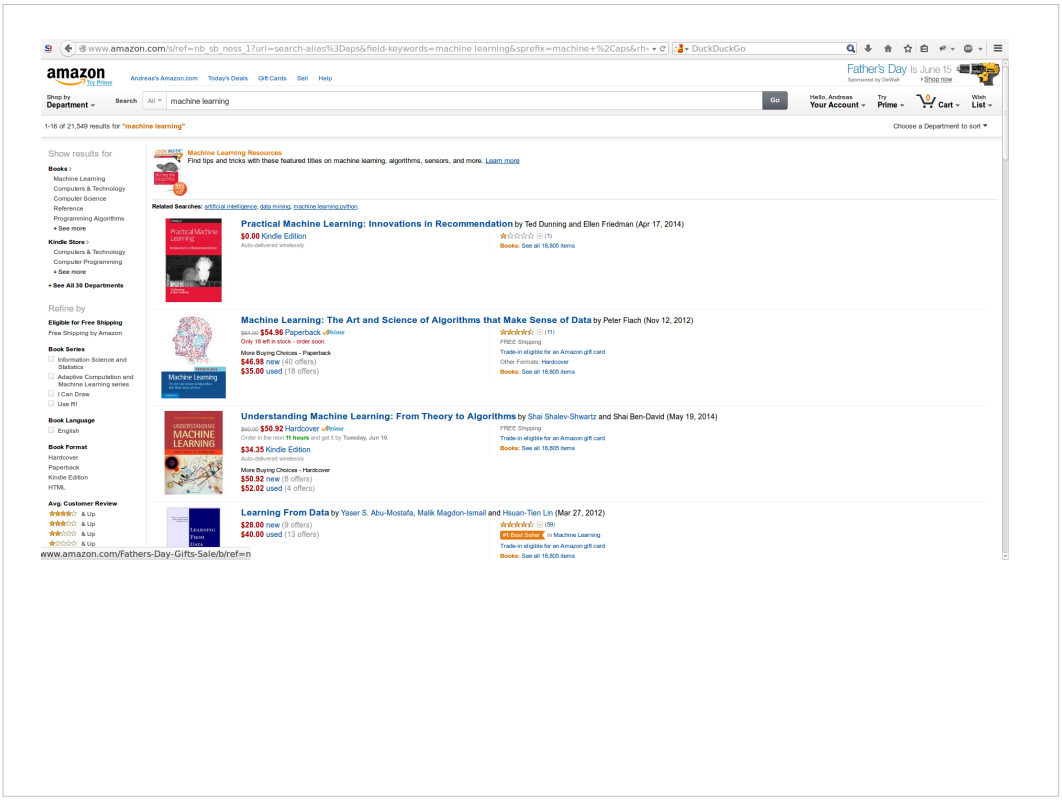
How about spotify? I’m German, so obviously I’m listening to Rammstein.

What machine learning can we see here?

There’s popular songs, which in this case are actually in the order of “likes” but in fact the algorithm also takes other features into account. And then theres Related Artists.

Is there other machine learning in spotify that you know of?

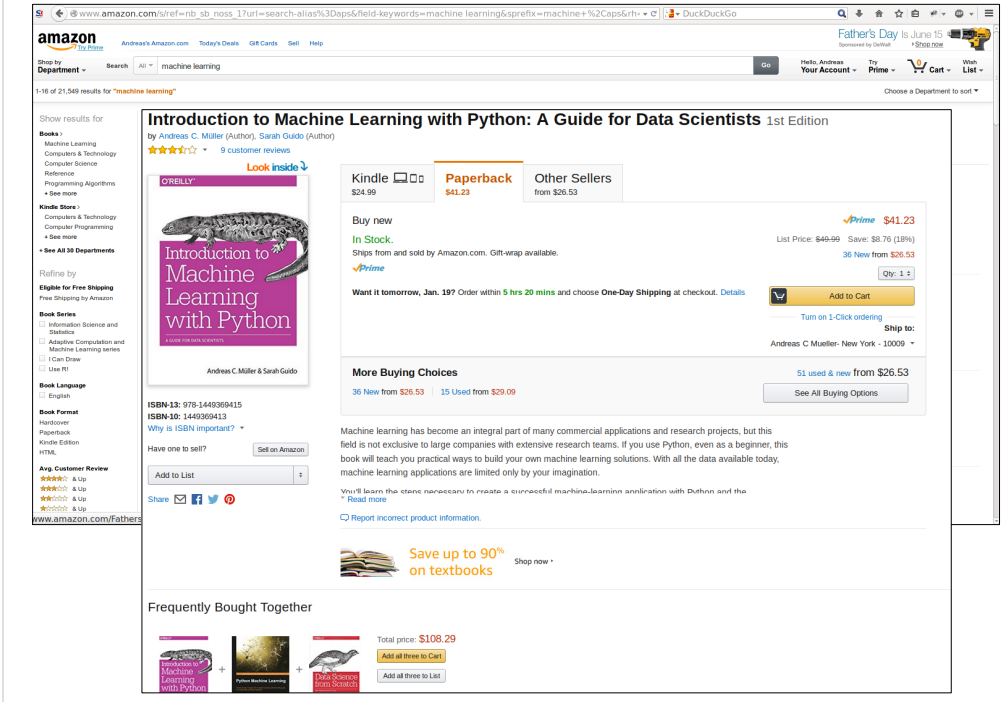
the artist radio, and of course, discover weekly.



And then as a last example, amazon. Because google would have been too easy.

Here I'm searching for machine learning. I get a ranked list, using machine learning. I get sub-categories to search in, via machine learning. Each book has some features, like top seller etc, added with machine learning.

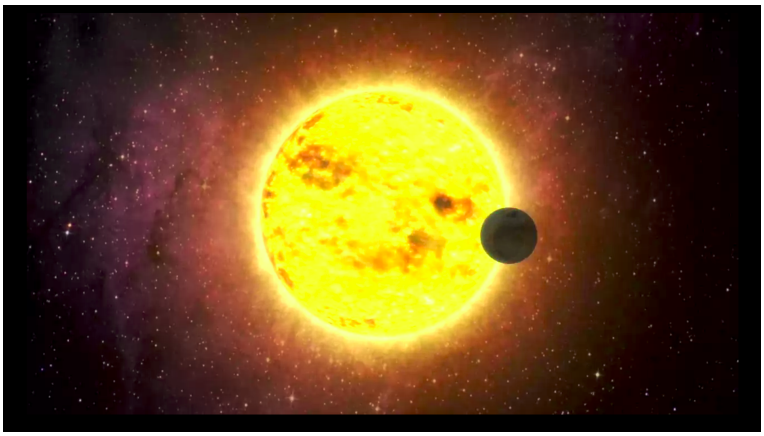
And there's an ad at the top, selected via machine learning.



And if I click on a book, more machine learning. There's an add below for textbooks, targetted to me. There's paperback as the default choice, again machine learning. There's frequently bought together, and maybe less obviously, there's a default seller! The price that is shown on the right that's selected for "add to cart" is also selected from a whole pool of possible amazon sellers via machine learning.

Ok, that's enough websites I think. While I went through all this, you were probably on your phone, looking at more output produced by machine learning. My point is, it's everywhere. And often non-obvious.

Science!



That was some of the flashy, every day live applications. Something that might get you VC funding. There's also a lot of machine learning applications in less visible, but equally important applications in science.

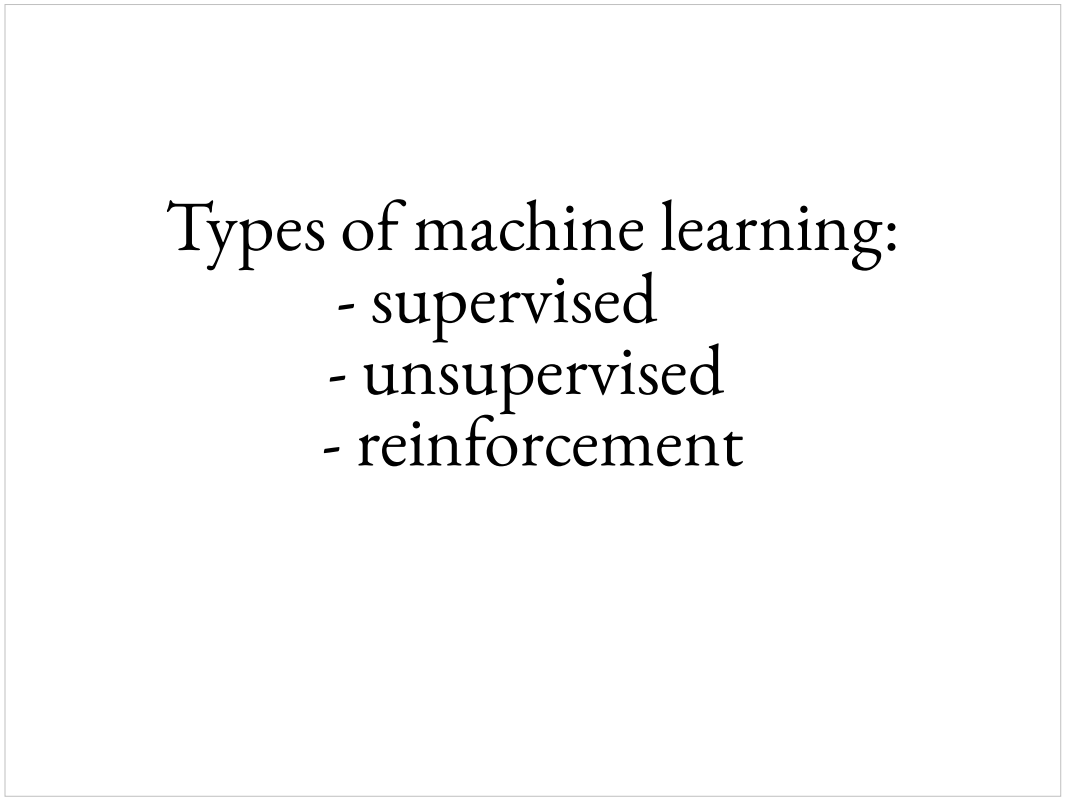
There is more and more personalized cancer treatment – via machine learning. More medical diagnosis, and more drug discovery are using machine learning. The higgs boson couldn't have been found without machine learning, and the same is true for many earth like planets in other solar systems. Which is shown using an artists illustration here, but in reality you would have a single pixel, containing the sun and the planet!

Machine learning is an essential in all data driven sciences now!

So no matter where you want to go with data, you need machine learning.

But what does that mean?

Next, I want to give you a little taxonomy of machine learning methods.



There are three main branches of machine learning.

Who can name them?

They called supervised learning, unsupervised learning and reinforcement learning.

What are they?

This course will heavily focus on supervised learning, but you should be aware of the distinction. Supervised learning is also the most commonly used type in industry and research right now, though reinforcement learning becomes increasingly important.

Supervised Learning

$$(x_i, y_i) \propto p(x, y) \quad \text{i.i.d.}$$

$$x_i \in \mathbb{R}^n$$

$$y_i \in \mathbb{R}$$

$$f(x_i) \approx y_i$$

In supervised learning, the dataset we learn form is input-output pairs (x_i, y_i), where x_i is some n_dimensional input, or feature vector, and y_i is the desired output we want to learn.

Generally, we assume these samples are drawn from some unknown joint distribution p(x, y).

What does iid mean?

We say they are drawn iid, which stands for independent identically distributed. In other words, the x_i, y_i pairs are independent and all come from the same distribution p.

You can think of this as there being some process that goes from x_i to y_i, but that we don't know. We write this as a probability distribution and not as a function since even if there is a real process creating y from x, this process might not be deterministic.

The goal is to learn a function f so that for new inputs x for which we don't observe y, f(x) is close to y.

This approach is very similar to function approximation.

The name supervised comes from the fact that during learning, a supervisor gives you the correct answers y_i.

Examples of Supervised Learning

Here are some examples of supervised learning.

- Given an array of test results from a patient, does this patient have diabetes? The x_i would be the different test results, and y_i would be diabetes or no diabetes.
- Given a piece of a satellite image, what is the terrain in this image? Here x_i would be the pixels of the image, and y_i would be the terrain types.

In your setting, it's probably good to think of each x_i as an experiment, and each entry of x_i as one sensor measurement.

- This is often used to automate manual labor. For example, you might annotate part of a dataset manually, then learn a machine learning model from this annotations, and use the model to annotate the rest of your data.

Unsupervised Learning

$x_i \propto p(x)$ i.i.d.

Learn about p

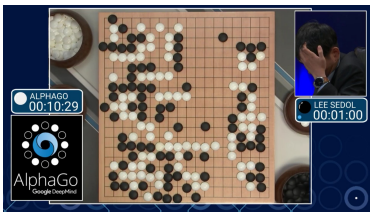
In unsupervised machine learning, we are just given data points x_i, that are assumed to be drawn from an unknown distribution. Usually we want to learn something about these, such as whether they lie on a low-dimensional subspace, or whether the data clusters in several groups, or find ways to represent the distribution compactly.

The goal in unsupervised learning is often much less clear than in supervised learning, and there is no-one providing a “correct” answers and no supervisor.

Common examples of unsupervised learning is discovering topics in news articles or on twitter, or grouping data into clusters for easier analysis.

Another one is outlier detection, where you ask “does this data look normal” which is important for fraud detection and security systems.

Reinforcement Learning

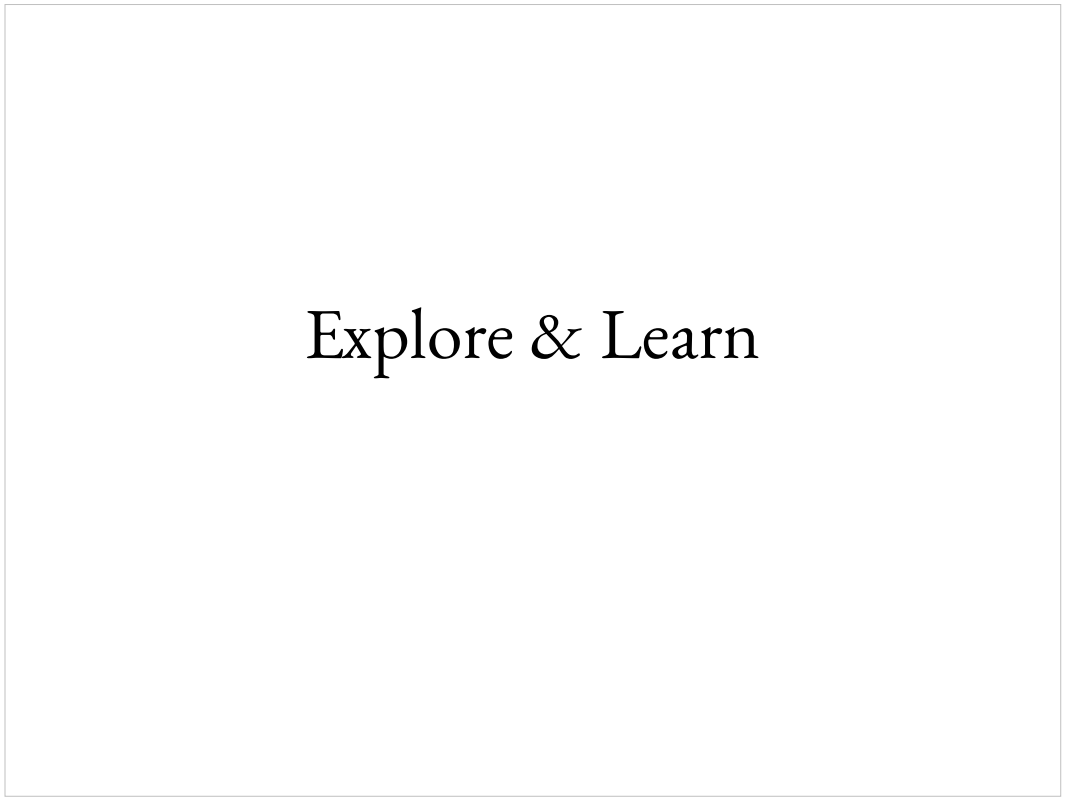


The third kind, reinforcement learning, recently had a great success. **Has anyone heard of that?** Alpha go beat the world champion in go.

Reinforcement learning is about an agent learning to interact with an environment, with some ultimate goal. The environment could be a go board, and the goal to win the game.

For self-driving cars, the the environment could be roads, sensed by cameras and laser sensors, and the goal would be to get you somewhere quickly and safely.

Or, the environment could be a social media platform, and the goal could be to provide you such great content that you never remove your eyes from your phone again!



- Reinforcement learning is quite different in that you usually don’t work with a dataset – you work with a whole world. This can be a video game, a simulation, or the real world. The actions of the agent influence which part of the world they see and which situations they encounter, and it’s usually impossible to look at all possible situations, even for something as limited as a go board.
- This means in reinforcement learning you can not separate data collection and learning, which you can do for unsupervised and supervised learning.
- There will be no reinforcement learning in this class, as real-world use of this technique is still quite limited.

Other kinds of learning

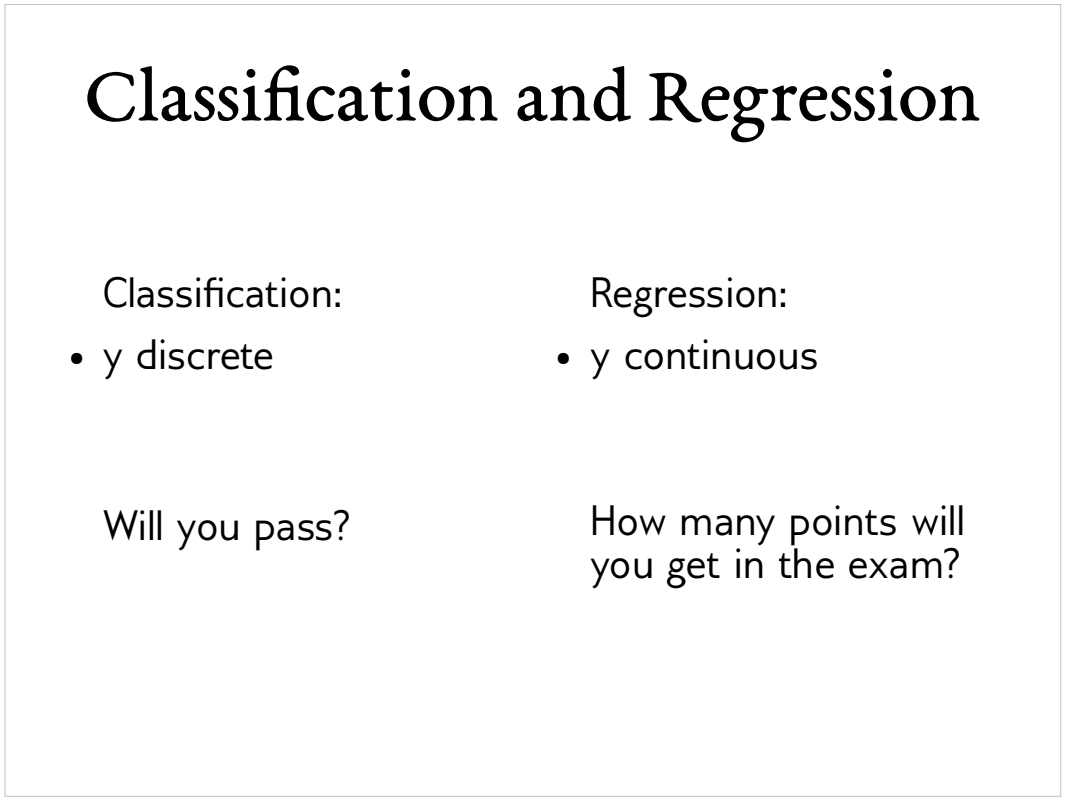
- Semi-Supervised
- Active Learning
- Forecasting
- ...

There are other kinds of learning that are somewhere between the three kinds I just explained. Semi-supervised learning for example is a combination of supervised and unsupervised learning. Active learning is somewhere between reinforcement learning and supervised learning. There are also many kinds of supervised learning where the assumption that data points are independent is dropped, for example for time series analysis and forecasting.

However, if you get the three main concepts, the rest will be easy to understand.

Some people, including Yann LeCun think that supervised learning is fundamentally limited. In particular it doesn't seem to be how humans learn. So now you can buy these shirts on redbubble

This might be true, but as long as we are trying to solve science and business problems, I think supervised learning is our best bet, even though it will probably not lead us to creating some superhuman general intelligence.



So getting back to supervised learning, there are two basic kinds, called classification and regression.
The difference is quite simple, if y is continuous, then it's regression, and if y is discrete, it's classification.

That's pretty simple, still let me give an example.
If I want to predict whether a student will pass the class, it's a classification problem. There are two possible answers, "yes" and "no".
If I want to predict how many points a student gets on an exam, it's a regression problem, there is a continuous, gradual output space.

There are generalizations of this where we try to predict more than one variable, but we won't go into that in this course. The main reason this distinction is important is because the way we measure how good a prediction is is very different for the two.

Generalization

Not only $f(x_i) \approx y_i$

Also for new data: $f(x) \approx y$

For both regression and classification, it's important to keep in mind the concept of generalization.

Let's say we have a regression task. We have features, that is data vectors x_i and targets y_i drawn from a joint distribution.

We now want to learn a function f, such that f(x) is approximately y, not on this training data, but on new data drawn from this distribution. This is what's called generalization, and this is a core distinction to function approximation. In principle we don't care about how well we do on x_i, we only care how well we do on new samples from the distribution

We'll go into much more detail about generalization in two weeks, when we dive into supervised learning.

Relationship to Statistics

Statistics	Machine Learning
• Model first	• Data first
• Inference	• Prediction
	• Generalization

Before I'll go into some general principles, I want to position machine learning in relation to statistics.

There's really no clear boundary between statistics and machine learning, and anyone that tells you otherwise is lying.

But I can tell you how the tools that I'm talking about in this course will differ from what you'd learn in a stats course.

Statistics is usually about inference, often phrased in terms of hypothesis testing.

An example might be a yes-no-question, such as "are women less likely to enroll in a Data Science Program", and you have a sample population, for example this classroom, and you can then try to make an inference about whether this statement is true.

Often this includes making assumptions on how your sample relates to the general population, say this class vs all of DSI or Columbia vs all of US. You might also have a specific model of how the process behind your question works.

Relationship to Statistics

Statistics

- Model first
- Inference

Machine Learning

- Data first
- Prediction
- Generalization

On the other hand, machine learning is about prediction and generalization. We want to learn from past data to predict outcomes on future, unseen data.

We usually want to make statements about individual data points, and we want to build a model that will work on new data that fulfills our assumptions, independent of the population we samples. Often we don't have or need a model of the process, but we rely on the assumption that our training data is generated from the same process as any future data will be.

There are statisticians that do predictions and there are machine learning scientists that do inference, but I find this distinction helpful. Even though the machine learning book I recommended to you earlier says statistics.

And I'm not saying one or the other is better, I'm just saying that you should know what kind of problem you are trying to solve, and what the right tool for the problem is. And then you can call it machine learning or statistics or probabilistic inference or data science.

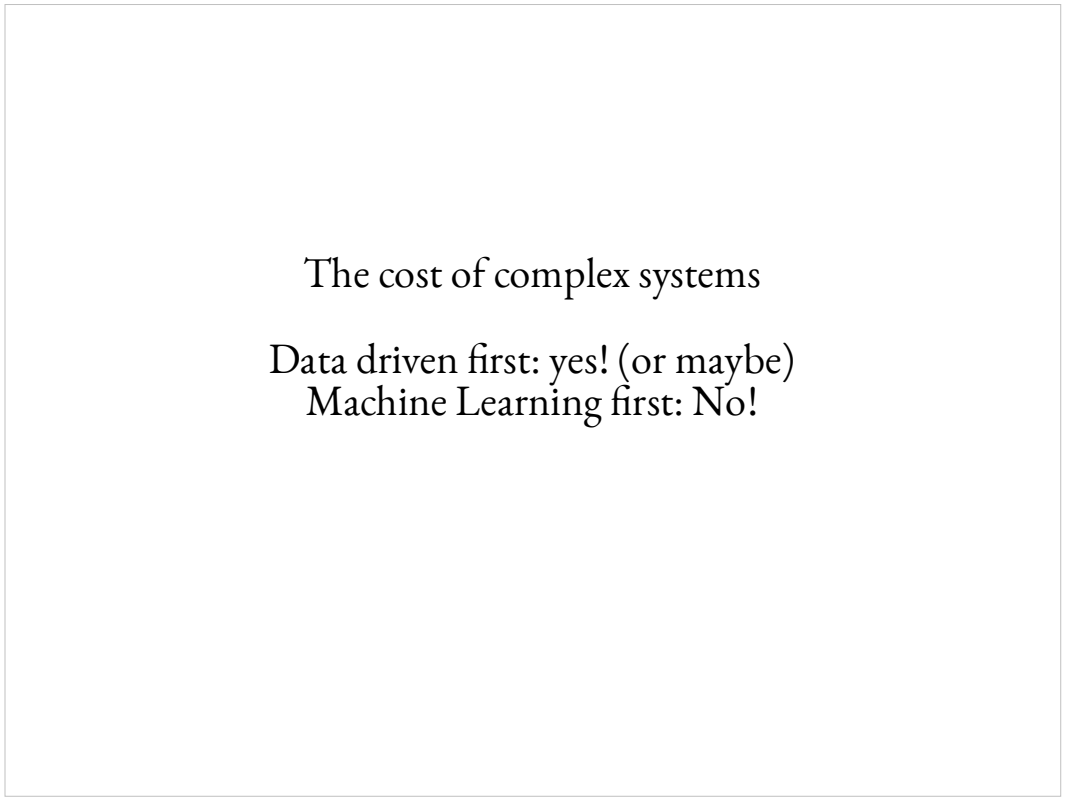
Guiding principles in machine learning

Probably for the rest of todays lecture, I want to talk about some guiding principles in machine learning applications, and they will hopefully be something you'll come back to later.

We will revisit some of them at the very end of the semester in more detail.

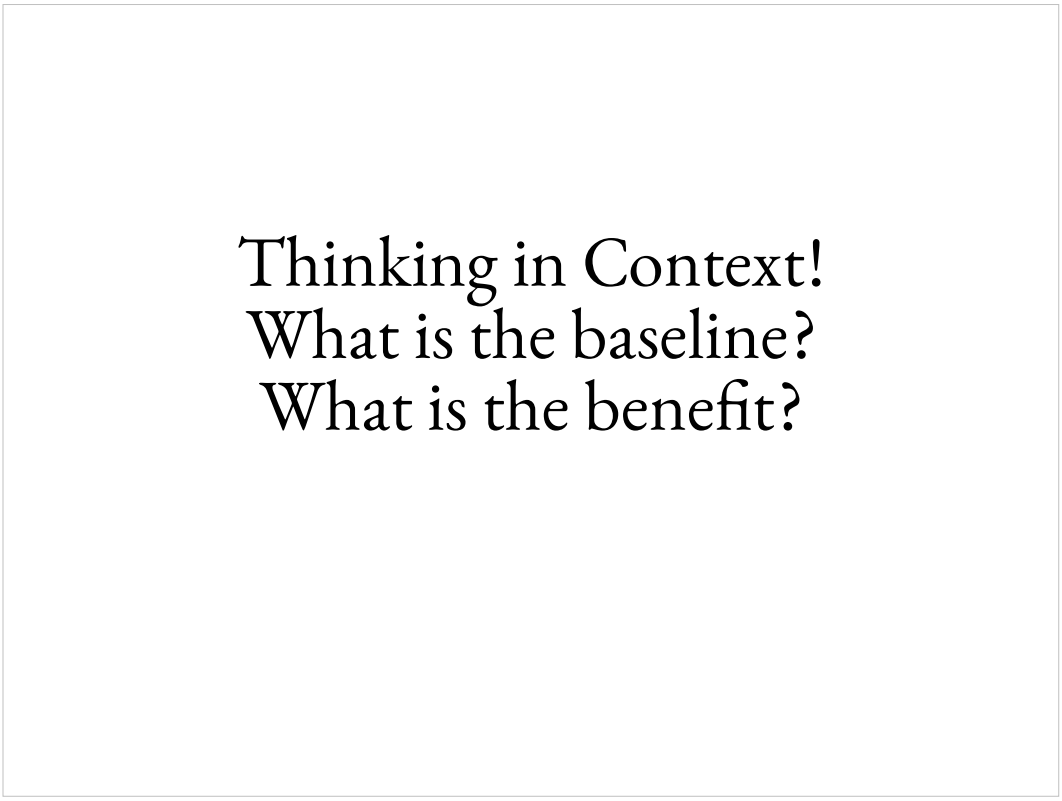
Goal considerations

- One of the most important parts of machine learning is defining the goal, and defining a way to measure that goal. In this way, Kaggle is a really bad way to prepare you for machine learning in the real world, because they already did that work for you.
- In the real world, people don’t tell you whether you should use unsupervised learning, supervised learning, classification or regression, and what’s the right way to cast something as a machine learning task – or whether to cast it as machine learning at all.



My first advice would be, don’t try machine learning. Machine learning systems are very complex and often fragile. Whether you’re in research or a startup, don’t immediately start with “oh we can apply deep learning to this”.

Often it’s good to collect data, and be able to use data to drive and evaluate decisions. But including a complex process like machine learning into whatever you’re trying to do will make it much harder to debug and much harder to understand.



So think in context of your problem. What do you want to achieve? What is the easiest way to achieve this? And what will improving over this baseline buy you?

Let’s go through an example:

Imagine you had an electronics store, but you realized retail is all digital and you’re closing your store with one big sale to move on. You want to do an email campaign to tell all your customers about the sale.

Is there a machine learning application here? You could try A/B testing the email message. But that will only work if you have many many customers. You could use ML to predict which customers might respond. **How do you get the data?** Let’s say you have data. **What’s the benefit?**

None! This is your closing sale. You’ll never email them again. It doesn’t matter if they’ll include you in your spam filter. Just email everybody!

Thinking in Context!
What is the baseline?
What is the benefit?

In particular, in industry, you need to ensure you make the right trade-offs. Maybe you can improve a decision making process by using machine learning. Or maybe there's a machine learning system you can improve.

How do you think you should decide whether a system is worth improving?

Only the ultimate metric matters (often revenue). If you can increase performance from 30% to 90% accurate but it doesn't increase revenue, who cares? Often revenue is hard to measure, and you need to measure a substitute like user engagement. Still, don't measure accuracy!

And if you measure a substitute metric, make sure that it's reasonable to assume the change will translate into your original metric.

Good and bad substitutes

The problem of metrics is not unique to machine learning, but a problem in any data driven decision making. And often you have no choice but to use a substitute metric, either because the effect you're interested in is too hard to measure, or because the influence is too indirect.

Imagine spotify improved their artist radio to be waaay better. The metric they care about is revenue. **Do you think better radio will increase revenue short-term?**

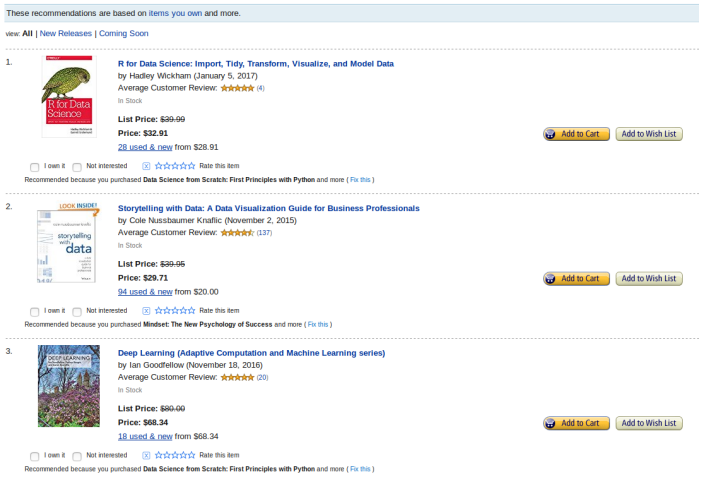
What would be a good substitute metric?

[give example of bad substitute!!]

Communicating Results

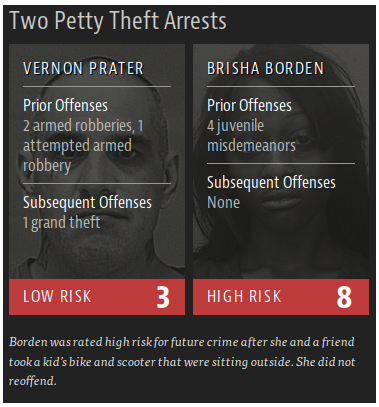
- Ok, so enough about goals and measurement. Another very important aspect of machine learning is to be able to communicate results, and to be able to explain methodologies.
- If you're not in computer science and you use machine learning in research, you need to be able to explain why it works. If you automatically make some decisions in your business applications, you need to convince your boss that nothing crazy will happen.
- How do you do that? It's an important skill to explain what particular results mean, what their impact will be, and how often a method will work or fail.
- Only if you can convince your manager your approach is sensible will your system go to production.

Explainable Results



- Sometimes, you even need to explain to the user why you made a decision. People found that recommendation engines work much better if the users are shown a reason for a particular recommendation. Netflix tells you based on what show it recommends another show, Amazon does the same for products.
- Interestingly, this also works even if the explanation is not true. There are systems that make a recommendation, and then provide a believable “explanation” after the fact. That gets more engagement than just providing the recommendation, though some people might question the ethics of that.
- Earlier I said in ML we are just interested in good predictions. If you need to be able to explain your predictions, this is no longer true.

Sidebar: Ethical Considerations



https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing

There is another area where explainability and transparency matter, and that is when people's lives are at stake.

One aspect of machine learning that only now gets some more attention is ethics. There was a recent article in propublica about racial bias in risk-assessment used in the criminal justice system. Spoiler alert: it's bad. I recommend reading the article, it's quite interesting.

This is a black-box machine learning system created by some company. If they had to provide explanations, or a more transparent system, the situation would likely be better.

But this is not the only place where ethics plays a role in machine learning. I'm not sure there's currently a course in data science ethics at the DSI, but there will be.

Ethics: It's in the application!

Some people think that ethics is not something that the technical people care about, but I disagree. I think if you build a machine learning system, you should know whether and how it is biased, and whether its application is ethical. Sometimes it's hard to decide that, though.

There's an example of two high-schools, both of whom tried to predict which of their students will underperform in the coming year. There is a lot of ways this could be biased based on race, financial background and other factors. But that's not the point.

The point is that one of the schools used the predictions, and kicked out these students before the annual evaluation, so that they got a better evaluation score.

The other school used the data to provide these students with targeted support and help. The algorithm could be the same, but the outcome is quite different.

Ok, that's enough about ethics, I hope you'll keep these considerations in mind.

The next thing I want to talk about is data!

Data and Data Collection

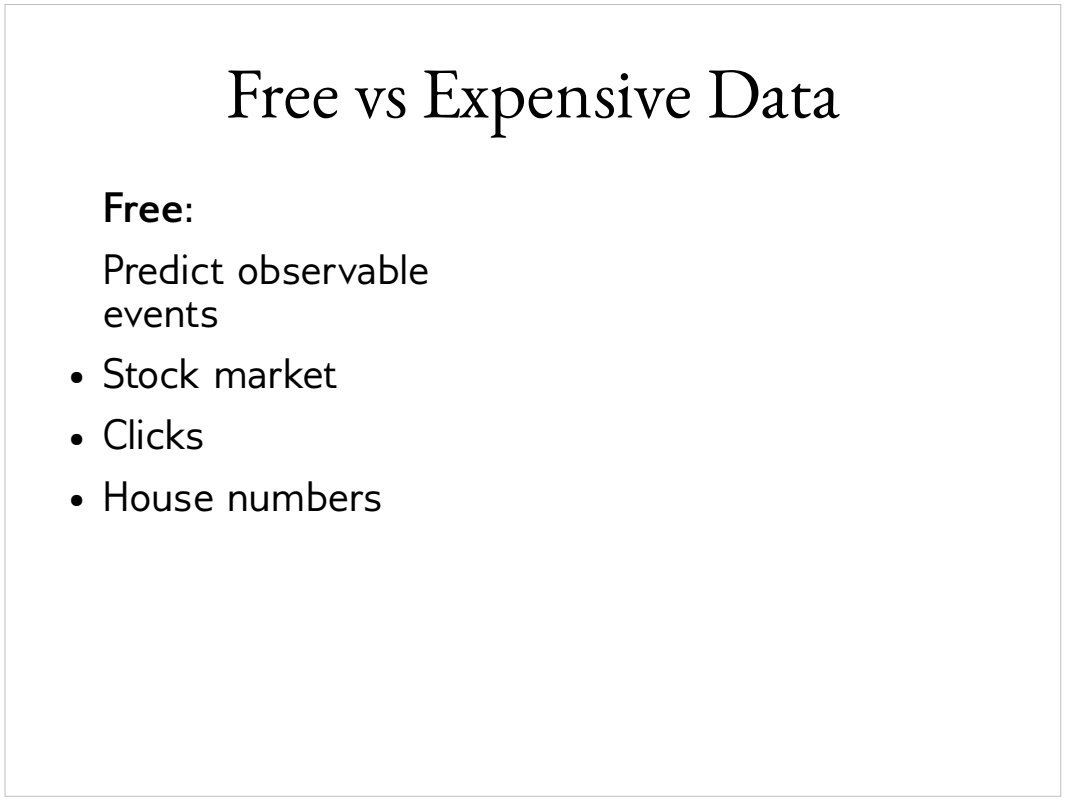
Clearly the data you use for building and applying machine learning systems is a critical component, and we will talk a lot about handling and transforming data this semester.

Clearly, if you don’t have data, you can’t use machine learning. Let’s say you have a dataset. A very important question you need to ask yourself is: should I get more data? That’s another reason why kaggle competitions are bad: usually you can’t get more data.

So what do you think? Should you get more data?

More data always improves the model if it’s from the right source. Depends: What’s the marginal cost of more data, what’s the marginal benefit to the model, what’s the marginal benefit of the model to your end-goal?

We will talk about how to assess the benefit to the model later in the course. But the other questions are also important.



The cost of data can be very different, and to kinds of data are particularly common: free data, and very expensive data. **What kind of data do you think is free?**

Free data is data that you'll just get more of. And that happens a lot. If you are running an ad company and want to do click prediction, every day you'll get so much new data, you'll barely be able to use it all.

The same is true for the stock-market. In general, if you want to predict the future, and the event is observable, you'll get more data just by waiting.

This can either be because the world just produces the data, or your business process produces them.

You can also be smart, and ensure your business is set up in a way so that it does produce the data.

Google used captchas to do OCR and to read house numbers, then they used the results for machine learning.

Free vs Expensive Data

Free:

Predict observable events

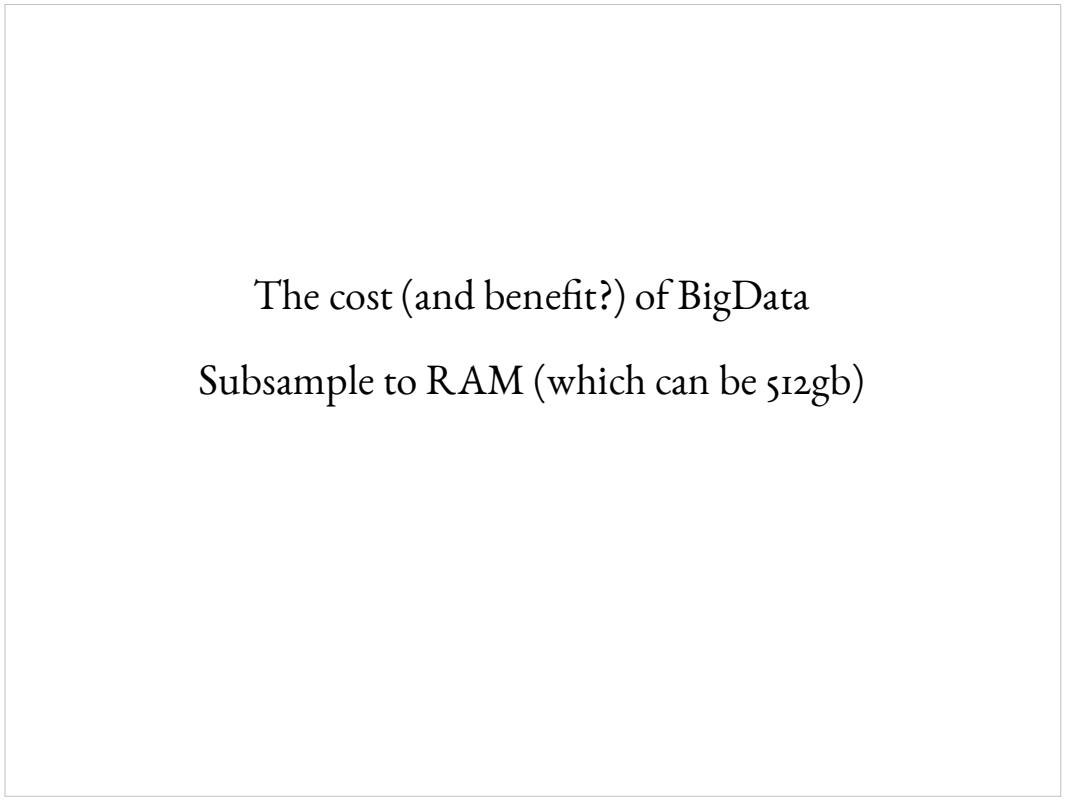
- Stock market
- Clicks
- House numbers

Expensive:

Automate complex process

- Diagnosis
- Drug Trial
- Chip Design

The other extreme of the spectrum is when you want to automate an expensive process with machine learning. This process could be an expert opinion, like a doctor's diagnosis, or a literary analysis. It could also be an experiment, like an initial drug-trial, or measuring the efficiency of a microchip.



There's another aspect to data collection and dataset size. More data might be more expensive to collect, but in particular it might also be more expensive to work with. With the available cloud services, storage might not be that much of an issue any more. But runtime is. And I'm less concerned about buying a bigger cluster, I'm more concerned about your time, the data scientists time and the machine learning analysis.

There's a reason we'll be using Python in this course. Python is easy to learn, has lots of tools and allows very close interactions with the data. If we would try to use SPARK instead, this would be whole other story. Working with distributed systems is hard, they are not responsive and the tooling is often not as good. So what I often do, no matter how big the dataset is, is to work with a subset of the data that fits into my RAM. Then I can use python, and everything is easy. And with AWS, I can easily get 512GB of RAM, if I really need to.

Cornerstones of this Course

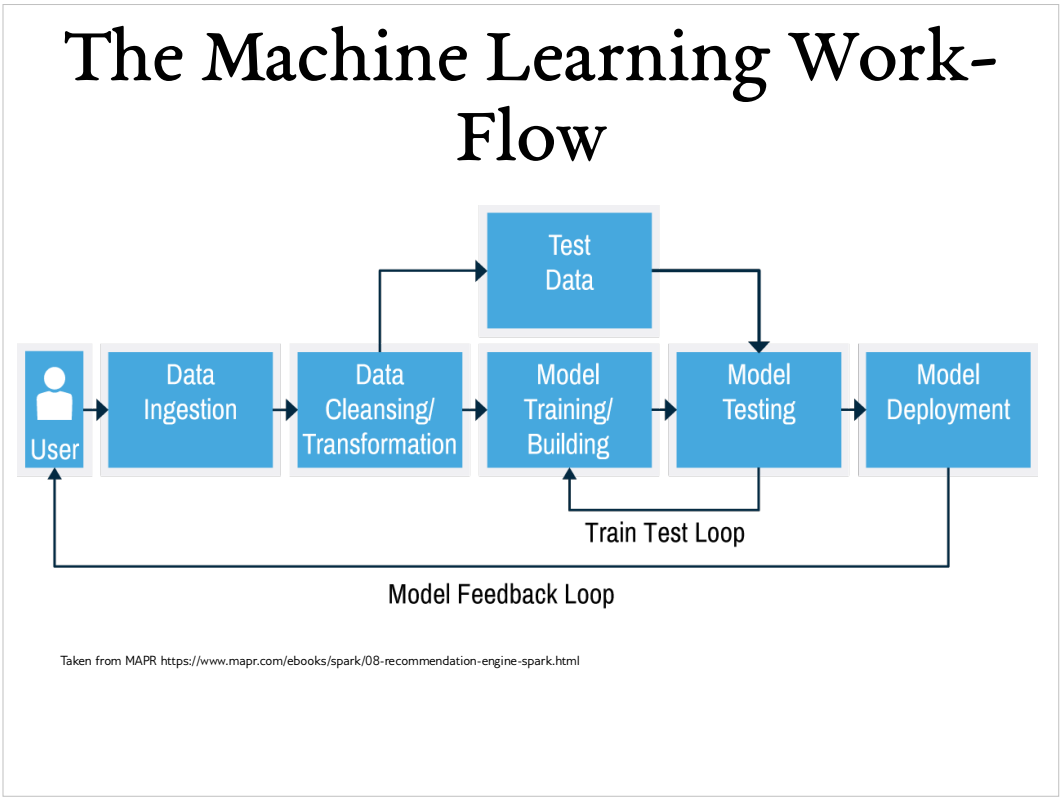
- Good software engineering practices
- Problem definition and success measures
- Feature engineering and data cleaning
- Strength and weaknesses of different algorithms
- Model selection best practices

So finally some notes about my philosophy for this course. There are a couple of things I want to emphasize. One is good software engineering practices. Machine learning systems are usually more complex than traditional software, and so good software engineering is even more important, and that's where we'll start in the next lecture.

One thing that we discussed today and that will also be part of the assignments is problem formulation and choosing the right evaluation measure. That's something that's critical in the real world, but not part of your usual course or ml competition.

There are things that are often not taught in class, but you can learn at machine learning competitions are feature engineering, data cleaning and model choice.

And it's not only important to know which models to try, it's also important to apply best practices in model selection to ensure your models will perform well in the real world.



We will be working on all of these different aspects of machine learning, but it's good to have the overall picture of the process in mind.

I stole this illustration from a book on recommendation engines, but I mostly agree with it.

Everything starts with the data generating process, here it's the user. You ingest the data, and preprocess it. Some of the preprocessed that is held out for model validation. The rest is used for actually building the model. Then, there's the model debugging loop. You get results and you want to improve them. That's what's shown as the train-test-loop. I would argue sometimes this loop goes further back, and maybe you discover an issue as early as in the ingestion. In the end, you find a model that you like, and you deploy it. However, that's not the end of the story! If you're working with a life system, the deployed model is likely to change user behavior, which means your input data changes, and your model might not be applicable any more! That feedback loop is really tricky, and we won't go into that in this course. It's good to keep in mind for your first startup, though.

Overview of the course

Infrastructure and basic tools (Wk 2)

- Python, Jupyter
- git, github
- testing, documentation, continuous integration
- Writing fast Python
- Numpy, matplotlib, pandas, seaborn (?)

Exploratory analysis & Supervised learning (week 3)

- Visualization techniques
- Exploration questions
- Generalization in practice
- Nearest Neighbors, Nearest Centroid

Linear Models (Week 4)

- Regression:
  Linear regression, Ridge Regression, Lasso
- Classification:
  Linear SVM, Logistic Regression
- Penalties, complexity and features

Preprocessing And Feature Engineering (Week 5)

- Scaling, normalization
- Variable types
- Binning, discretization, aggregation
-

Feature Selection, Model validation (Week 6)

- Statistics for feature selection
- Model-driven feature selection
- Mutual-Information based feature selection
- Cross-validation
- Grid-Search

Week 7 & 8

- Non-linear support vector machines
- Support Vector regression
- Decision Trees
- Random Forests
- Gradient Boosting

Model evaluation and Introspection (Wk 10)

- Model interpretation
- Variable Importance
- Model evaluation metrics
- Imbalanced datasets
- Model debugging

Unsupervised learning (Week 11)

- Decomposition and dimensionality reduction:

  PCA, NMF, Sparse Coding, Manifold Learning

- Clustering:

  k-means, DBSCAN, Gaussian Mixture Models, Agglomerative Clustering, Spectral Clustering

Text Data (Week 12)

- N-grams and Bag of Words models
- Preprocessing text
- Word Vectors
- Text classification
- Topic models

Neural Networks (Week 13)

- Backpropagation
- Tensorflow
- Learning Algorithms and tuning them

Common Data Types (week 14 & 15)

- Image data and image tasks
- Convolutional neural networks for image analysis
- Time series data
- Forecasting and time-series models
- Gaussian Processes (?)