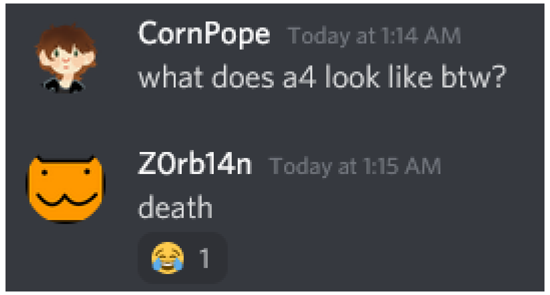
CPSC 340:
Machine Learning and Data Mining

Feature Selection

Summer 2021

Admin

- Assignment 4 is out. Start early!
  - Due Monday, June 7, 2021
  - You can do Q1 already
  - You will be able to do Q2 after today
  - A4 Q3.4 (Softmax Classifier) is notoriously hard
    - You can do it without knowing what softmax is
    - We will cover more relevant information on Monday
- Midterm is this coming Tuesday
  - Tutorials on Monday will be midterm prep
  - No office hours Tuesday, see course calendar



CornPope Today at 1:14 AM
what does a4 look like btw?

Z0rb14n Today at 1:15 AM
death

😂 1

In This Lecture

- Model Selection (10 minutes)
- Complexity Penalties (15 minutes)
- Feature Selection (10 minutes)
- Forward Selection (15 minutes)

Coming Up Next

FINDING THE “TRUE MODEL”

Back to Food Allergy Example

Egg	Milk	Fish	Wheat	Shellfish	Peanuts
0.8	0.2	0	0.3	0	0

→

Sick?
?

Input food

Outcome

Q: What is the "most certain" way
to determine the outcome for this input?

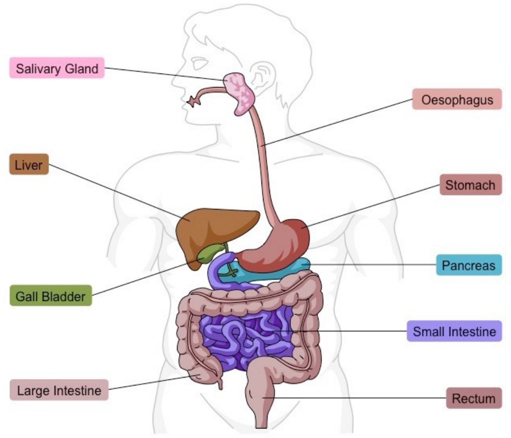
- Eat the food and see what happens (experiment)
  - Risky: I don't want to get sick
  - Expensive: I can't eat many different combinations at once
  - Ignores the historical data available

What is a "Model"?

- Model := Simple approximation of Complex process
- Assumption: data is generated from a process

Egg	Milk	Fish	Wheat	Shellfish	Peanuts
0.8	0.2	0	0.3	0	0

$\longrightarrow$



$\longrightarrow$

Sick?
1

Input food

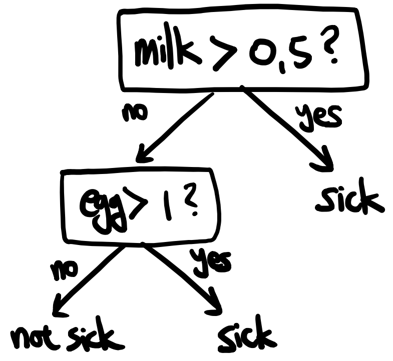
Digestive process

Outcome

What is a “Model”?

- Model := simpler approximation of complex process
- Assumption: data is generated from a process

Egg	Milk	Fish	Wheat	Shellfish	Peanuts
0.8	0.2	0	0.3	0	0



Sick?
1

Input food

Model

Outcome

- A model’s predictions are less accurate than the actual process
  - We lose some information from simplifying (Compression)

Q: When is the model most accurate?

Recall: Polynomial Regression

- Let's say my process is actually a mathematical function

$$y_i : \mathbb{R} \longrightarrow \mathbb{R}$$

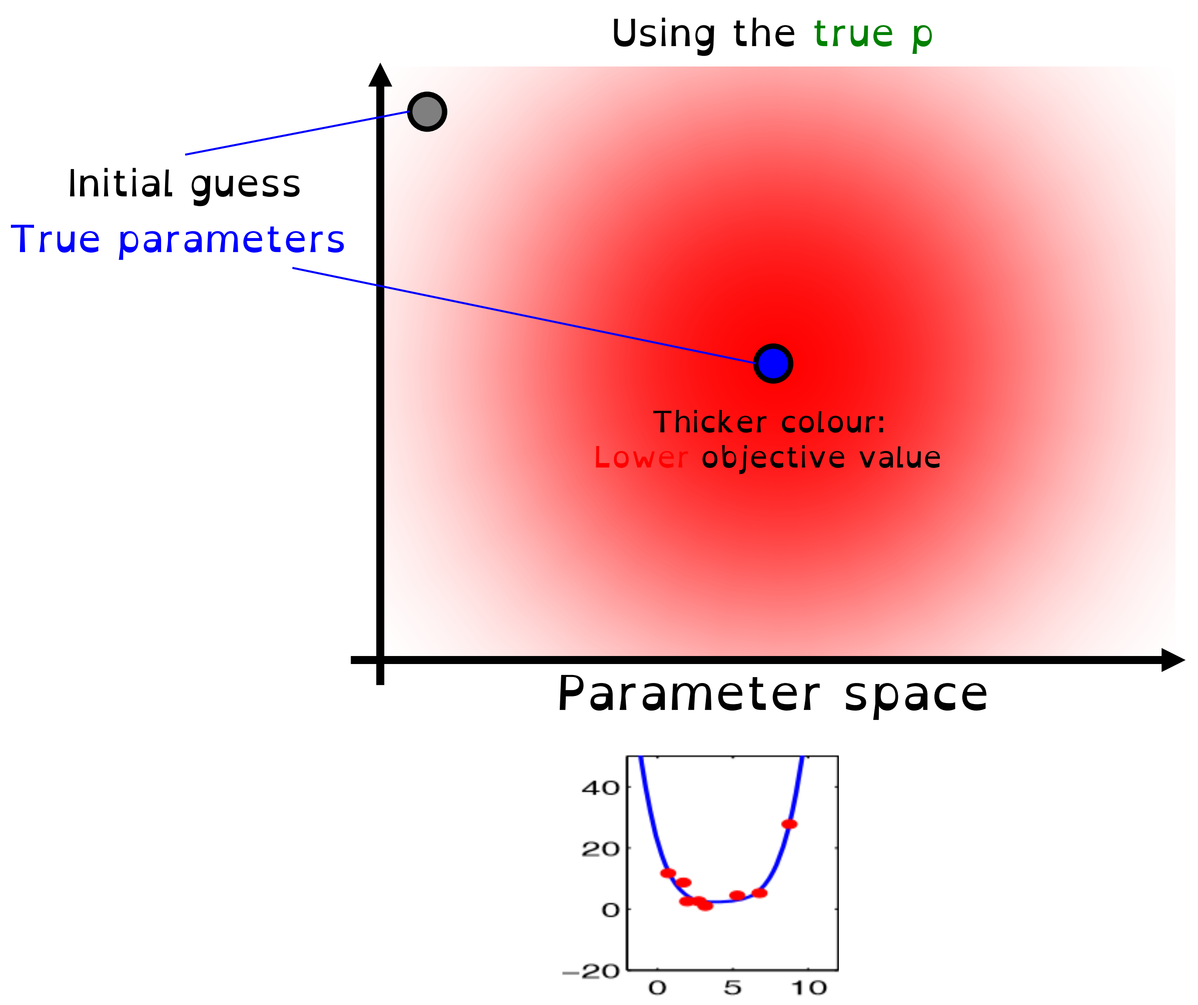
"True p" of the process

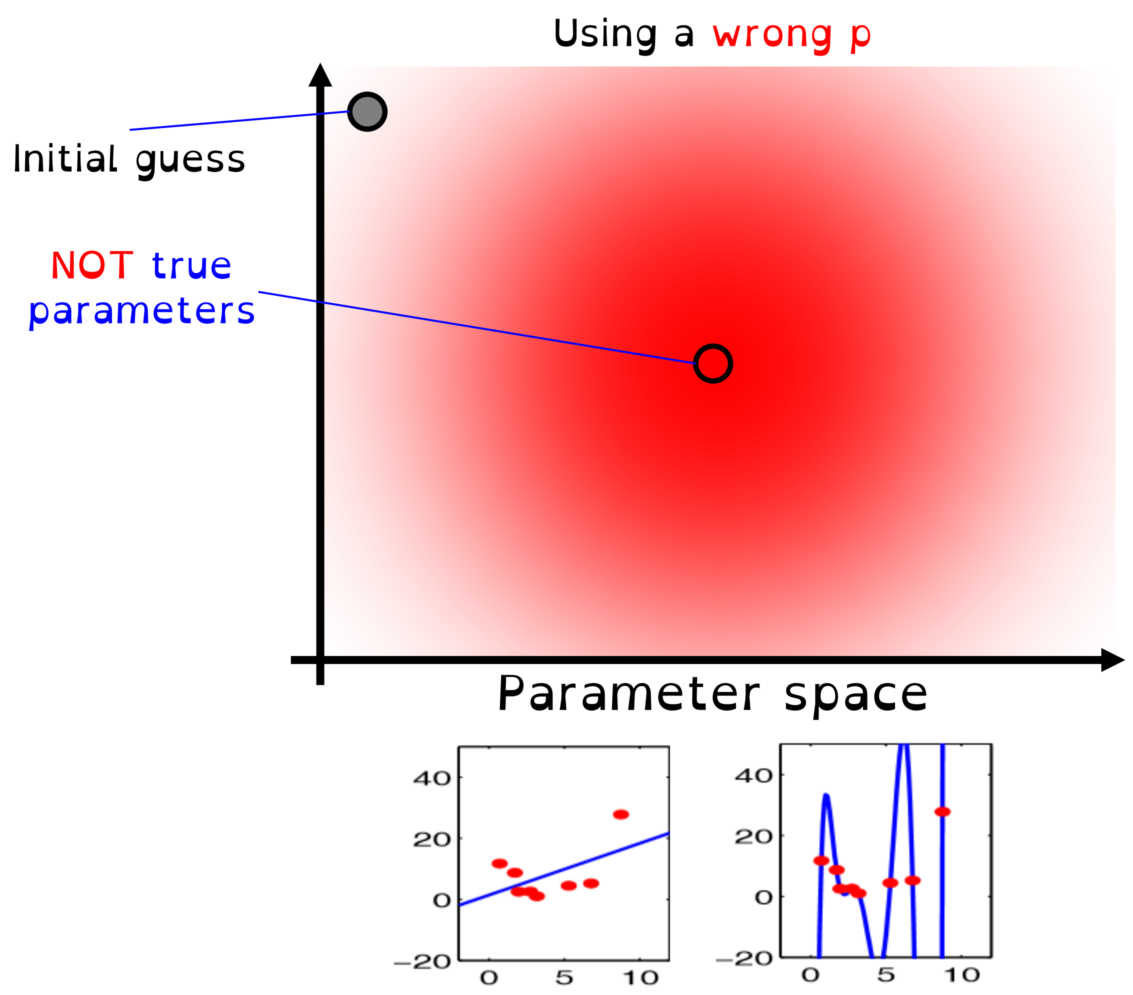
$$y_i(x_i) = w_0^* + w_1^* x_i + w_2^* x_i^2 + \cdots + w_p^* x_i^p$$

"True parameters" of the process

- Then the most accurate polynomial regression model would:
  - Choose the true p as the degree of polynomial
  - Make the learned parameters very close to the true parameters

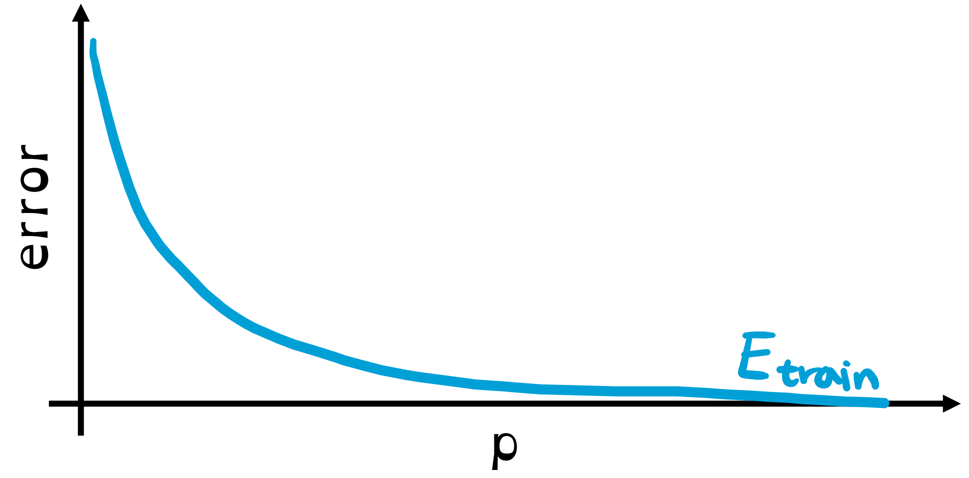
Visualizing "True Model"





Q: How should we choose our p?

How Should We Choose P?



- Training error does not work:
  - It goes down as ‘p’ goes up. (within reasonable range)
- Cross-validation may also not work:
  - Tends to overestimate ‘p’.
  - Subject to optimization bias.

For example, imagine that the true model is $y_i = 2x_i^2 - 5 + (\text{noise})$

We might choose ~~d~~ p=3 and a model like $\hat{y}_i = \underline{0.001}x_i^3 + 2x_i^2 - 5$,

since it might get a slightly smaller validation error.

Coming Up Next

COMPLEXITY PENALTIES

Discouraging Complexity

- There are a lot of "scores" people use to find the "true" model.
- Basic idea behind them: put a penalty on the model complexity.
  - Want to fit the data and have a simple model.
- For example, minimize training error plus the degree of polynomial.

Let $\underline{Z_p} = \begin{bmatrix} 1 & x_1 & (x_1)^2 & \cdots & (x_1)^p \\ 1 & x_2 & (x_2)^2 & \cdots & (x_2)^p \\ 1 & x_3 & (x_3)^2 & \cdots & (x_3)^p \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & (x_n)^2 & \cdots & (x_n)^p \end{bmatrix}$

Find 'p' that minimizes:

$$\text{score}(p) = \underbrace{\frac{1}{2}\|Z_p v - y\|^2}_{\substack{\text{train error for} \\ \text{best 'v' with this basis.}}} + \underbrace{p}_{\substack{\text{degree of} \\ \text{polynomial}}}$$

  - If we use p=4, use "training error plus 4" as error.
- If two 'p' values have similar error, this prefers the smaller 'p'.

Choosing Degree of Polynomial Basis

- How can we optimize this score?

$$\text{score}(p) = \frac{1}{2}\|Z_p v - y\|^2 + p$$

  - Form Z_0, solve for 'v', compute score(0) = $\frac{1}{2}\|Z_0 v - y\|^2 + 0$.
  - Form Z_1, solve for 'v', compute score(1) = $\frac{1}{2}\|Z_1 v - y\|^2 + 1$.
  - Form Z_2, solve for 'v', compute score(2) = $\frac{1}{2}\|Z_2 v - y\|^2 + 2$.
  - Form Z_3, solve for 'v', compute score(3) = $\frac{1}{2}\|Z_3 v - y\|^2 + 3$.

  - Choose the degree with the lowest score.
    - "You need to decrease training error by at least 1 to increase degree by 1."

Q: Does this mean
optimization bias is not a problem anymore?

Information Criteria

- There are many scores, usually with the form:

$$\text{score}(p) = \frac{1}{2}\|Z_p v - y\|^2 + \lambda k$$

  - The value 'k' is the "number of estimated parameters" ("degrees of freedom").
    - For polynomial basis with d=1, we have k = (p+1).
  - The parameter λ > 0 controls how strong we penalize complexity.
    - "You need to decrease the training error by least λ to increase 'k' by 1".

- Using (λ = 1) is called Akaike information criterion (AIC).
- Other choices of λ (not necessarily integer) give other criteria:
  - Mallow's C_p.
  - Adjusted R^2.
  - ANOVA-based model selection.

Choosing Degree of Polynomial Basis

- How can we optimize this score in terms of ‘p’?

$$\text{score}(p) = \frac{1}{2}\|Z_p v - y\|^2 + \lambda k$$

  - Form Z_0, solve for ‘v’, compute score(0) = $\frac{1}{2}\|Z_0 v - y\|^2 + \lambda$.
  - Form Z_1, solve for ‘v’, compute score(1) = $\frac{1}{2}\|Z_1 v - y\|^2 + 2\lambda$.
  - Form Z_2, solve for ‘v’, compute score(2) = $\frac{1}{2}\|Z_2 v - y\|^2 + 3\lambda$.
  - Form Z_3, solve for ‘v’, compute score(3) = $\frac{1}{2}\|Z_3 v - y\|^2 + 4\lambda$.

  - So we need to improve by “at least λ” to justify increasing degree.
    - If λ is big, we’ll choose a small degree. If λ is small, we’ll choose a large degree.

Bayesian Information Criterion (BIC)

- A disadvantage of these methods:
  - Still prefers a larger ‘p’ as ‘n’ grows.

- Solution: make λ depend on ‘n’.
- For example, the Bayesian information criterion (BIC) uses:

$$\lambda = \frac{1}{2}\log(n)$$

- BIC penalizes a bit more than AIC for large ‘n’.
  - As ‘n’ goes to ∞, recovers “true” model (“consistent” for model selection).
- In practice, we usually just try a bunch of different λ values.
  - Picking λ is like picking ‘k’ in k-means.

Discussion of other Scores for Model Selection

- There are many other selection and scoring methods:
  - Elbow method (corresponds to specific choice of λ).
    - You could also use BIC for choosing 'k' in k-means.
  - Methods based on validation error.
    - "Take smallest 'p' within one standard error of minimum cross-validation error".
  - Minimum description length.
  - Risk inflation criterion.
  - False discovery rate.
  - Marginal likelihood (CPSC 540).

- These can adapted to use the L1-norm and other errors.

Coming Up Next

FEATURE SELECTION

Motivation: Discovering Food Allergies

- Recall the food allergy example:

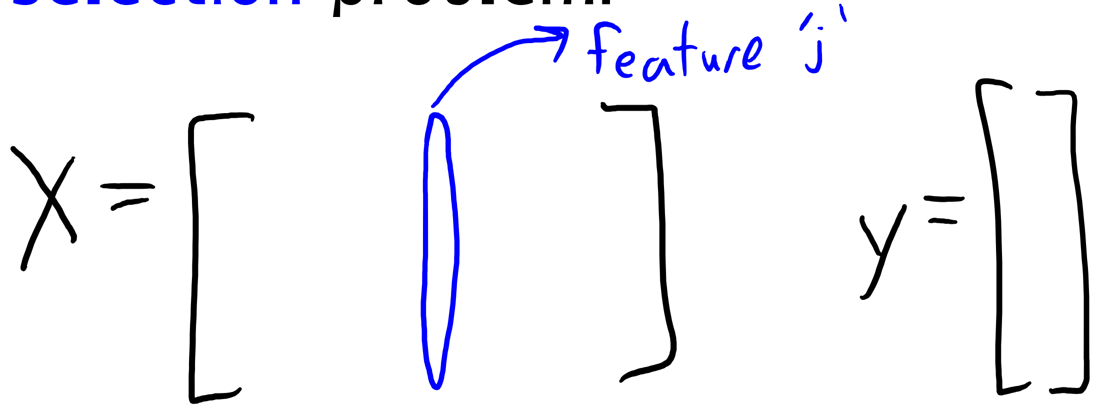
Egg	Milk	Fish	Wheat	Shellfish	Peanuts	…
0	0.7	0	0.3	0	0	
0.3	0.7	0	0.6	0	0.01	
0	0	0	0.8	0	0	
0.3	0.7	1.2	0	0.10	0.01	

Sick?
1
1
0
1

Q: Which features are most "relevant" for predicting the outcome?

Feature Selection

- General feature selection problem:



  - Find the features (columns) of 'X' that are important for predicting 'y'.
    - "What are the relevant factors?"
    - "Which basis functions should I use among these choices?"
    - "What types of new data should I collect?"
    - "How can I speed up computation?"

- One of most important problems in ML/statistics, but very messy.
  - For now, we'll say a feature is "relevant" if it helps predict y_i from x_i.

"Association" Approach

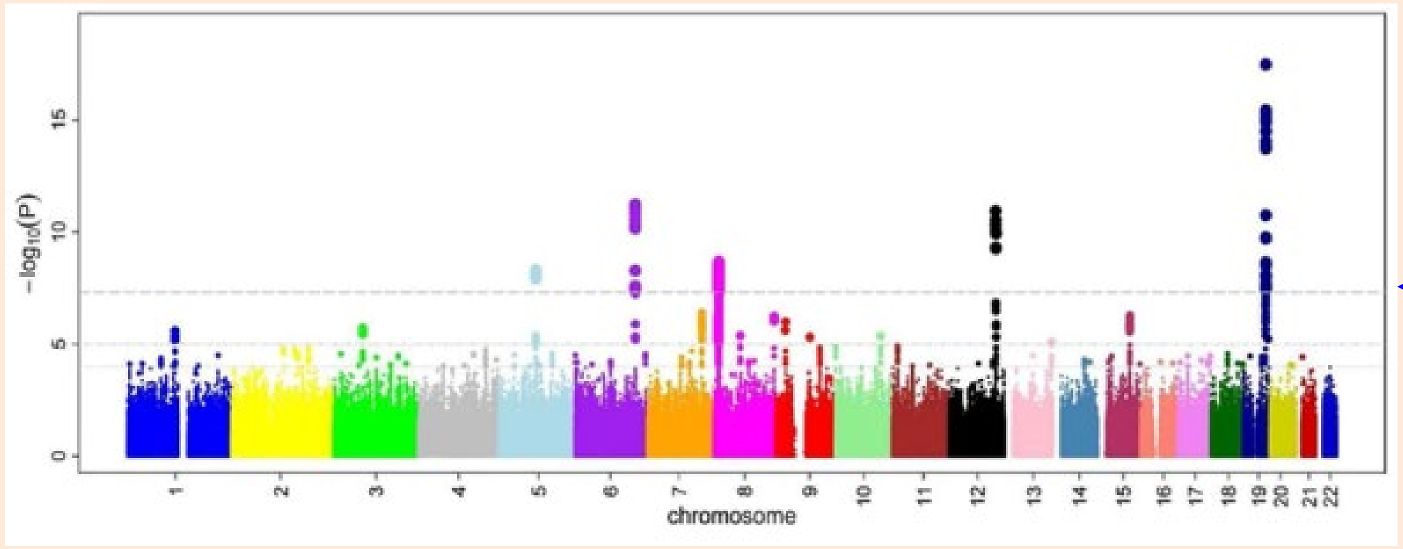
- A simple/common way to do feature selection:
  - For each feature 'j', compute Sample correlation between feature values x^j and 'y'.
    - Say that 'j' is relevant if correlation is above 0.9 or below -0.9.

- Turns feature selection into hypothesis testing for each feature.
    - There are many other measures of "dependence" (Wikipedia).

- Usually gives unsatisfactory results as it ignores feature interaction:
  - Includes irrelevant features: "Taco Tuesdays".
    - If tacos make you sick, and you often eat tacos on Tuesdays, it will say "Tuesday" is relevant.
  - Excludes relevant features : "Diet Coke + Mentos Eruption".
    - Diet coke and Mentos don't make you sick on their own, but *together* they make you sick.

Genome-Wide Association Studies

- Genome-wide association studies:
  - Measure if there exists a dependency between each individual "single-nucleotide polymorphism" in the genome and a particular disease.



"Manhattan plot"

  - Has identified thousands of genes "associated" with diseases.
    - But by *design* this has a huge numbers of false positives (and many false negatives).

“Regression Weight” Approach

- A simple/common approach to feature selection:
  - Fit regression weights ‘w’ based on **all** features (maybe with least squares).
  - Take all features ‘j’ where weight $|w_j|$ is greater than a threshold.

- For example: you fit a least squares model with 5 features and get:

$$w = \begin{bmatrix} 0.01 \\ -0.2 \\ 10 \\ -3 \\ 0.0001 \end{bmatrix}$$

Q: Which features look relevant?

  - Feature 3 looks the most relevant.
  - Feature 4 also looks relevant.
  - Feature 5 seems irrelevant.

“Regression Weight” Approach

- A simple/common approach to feature selection:
  - Fit regression weights ‘w’ based on **all** features (maybe with least squares).
  - Take all features ‘j’ where weight $|w_j|$ is greater than a threshold.

- This could recognize that “Tuesday” is irrelevant.
  - If you get enough data, and you sometimes eat tacos on other days.
    (And the relationship is actually linear.)

- This could recognize that “Diet Coke” and “Mentos” are relevant.
  - Assuming this combination occurs enough times in the data.

“Regression Weight” Approach

- A simple/common approach to feature selection:
  - Fit regression weights ‘w’ based on **all** features (maybe with least squares).
  - Take all features ‘j’ where weight $|w_j|$ is greater than a threshold.

- Has major problems with collinearity (e.g. “Taco Tuesday”):
  - If the “Tuesday” feature always equals the “taco” feature, it could say that Tuesdays are relevant but tacos are not.

  - If you have two copies of an irrelevant feature, it could take both irrelevant copies.

$$\hat{y}_i = w_1 * taco + w_2 * Tuesday = 0 * taco + (w_1 + w_2) * Tuesday$$

$$\hat{y}_i = 0 * irrelevant + 0 * irrelevant = 10000 * irrelevant + (-10000) * irrelevant$$

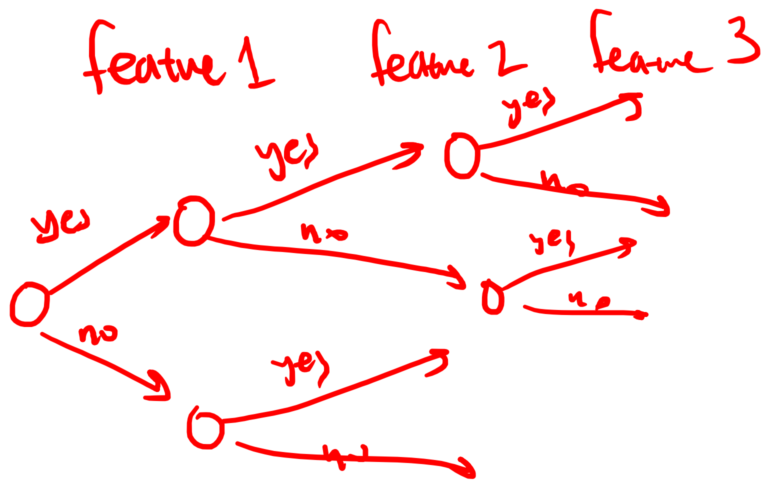
Coming Up Next

FORWARD SELECTION

Search and Score Methods

- Most common feature selection framework is search and score:
  1. Define score function f(S) that measures quality of a set of features 'S'.
  2. Now search for the features 'S' with the best score.

- Example with 3 features:
  - Compute "score" of using feature 1.
  - Compute "score" of using feature 2.
  - Compute "score" of using feature 3.
  - Compute "score" of using features {1,2}.
  - Compute "score" of using features {1,3}.
  - Compute "score" of using features {2,3}.
  - Compute "score" of using features {1,2,3}.
  - Compute "score" of using features {}. ← return-the-mode
  - Return the set of features 'S' with the best "score".



2^d

Q: How big is this search space?

Which Score Function?

- The score can't be the training error.
  - Training error goes down as you add features, so will select all features.
- A more logical score is the validation error.
  - "Find the set of features that gives the lowest validation error."
  - To minimize test error, this is what we want.
- But there are problems due to the large number of sets of features:
  - If we have 'd' features, there are 2^d sets of features.
  - __optimization bias__ is high: we're optimizing over 2^d models (not 10).
  - Prone to false positives: irrelevant features will sometimes help by chance.

"Number of Features" Penalties

- To reduce false positives, we can again use complexity penalties:

$$\text{score}(S) = \frac{1}{2}\sum_{i=1}^{n}(w_s^T x_{is} - y_i)^2 + \text{size}(S)$$

  - E.g., we could use squared error and number of non-zeroes.
  - We're using 'x_{is}' as the features 'S' of example x_i.

- If two 'S' have similar error, this prefers the smaller set.
  - It prefers removing feature 3 instead of having $w_3 = 0.00001$.

- Instead of "size(S)", we usually write this using the "L0-norm"...

L0-Norm and "Number of Features We Use"

- In linear models, setting $w_j = 0$ is the same as removing feature 'j':

$$\hat{y}_i = w_1 x_{i1} + \underbrace{w_2 x_{i2}}_{\text{set } w_2 = 0} + w_3 x_{i3} + \cdots + w_d x_{id}$$

$$\hat{y}_i = w_1 x_{i1} + \underbrace{0}_{\text{ignore } x_{i2}} + w_3 x_{i3} + \cdots + w_d x_{id}$$

- The L0 "norm" is the number of non-zero values ($||w||_0$ = size(S)).

If $w = \begin{bmatrix} 1 \\ 0 \\ 2 \\ 0 \\ 3 \end{bmatrix}$ then $||w||_0 = 3$ If $w = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$ then $||w||_0 = 0$.

  - Not actually a true norm.
  - If 'w' has a small L0-norm, then it doesn't use many features.

L0-penalty: optimization

- L0-norm penalty for feature selection:

$$f(w) = \underbrace{\frac{1}{2}\|Xw - y\|^2}_{\text{training error}} + \lambda \underbrace{\|w\|_0}_{\text{degrees of freedom 'k'}}$$

- Suppose we want to use this to evaluate the features S = {1,2}:
  - First fit the 'w' just using features 1 and 2.
  - Now compute the training error with this 'w' and features 1 and 2.
  - Add λ*2 to the training error to get the score.

- We repeat this with other choices of 'S' to find the "best" features.

L0-penalty: interpretation

- L0-norm penalty for feature selection:

$$f(w) = \frac{1}{2}\|Xw - y\|^2 + \lambda\|w\|_0$$

- Balances between training error and number of features we use.
  - With λ=0, we get least squares with all features.
  - With λ=∞, we must set w=0 and not use any features.

  - With other λ, balances between training error and number of non-zeroes.
    - Larger λ puts more emphasis on having zeroes in 'w' (more features removed).
    - Different values give AIC, BIC, and so on.

Forward Selection (Hill-Climbing with Heuristics)

- In search and score, it’s also just hard to search for the best ‘S’.
  - There are 2^d possible sets.
- A common greedy search procedure is forward selection (A4):

Post-lecture slides:
more formal description

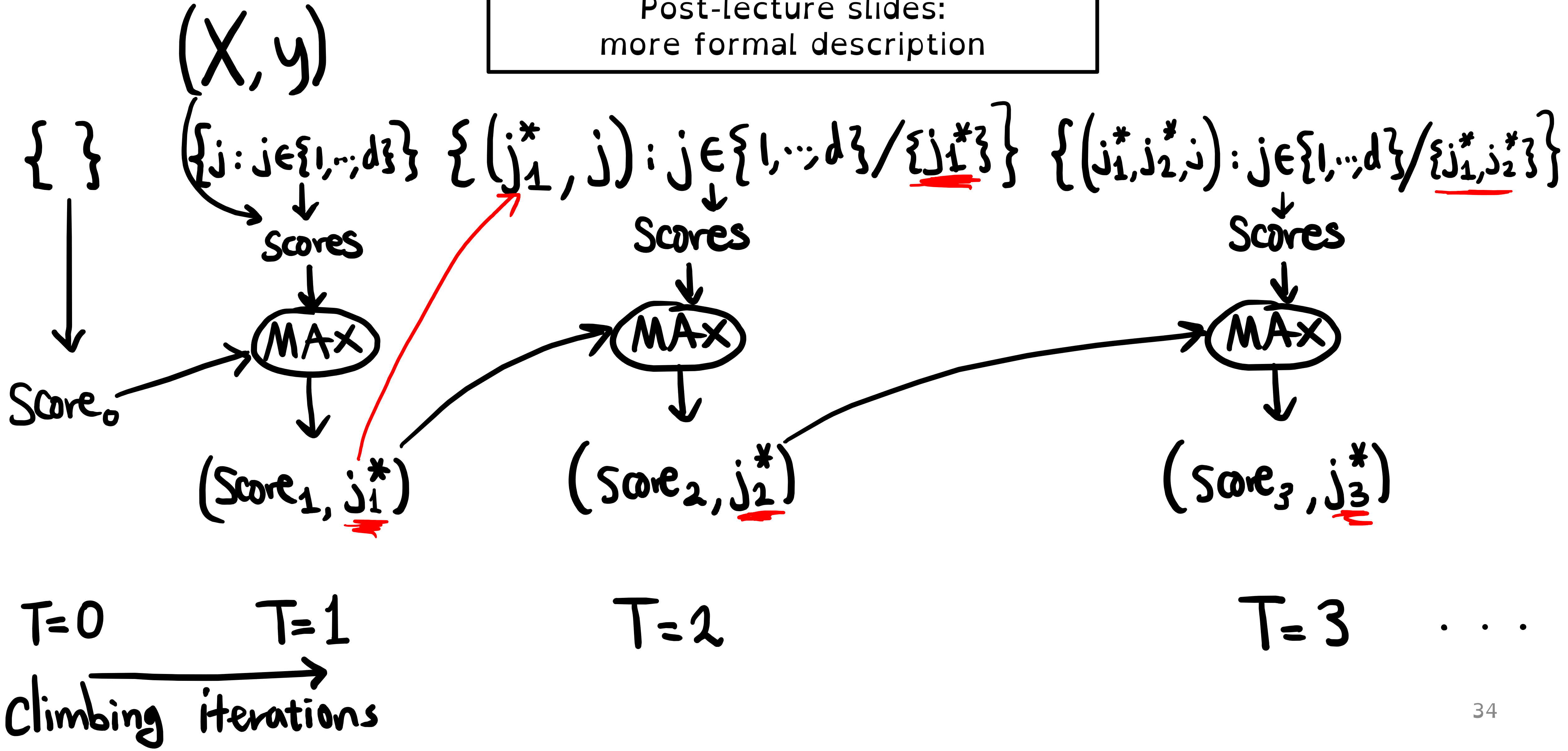
"Score" given X, y, and feature set $(j_1, j_2, \cdots, j_T)$

$$\begin{tikzcd}
(j_1, j_2, \cdots, j_T) \arrow[dr] & X = {}_n[\quad]_d \arrow[d] & y = {}_n[\quad]_1 \arrow[d] \arrow[dr] & \\
& X_{sub} = {}_n[\quad]_T \arrow[r] \arrow[rr, bend right] & \underset{T \times 1}{w} \arrow[r] & \text{"score"}
\end{tikzcd}$$

$f(w) = \frac{1}{2}\|X_w - y\|^2 + \lambda\|w\|_0$

Forward Selection (Hill-Climbing with Heuristics)

Post-lecture slides:
more formal description

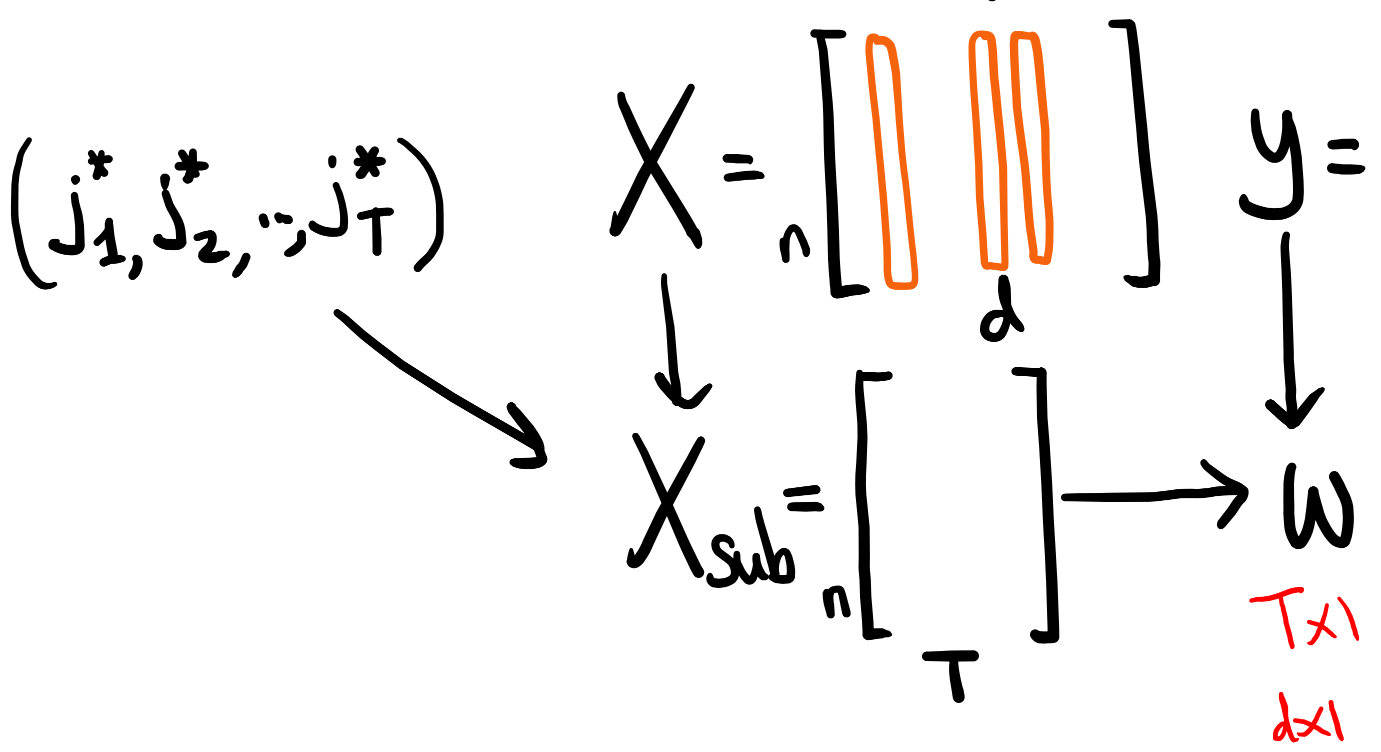

(X, y)
$\{\ \}$
$\{j : j \in \{1, \dots, d\}\}$
$\{(j_1^*, j) : j \in \{1, \dots, d\} / \{j_1^*\}\}$
$\{(j_1^*, j_2^*, j) : j \in \{1, \dots, d\} / \{j_1^*, j_2^*\}\}$
Scores
Scores
Scores
MAX
MAX
MAX
Score_0
(Score_1, j_1^*)
(Score_2, j_2^*)
(Score_3, j_3^*)
T=0
T=1
T=2
T=3
. . .
Climbing iterations

Forward Selection (Hill-Climbing with Heuristics)

Post-lecture slides:
more formal description

Stop climbing when we can't improve

$(\text{score}_{T+1} = \text{score}_T)$



(or expand into $d \times 1$ vector with 0s for excluded js)

Backward Selection and RFE

- Forward selection often works better than naïve methods.

- A related method is backward selection:
  - Start with all features, compute score after removing each feature, remove the one that improves the score the most.

- If you consider adding or removing features, it's called stagewise.

- Stochastic local search is a class of fancier methods.
  - Simulated annealing, genetic algorithms, ant colony optimization, etc.

- Recursive feature elimination is another related method:
  - Fit parameters of a regression model.
  - Prune features with small regression weights.
  - Repeat.

Summary

- Model selection: choose the best approximation to the real process
- Feature selection: choose the "relevant" features.
  - Obvious simple approaches have obvious simple problems.
- Search and score: find features that optimize some score.
  - L0-norm penalties are the most common scores.
  - Forward selection is a heuristic to search over a smaller set of features.
- Post-lecture bonus slides:
  - "Relevance" is really hard to define.
  - Mark's "rough guide" to how different methods deal with "relevance" issues.

- Next time: probably the most important topic of this course.

Review Questions

- Q1: Why is looking at the magnitude of regression weights not a good idea for feature selection?
- Q2: Why does selecting the degree of polynomial based on the validation set prefer a larger ‘p’ value?
- Q3: What happens to the predictions of my linear model when there are too many irrelevant features?
- Q4: What does it mean that the forward selection algorithm is a greedy heuristic search?

Forward Selection (Hill-Climbing with Heuristics)

- In search and score, it's also just hard to search for the best 'S'.
  - There are 2^d possible sets.

- A common greedy search procedure is forward selection:

1. Compute score if we use no features.
2. Try adding "taco", "milk", "egg", and so on (computing score of each)
3. Add "milk" because it got the best score.
4. Try {milk, taco}, {milk, egg}, and so on (computing score of each variable with milk)
5. Add "egg" becaus it got the best score combined with "milk"
6. Try {milk, egg, taco}, {milk, egg, pizza}, and so on...

Forward Selection (Hill-Climbing with Heuristics)

- Forward selection algorithm for feature selection:
  1. Start with an empty set of features, S = [ ].
  2. For each possible feature 'j':
     - Compute scores of features in 'S' combined with feature 'j'.
  3. Find the 'j' that has the best score when added to 'S'.
  4. Check if {S ∪ j} improves on the best score found so far.
  5. Add 'j' to 'S' and go back to Step 2.
     - A variation is to stop if no 'j' improves the score over just using 'S'.

- Not guaranteed to find the best set, but reduces many problems:
  - Considers $O(d^2)$ models: cheaper, overfits less, has fewer false positives.

Mark's advice if you want the "relevant" features.

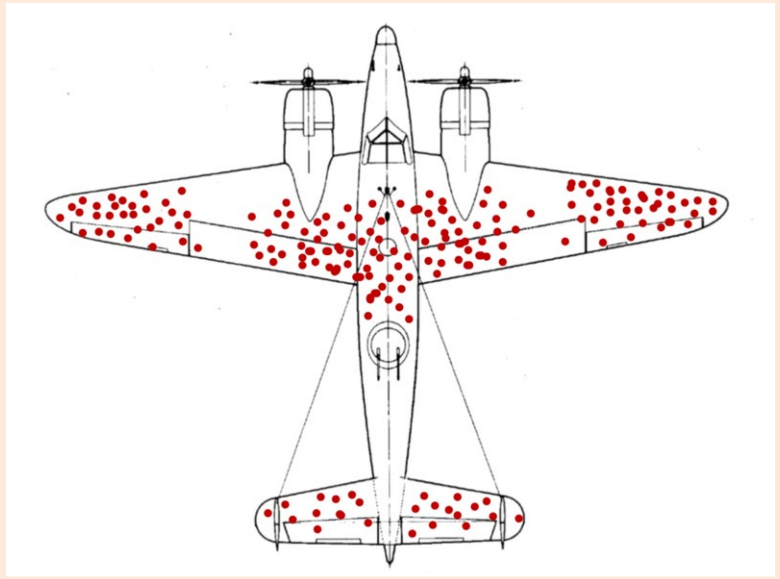
- Try the association approach.
- Try forward selection with different values of λ.
- Try out a few other feature selection methods too.

- Discuss the results with the domain expert.
  - They probably have an idea of why some features might be relevant.

- Don't be overconfident:
  - These methods are probably not discovering how the world truly works.
  - "The algorithm has found that these features are helpful in predicting y_i."
    - Then a warning that these models are not perfect at finding relevant features.

Related: Survivorship Bias

- Plotting location of bullet holes on planes returning from WW2:

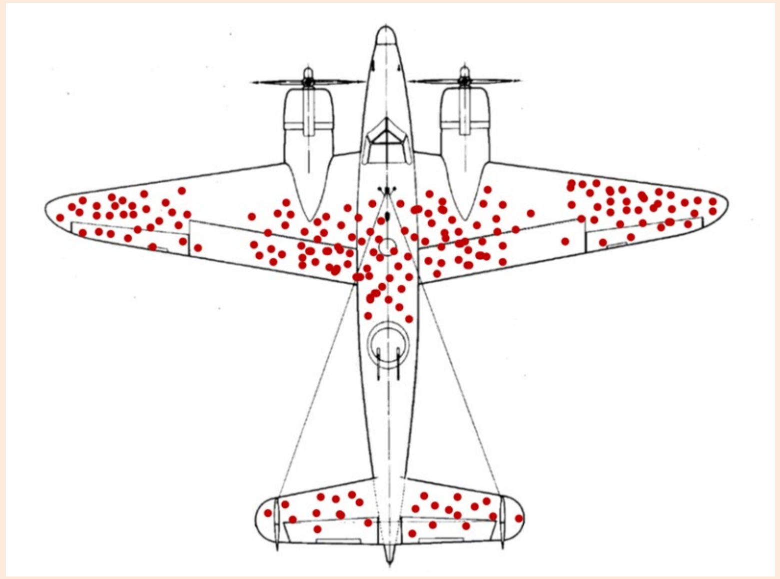


- Where are the "relevant" parts of the plane to protect?
  - "Relevant" parts are actually where there are no bullets.
  - Planes shot in other places did not come back (armor was needed).

https://en.wikipedia.org/wiki/Survivorship_bias

Related: Survivorship Bias

- Plotting location of bullet holes on planes returning from WW2:

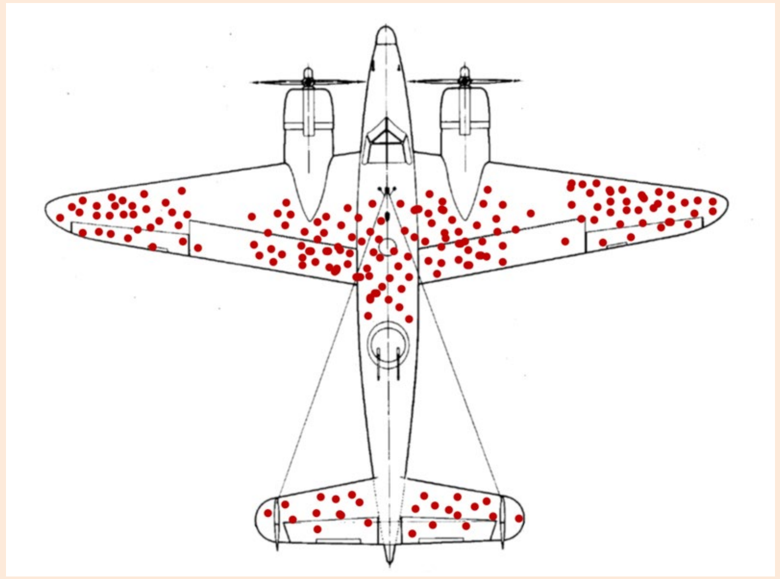


- This is an example of "survivorship bias":
  - Data is not IID because you only sample the "survivors".
  - Causes havoc for feature selection, and ML methods in general.

https://en.wikipedia.org/wiki/Survivorship_bias

Related: Survivorship Bias

- Plotting location of bullet holes on planes returning from WW2:



- People come to wrong conclusions due to survivor bias all the time.
  - Article on “secrets of success”, focusing on traits of successful people.
    - But ignoring the number of non-super-successful people with the same traits.
  - Article hypothesizing about various topics (allergies, mental illness, etc.).

https://en.wikipedia.org/wiki/Survivorship_bias

Is “Relevance” Clearly Defined?

- Consider a supervised classification task:

gender	mom	dad
F	1	0
M	0	1
F	0	0
F	1	1

SNP
1
0
0
1

- Predict whether someone has particular genetic variation (SNP).
  - Location of mutation is in “mitochondrial” DNA.
    - “You almost always have the same value as your mom”.
  - For simplicity we’ll assume 1950s-style gender and parentage.

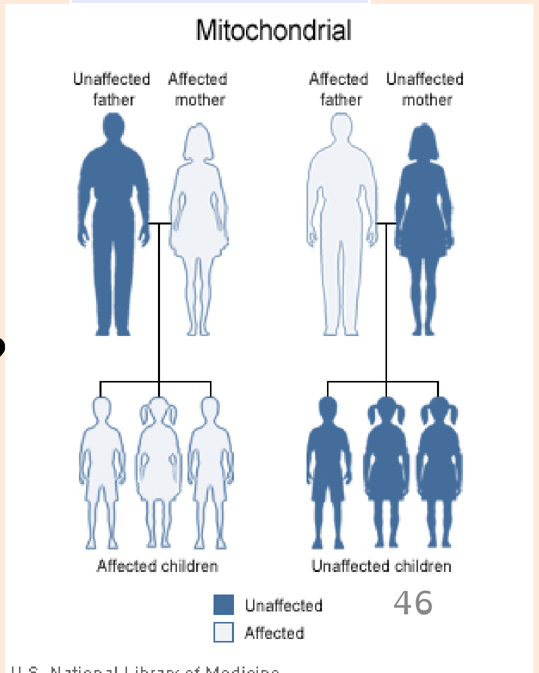
Is “Relevance” Clearly Defined?

- Consider a supervised classification task:

gender	mom	dad
F	1	0
M	0	1
F	0	0
F	1	1

SNP
1
0
0
1

- True model:
  - (SNP = mom) with very high probability.
  - (SNP != mom) with some very low probability.
- What are the “relevant” features for this problem?
  - Mom is relevant and {gender, dad} are not relevant.



https://en.wikipedia.org/wiki/Human_mitochondrial_genetics

Is "Relevance" Clearly Defined?

- What if "mom" feature is repeated?

gender	mom	dad	mom2
F	1	0	1
M	0	1	0
F	0	0	0
F	1	1	1

SNP
1
0
0
1

Neither of these is "correct", but not picking either is <u>incorrect</u>.

- Are "mom" and "mom2" relevant?
  - Should we pick them both?
  - Should we pick one because it predicts the other?
- If features can be predicted from features, can't know which to pick.
  - Collinearity is a special case of "dependence" (which may be non-linear).

Is "Relevance" Clearly Defined?

- What if we add (maternal) "grandma"?

gender	mom	dad	grandma
F	1	0	1
M	0	1	0
F	0	0	0
F	1	1	1

SNP
1
0
0
1

- Is "grandma" relevant?
  - You can predict SNP very accurately from "grandma" alone.
  - But "grandma" is irrelevant if I know "mom".
- A feature is only "relevant" in the context of available features.
  - Adding/removing features can make features relevant/irrelevant.

Is “Relevance” Clearly Defined?

- What if we don’t know “mom”?

gender	grandma	dad
F	1	0
M	0	1
F	0	0
F	1	1

SNP
1
0
0
1

- Now is “grandma” is relevant?
  - Without “mom” feature, using “grandma” is the best you can do.

- A feature is only “relevant” in the context of available features.
  - Adding/removing features can make features relevant/irrelevant.

Is “Relevance” Clearly Defined?

- What if we don’t know “mom” or “grandma”?

gender	dad
F	0
M	1
F	0
F	1

SNP
1
0
0
1

- Now there are no relevant features, right?
  - But “dad” and “mom” must have some common maternal ancestor.
  - “Mitochondrial Eve” estimated to be ~200,000 years ago.
- A “relevant” feature may have a tiny effect.

Is “Relevance” Clearly Defined?

- What if we don’t know “mom” or “grandma”?

gender	dad
F	0
M	1
F	0
F	1

SNP
1
0
0
1

- Now there are no relevant features, right?
  - What if “mom” likes “dad” because he has the same SNP as her?

- Confounding factors can make “irrelevant” features “relevant”.

Is “Relevance” Clearly Defined?

- What if we add “sibling”?

gender	dad	sibling
F	0	1
M	1	0
F	0	0
F	1	1

SNP
1
0
0
1

- Sibling is “relevant” for predicting SNP, but it’s not the cause.

- “Relevance” for prediction does not imply a causal relationship.
  - Causality can even be reversed…

Is “Relevance” Clearly Defined?

- What if don’t have “mom” but we have “baby”?

gender	dad	baby
F	0	1
M	1	1
F	0	0
F	1	1

SNP
1
0
0
1

- “Baby” is relevant when (gender == F).
  - “Baby” is relevant (though causality is reversed).
  - Is “gender” relevant?
    - If we want to find relevant causal factors, “gender” is not relevant.
    - If we want to predict SNP, “gender” is relevant.
- “Relevance” may depend on values of certain features.
  - “Context-specific” relevance.

Is "Relevance" Clearly Defined?

- Warnings about feature selection:
  - If features can be predicted from features, you can't know which to pick.
  - A feature is only "relevant" in the context of available features.
  - A "relevant" feature may have a tiny effect.
  - Confounding factors can make "irrelevant" features the most "relevant".
  - "Relevance" for prediction does not imply a causal relationship.
  - "Relevance" may depend on values of certain features.

Is this hopeless?

- We often want to do feature selection we so have to try!
- Different methods are affected by problems in different ways.
- These “problems” don’t have right answers but have wrong answers:
  - feature dependence (“mom” and “mom2” have same information).
    - But should take at least one.
  - Conditional independence (all “grandma” information is captured by “mom”).
    - Should take “grandma” only if “mom” missing.
- These “problems” have application-specific answers:
  - Tiny effects.
  - Context-specific relevance (is “gender” relevant if given “baby”?).
- See slides for discussion on causality and confounding issues.
  - Unless you control data collection, standard feature selection methods cannot address those issues.

Rough Guide to Feature Selection

Method\Issue	Dependence	Conditional Independence	Tiny effects	Context-Specific Relevance
Association (e.g., measure correlation between features 'j' and 'y')	Ok (takes "mom" and "mom2")	Bad (takes "grandma", "great-grandma", etc.)	Ignores	Bad (misses features that must interact, "gender" irrelevant given "baby")

Rough Guide to Feature Selection

Method\Issue	Dependence	Conditional Independence	Tiny effects	Context-Specific Relevance
Association (e.g., measure correlation between features 'j' and 'y')	Ok (takes "mom" and "mom2")	Bad (takes "grandma", "great-grandma", etc.)	Ignores	Bad (misses features that must interact, "gender" irrelevant given "baby")
Regression Weight (fit least squares, take biggest $\|w_j\|$)	Bad (can take irrelevant but collinear, can take none of "mom1-3")	Ok (takes "mom" not "grandma", if linear and 'n' large.	Ignores (unless collinear)	Ok (if linear, "gender" relevant give "baby")

Rough Guide to Feature Selection

Method\Issue	Dependence	Conditional Independence	Tiny effects	Context-Specific Relevance
Association (e.g., measure correlation between features 'j' and 'y')	Ok (takes "mom" and "mom2")	Bad (takes "grandma", "great-grandma", etc.)	Ignores	Bad (misses features that must interact, "gender" irrelevant given "baby")
Regression Weight (fit least squares, take biggest $\|w_j\|$)	Bad (can take irrelevant but collinear, can take none of "mom1-3")	Ok (takes "mom" not "grandma", if linear and 'n' large.	Ignores (unless collinear)	Ok (if linear, "gender" relevant give "baby")
Search and Score w/ Validation Error	Ok (takes at least one of "mom" and "mom2")	Bad (takes "grandma", "great-grandma", etc.)	Allows	Ok ("gender" relevant given "baby")

Rough Guide to Feature Selection

Method\Issue	Dependence	Conditional Independence	Tiny effects	Context-Specific Relevance
Association (e.g., measure correlation between features 'j' and 'y')	Ok (takes "mom" and "mom2")	Bad (takes "grandma", "great-grandma", etc.)	Ignores	Bad (misses features that must interact, "gender" irrelevant given "baby")
Regression Weight (fit least squares, take biggest $\|w_j\|$)	Bad (can take irrelevant but collinear, can take none of "mom1-3")	Ok (takes "mom" not "grandma", if linear and 'n' large.	Ignores (unless collinear)	Ok (if linear, "gender" relevant give "baby")
Search and Score w/ Validation Error	Ok (takes at least one of "mom" and "mom2")	Bad (takes "grandma", "great-grandma", etc.)	Allows (many false positives)	Ok ("gender" relevant given "baby")
Search and Score w/ L0-norm	Ok (takes exactly one of "mom" and "mom2")	Ok (takes "mom" not grandma if linear-ish).	Ignores (even if collinear)	Ok ("gender" relevant given "baby")

Feature Selection in Tree-Based Methods

- Decision trees naturally do feature selection while learning:
  - The features used for the splits are the ones that are "selected".

- There are a variety of ways evaluate features in random forests:
  - Compute proportion of trees that use feature 'j'.
  - Compute average infogain increase when using feature 'j'.
  - Permute all values of feature 'j', and see how "out of bag" error increases.
- You could use any of above to select features from random forest.

Mallow's Cp

- Older than AIC and BIC is Mallow's Cp:

$$f(w) = \frac{\|Xw - y\|^2}{\frac{1}{n}\|X\hat{w} - y\|^2} - n + 2\|w\|_0$$

$\hat{w}$ ← least squares weights if we used <u>all</u> features.

- Minimizing this score is equivalent to L0-regularization:

$$f(w) = \frac{1}{2}\|Xw - y\|^2 + \lambda\|w\|_0$$

with $\lambda = \frac{\|X\hat{w} - y\|^2}{n}$

- So again, viewing λ as hyper-parameter, this score is special case.

Adjusted R²

- Older than AIC and BIC and Mallow's Cp is adjusted R²:

$$f(w) = 1 - (1 - R^2)\frac{n-1}{n - \|w\|_0 - 1} \quad \text{where} \quad R^2 = 1 - \frac{\|Xw - y\|^2}{\|X\hat{w} - y\|^2}$$

- Maximizing this score is equivalent to L0-regularization:

$$= \frac{1}{2}\|Xw - y\|^2 + \lambda \|w\|_0$$

$$\text{with} \quad \lambda = \frac{\|X\hat{w} - y\|^2}{2(n-1)}$$

- So again, viewing λ as hyper-parameter, this score is special case.

ANOVA

- Some people also like to compute this “ANOVA” quantity:

$$f(w) = \frac{\|Xw - \bar{y}\|^2}{\|y - \underset{\uparrow}{\bar{y}}\|^2}$$

mean of y_i values repeated 'n' times

- This is based on the decomposition of “total squared error” as:

$$\|\underbrace{y - \bar{y}}_{\text{"total" error}}\|^2 = \|\underbrace{Xw - \bar{y}}_{\text{"explained" error}}\|^2 + \|\underbrace{Xw - y}_{\text{"residual" (usual) error.}}\|^2$$

- Notice that “explained error” goes up as our usual (“residual”) error goes down.
- Trying to find the ‘k’ features that maximize ‘f’ (“explain the most variance”) is equivalent to L0-regularization with a particular λ (so another special case).

Information Criteria with Noise Variance

- We defined AIC/BIC for feature selection in least squares as:

$$f(w) = \frac{1}{2}\|Xw - y\|^2 + \lambda\|w\|_0$$

- The first term comes from assuming $y_i = w^T x_i + \varepsilon$, where ε comes from a normal distribution with a variance of 1.
  - We'll discuss why when we discuss MLE and MAP estimation.
  - If you aren't doing least squares, replace first term by "log-likelihood".
- If you treat variance as a parameter, then after some manipulation:

$$f(w) = \frac{n}{2}\log(\|Xw - y\|^2) + \lambda\|w\|_0$$

- However, this is again equivalent to just changing λ.

Complexity Penalties for Other Models

- Scores like AIC and BIC can also be used in other contexts:
  - When fitting a decision tree, only split a node if it improves BIC.
  - This makes sense if we're looking for the "true tree", or maybe just a simple/interpretable tree that performs well.

- In these cases we replace "L0-norm" with "degrees of freedom".
  - In linear models fit with least squares, degrees of freedom is number of non-zeroes.
  - Unfortunately, it is not always easy to measure "degrees of freedom".

Alternative to Search and Score: good old p-values

- Hypothesis testing ("constraint-based") approach:
  - Generalization of the "association" approach to feature selection.
  - Performs a sequence of conditional independence tests.

$$x_{ij} \perp y_i \mid x_{is}$$

feature 'j' label other features 's'

"If I know features in 's' does feature 'j' tell me anything about label?"

  - If they are independent (like "p < .05"), say that 'j' is "irrelevant".
- Common way to do the tests:
  - "Partial" correlation (numerical data).
  - "Conditional" mutual information (discrete data).

Testing-Based Feature Selection

- Hypothesis testing ("constraint-based") approach:
- Two many possible tests, "greedy" method is for each 'j' do:

First test if $x_{ij} \perp y_i$

If still dependent test $x_{ij} \perp y_i \mid x_{is}$ where 's' has one feature

If still dependent test $x_{ij} \perp y_i \mid x_{is}$ where 's' now has two features

Often choose features to minimize dependence.

⋮

If still dependent when 's' includes all other features, declare 'j' relevant.

- "Association approach" is the greedy method where you only do the first test (subsequent tests remove a lot of false positives).

Hypothesis-Based Feature Selection

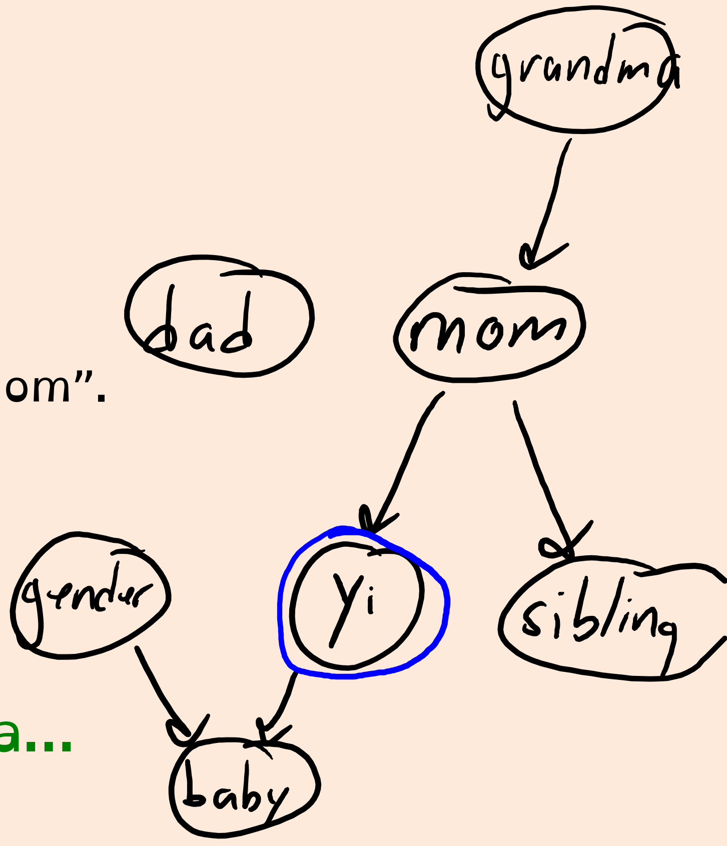
- Advantages:
  - Deals with conditional independence.
  - Algorithm can explain why it thinks ‘j’ is irrelevant.
  - Doesn’t necessarily need linearity.
- Disadvantages:
  - Deals badly with exact dependence: doesn’t select “mom” or “mom2” if both present.
  - Usual warning about testing multiple hypotheses:
    - If you test $p < 0.05$ more than 20 times, you’re going to make errors.
  - Greedy approach may be sub-optimal.
- Neither good nor bad:
  - Allows tiny effects.
  - Says “gender” is irrelevant when you know “baby”.
  - This approach is sometimes better for finding relevant factors, not to select features for learning.

Causality

- None of these approaches address causality or confounding:
  - "Mom" is the only relevant direct causal factor.
  - "Dad" is really irrelevant.
  - "Grandma" is causal but is irrelevant if we know "mom".

- Other factors can help prediction but aren't causal:
  - "Sibling" is predictive due to confounding of effect of same "mom".
  - "Baby" is predictive due to reverse causality.
  - "Gender" is predictive due to common effect on "baby".

- We can sometimes address this using interventional data…



Interventional Data

- The difference between observational and interventional data:
  - If I see that my watch says 10:45, class is almost over (observational).
  - If I set my watch to say 10:45, it doesn't help (interventional).

- The intervention can help discover causal effects:
  - "Watch" is only predictive of "time" in observational setting (so not causal).

- General idea for identifying causal effects:
  - "Force" the feature to take a certain value, then measure the effect.
    - If the dependency remains, there is a causal effect.
    - We "break" connections from reverse causality, common effects, or confounding.

Causality and Dataset Collection

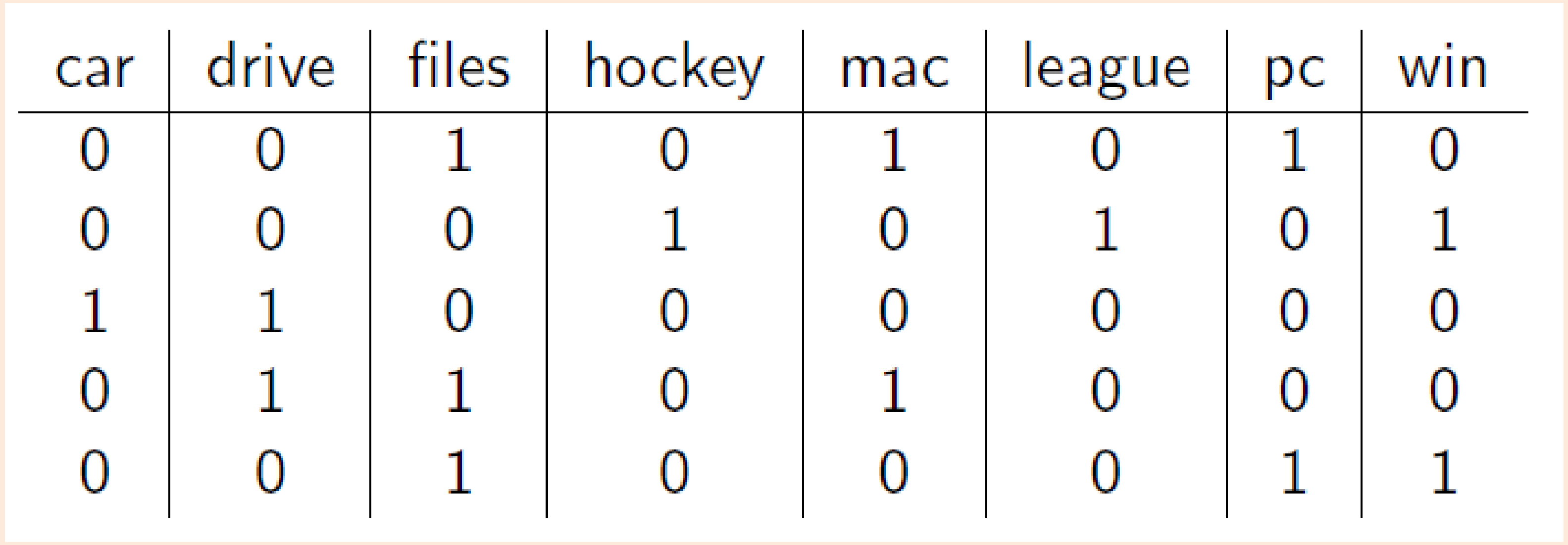
- This has to do with the way you collect data:
  - You can't "look" for features taking the value "after the fact".
  - You need to manipulate the value of the feature, then watch for changes.

- This is the basis for randomized control trial in medicine:
  - Randomly assigning pills "forces" value of "treatment" feature.
    - Randomization means they aren't taking the pill due to confounding factors.
    - Differences between people who did and did not take pill should be caused by pill.
  - Include a "control" as a value to prevent placebo effect as confounding.

- See also Simpson's Paradox:
  - https://www.youtube.com/watch?v=ebEkn-BiW5k

Structure Learning: Unsupervised Feature Selection

- "News" data: presence of 100 words in 16k newsgroup posts:



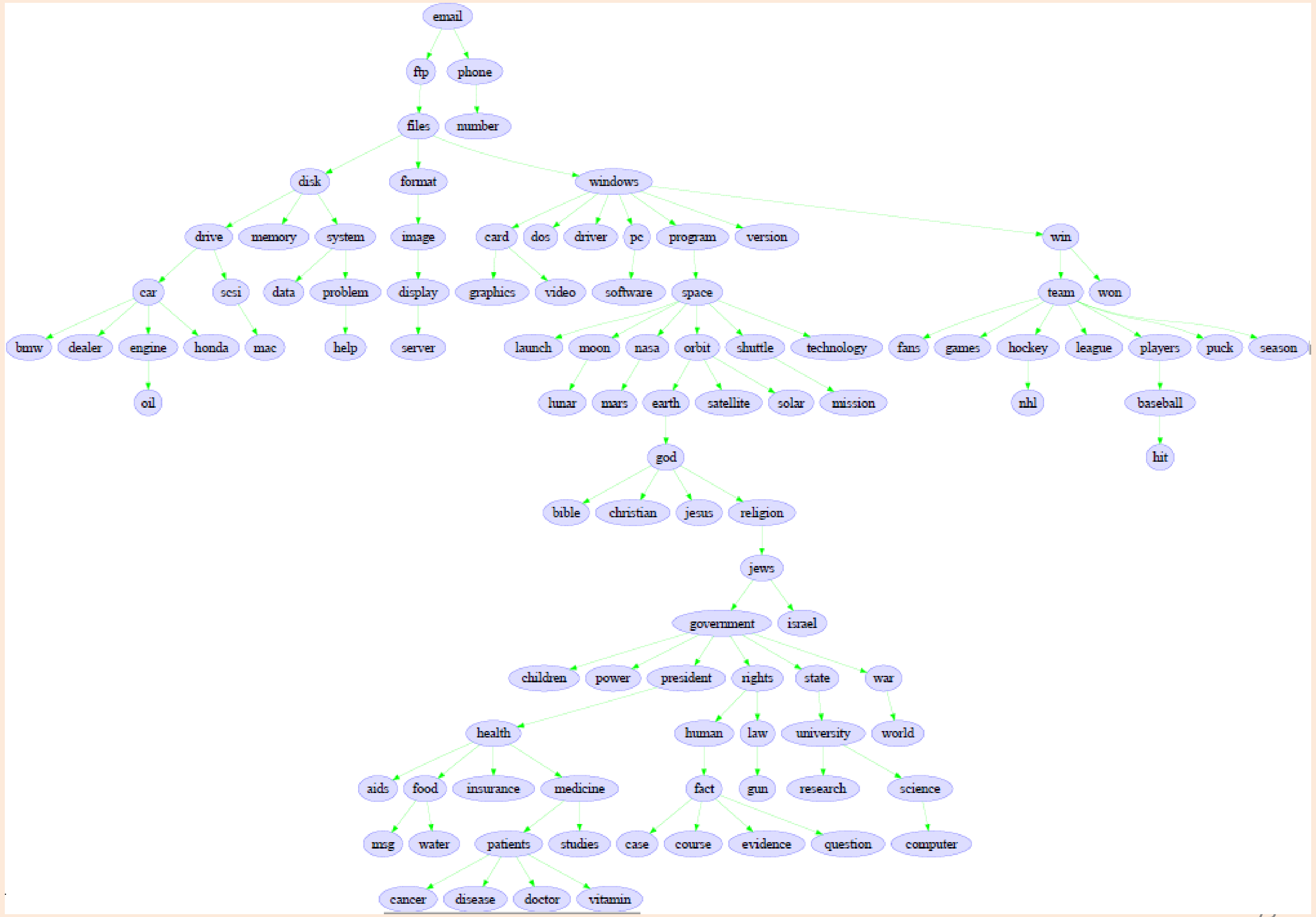
car	drive	files	hockey	mac	league	pc	win
0	0	1	0	1	0	1	0
0	0	0	1	0	1	0	1
1	1	0	0	0	0	0	0
0	1	1	0	1	0	0	0
0	0	1	0	0	0	1	1

- Which words are related to each other?

- Problem of structure learning: unsupervised feature selection.

Structure Learning: Unsupervised Feature Selection

- Optimal tree structure: (ignore arrows)



Naïve Approach: Association Networks

- A naïve approach to structure learning ("association networks"):
  - For each pair of features, compute a measure of similarity or dependence.

- Using these n^2 similarity values either:
  - Select all pairs whose similarity is above a threshold.
  - Select the "top k" most similar features to each feature 'j'.

- Main problems:
  - Usually, most features are dependent (too many edges).
    - "Sick" is getting connected to "Tuesdays" even if "tacos" are a feature.
  - "True" neighbours may not have the highest dependence.
    - "Sick" might get connected to "Tuesdays" before it gets connected to "milk".

- (Variation: best tree can be found as minimum spanning tree problem.)

Example: Vancouver Rain Data

- Consider modeling the "Vancouver rain" dataset.

	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8	Day 9	…
Month 1	0	0	0	1	1	0	0	1	1	
Month 2	1	0	0	0	0	0	1	0	0	
Month 3	1	1	1	1	1	1	1	1	1	
Month 4	1	1	1	1	0	0	1	1	1	
Month 5	0	0	0	0	1	1	0	0	0	
Month 6	0	1	1	0	0	0	0	1	1	

- The strongest signal in the data is the simple relationship:
  - If it rained yesterday, it's likely to rain today (> 50% chance that $x^{t-1} = x^t$).
  - But an "association network" might connect all days (all dependent).

Dependency Networks

- A better approach is dependency networks:
  - For each feature ‘j’, make it the target in a supervised learning problem.

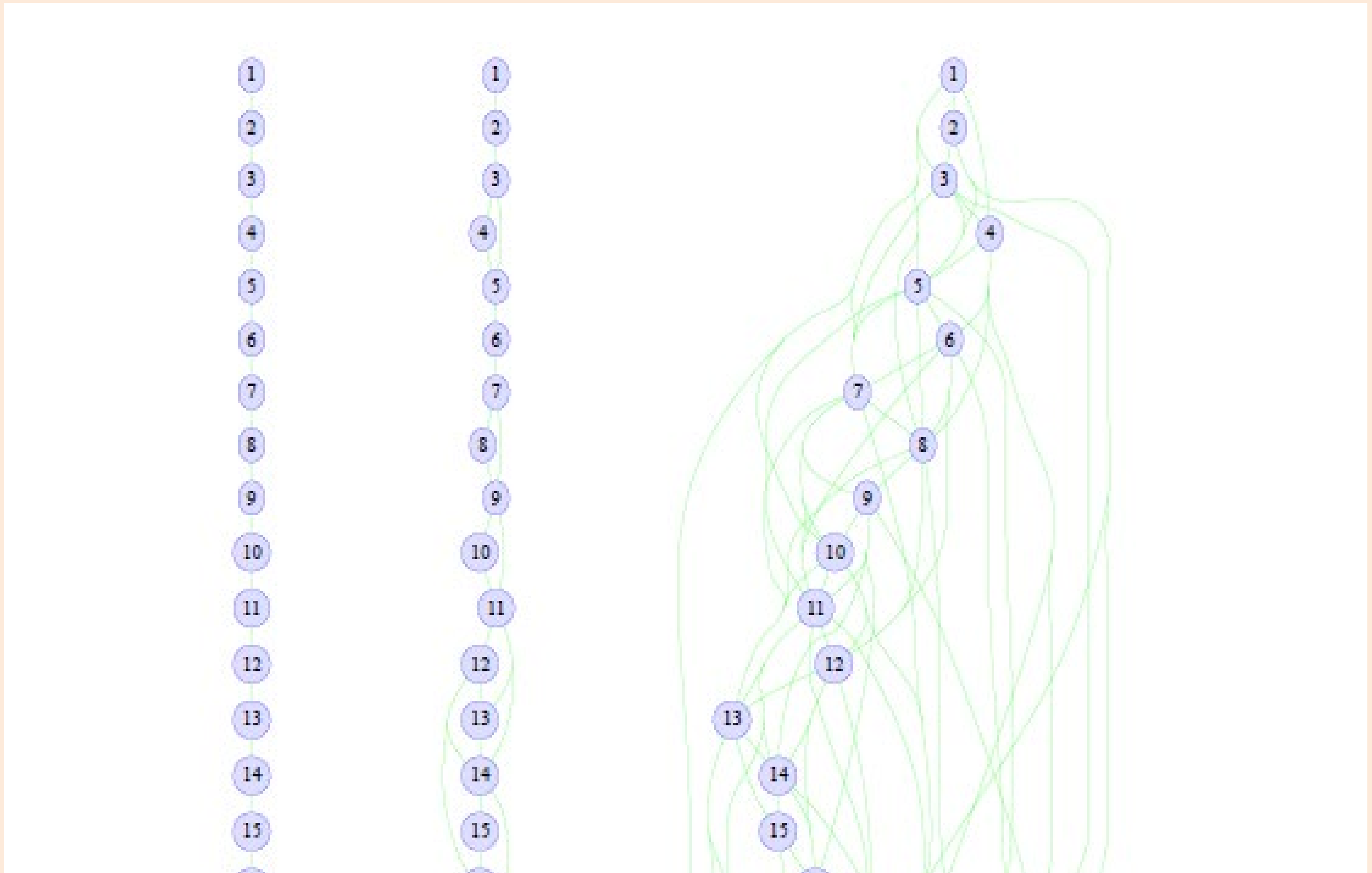
$$X = \begin{bmatrix} | & | & | & | & | \\ x^1 & x^2 & x^3 & \boxed{x^4} & x^5 \\ | & | & | & | & | \end{bmatrix} \Longrightarrow \bar{X} = \begin{bmatrix} | & | & | & | \\ x^1 & x^2 & x^3 & x^5 \\ | & | & | & | \end{bmatrix} \quad y = \begin{bmatrix} | \\ x^4 \\ | \end{bmatrix}$$

  - Now we can use any feature selection method to choose j’s “neighbours”.
    - Forward selection, L1-regularization, ensemble methods, etc.

- Can capture conditional independence:
  - Might connect “sick” to “tacos”, and “tacos” to “Tuesdays” (w/o sick-tacos).

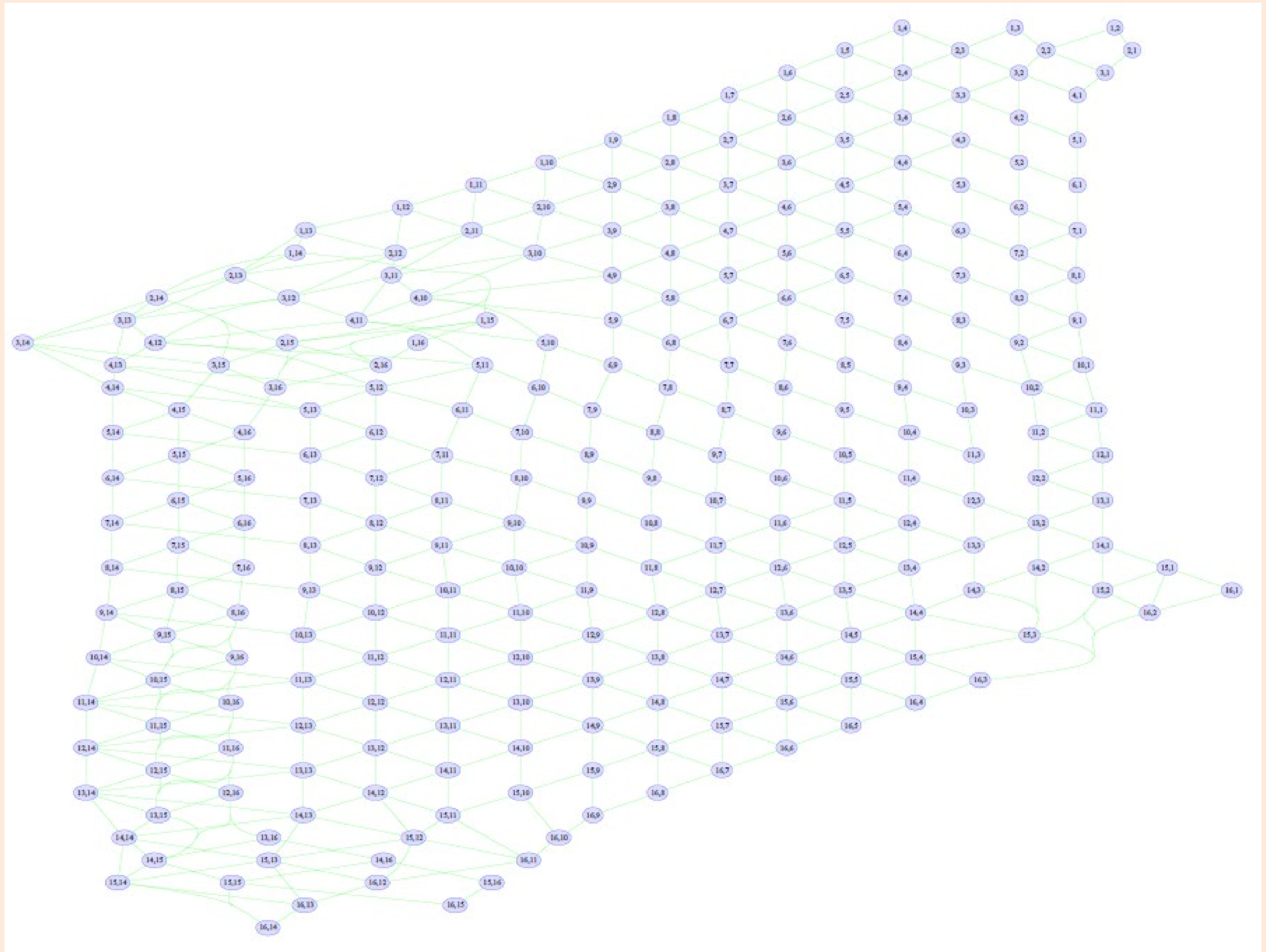
Dependency Networks

- Dependency network fit to Vancouver rain data (different λ values):



Dependency Networks

- Variation on dependency networks on digit image pixels:



Another popular structure learning method is the "PC" algorithm.