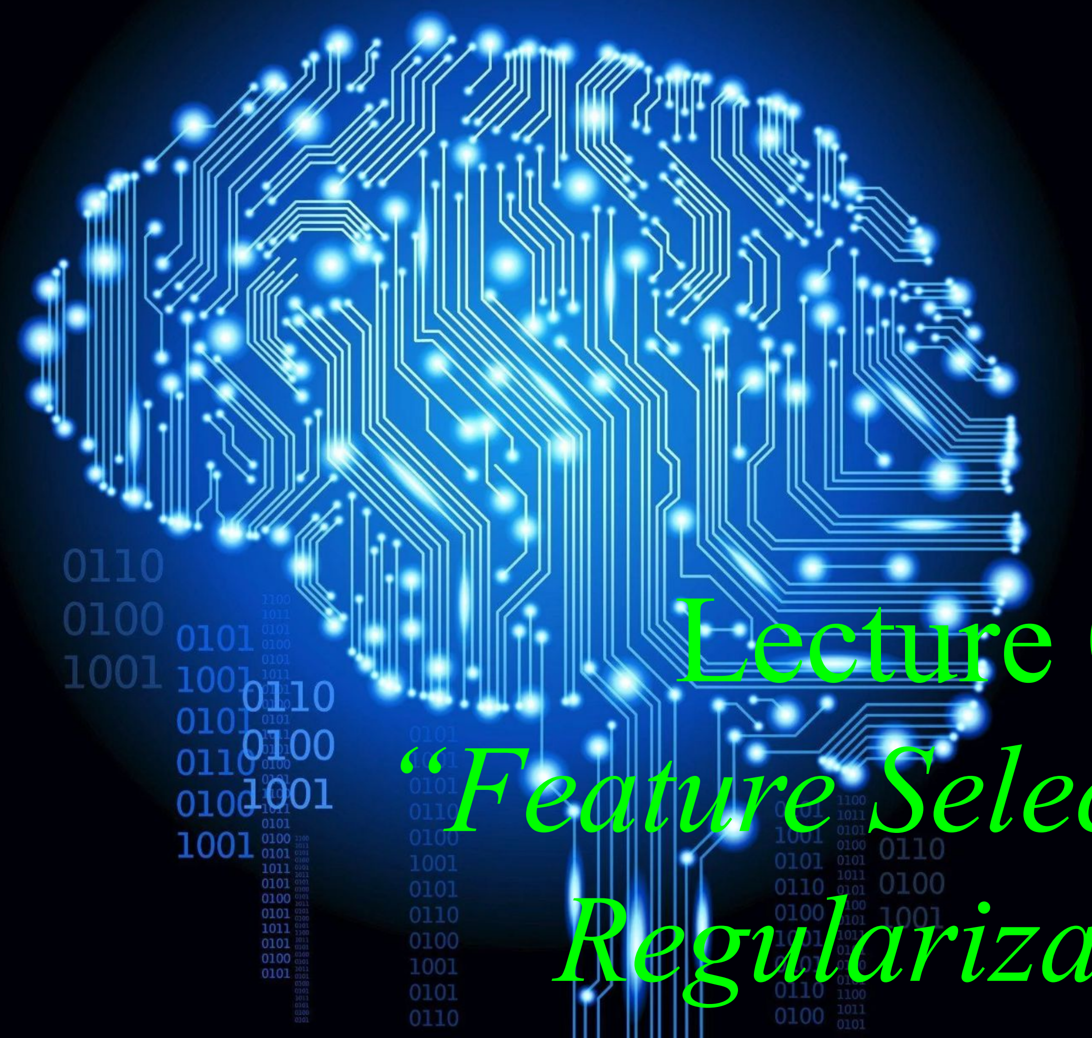


Lecture 06:

“Feature Selection and Regularization”

Course Logistics

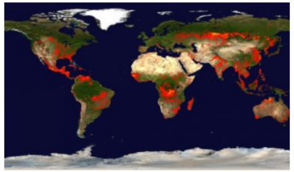
Schedule (updated)

Week	Lecture (Tues, 2:30 @ MCB 113)	Lab (Thurs, 2:30 @ MCB 113)	Assignment
7 (Oct 26, 28)	Midterm Review (L01-L06)	**MIDTERM 2:30 - 4:30 PM**	...
8 (Nov 2, 4)	READING WEEK		
9 (Nov 9, 11)	L09: Trees and Random Forest	Code Review 9	A5: released Nov 11, due Nov 21
10 (Nov 16, 18)	L10: Neural Networks, Deep Learning	Code Review 10	**A6: released Nov 21, due Dec 01**
11 (Nov 23, 25)	L11: Unsupervised Learning, RL	Code Review 11	...
12 (Nov 30, Dec 02)	L12: Dimensionality Reduction	Project Presentations	...
13 (Dec 07)	Exam Review	...	...

Assignment Solutions (for midterm studying)

- A1-A3 Solutions released tonight after the lecture
- A4 Solutions released next Tuesday night

A4: Model Selection, Cross-validation, Confidence Intervals [__ /70 marks]



In this assignment we will compare 3 different models (and select one) on a modified version of the "forest fires" dataset. Specifically, given some input features (temperature, relative humidity, etc.) and an output y (`area`) we wish to build models, select a particular model, and make predictions on unseen data. We also want to bound our prediction with a 95% confidence interval (CI); for this confidence interval we will use the Central Limit Theorem (CLT).

Before you start...

- see relevant lecture code (`L5_CF.ipynb` , `L4_CF.ipynb`)

Before you submit...

- restart the kernel, then re-run the whole notebook to ensure no errors

In [9]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.linear_model import LinearRegression
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler, PolynomialFeatures
from sklearn.metrics import make_scorer
from sklearn.base import BaseEstimator, TransformerMixin
%matplotlib inline
```

Question 1.1 [_ /4 marks]

Read the file `ffd.csv` into a dataframe. Display the first 5 rows of this dataframe.

In [1]:
```
# Read ffd.csv into a dataframe [ /2 marks]
# ****** your code here ******
```

Project Groups (Graduate)

- We have ~½ the groups signed up
- I will start to assist here

Rate Me (feedback.uwo.ca)

- Any comments/feedback are appreciated
  - Whether you like the course or think it sucks
- I’ll make sure to use feedback to adjust where I can

How to use Reshape

- What does reshape(-1) do?
  - From 2d array to 1d vector
- What does reshape(-1,1) do?
  - From 1d vector to 2d column array/vector

Feature Construction

Construct Features?

This dataset has **6 features**

- Sometimes it may be useful to give *combinations of features* to our models
- Ex: Construct a feature called "LotArea"

$$\hat{y}_i = b_0 + b_1(LotDepth) + b2(LotFrontage)$$

vs.

$$\hat{y}_i = b_0 + b_1(LotArea)$$

	Id	LotDepth	LotFrontage	Street	Utilities	HasPoolTable	SalePrice
0	1	53.0	65.0	Pave	AllPub	True	208500
1	2	76.0	80.0	Pave	AllPub	False	181500
2	3	56.0	68.0	Pave	AllPub	False	223500
3	4	46.0	60.0	Pave	AllPub	False	140000
4	5	72.0	84.0	Pave	AllPub	True	250000
...	...	...	...	...	...	...	...
1455	1456	50.0	62.0	Pave	AllPub	False	175000
1456	1457	81.0	85.0	Pave	AllPub	True	210000
1457	1458	52.0	66.0	Pave	AllPub	False	266500
1458	1459	64.0	68.0	Pave	AllPub	True	142125
1459	1460	71.0	75.0	Pave	AllPub	True	147500

Beyond Linearity

- As seen in previous weeks, we can add nonlinearity to our features
- We can extend this by adding further augmentations to features with **basis functions** (also called “feature functions”). With these we are defining new features.

Ex:

$h_m(X) = x_m, m = 1, \ldots, p$ $\longrightarrow$ Original features

$h_m(X) = x^2_m, \ m = 1, \ldots, p$ $\longrightarrow$ Square each feature

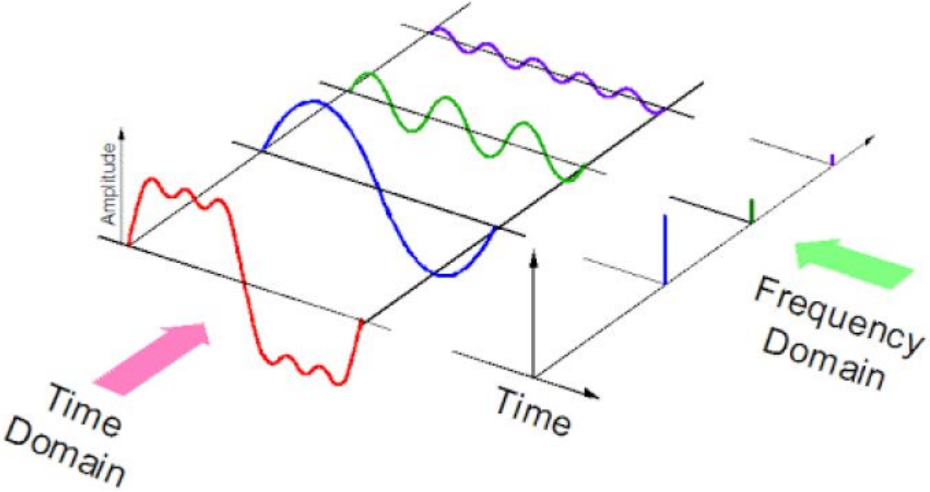
$h_m(X) = x_m\left(x_{m+1}\right), \ m = 1, 4, 9$ $\longrightarrow$ Square particular features

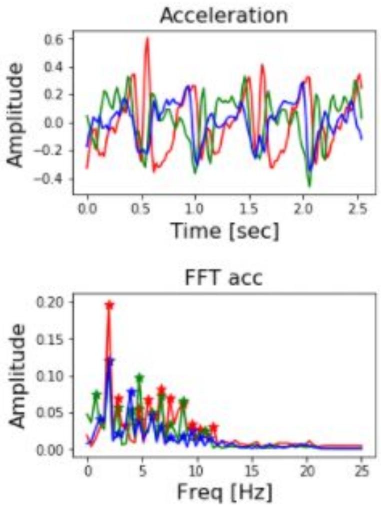
$h_m(X) = \log\left(x_m\right), \ m = 1, \ldots, p$ $\longrightarrow$ Apply other nonlinear transformation

Example: Fourier Basis

Ex: $h_0(x) = 1$ $h_j(x) = \cos\left(\omega_j x + \Psi_j\right)$ for $j > 0$

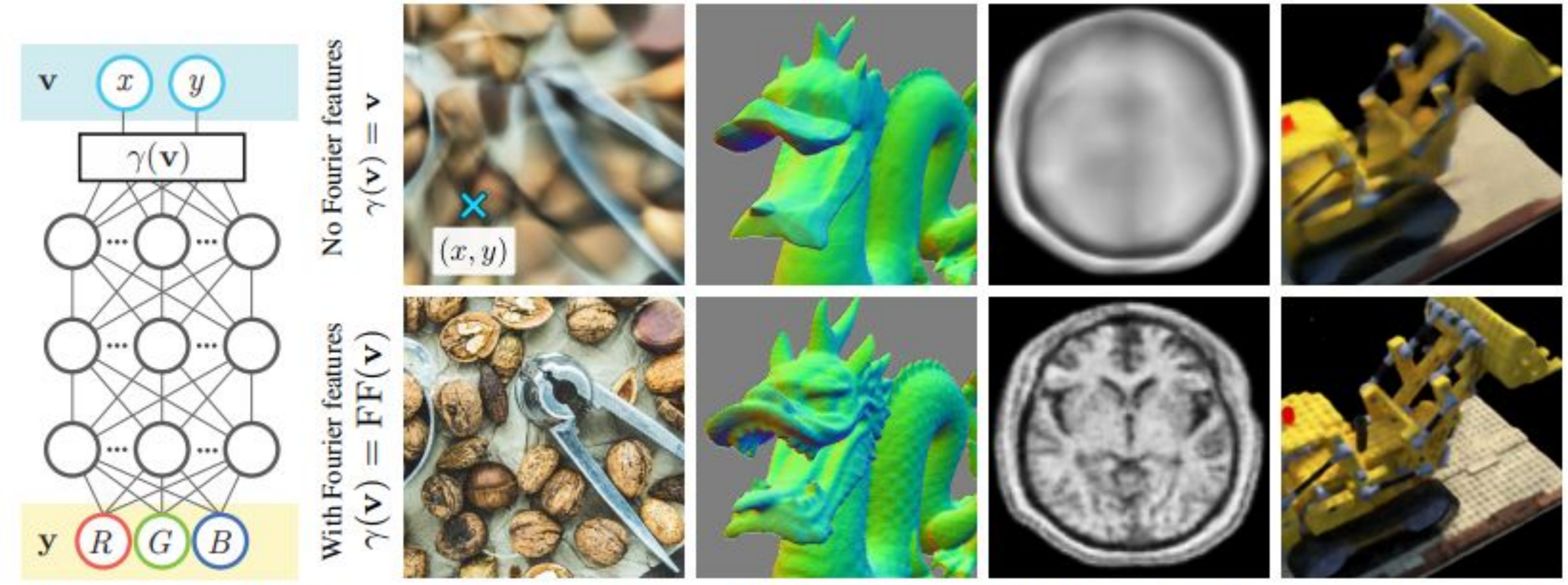
- This is useful for various temporal/periodic and signal processing tasks (ECG, audio)





- Allows us to extract features from time domain and construct them for frequency domain.

Example: Fourier Features



(a) Coordinate-based MLP (b) Image regression $(x,y) \rightarrow$ RGB (c) 3D shape regression $(x,y,z) \rightarrow$ occupancy (d) MRI reconstruction $(x,y,z) \rightarrow$ density (e) Inverse rendering $(x,y,z) \rightarrow$ RGB, density

https://proceedings.neurips.cc/paper/2020/file/55053683268957697aa39fba6f231c68-Paper.pdf

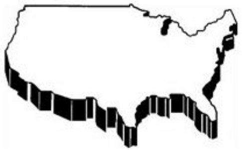
Feature Selection

Select Features?

Communities and Crime Data Set

Download: Data Folder, Data Set Description

Abstract: Communities within the United States. The data combines socio-economic data from the 1990 US Census, law enforcement data from the 1990 US LEMAS survey, and crime data from the 1995 FBI UCR.



Data Set Characteristics:	Multivariate	Number of Instances:	1994	Area:	Social
Attribute Characteristics:	Real	Number of Attributes:	128	Date Donated	2009-07-13
Associated Tasks:	Regression	Missing Values?	Yes	Number of Web Hits:	331742

	state	county	community	communityname	fold	population	householdsize	racepctblack	racePctWhite	racePctAsian	...	LandArea	PopDens
16	36	1	1000	Albanycity	1	0.15	0.31	0.40	0.63	0.14	...	0.06	0.39
23	19	193	93926	SiouxCitycity	1	0.11	0.43	0.04	0.89	0.09	...	0.16	0.12
33	51	680	47672	Lynchburgcity	1	0.09	0.43	0.51	0.58	0.04	...	0.14	0.11
68	34	23	58200	PerthAmboycity	1	0.05	0.59	0.23	0.39	0.09	...	0.01	0.73
74	9	9	46520	Meridentown	1	0.08	0.39	0.08	0.85	0.04	...	0.07	0.21
...	...	...	...	...	...	...	...	...	...	...	...	...	...
1880	34	39	40350	Lindencity	10	0.04	0.39	0.39	0.65	0.09	...	0.03	0.28
1963	36	27	59641	Poughkeepsiecity	10	0.03	0.32	0.61	0.47	0.09	...	0.01	0.47
1981	9	9	35650	Hamdentown	10	0.07	0.38	0.17	0.84	0.11	...	0.09	0.13
1991	9	9	80070	Waterburytown	10	0.16	0.37	0.25	0.69	0.04	...	0.08	0.32
1992	25	17	72600	Walthamcity	10	0.08	0.51	0.06	0.87	0.22	...	0.03	0.38

This dataset has **128 features**

- Can we isolate a subset of these features?

Feature Selection

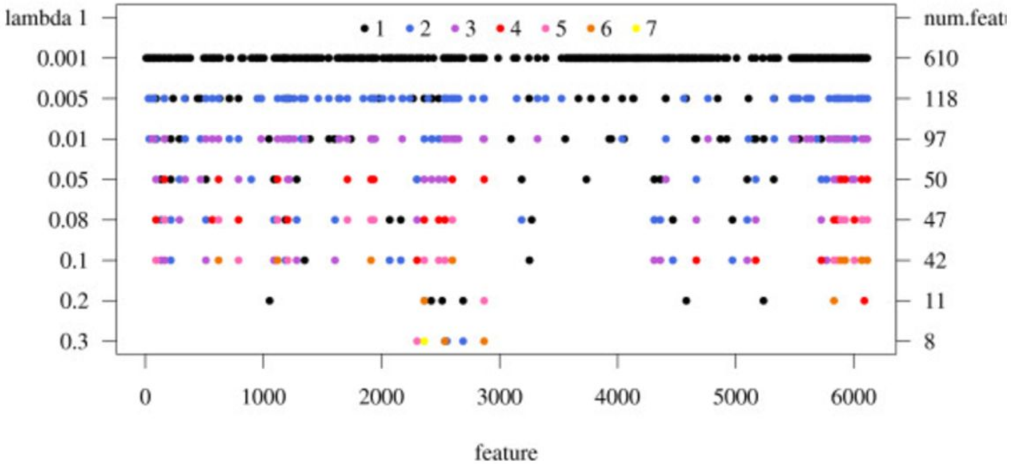
Why?

- Some of the features are not associated with the target/output variable y
- If we have many irrelevant features → unnecessary complexity in model
- Why is that a problem? (1) Computation costs/time; (2) Variance → Overfitting

	Id	LotDepth	LotFrontage	Street	Utilities	HasPoolTable	SalePrice
0	1	53.0	65.0	Pave	AllPub	True	208500
1	2	76.0	80.0	Pave	AllPub	False	181500
2	3	56.0	68.0	Pave	AllPub	False	223500
3	4	46.0	60.0	Pave	AllPub	False	140000
4	5	72.0	84.0	Pave	AllPub	True	250000
...	...	...	...	...	...	...	...

3 Common Approaches

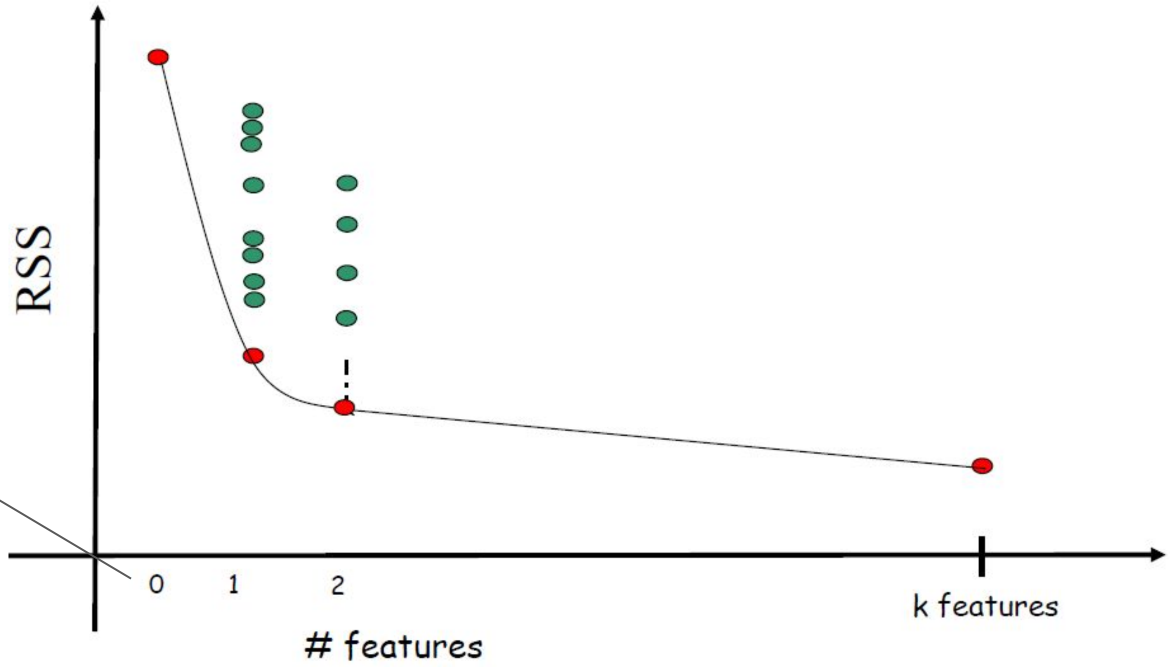
- **Subset Selection**: Identify a subset of k features (predictors) which we believe are related to the output (response/target).
- **Regularization**: Keep all k features, but shrink some number of these features towards zero ~ eliminating influence of the features
- **Dimensionality reduction**: Project the k features into a lower-dimensional subspace



Subset Selection

Approach 1.1: Best Subset Selection

- Search over all features and combinations of features → get all model performances



Null model

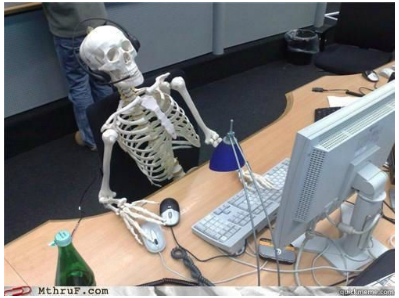
https://link-springer-com.proxy1.lib.uwo.ca/content/pdf/10.1007%2F978-1-4614-7138-7.pdf

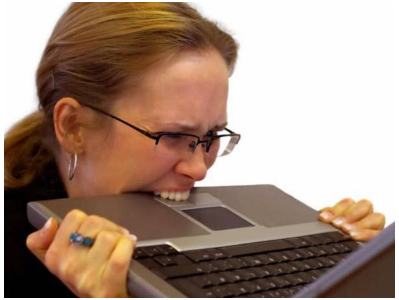
$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

Approach 1.1: Best Subset Selection

- For k features and models containing 1 of those features → $\binom{k}{1}$ models considered
- For k features and models containing 2 of those features → $\binom{k}{2}$ models considered
- For k features and models containing n of those features → $\binom{k}{n}$ models considered

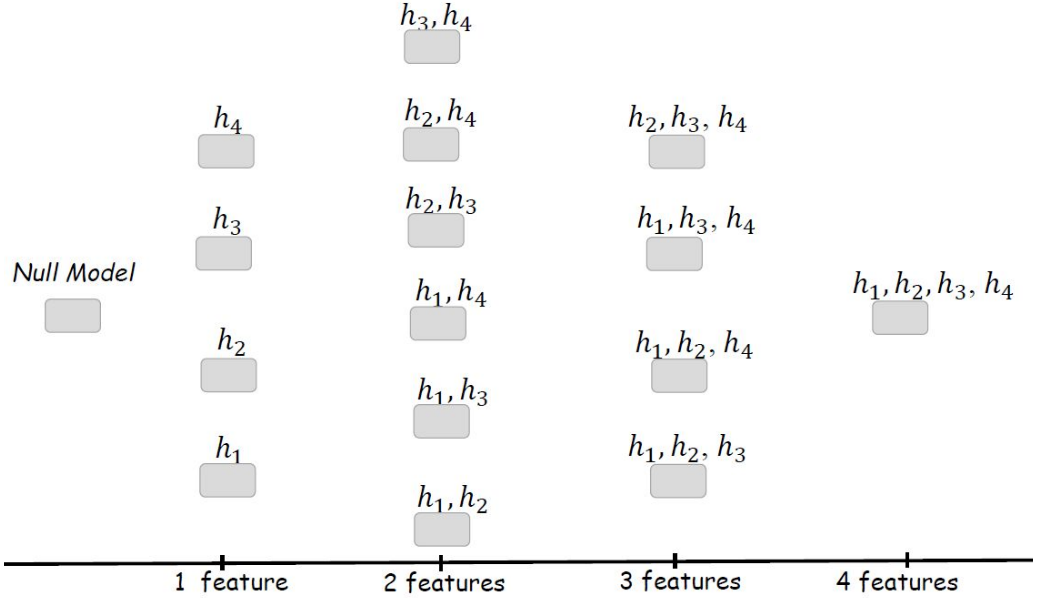
The complexity of "Best Subset Selection" → $1 + k + \binom{k}{2} + \binom{k}{3} + \ldots + \binom{k}{n} \approx 2^k$





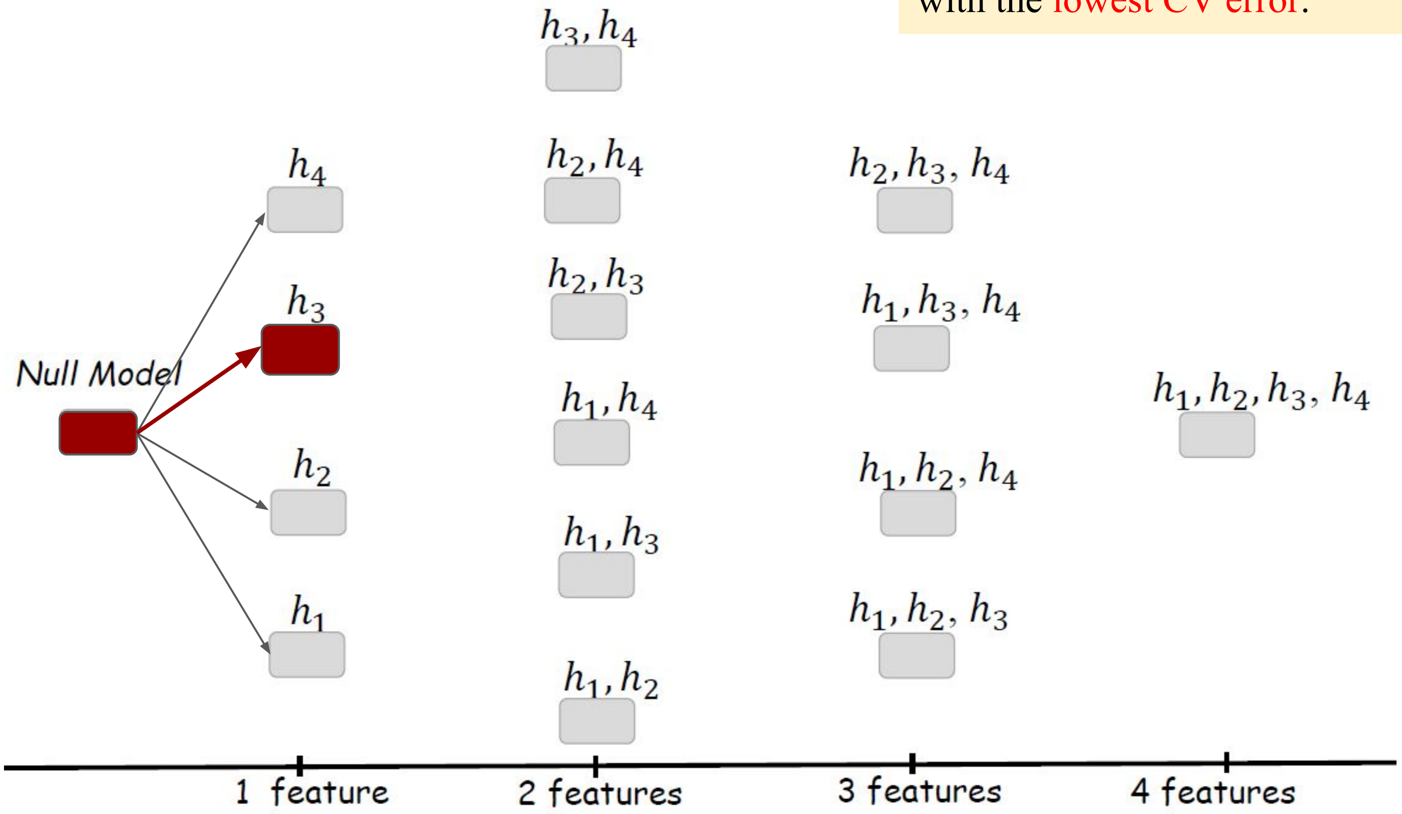
Approach 1.2: Stepwise Selection

1. Forward Stepwise Selection
2. Backward Stepwise Selection



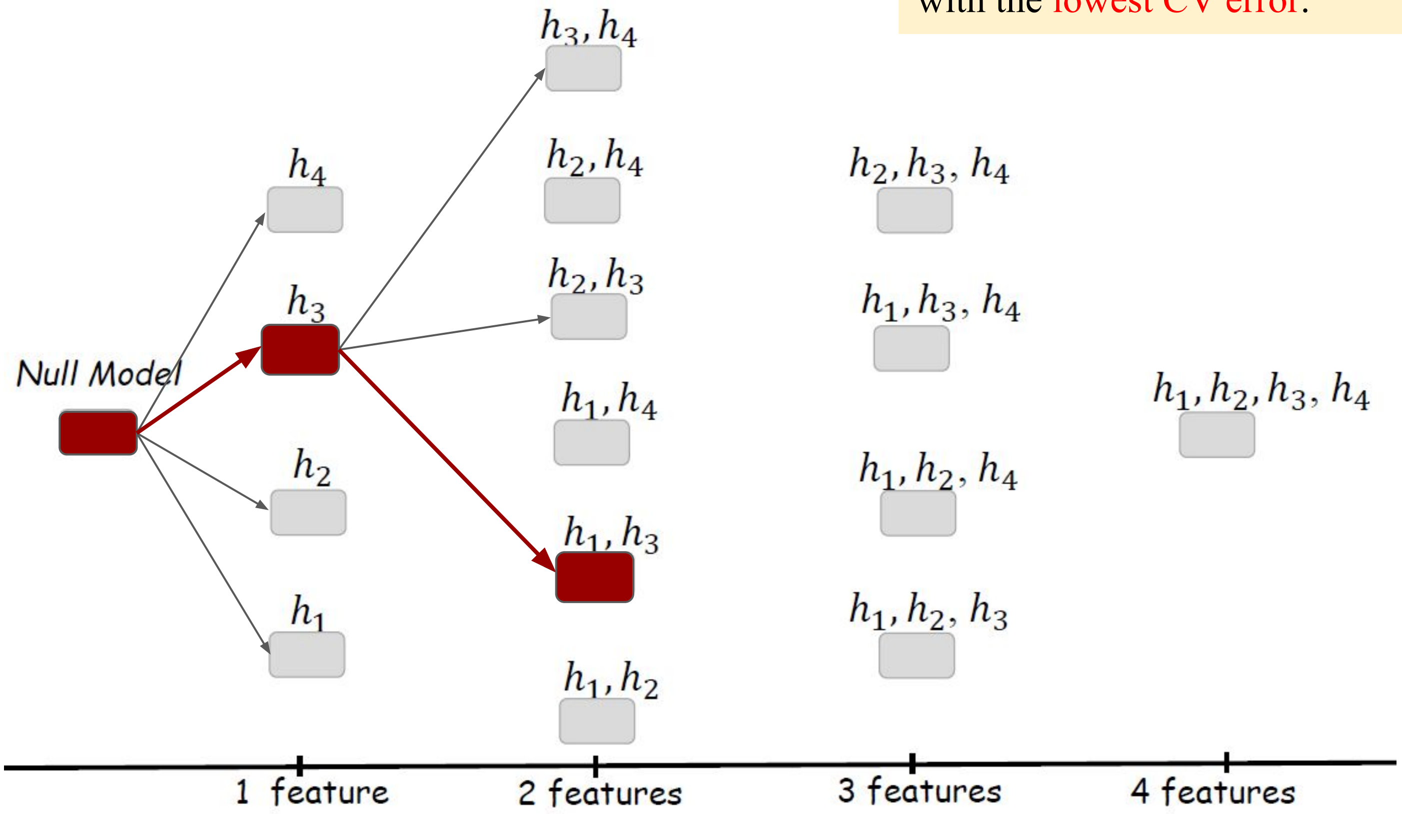
Forward Stepwise Selection

At each step, choose the **best model** with the lowest CV error.



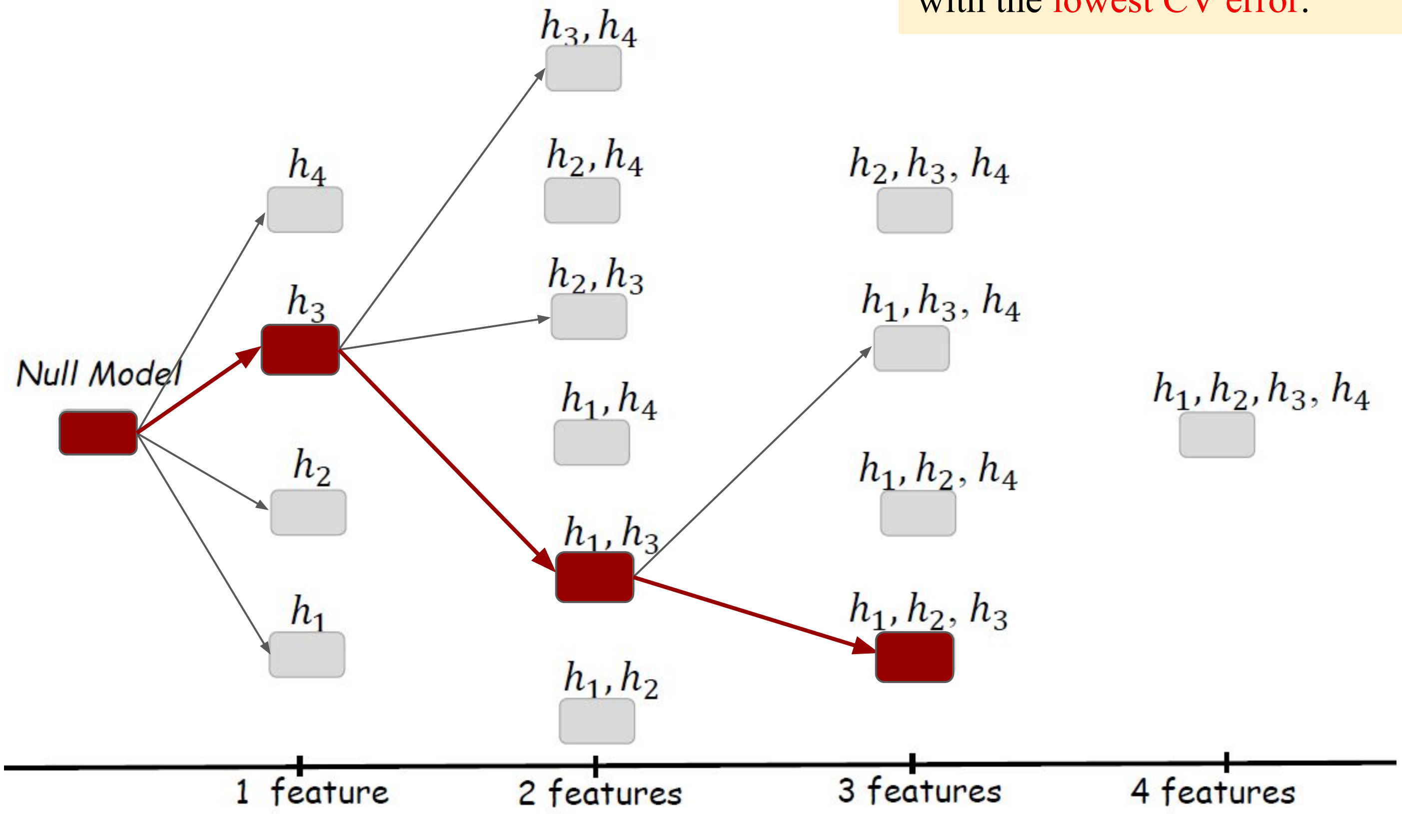
Forward Stepwise Selection

At each step, choose the **best model** with the lowest CV error.



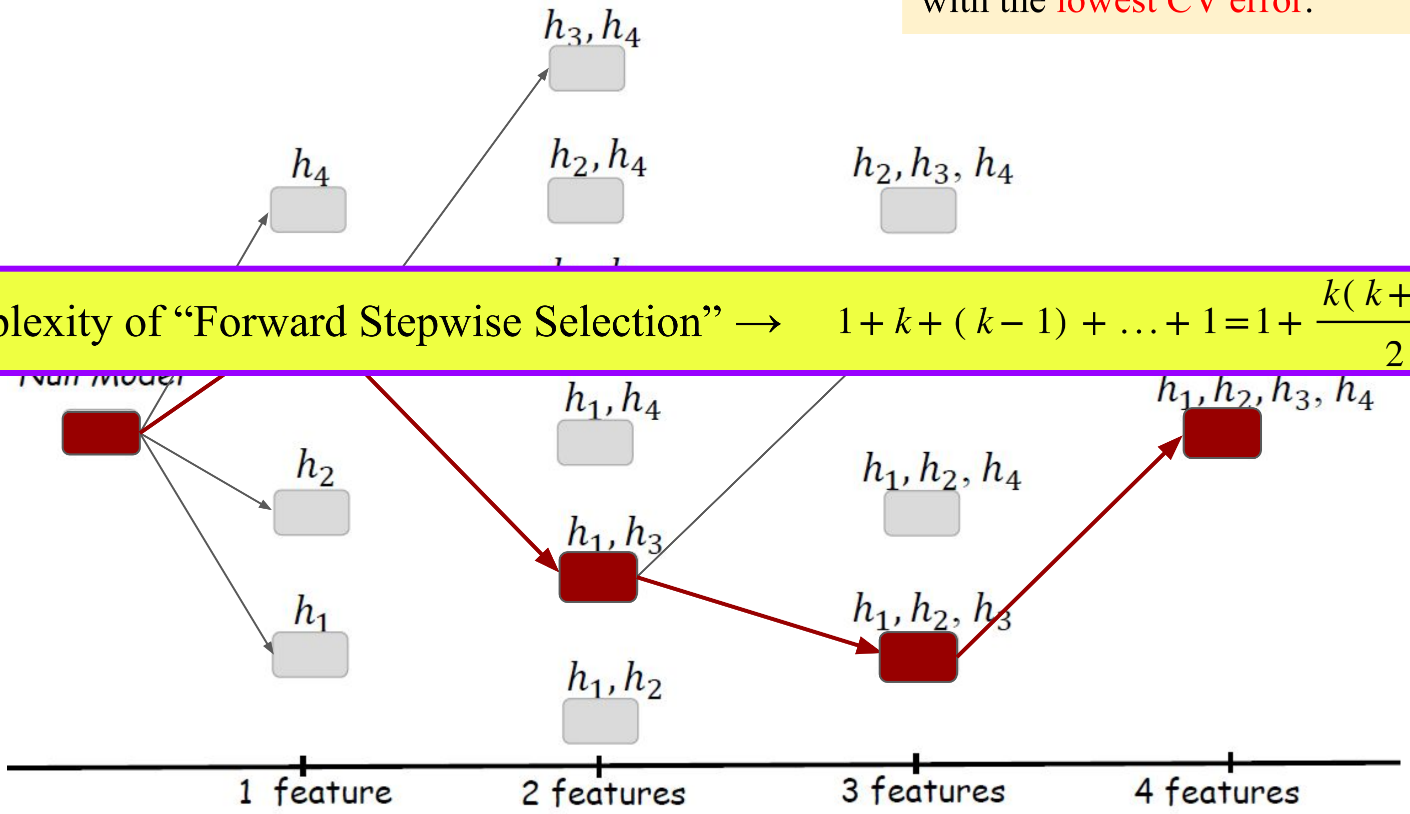
Forward Stepwise Selection

At each step, choose the **best model** with the lowest CV error.



Forward Stepwise Selection

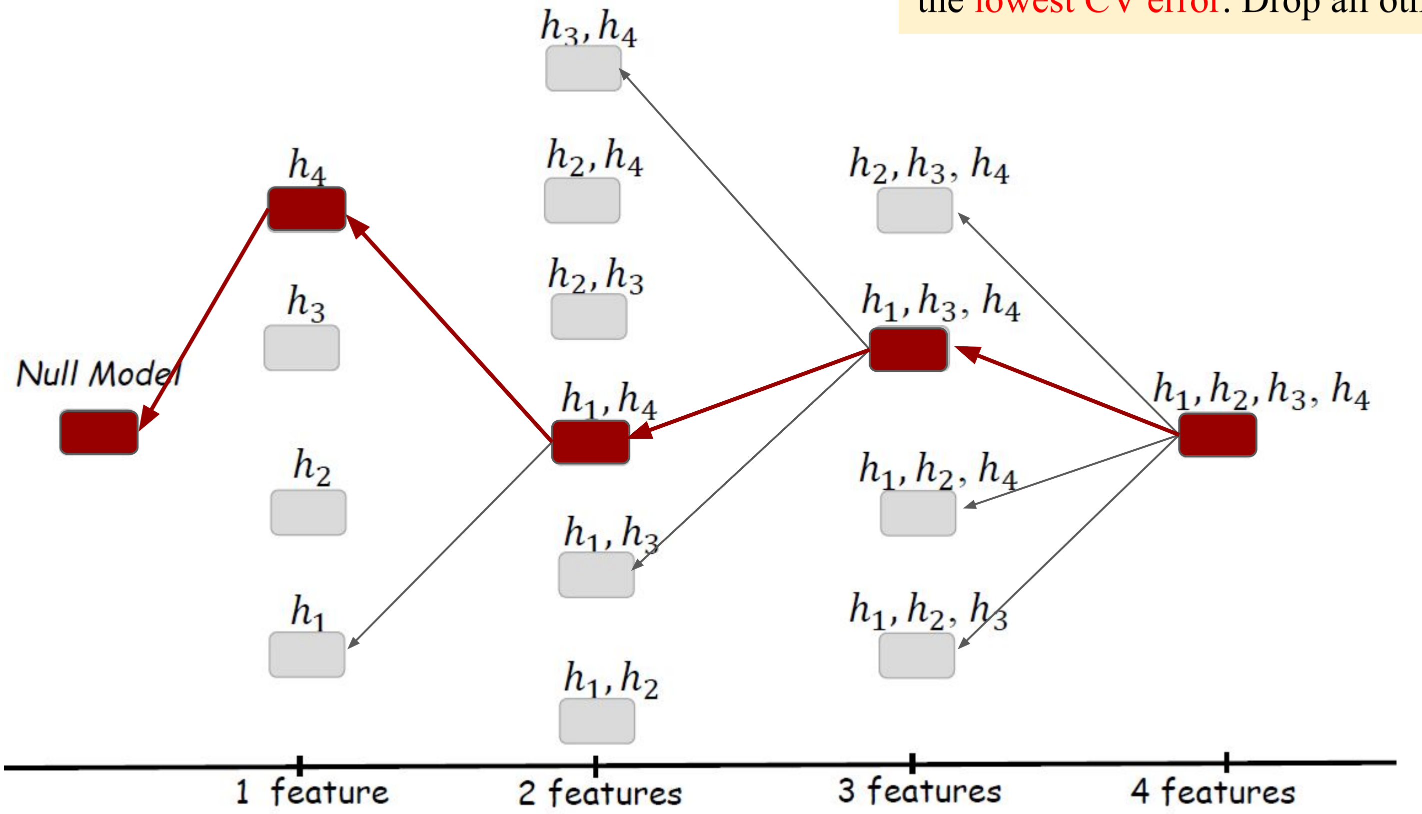
At each step, choose the **best model** with the lowest CV error.



The complexity of "Forward Stepwise Selection" $\longrightarrow$ $1 + k + (k-1) + \ldots + 1 = 1 + \frac{k(k+1)}{2}$

Backward Stepwise Selection

At each step, keep the **best model** with the lowest CV error. Drop all others.

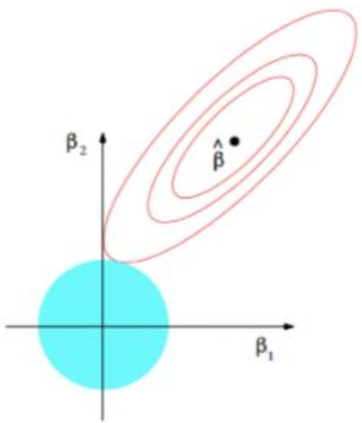


Regularization

Regularization

- Unlike with subset selection and stepwise selection, here we **keep all features**
- *How do we select*? The coefficients of particular features shrink towards 0
- Ex: We can consider all features and all quadratic feature combinations at first… then when we regularize certain features and feature combinations will be "removed"

Objective = Measure of fit + Measure of Magnitude of Coefficients

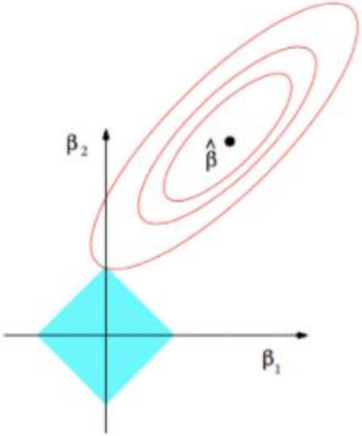


$$\lambda \sum_j \beta_j^2$$

Ridge

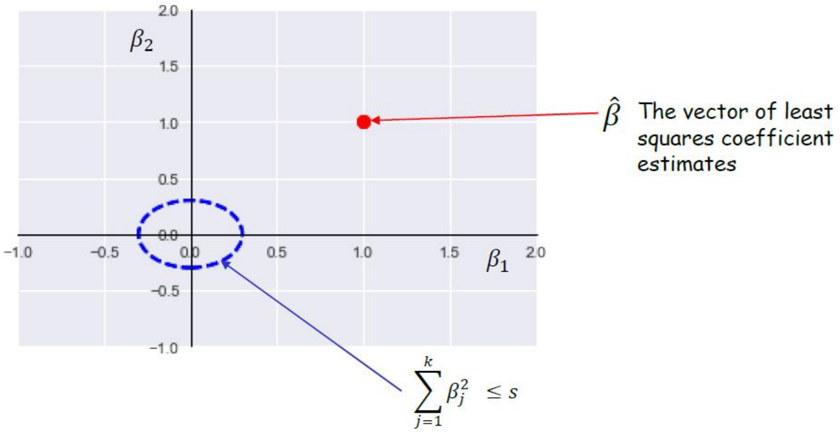
$$\lambda \sum_j |\beta_j|$$

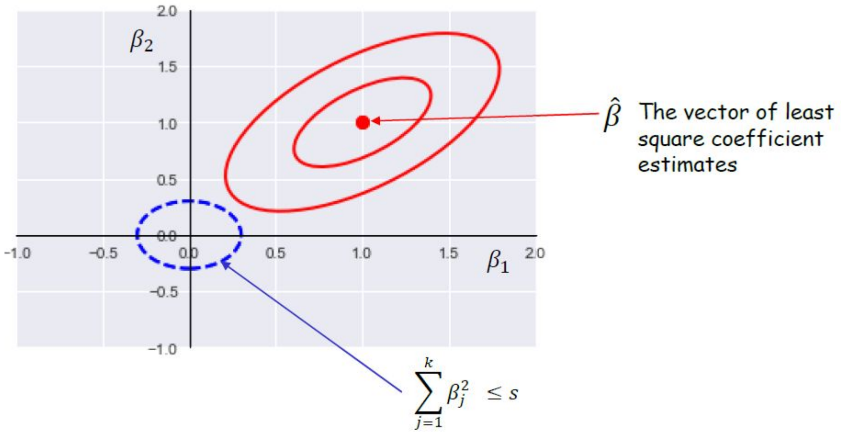
Lasso



Approach 2.1: Ridge Regression

$$J_{ridge} = \sum_{i=1}^{n} \left(y_i - \left(\beta_0 + \sum_{j=1}^{k} \beta_j x_{ij} \right) \right)^2 + \lambda \sum_{j=1}^{k} \beta_j{}^2$$





$$min_{\beta} \left\{ \sum_{i=1}^{n} \left(y_i - \left(\beta_0 + \sum_{j=1}^{k} \beta_j x_{ij} \right) \right)^2 \right\}, \text{ subject to } \sum_{j=1}^{k} \beta_j{}^2 \leq s$$

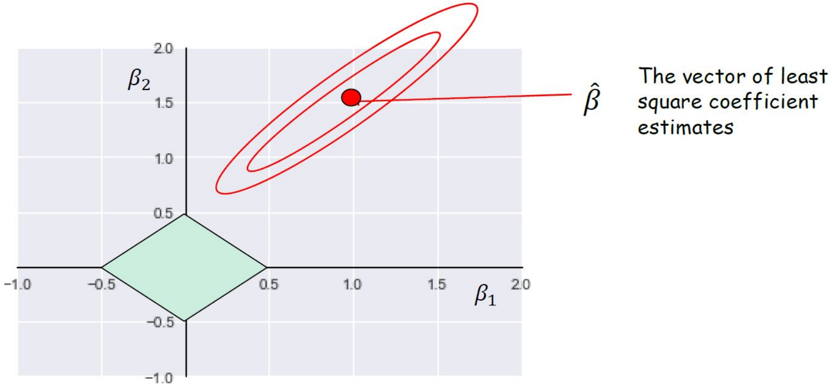
Approach 2.1: Ridge Regression

- In **un-regularized regression**, scaling inputs by constant C results in scaling of corresponding coefficient by 1/C; we can therefore say it's *scale-invariant*.
- In **regularized regression**, scaling inputs by constant can **dramatically affect the objective function**
- To ensure that features are scaled uniformly, we can standardize them (i.e. z-standardize)
- Alternatively, we can use different λ for different features (Tikhonov regularization)

Standardize $$\tilde{x} = \frac{x - \mu}{\sigma}$$

Approach 2.2: Lasso

$$J_{lasso} = \sum_{i=1}^{n}\left(y_i - \left(\beta_0 + \sum_{j=1}^{k}\beta_j x_{ij}\right)\right)^2 + \lambda \sum_{j=1}^{k}|\beta_j|$$



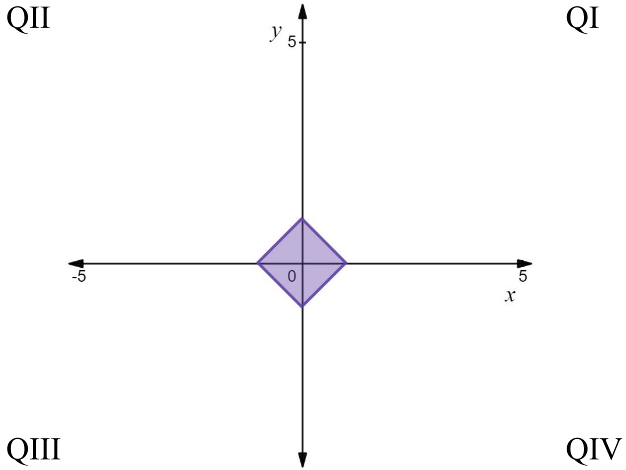
- **Ridge** generates a model including all features
- **Lasso** forces some coefficients to exactly 0

$$min_{\beta}\left\{\sum_{i=1}^{n}\left(y_i - \left(\beta_0 + \sum_{j=1}^{k}\beta_j x_{ij}\right)\right)^2\right\}, \text{ subject to } \sum_{j=1}^{k}|\beta_j| \le s$$

$$|x| = \begin{cases} x & \text{if } x \geq 0 \\ -x & \text{if } x < 0 \end{cases}$$

Aside: The graph of |x|+|y|=1

- In Quadrant I only, |x|=x, |y|=y → the line plotted is x+y=1
- In Quadrant II only, |x|=-x, |y|=y → the line plotted is -x+y=1
- In Quadrant III only, |x|=-x, |y|=-y → the line plotted is -x-y=1
- In Quadrant IV only, |x|=x, |y|=-y → the line plotted is x-y=1



Ridge vs. Lasso

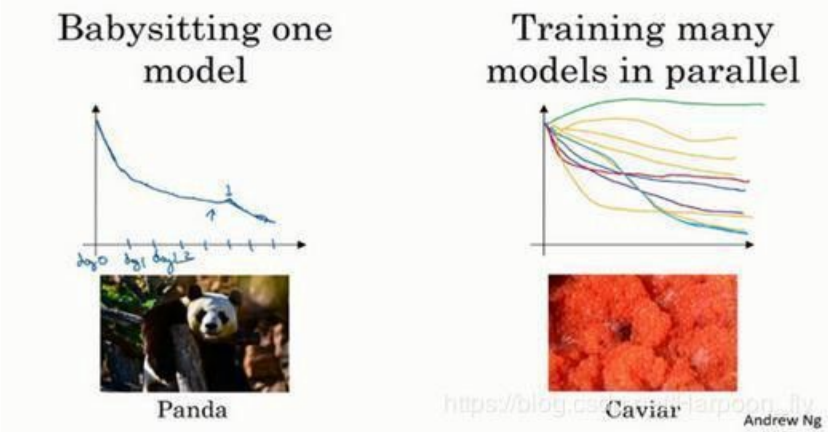
- Lasso performs feature selection, whereas Ridge doesn't set features to 0
- Lasso works better when few predictors, Ridge better when many predictors
- Lasso lends to better interpretability → selects variables instead of small coefficients

Can we combine these? **Yes** → **Elastic Net**

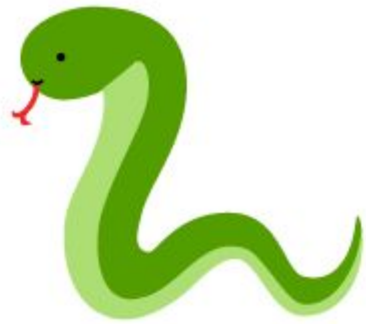
$$J_{elastic} = \sum_{i=1}^{n}\left(y_i - \left(\beta_0 + \sum_{j=1}^{k}\beta_j x_{ij}\right)\right)^2 + \lambda\left(\alpha\sum_{j=1}^{k}\left|\beta_j\right| + (1-\alpha)\sum_{j=1}^{k}\beta_j^{\,2}\right)$$

How to select λ?

- Cross-validation: Pick a range of values for λ, then get CV error for each



Let’sss try it in Python...



Summary

- Constructing Features
- Selecting Features
  - Best Subset selection
  - Stepwise Selection (Forward)
  - Stepwise Selection (Backward)
- Regularization
  - Ridge regression
  - Lasso
  - Elastic Net
  - Selecting λ (Babysitting vs. Caviar)