APMA E4990.02: Introduction to Regression

Lecture 2

Checklist

- Finish the basic UNIX example I made an announcement for?
- Install Github, Anaconda, Cygwin (if using Windows).
- Clone Github repo AND repo for the assignment - see Homework 1 in the homework folder. Due next week.

Unix Importance

- All production level data science code is run on Linux/Unix servers. The scripts used to pull in data, merge it and run models is all run via scripts on these machines.

- Commonly need to move, edit files on remote servers, scp various pieces of code/data, start AWS clusters, etc. All of this requires basic knowledge of the command line.

- My opinion is that learning without doing is pointless, so we will learn Linux by example when you set up your own web server.

Unix Knowledge

File manipulation first
- mkdir, ls, cd, touch, mv (-r), cp (-r)

Making life on the command line easier
- emacs shortcuts on the command line
- up arrow, CTRL-R history search, CTRL-E, CTRL-U

A terminal based editor
- pico, nano, vim or emacs.

Text manipulation and display
- cat head tail grep (-r)

Network
- curl, wget

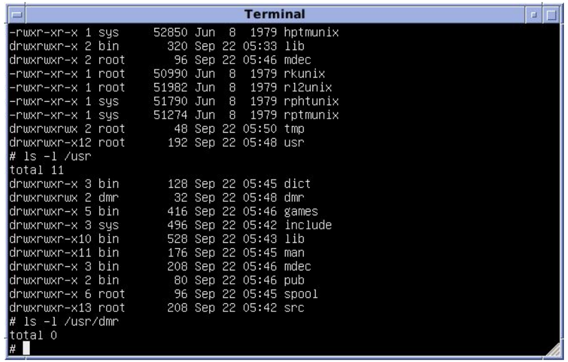
STDIN/OUT
- redirection (>1 >2),

Process chaining and standard in/out
- pipe (|), redirecting

Process manipulation
- bg, CTRL-Z, CTRL-C
- environmental variables
- ps (list of processes)
- what is TOP
- lsof (what files are open by what processes)

Common tasks:
- operating on a bunch of files using xargs or a loop
- shell scripting
- .bashrc file
- working on remote machines with sshd
- awk/sed for file processing.

http://ryanstutorials.net/linuxtutorial/cheatsheet.php



Basic Navigation

pwd
Where am I in the system.

ls [path]
Perform a listing of the given path or your current directory.

Common options: -l, -h, -a

cd [path]
Change into the given path or into your home directory.

Path
A description of where a file or directory is on the filesystem.

Absolute Path
One beginning from the root of the file system (eg. /etc/sysconfig).

Relative Path
One relative to where you currently are in the system (eg. Documents/music).

~ (tilde)
Used in paths as a reference to your home directory (eg. ~/Documents).

. (dot)

Used in paths as a reference to your current directory (eg. ./bin).

.. (dot dot)
Used in paths as a reference to your current directories parent directory (eg. ../bin).

TAB completion
Start typing and press TAB. The system will auto complete the path. Press TAB twice and it will show you your alternatives.

More About Files

file [path]
Find out what type of item a file or directory is.

Spaces in names
Put whole path in quotes (") or a backslash (\ ) in front of spaces.

Hidden files and directories
A name beginning with a . (dot) is considered hidden.

Permissions

r (read) w (write) x (execute)

Owner or User, Group and Others

ls -l [path]
View the permissions of a file or all items in a directory.

chmod <permissions> <path>
Change permissions. Permissions can be either shorthand (eg. 754) or longhand (eg. g+x).

Manual Pages

man <command>
View the man page for a command.

man -k <search term>
Search for man pages containing the search term.

Press q to exit man pages

Vi / Vim

View our Vim Cheat sheet

Filters

head
Show the first n lines.

tail
Show the last n lines.

sort
Sort lines in a given way.

wc
How many words, characters and lines.

grep
Search for a given pattern.

See more on our Grep Cheat sheet

More filters can be found here.

File Manipulation

mkdir <directory name>
Create a directory

rmdir <directory name>
Remove a directory (only if empty).

touch <file name>
Create a blank file.

cp <source> <destination>
Copy the source file to the destination.

mv <source> <destination>
Move the source file to the destination.

May also be used to rename files or directories.

rm <path>
Remove a file or directory.

Common options: -r -f

Wildcards

May be used anywhere in any path.

*****
Zero or more characters (eg. b*).

?
Single character (eg. file.???).

[ ]
Range (eg. b[aio]t).

Wildcards

May be used anywhere in any path.

*****
Zero or more characters (eg. b*).

?
Single character (eg. file.???).

[ ]
Range (eg. b[aio]t).

Piping and Redirection

>
Redirect STDOUT to a file.

>>
Append STDOUT to the end of a file.

2>
Redirect the STDERR to a file.

<
Pass the contents of a file to a program as STDIN.

|
Feed the STDOUT of the program on the left as STDIN to the program on the right.

Process Management

CTRL + C
Cancel the currently running process.

kill <process id>
Cancel the given process.

Include the option -9 to kill a stubborn process.

ps
Obtain a listing of processes and their id's.

Including the option aux will show all processes.

CTRL + Z
Pause the currently running process and put it in the background.

jobs
See a list of current processes in the background.

fg <job number>
Move the given process from the background to the foreground.

Review from Last Time

- Introduced examples of machine learning applications and methods.
    - Used examples from Amazon, Netflix, The New York Times, Booking.com and Tinder.
    - Discussed **Predictive**, **Descriptive** and **Prescriptive** methods of machine learning.
    - Linear Regression, Decision Trees, Graph Diffusion models.
- How do we learn from data?
    - Overall pipeline: Data pulling -> Merging/Cleaning -> Model Training -> Evaluation.
    - MNIST digit recognition via KNN.
    - Logistic Regression for churn at The New York Times.
- Model Evaluation and Complexity
    - Cross Validation - having a hold-out set for testing.
    - Hyperparameter optimization and overfitting. How complex is “just right”?

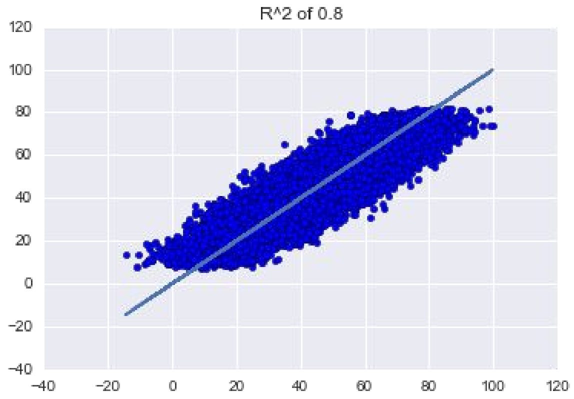
Outline

- Introduction to Linear Regression
  - Cross Validation reminder - the test/train split.
  - Definition, and comparison of Lp norms.
  - How do we find the solution? The special case of p=2.
  - When do unique solutions exist?
  - The problem of dependent features.
- General Solution Finding Procedure - Gradient Descent
  - Gradient Descent, and Introduction to Convex Optimization.
  - Concrete Examples of Linear Regression and Gradient Descent in Python in an iPython Notebook.
- Introduction to Decision Trees
  - How decision trees are constructed - variance reduction.
  - Concrete Examples of Decision Trees in Python with an iPython Notebook.

Introduction to Regression

Given a collection of points to learn from: (x_i, y_i)

Can we find a function $f : X \to Y$ minimizing the distance to the data.



$$\mathcal{L}(y, f) := \sum_{i=1}^{N} d(y_i, f(x_i))$$

$d(x, y)$ - Is a metric (distance between two points)

Testing/Training Split

We always learn from training data

Splitting data into Testing and Training

In [72]: `X_train.head()`

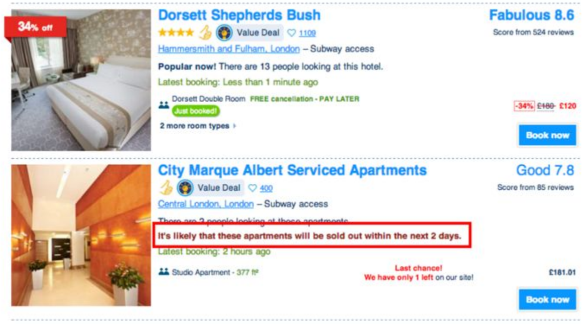
Out[72]:

	account	const	hotel_rating	location	price_per_night_avg	purchase_velocity_lastweek	rooms_left	sellouts_total
0	34961	1	8.9	8.5	409	232	226	3
1	25510	1	4.5	7.1	189	325	265	4
2	95668	1	7.7	6.0	377	287	475	4
3	38365	1	1.7	8.9	350	205	289	8
4	51131	1	0.6	5.2	542	301	66	2

In [73]: `X_test.head()`

Out[73]:

	account	const	hotel_rating	location	price_per_night_avg	purchase_velocity_lastweek	rooms_left	sellouts_total
40	33846	1	7.8	3.1	294	85	371	1
41	17771	1	7.8	6.8	572	377	384	6
42	10967	1	0.9	6.9	191	95	79	8
43	37812	1	2.9	3.2	418	65	4	5
44	61968	1	3.5	4.2	222	201	356	3



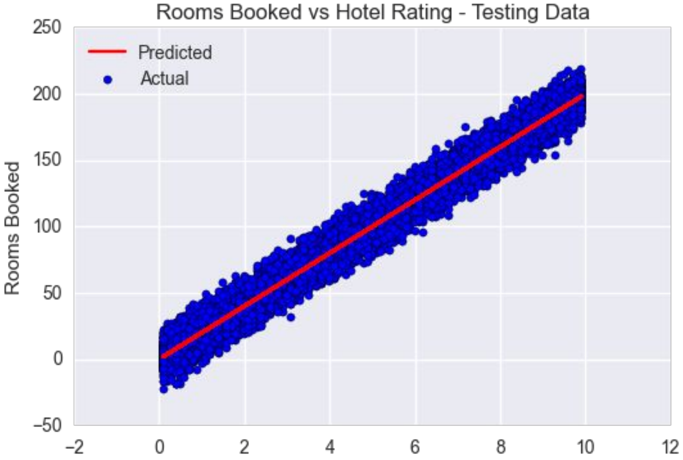
Let y be the number of rooms booked at a hotel tomorrow.

- First, we randomly split our data into **testing and training subsets.**
- The most important thing in machine learning is **generalizability** - we must be able to **make predictions on unseen data** that works.
- If you make a model **complex enough**, you can **often get very good accuracy** on the **training data**, but it **won't generalize** (next lecture).

Splitting data into Testing and Training

In [72]:
```
X_train.head()
```
Out[72]:

	account	const	hote
0	34961	1	8.9
1	25510	1	4.5
2	95668	1	7.7
3	38365	1	1.7
4	51131	1	0.6



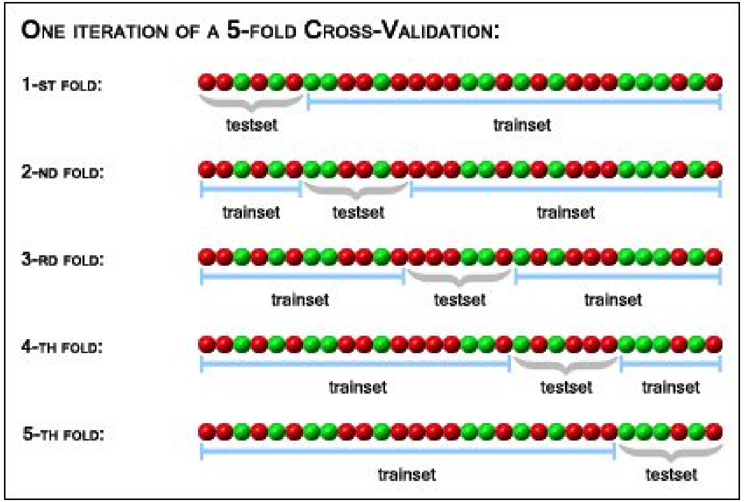
In [73]:
```
X_test.head()
```
Out[73]:

	account	const	hotel_
40	33846	1	7.8
41	17771	1	7.8
42	10967	1	0.9
43	37812	1	2.9
44	61968	1	3.5

- We must use the **training data to learn the model.**
- Then from the model we've learned, we **evaluate it on the testing data.**

$$y = 20 * \text{hotel-rating} + \text{noise}$$

Cross Validation



- Generally instead of just taking say **80% of your data for training**, **20% for testing (for example)**, we randomly split the data into several 'folds'.

- In **k-fold cross-validation**, the original sample is randomly partitioned into **k equal sized subsamples**. Of the k subsamples, **a single subsample is retained as the validation data** for testing the model, and t**he remaining k − 1 subsamples are used as training data.**

Linear Regression

The special case of Lp

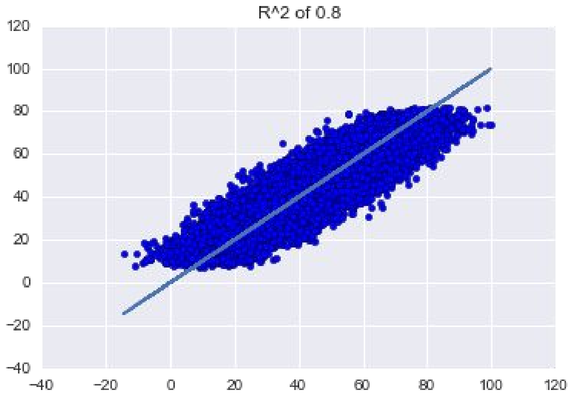
$$\mathcal{L}(y, \beta \cdot x) = \frac{1}{N} \sum_{i=1}^{N} |y_i - \beta \cdot x_i|^p$$

Most common case:

- p=2
  - Penalizes outliers much more.
  - Less sparse coefficients.
  - Has an **analytical solution.**
  - Unique solution when features are linearly independent.
  - Compatible with CLT (will see later!)
- p=1
  - More sparse coefficients (zero valued).
  - Helps eliminate collinear features.
  - Degenerate solutions.
  - Deals with outliers better (why?)

Linear assumption:

$$f(x_i) = \beta \cdot x_i$$



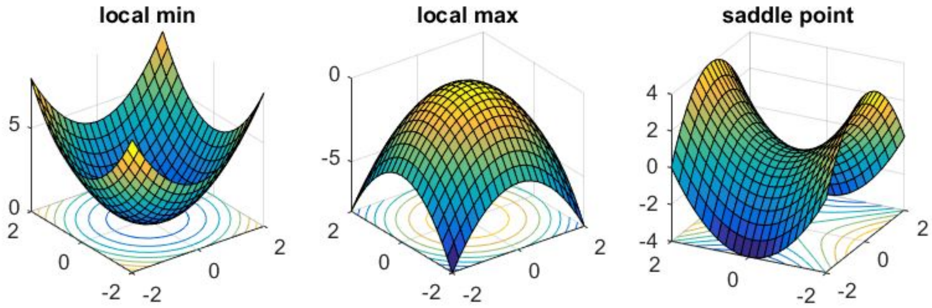
Question: Why would $p = \infty$ be a bad choice for machine learning?

Convex Analysis (Calculus)

- Recall from last time that our goal is often find some convex function $\mathcal{L} : \mathbf{S} \rightarrow \mathbb{R}$ such that we learn the rules of our model by its minimization (ie. coefficients). These are referred to as **parametric methods.**

- We can also "fit" data to a probability by **maximizing** the **likelihood** - concave problem.

- This is **not always the case**, with algorithms such as **decision trees** or **neural nets**. However, it covers many cases in supervised learning. These are called **non-parametric methods** (to be covered in this class or next).



Convex Analysis (Calculus)

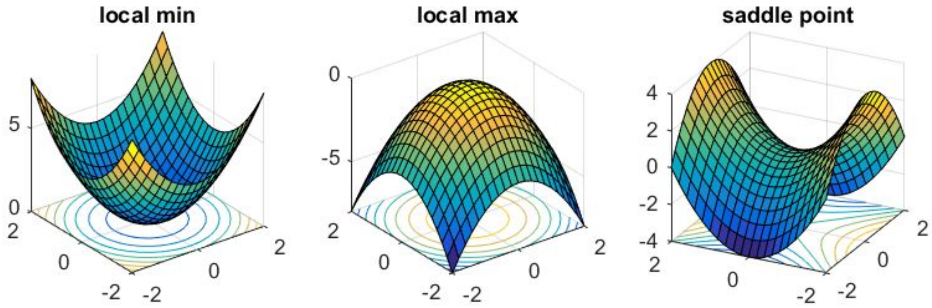
- Assume $\mathcal{L} : \mathbf{S} \rightarrow \mathbb{R}$ is continuously differentiable and convex, ie.

$$\mathcal{L}''(\beta) \geq 0 \text{ for all } \beta \in \mathbf{S}$$

$$\exists \beta_0 \text{ such that } \mathcal{L}(\beta_0) \leq \mathcal{L}(\beta) \text{ for all } \beta \in S$$

- There f has a minimum value. Moreover, the minimum is unique when f is strictly convex, ie.

$$\mathcal{L}''(\beta) \geq c > 0 \text{ for all } \beta \in \mathbf{S}$$



Ordinary Least Squares (p=2)

Analytical Solutions and Stability Analysis

Analytical Solution to OLS when p=2

$$\mathcal{L}(\beta) = \frac{1}{N}\sum_{i=1}^{N} |y_i - \beta \cdot x_i|^2$$

Claim: When p = 2 and the matrix $X^T X$ is invertible, the above is minimized uniquely by

$$\beta = (X^T X)^{-1} X^T y$$

The solution can make sense when $X^T X$ isn't invertible (in practice it often can be "almost not invertible", but solutions will be unstable and degenerate (more later)

Analytical Solution to OLS when p=2

Proof:

$$\mathcal{L}(\beta) = \frac{1}{N}\sum_{i=1}^{N} |y_i - \beta \cdot x_i|^2$$

Differentiating, we have

$$\frac{\partial \mathcal{L}}{\partial \beta_j} = -\frac{2}{N}\sum_{i=1}^{N}(y_i - \beta \cdot x_i)x_j$$

$$\frac{\partial \mathcal{L}}{\partial \beta_j} = 0 \Rightarrow \boxed{\frac{1}{N}\sum_{i=1}^{N}(\beta \cdot x_i)x_{ij} = \frac{1}{N}\sum_{i=1}^{N} y_i x_{ij}.} \Rightarrow \quad X^T X \beta = X^T y.$$

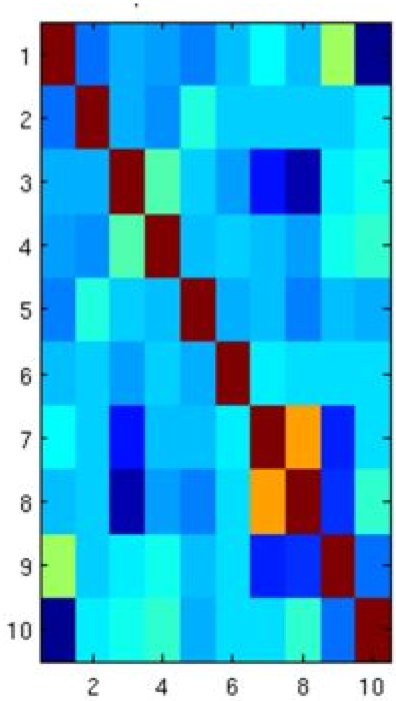
$$\Rightarrow \qquad \beta = (X^T X)^{-1} X^T y$$

- Need linearly independent features for a unique inversion.

Analytical Solution to OLS when p=2

It easily follows that $$\frac{d^2\mathcal{L}}{d^2\beta} = \frac{2}{N} X^T X$$

- From this, it follows there is a **unique solution** iff the **eigenvalues** of the above matrix are **strictly positive**.
- This occurs precisely when **X has linearly independent features**.
- If X is **mean centered and normalized**, the above matrix is equivalent to the **correlation matrix**.



Recall the Advertisement Example

What are the features?

TV: advertising dollars spent on TV for a single product in a given market (in thousands of dollars)

Radio: advertising dollars spent on Radio

Newspaper: advertising dollars spent on Newspaper

Goal: Predict the number of sales in a given market based on the advertising in TV, Radio and Newspaper.

In [263]:
```
# read data into a DataFrame
import pandas as pd
import pylab as plt
import seaborn
from sklearn.linear_model import LinearRegression
import numpy.random as nprnd
import random
import json
pd.set_option('display.max_columns', 500)
%matplotlib inline

df = pd.read_csv('http://www-bcf.usc.edu/~gareth/ISL/Advertising.csv', index_col=0)
df.head()
```

Out[263]:

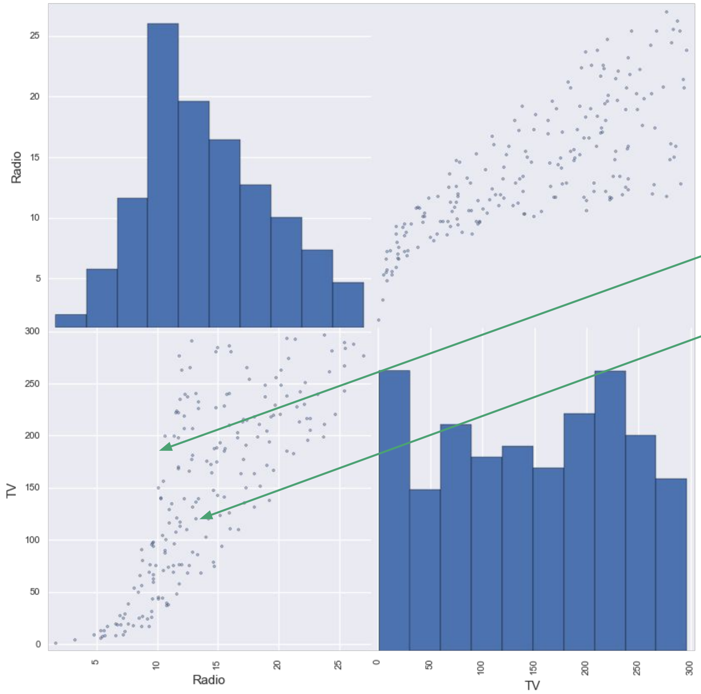
	TV	Radio	Newspaper	Sales
1	230.1	37.8	69.2	22.1
2	44.5	39.3	45.1	10.4
3	17.2	45.9	69.3	9.3
4	151.5	41.3	58.5	18.5
5	180.8	10.8	58.4	12.9

Understanding correlation

Note: This is not the same as the data you will do for the homework!

$$\mathbb{R}^{2\times n} \times \mathbb{R}^{n\times 2}$$

$$X^T X = \begin{bmatrix} 150 & 160 & \cdots \\ 17 & 12 & \cdots \end{bmatrix} \begin{bmatrix} 150 & 17 \\ 160 & 12 \\ \cdots & \cdots \end{bmatrix}$$



$$\overset{\mathbb{R}^{2\times 2}}{[X^T X]_{ij}} = \mathbf{x_i} \cdot \mathbf{x}_j$$

$$i = \text{ Radio}$$

$$j = \text{ TV}$$

Vectors are in n dimensions (number of data points)

$$[\mathrm{Corr}]_{ij} = \frac{\mathbf{x_i} \cdot \mathbf{x}_j}{\|\mathbf{x_i}\| \|\mathbf{x_j}\|} = \cos(\delta_{ij})$$

Eigenvalues of Covariance Matrix

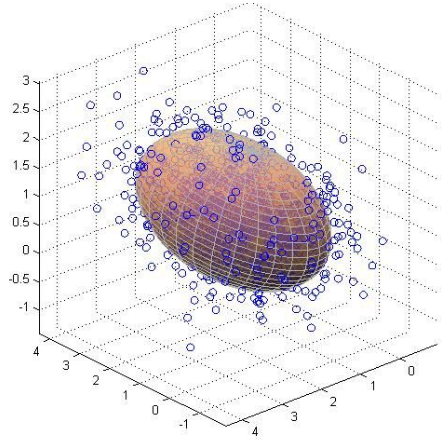
$$\lambda_1 = \max_{\|y\|=1} y^T A y$$

$$y^T A y = \langle y, Ay \rangle$$

$$y = v + \epsilon w$$

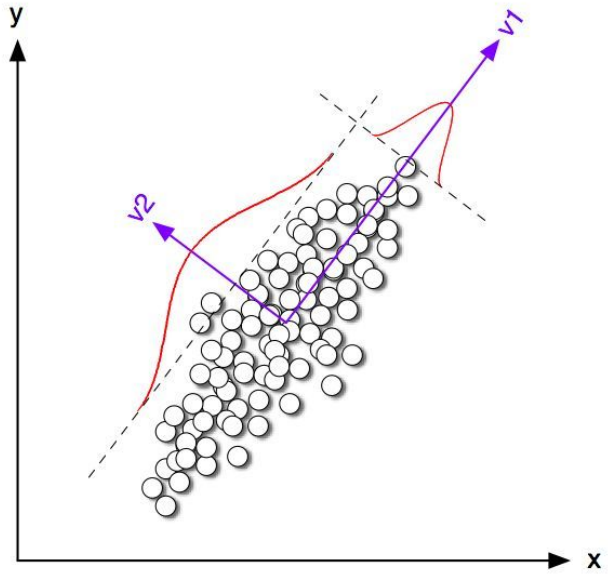
$$\nabla_\epsilon (y^T A y)\Big|_{\epsilon=0} = \langle w, Av \rangle + \langle v, Aw \rangle = \lambda \langle v, w \rangle$$

$$2\langle Av, w \rangle = \lambda \langle v, w \rangle \longrightarrow Av = \tilde{\lambda} v$$



The others are obtained by maximizing in the orthogonal complement to the vector v.

Eigenvalues and Instability



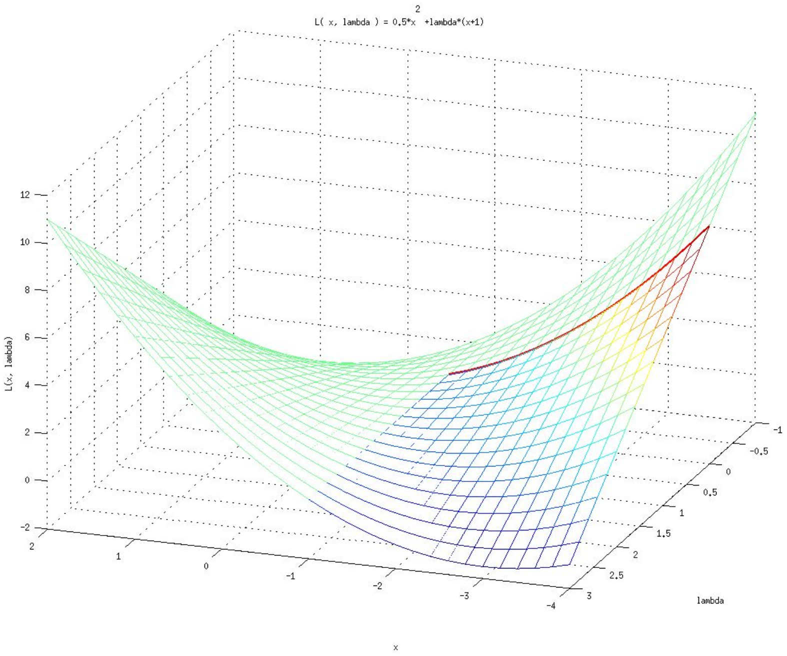
- Strongly correlated variables result in an orthogonal direction with a small eigenvalue.
- This creates the instability we have been discussing.

$$Av = \tilde{\lambda} v$$

Dependent features - what goes wrong

Why we want to eliminate collinear/dependent features

Correlation results in instability



- Flat directions in the feature space mean less stability and more uncertainty about what the true coefficients are!

- When we see gradient descent, the parameters involved can significantly affect solutions when there is no stability.

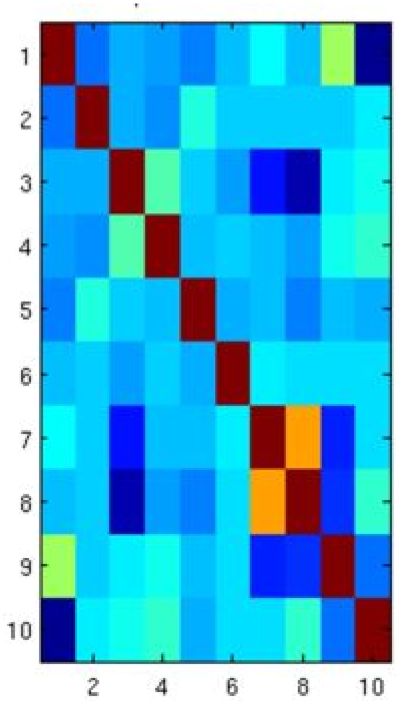
$$\frac{d^2\mathcal{L}}{d^2\beta} = \frac{2}{N}X^TX$$

Correlation results in instability

- X^TX is **symmetric** and therefore has **nonnegative eigenvalues.**

- They are **positive** precisely when the features of X are **linearly independent.**

- Let's assume for simplicity that **X is mean centered (fine but sometimes reasons why you might not).**



Imagine that X has two columns which are factors of one another. What can go wrong?

$$y = \alpha x_1 + \beta x_2$$

But our real rule is: $y = 5x_1$

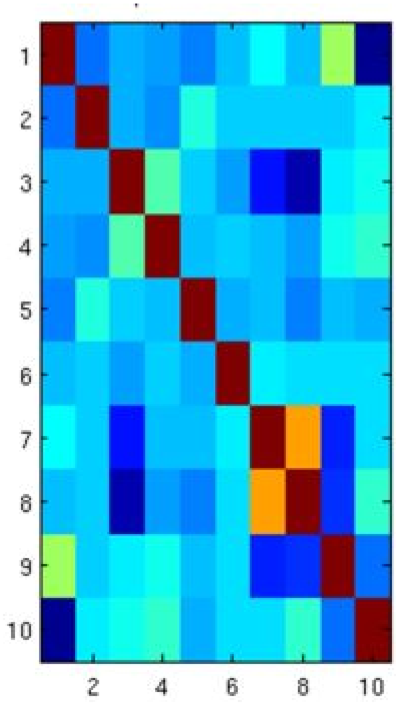
Correlation results in instability

$$y = \alpha x_1 + \beta x_2$$

But our real rule is: $y = 5x_1$

Then $y = -1000x_1 + 10005x_2$ is also a solution

Why is this a problem?

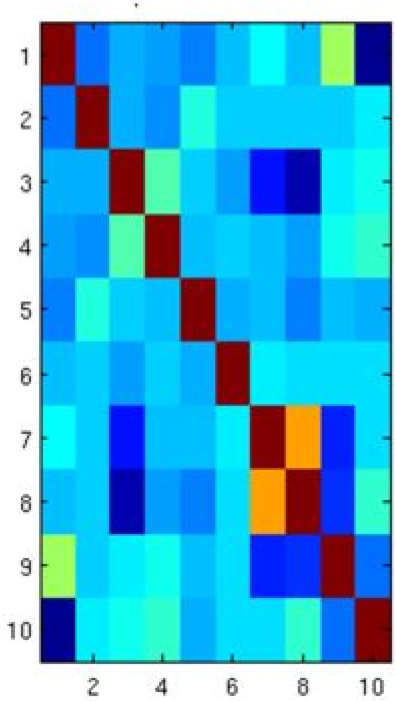


Correlation results in instability

$$y = \alpha x_1 + \beta x_2$$

But our real rule is: $y = 5x_1$

Then $y = -1000x_1 + 10005x_2$ is also a solution



Why is this a problem?

- Creates problems when finding the minimum (next).
- Causes more uncertainty in coefficient estimates (after gradient descent).

Gradient Descent

How do we minimize a function when there is no analytical solution?

The New York Times

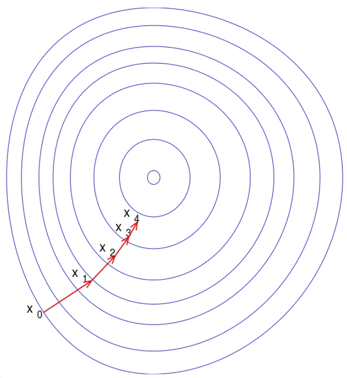
We can't always solve analytically!

The logistic regression model for churn at The New York Times is **parametric:**

$$p_\beta(\mathbf{x}) = \frac{1}{1 + \exp(-\beta^T \cdot \mathbf{x})}$$

$$\text{Cost}_2(p_\beta(\mathbf{x}), y) = \begin{cases} -\log p_\beta(\mathbf{x}) & \text{if } y \text{ is } 1 \\ -\log(1 - p_\beta(\mathbf{x})) & \text{if } y \text{ is } 0 \end{cases}$$

$$\min_\beta \frac{1}{N} \sum_{i=1}^{N} \text{Cost}_2(p_\beta(\mathbf{x}_i), y_i)$$



We minimize this convex cost function via gradient descent.

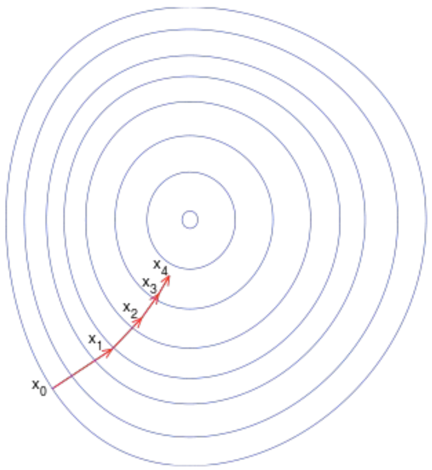
- When y is 1, $p_\beta(\mathbf{x})$ tries to be close to 1
- When y is 0, $p_\beta(\mathbf{x})$ tries to be close to 0.

$$\mathbf{x} = [\text{days\_active}, \text{pages\_viewed}]$$
$$y = 1 \text{ if churned}$$
$$y = 0 \text{ if retained}$$

Continuous Gradient Descent

Assume that $\beta(t)$ solves the equation:

$$\dot{\beta}(t) = -\nabla \mathcal{L}(\beta(t))$$



And that: $\beta \mapsto \mathcal{L}(\beta)$

- Is strictly convex.
- Twice differentiable.

Then: $\beta(t)$ converges to the minimum of f exponentially fast.

Proof of Convergence of Gradient Descent

Differentiating and using the gradient flow, we have

$$\frac{d}{dt}(\beta(t) - \beta_0)^2 = -2\nabla\mathcal{L}(\beta(t))(\beta(t) - \beta_0)$$

Next: How do we use strict convexity to obtain an estimate?

***Note*:** In reality we alway shave a discrete version of the above, and must choose a ‘learning rate’ (ie. time step size) - we will gloss over this for now.

Proof of Convergence of Gradient Descent

Using Taylor's Law we have

$$\mathcal{L}(\beta) - \mathcal{L}(\beta_0) = \nabla\mathcal{L}(\beta) \cdot (\beta - \beta_0) + \frac{1}{2}(\beta - \beta_0)^T D^2\mathcal{L}(\xi)(\beta - \beta_0)$$

$$\mathcal{L}(\beta_0) - \mathcal{L}(\beta) = \nabla\mathcal{L}(\beta_0) \cdot (\beta_0 - \beta) + \frac{1}{2}(\beta - \beta_0)^T D^2\mathcal{L}(\tilde{\xi})(\beta - \beta_0)$$

Therefore:

$$-\nabla\mathcal{L}(\beta - \beta_0) = -(\beta - \beta_0)^T(D^2\mathcal{L}(\xi) + D^2\mathcal{L}(\tilde{\xi}))(\beta - \beta_0) \leq -M|\beta - \beta_0|^2$$

$$\frac{d}{dt}(\beta(t) - \beta_0)^2 = -2\nabla\mathcal{L}(\beta(t))(\beta(t) - \beta_0)$$

$$\frac{d}{dt}(\beta - \beta_0)^2 \leq -m|\beta - \beta_0|^2$$

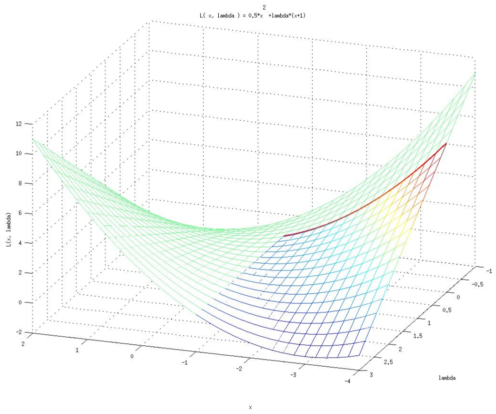
$$\Rightarrow |\beta(t) - \beta_0)| \leq Ce^{-mt}$$

Proof of Convergence of Gradient Descent

$$-\nabla\mathcal{L}(\beta-\beta_0) = -(\beta-\beta_0)^T(D^2\mathcal{L}(\xi)+D^2\mathcal{L}(\tilde{\xi}))(\beta-\beta_0) \leq -M|\beta-\beta_0|^2$$

$$\Rightarrow |\beta(t)-\beta_0)| \leq Ce^{-mt}$$

Stability is directly related to how large the eigenvalues are of the correlation matrix!



How is it computed in python?

$$\frac{\partial \mathcal{L}}{\partial \beta_j} = -\frac{2}{N}\sum_{i=1}^{N}(y_i - \beta \cdot x_i)x_j$$

$$\beta_n = \beta_{n-1} - \kappa \nabla_\beta \mathcal{L}(\beta_{n-1})$$

κ is known as the learning rate - when computing we must choose a time step size - Homework 1 covers this.

```
def step_gradient(b_current, m_current, points, learningRate):
    b_gradient = 0
    m_gradient = 0
    N = float(len(points))
    for i in range(0, len(points)):
        x = points[i, 0]
        y = points[i, 1]
        b_gradient += -(2/N) * (y - ((m_current * x) + b_current))
        m_gradient += -(2/N) * x * (y - ((m_current * x) + b_current))
    new_b = b_current - (learningRate * b_gradient)
    new_m = m_current - (learningRate * m_gradient)
    return [new_b, new_m]
```

But if the learning rate is too big then, gradient descent may overshoot the minimum and oscillate back and forth. Why?

The importance of gradient descent.

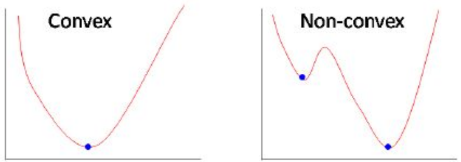
- All **parameric problems** in predictive machine learning seek to **minimize the distance of some a priori function of the data to the observed values.**

- It only makes sense to try to **minimize functions** which are **convex** (at least locally). Otherwise it's common to get stuck in local minima.

- When there is no analytical solution, the solution must be obtained by following the **steepest descent from a starting point to the minimum of the function.**

- This will be true when we work with more advanced probabilistic methods as well, and more important to understand.

- Stability is directly related to the eigenvalues of the covariance matrix, and also depends on how we choose our learning rate.



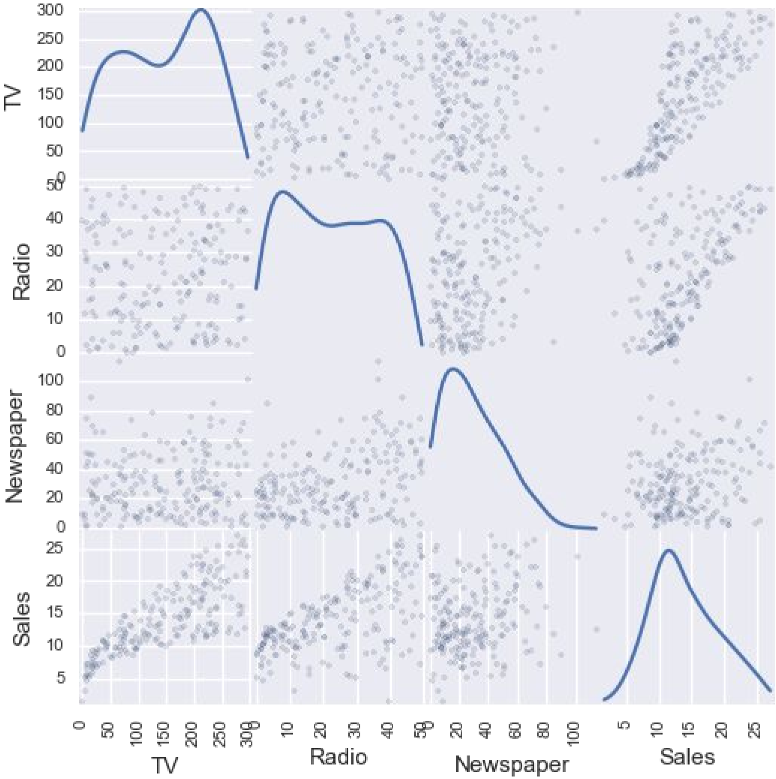
Coefficient Estimate Confidence

How do we know which variables are important, and how significant they are?

Inference vs Prediction

- Sometimes people care more about inference - which is understanding how variables affect the outcome variable.

- **Example: Does advertising on TV, Radio or Newspaper positively affect sales?**



Homework 1

Feature importance & Inference

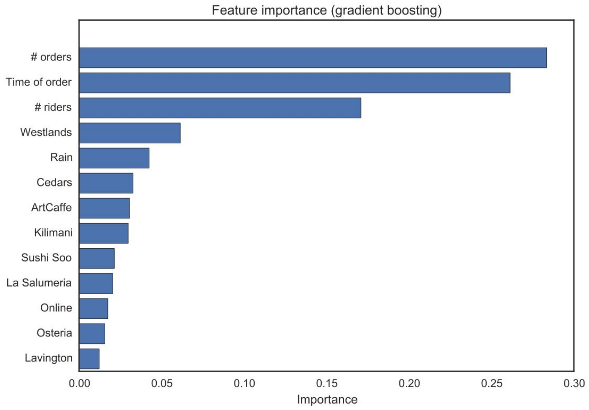
$$y_i = \sum_{j=1}^{N} \beta_j X_{ij}$$

When there is no relationship between the features, we can conclude that

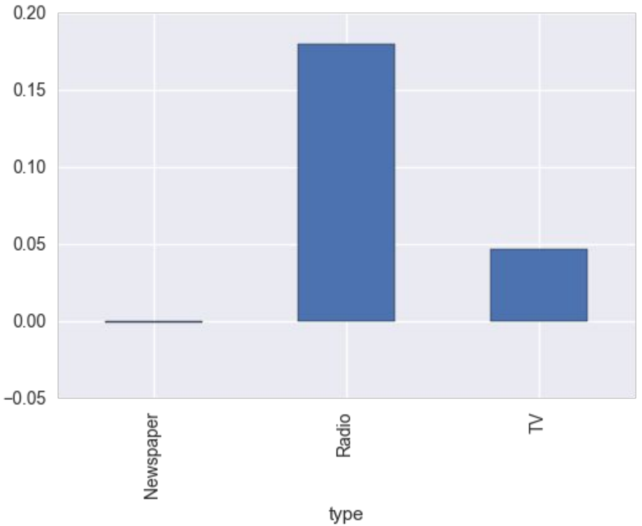
$$\frac{\partial y_i}{\partial X_{ik}} = \beta_k$$

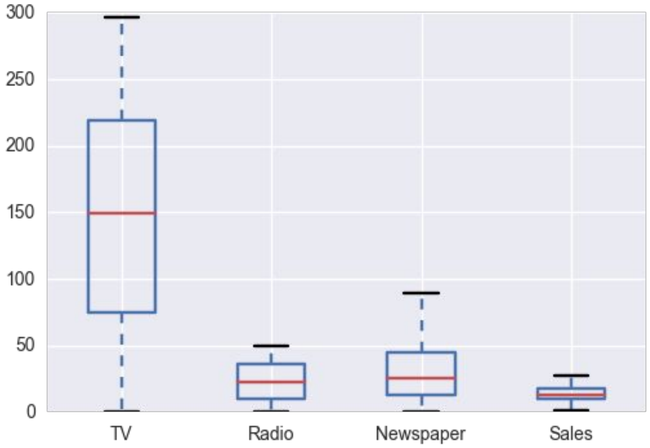
- But how does this show confidence?

- This allows us to interpret the significance of each variable, in terms of how much it changes the response variable.
- **Does not work if there is any interaction or dependence between the features however**.
- To **compare coefficients,** we need to **normalize the columns to have the same scale**.



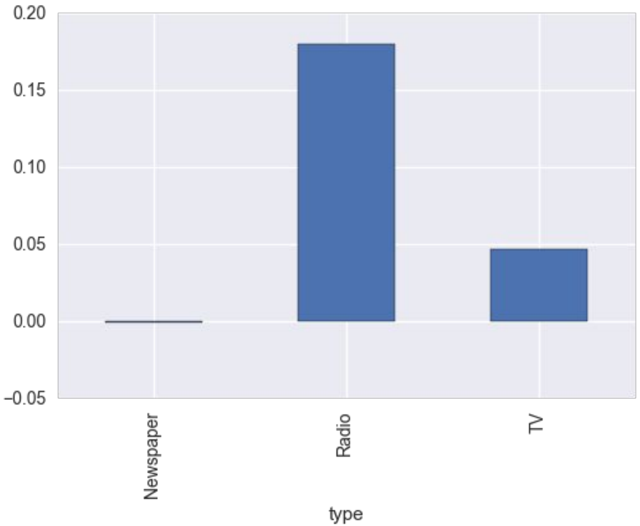
What’s wrong with this picture?



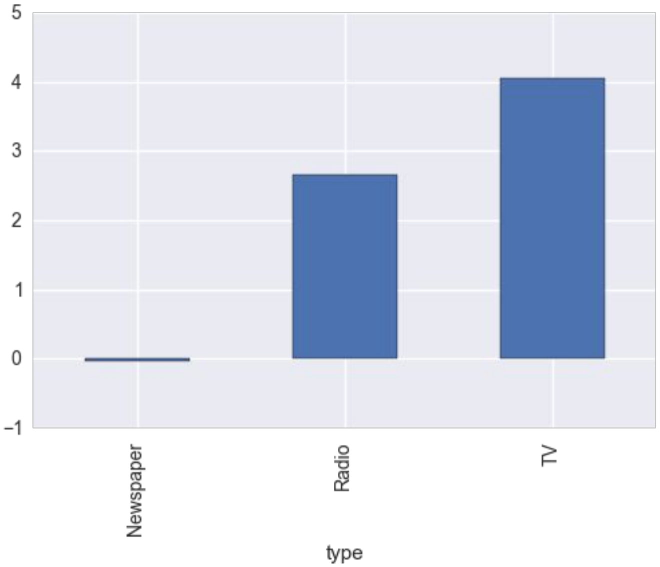


Conclusion: Radio is the most significant variable. ?

After standardizing



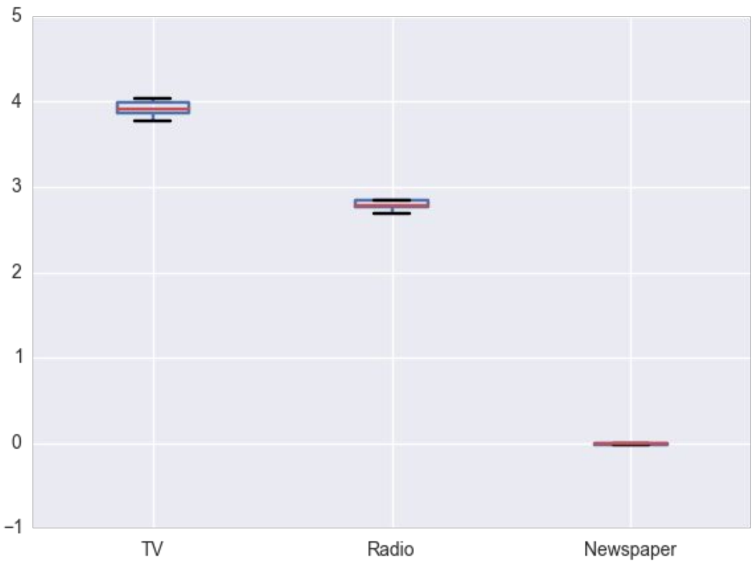
Non-Standardized



Standardized

- Each feature is normalized to have mean zero and unit variance

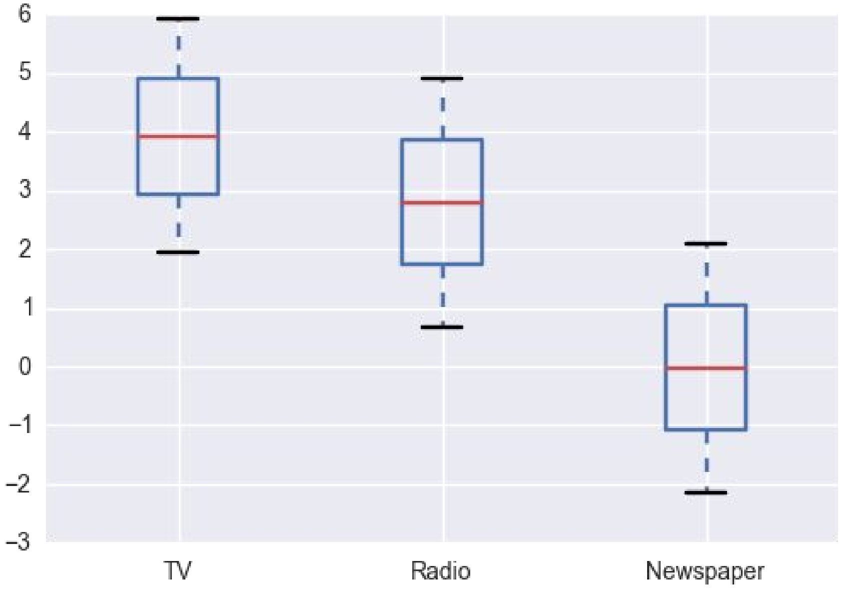
Confidence: via Cross Validation



Average over 5-fold cross validation.

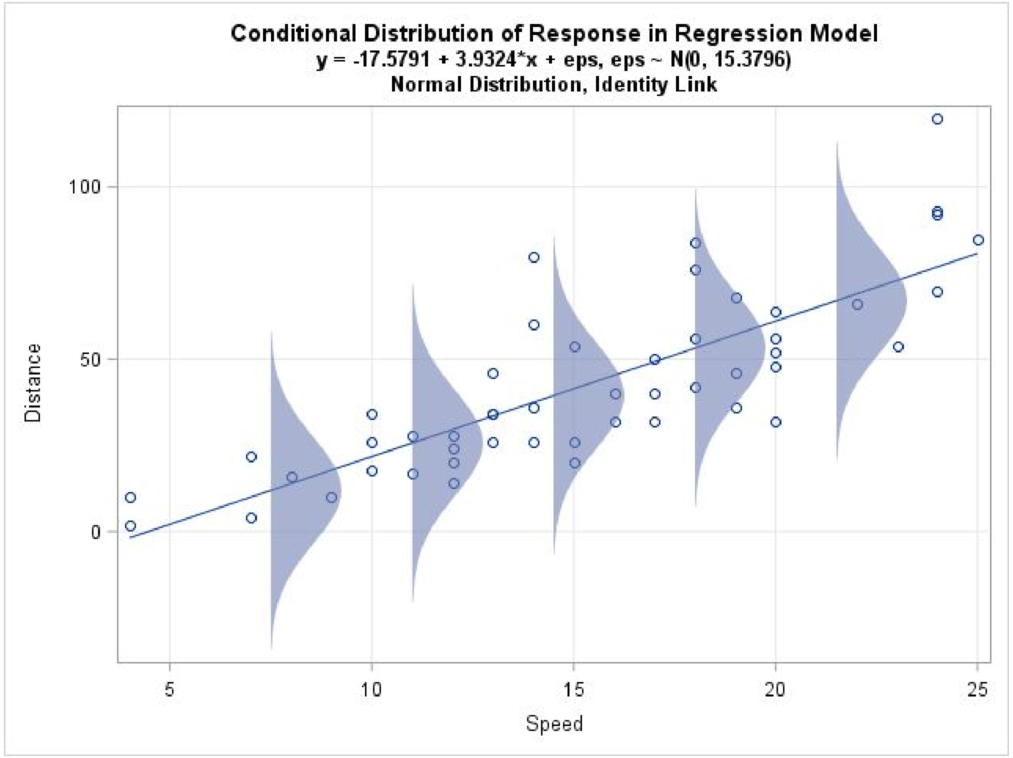
- Average your model **over many random folds, and standardize.**
- Look at the variance in the coefficients you obtain over many folds.

Confidence: via p-values



- Here the range represents a 95% confidence interval based on the standard error we computed.
- See how newspaper still seems irrelevant and we have no confidence that it’s significant.
- **Where do p values come from?**

Errors are normally distributed

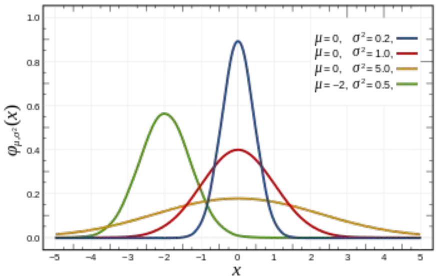


$$y_i = \beta \cdot x_i + \epsilon_i$$

$$\epsilon_i \sim \mathcal{N}(0, \sigma^2)$$

Confidence via p-values

$$y_i = \beta \cdot x_i + \epsilon_i$$



$$f(x \mid \mu, \sigma^2) = \frac{1}{\sqrt{2\sigma^2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

ϵ_i is normally distributed

$$\epsilon_i \sim \mathcal{N}(0, \sigma^2)$$

$$\hat{\beta}_1 = \frac{\sum_i x_i y_i}{\sum x_i^2}$$

$$= \frac{\sum x_i(\beta_1 x_i + \epsilon)}{\sum_i x_i^2}$$

$$= \beta_1 + \frac{\sum \epsilon_i x_i}{\sum_i x_i^2}$$

$$\frac{\sum_i \epsilon_i x_i}{\sum x_i^2} \sim \mathcal{N}\left(0, \frac{\sigma^2}{\sum x_i^2}\right)$$

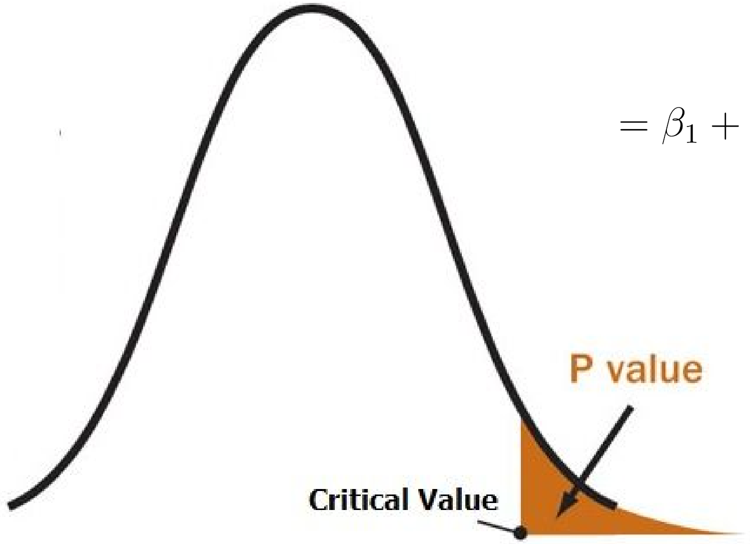
$$\mathcal{N}(0, (X^T X)^{-1}\sigma^2)$$

This is how you measure p values of regression coefficients.

Confidence via p values

$$\frac{\sum_i \epsilon_i x_i}{\sum x_i^2} \sim \mathcal{N}\left(0, (X^T X)^{-1}\sigma^2\right)$$

$$= \beta_1 + \frac{\sum \epsilon_i x_i}{\sum_i x_i^2}$$



$$t = \frac{\beta_1 - 0}{\mathrm{SE}(\beta_1)}$$

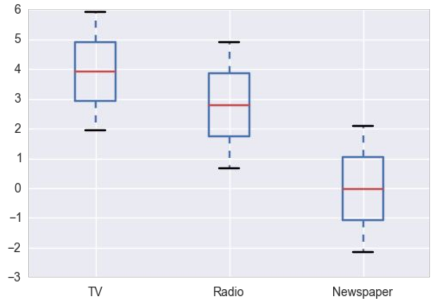
Number of standard deviations away from 0 - how rare is it to observe this value under the hypothesis H0?

H0: The coefficient is zero and has no effect on y.

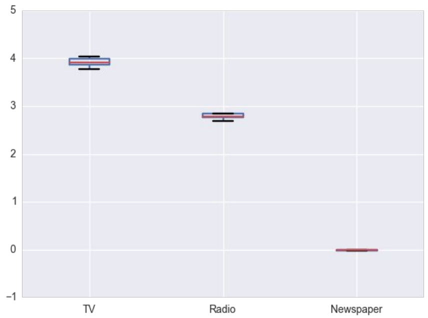
H1: The coefficient is non-zero and has an impact on y.

$$\mathrm{Var}(\beta_1) \sim \frac{1}{N}\frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (x_i - \bar{x})^2}$$

Confidence via p-values



p values



Cross validation

- Both p values and cross validation try to achieve the same goal - **to give us confidence in our parameter estimates**.
- It's good practice to both, **but at least do one!**
- Cross validation seems more common in modern machine learning applications.
- No assumptions about normality are made with cross validation.

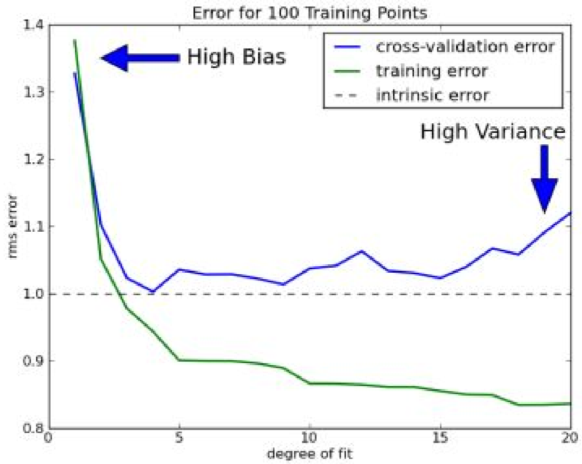
Eliminating bad variables?

$$\beta = (X^T X)^{-1} X^T y$$

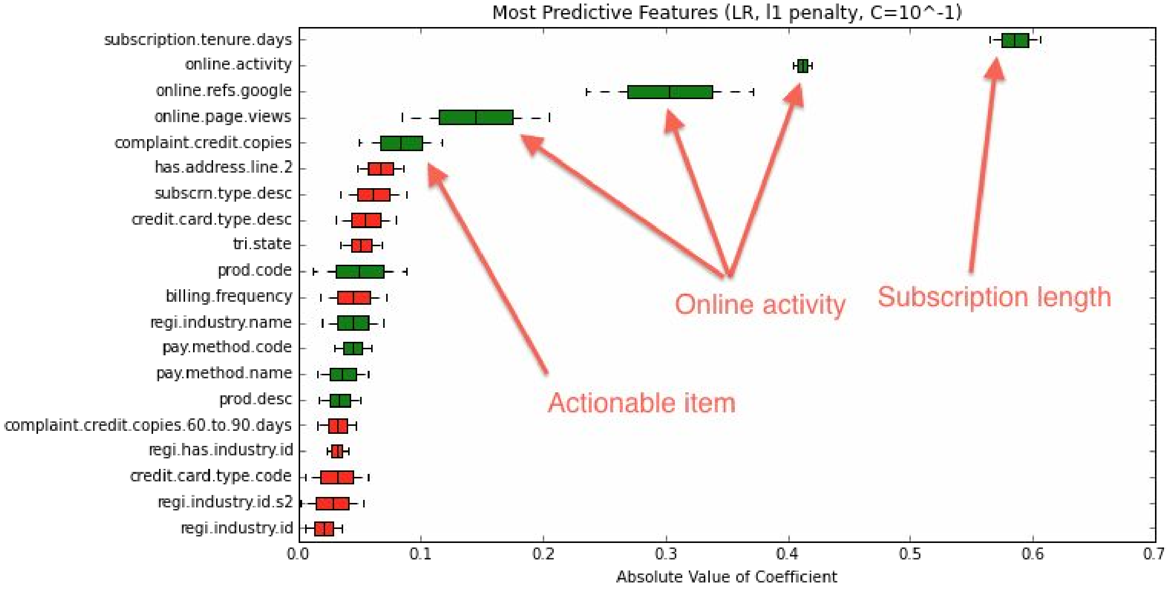
- 1) For building predictive models, the best strategy is to **optimize for performance by restricting the size of the coefficients, or dimensionality reduction.**

- 2) **If you don't have a lot of data**, **use many fold cross validation** to avoid the chance of having a spurious model.

  *Both will be covered in detail **next lecture.** *



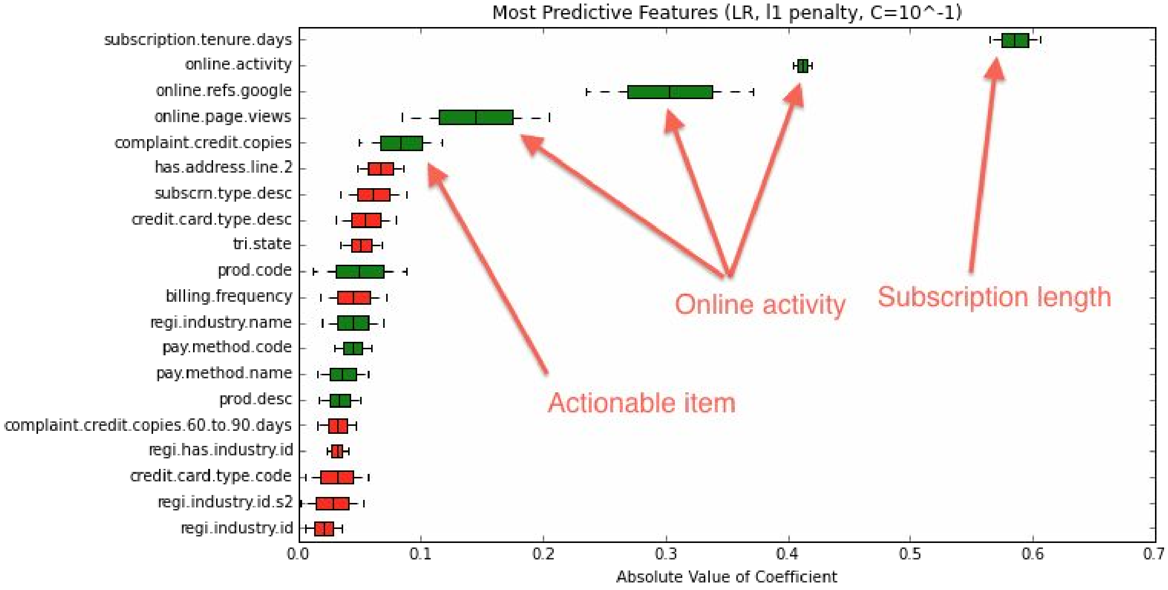
The New York Times - a real example



Example: What the widths and colors here represent do you think?

Predicting user churn

The New York Times



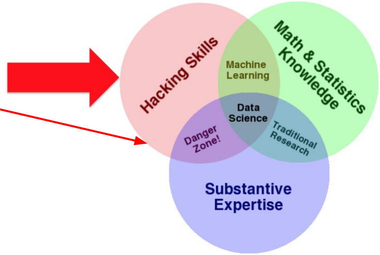
- **Approach 1:** p values represent the width (classic stats)
- **Approach 2:** Average coefficients over many folds in K-fold cross validation (data science)

Quick Summary

- Derived the analytical solution for L2 linear regression.

- Related the stability of the solution and confidence of the estimate of the coefficients to the dependence of features on one another (covariance/correlation). **P values or cross validation help to understand the confidence in our coefficients.**

- Under all of these assumptions, we can make sense of ‘feature importance’ - but we have to be very careful!



Evaluating on testing data

We have a model - does it generalize?

How to evaluate on testing data?

For regression problems, we generally use two possible metrics:

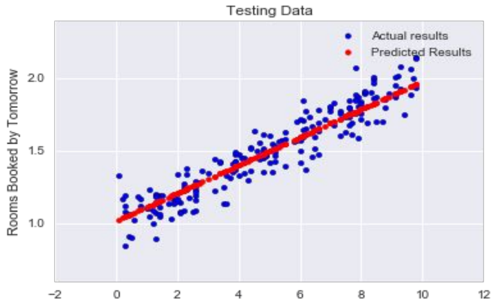
- **Root mean squared error:**

$$\sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - f(x_i))^2}$$

- **R-squared:**

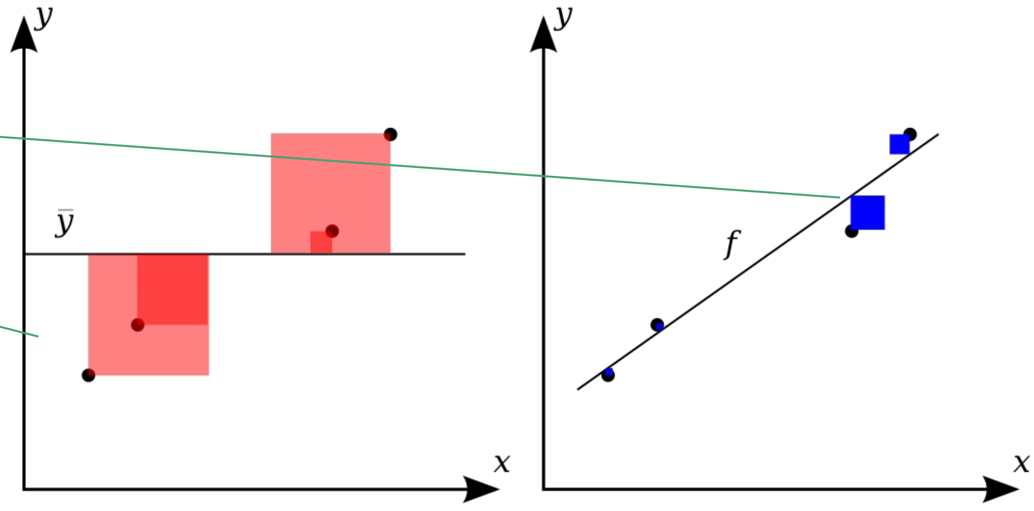
$$1 - \frac{\sum_{i=1}^{N}(f(x_i) - y_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2}$$

- This is the fraction of variance that our model is able to explain.



Coefficient of Determination (R^2)

$$1 - \frac{\sum_{i=1}^{N} (f(x_i) - y_i)^2}{\sum_{i=1}^{N} (y_i - \bar{y})^2}$$



- This is the percentage of variance that our model is able to explain
- A perfect model would have the second term be 0, resulting in 1.
- If the model was simply the mean, the result would be 0.

What have we shown?

- Given any norm, by iterating under its gradient flow, we can obtain a solution to the minimization problem (it's unique iff the norm is strictly convex).
- Most minimization problems don't have analytical solutions like the L2 norm - **this is how scikit-learn codes these algorithms (with GD).**
- We will now see a live demo of this in Python.
- https://github.com/Columbia-Intro-Data-Science/APMAE4990-/blob/master/notebooks/Lecture1%20-%20Introduction-to-Regression.ipynb
- You will be asked to apply these methods that we work through now in the homework assignment: https://github.com/Columbia-Intro-Data-Science/APMAE4990-/blob/master/homework/Homework%201%20-%20Github%20and%20Pandas.ipynb

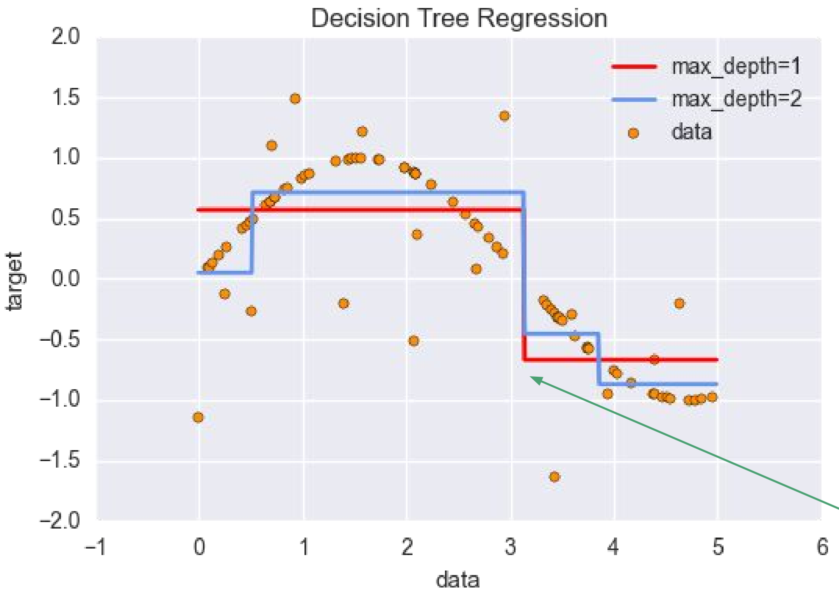
Decision Tree Regression

First: A 1d example

$$y = \sin(x) + \text{noise}$$

- Decision trees make a recursive series of decisions which lead to an outcome, real valued or labeled (for regression, real valued)
- The goal is to make splitting decisions on the data to minimize a norm, in particular, the L2 distance to the mean on that subset of the data, also known as the **variance.**

First: A 1d example



$$y = \sin(x) + \text{noise}$$

- For a depth of **zero**, one chooses the **mean**.
- How does one choose the splitting point when the depth is larger than zero?
- For a depth of one, the algorithm searches through all values between (-1,6) (in this example), and chooses the split which **minimizes the variance on the two segments which it creates.**

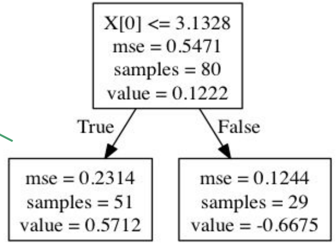
Increased depth can be bad

$$y = \sin(x) + \text{noise}$$

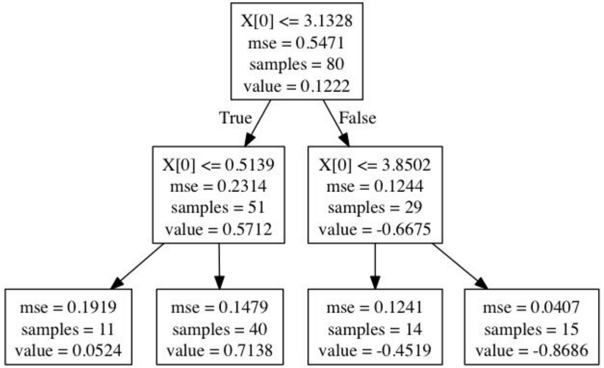
- In general, if one adds more depth to the tree, there is a risk of overfitting. Look at the unwanted variance we have picked up in this example.
- We will learn next lecture how to choose the perfect depth number!

How do the decisions work?

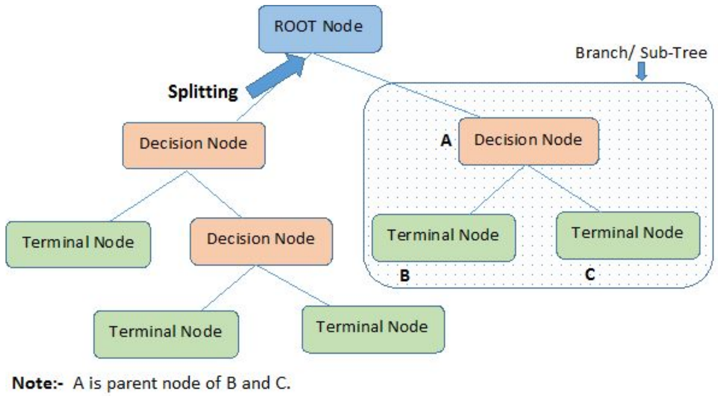
max_depth=1



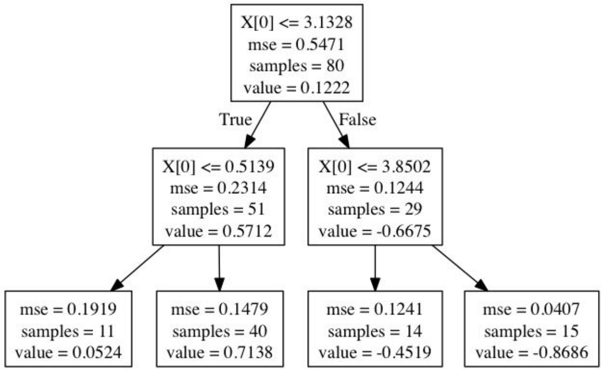
max_depth=2



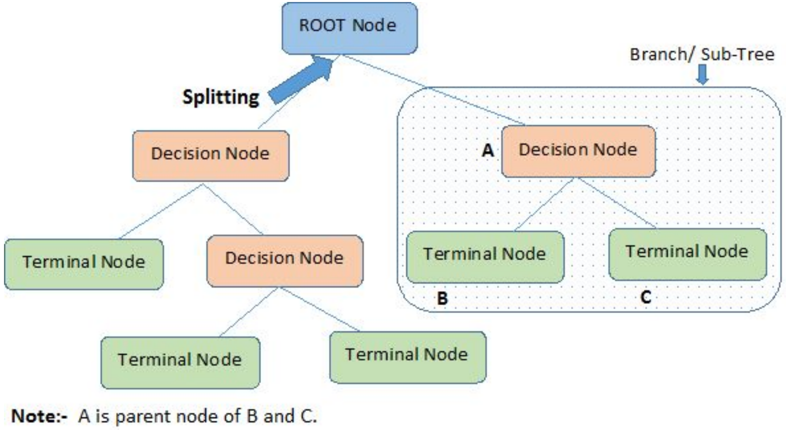
Some terminology



max_depth=2



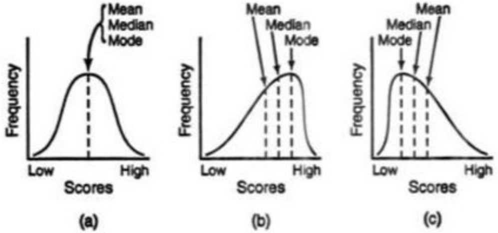
Some terminology



1. **Root Node:** It represents entire population or sample and this further gets divided into two or more homogeneous sets.
2. **Splitting:** It is a process of dividing a node into two or more sub-nodes.
3. **Decision Node:** When a sub-node splits into further sub-nodes, then it is called decision node.
4. **Leaf/ Terminal Node:** Nodes do not split is called Leaf or Terminal node.
5. **Pruning:** When we remove sub-nodes of a decision node, this process is called pruning. You can say opposite process of splitting.
6. **Branch / Sub-Tree:** A sub-section of entire tree is called branch or sub-tree.
7. **Parent and Child Node:** A node, which is divided into sub-nodes is called parent node of sub-nodes where as sub-nodes are the child of parent node.

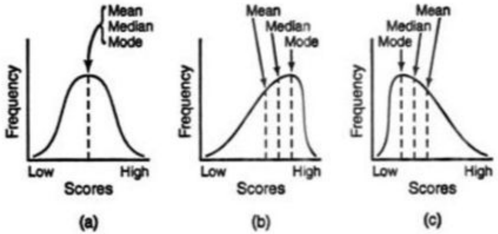
What are the advantages of decision trees?

- Very easy to interpret.
- Makes **no a-priori assumptions about the structural form of the data** (as linear models do). For instance, works well with non-linear data.
- More “robust” than linear models, meaning **less sensitive to outliers**. This is for the same reason that the median is less sensitive to outliers than the median.
- Well defined notion of ‘**most significant variables**’, so good for data exploration (will explain).
- Handles categorical and real-valued variables (doesn’t require one-hot encoding as linear models always do - Exercise: Why?)



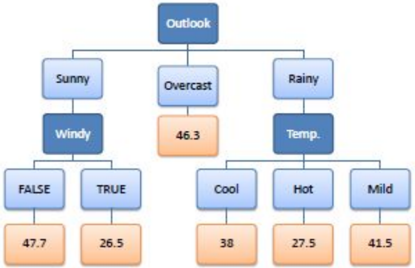
What are the disadvantages?

- Prone to **overfitting** (more on this next lecture).
- Can lose information when dealing with continuous variables, since it needs to make a finite number of splits.



More than one variable?

Predictors				Target
Outlook	Temp	Humidity	Windy	Hours Played
Rainy	Hot	High	False	26
Rainy	Hot	High	True	30
Overcast	Hot	High	False	48
Sunny	Mild	High	False	46
Sunny	Cool	Normal	False	62
Sunny	Cool	Normal	True	23
Overcast	Cool	Normal	True	43
Rainy	Mild	High	False	36
Rainy	Cool	Normal	False	38
Sunny	Mild	Normal	False	48
Rainy	Mild	Normal	True	48
Overcast	Mild	High	True	62
Overcast	Hot	Normal	False	44
Sunny	Mild	High	True	30

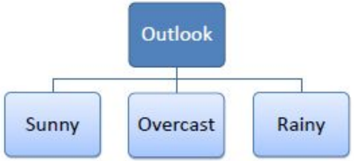


- A decision tree is simply a recursive set of decisions which leads to a result.
- It can be a real value or a class.
- Decision trees are very simple to understand, but are somewhat complex to explain when multiple features are involved.

Machine Learning Objective: Let’s try to predict the number of hours played on a golf course in a single day by all of the golfers.

Example taken from: http://chem-eng.utoronto.ca/~datamining/dmc/decision_tree_reg.htm

How do we choose the root node?



Outlook	Temp	Humidity	Windy	Hours Played
Sunny	Mild	High	FALSE	45
Sunny	Cool	Normal	FALSE	52
Sunny	Cool	Normal	TRUE	23
Sunny	Mild	Normal	FALSE	46
Sunny	Mild	High	TRUE	30
Rainy	Hot	High	FALSE	25
Rainy	Hot	High	TRUE	30
Rainy	Mild	High	FALSE	35
Rainy	Cool	Normal	FALSE	38
Rainy	Mild	Normal	TRUE	48
Overcast	Hot	High	FALSE	46
Overcast	Cool	Normal	TRUE	43
Overcast	Mild	High	TRUE	52
Overcast	Hot	Normal	FALSE	44

		Hours Played (StDev)
Outlook	Overcast	3.49
	Rainy	7.78
	Sunny	10.87
SDR=1.66		

		Hours Played (StDev)
Temp.	Cool	10.51
	Hot	8.95
	Mild	7.65
SDR=0.17		

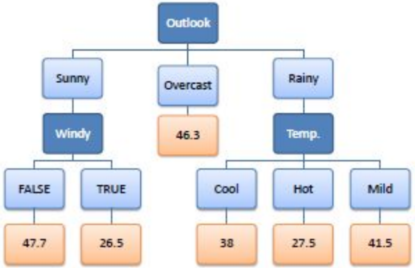
		Hours Played (StDev)
Humidity	High	9.36
	Normal	8.37
SDR=0.28		

		Hours Played (StDev)
Windy	False	7.87
	True	10.59
SDR=0.29		

- Here we've tried splitting first by the variable "Outlook".
- From here, we can compute the conditional standard deviations in the three subsets created.
- Our goal is to find the variable that, when split, reduces the stdev the most.
- Original stdev of "Hours Played" is 9.33.
- We can try this for every variable, and then choose the one which reduces the stdev the most.
- Subsequent decisions are made by recursively iterating this procedure.

More precise description

Predictors				Target
Outlook	**Temp.**	**Humidity**	**Windy**	**Hours Played**
Rainy	Hot	High	False	26
Rainy	Hot	High	True	30
Overcast	Hot	High	False	48
Sunny	Mild	High	False	46
Sunny	Cool	Normal	False	62
Sunny	Cool	Normal	True	23
Overcast	Cool	Normal	True	43
Rainy	Mild	High	False	36
Rainy	Cool	Normal	False	38
Sunny	Mild	Normal	False	48
Rainy	Mild	Normal	True	48
Overcast	Mild	High	True	62
Overcast	Hot	Normal	False	44
Sunny	Mild	High	True	30



For zero depth we minimize:

$$\mathrm{Var}(y) := \frac{1}{N}\sum_{i=1}^{N}(y_i - \bar{y})^2$$

Solution is: $\hat{y} = \bar{y}$

$$\mathrm{Var}(Y|X = x_j) = \sum_{i=1}^{N}(y_i - \bar{y}_j)^2 p(y_i|x_j)$$

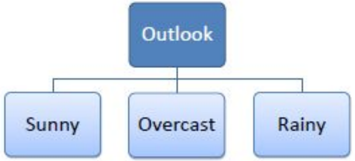
$$= \frac{1}{N}\sum_{i, x_i = x_j}(y_i - \bar{y}_j)^2 |X = x_j|$$

For depth one we minimize the conditional variance for each variable:

$$\mathrm{VarRed}(Y, X) = \mathrm{Var}(Y) - \sum_j \mathrm{Var}(Y|X = x_j)$$

Variance Reduction

Let’s try Outlook first



Outlook	Temp	Humidity	Windy	Hours Played
Sunny	Mild	High	FALSE	45
Sunny	Cool	Normal	FALSE	52
Sunny	Cool	Normal	TRUE	23
Sunny	Mild	Normal	FALSE	46
Sunny	Mild	High	TRUE	30
Rainy	Hot	High	FALSE	25
Rainy	Hot	High	TRUE	30
Rainy	Mild	High	FALSE	35
Rainy	Cool	Normal	FALSE	38
Rainy	Mild	Normal	TRUE	48
Overcast	Hot	High	FALSE	46
Overcast	Cool	Normal	TRUE	43
Overcast	Mild	High	TRUE	52
Overcast	Hot	Normal	FALSE	44

		Hours Played (StDev)	Count
Outlook	Overcast	3.49	4
	Rainy	7.78	5
	Sunny	10.87	5
			14

$$\mathrm{Var}(y) := \frac{1}{N}\sum_{i=1}^{N}(y_i - \bar{y})^2 = 9.32^2$$

$$\sum_j \mathrm{Var}(Y|\mathrm{Outlook}) = \frac{4}{14}(3.49)^2 + \frac{5}{14}(7.78)^2 + \frac{5}{14}(10.87)^2$$

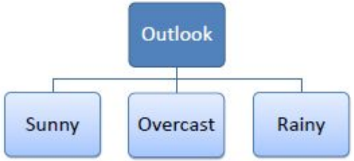
$$\mathrm{Var}(Y|X = x_j) = \sum_{i=1}^{N}(y_i - \bar{y}_j)^2 p(y_i|x_j)$$

$$= \frac{1}{N}\sum_{i, x_i = x_j}(y_i - \bar{y}_j)^2 |X = x_j|$$

$$\mathrm{VarRed}(Y, X) = \mathrm{Var}(Y) - \sum_j \mathrm{Var}(Y|X = x_j)$$

- Therefore there is a variance reduction of **19.56 for the outlook variable.**

Now we've split based on Outlook



Outlook	Temp	Humidity	Windy	Hours Played
Sunny	Mild	High	FALSE	45
Sunny	Cool	Normal	FALSE	52
Sunny	Cool	Normal	TRUE	23
Sunny	Mild	Normal	FALSE	46
Sunny	Mild	High	TRUE	30
Rainy	Hot	High	FALSE	25
Rainy	Hot	High	TRUE	30
Rainy	Mild	High	FALSE	35
Rainy	Cool	Normal	FALSE	38
Rainy	Mild	Normal	TRUE	48
Overcast	Hot	High	FALSE	46
Overcast	Cool	Normal	TRUE	43
Overcast	Mild	High	TRUE	52
Overcast	Hot	Normal	FALSE	44

		Hours Played (StDev)
Outlook	Overcast	3.49
	Rainy	7.78
	Sunny	10.87
SDR=1.66		

		Hours Played (StDev)
Temp.	Cool	10.51
	Hot	8.95
	Mild	7.65
SDR=0.17		

		Hours Played (StDev)
Humidity	High	9.36
	Normal	8.37
SDR=0.28		

		Hours Played (StDev)
Windy	False	7.87
	True	10.59
SDR=0.29		

- These groups now have less variance in the data. And Outlook reduced variance the most. Now we continue recursively.

- A branch set with standard deviation more than 0 needs further splitting - we need pure classes on the final leaf nodes.

Common Questions

- **What about real valued inputs?** In general these splits are binary, then iterated through levels in the tree.

- **Are splits always binary?** In Python, yes, but not for a general algorithm. Any 3 way split can be seen as a one versus all split (ie. A&B or C versus A, B or C). It depends on the algorithm.

Follow the notebook here:

https://github.com/doriang102/Columbia_Data_Science/blob/master/notebooks/Lecture1%20-%20Introduction-to-Regression.ipynb

Appendix - mean center without loss of generality

$$y = \beta \cdot X + c$$

$$y = \beta \cdot (X - \bar{X} + \bar{X}) + c$$

$$y = \beta \cdot (X - \bar{X}) + c + \beta \cdot \bar{X}$$

$$y = \beta \cdot (X - \bar{X}) + d$$