Introduction to Statistical Machine Learning CSC/DSCC 265/465

Lecture 9: Regularization

Cantay Caliskan

Notes and updates

Notes and updates

- We will be posting the grades for **PS2** soon.
  - Average is **91.75**.
- **PS4** posted, deadline is **Sunday, Feburary 20, 11:59 PM**
- **Rings of Power Trailer has been released!**
- **Quick advice:**
  - Please start on the homework before we help you!

Plan for today

- ***Multinomial Logistic Regression***
- ***Regularization***

Plan for today

- ***Multinomial Logistic Regression***
- ***Regularization***

Multinomial Logistic Regression

Multinomial Logistic Regression

- Definition: A classification method that generalizes logistic regression to ***multiclass*** problems.

$$y_i \in \{1, 2, \dots, K\}$$

- Note: One of the labels becomes the reference category!

- Different names exist in the literature:
  - *Multiclass logistic regression*
  - *Softmax regression*
  - *Multinomial logit*
  - *Multinomial logistic regression*
- Question: What can be the puzzles we are trying to solve?
- Question: What do we need to modify to expand the two-class logistic regression model?

Multinomial Logistic Regression

- Dependent variable: Nominal (=*categorical*).
  - Cannot be ordinal or nested or
  - **Assumption**: Categorical choices need to be independent

- Puzzles we may be interested in solving:
  - Which major will a student choose at college?
  - Which language is being typed in at the moment?
  - Which speaker is currently speaking?
  - Which party will the electorate choose in elections?
  - Which country is the best to invest in?

- Note: Assumption of Independence of Irrelevant Alternatives (IIA)

Independence or Irrelevant Alternatives (IIA)

- Idea: If the preference ratio between **outcome A** and **outcome B** is ***x***
  - The preference ratio should still stay ***x*** when a new **outcome C** is added to the dataset

- Example: Travelling to work 10 times
  - Prefer bus 6 out of 10 times
  - Prefer walking 4 out of 10 times
  - Ratio: 1.5
  - Let's add ***car*** as an option
    - Bus: 4 out of 10 times, walking: 2 out of 10 times, car: 4 out of 10 times
    - ***IIA*** does not hold here!

Multinomial Distribution

- Multinomial distribution:
  - A generalization of the ***binomial distribution***

  - Let’s say we have a response variable y_i that may take ***k*** many class labels such that $y_i \in \{1, 2, \ldots, K\}$
  - When ***k = 2*** and ***n > 1***, multinomial distribution becomes binomial distribution
  - If ***k*** is finite, and we have ***k*** many mutually exclusive outcomes with corresponding probabilities $p_1, p_2, \ldots, p_k$ and n independent trials:

$$\sum_{j=1}^{J} p_i = 1$$

Question: What is a random variable?

  - If the random variables X_i indicate the number of times outcome ***i*** is observed over ***n*** trials, $X = (X_1, X_2, \ldots, X_k)$ follows a multinomial distribution.

Multinomial Logit

- Idea: Run 'several' models to determine the outcome for ***c*** categories
  - Question: How many models do we need to run?
  - Answer: Run ***K-1*** many models
- Setup:
  - One outcome is chosen as the 'reference'
  - Other ***K-1*** outcomes are separately regressed against the reference
- Example: If the last class label is chosen as the reference, we get the following log-odds:

$$\ln \frac{\Pr(Y_i = 1)}{\Pr(Y_i = K)} = \boldsymbol{\beta}_1 \cdot \mathbf{X}_i$$

$$\ln \frac{\Pr(Y_i = 2)}{\Pr(Y_i = K)} = \boldsymbol{\beta}_2 \cdot \mathbf{X}_i$$

$$\cdots\cdots$$

$$\ln \frac{\Pr(Y_i = K-1)}{\Pr(Y_i = K)} = \boldsymbol{\beta}_{K-1} \cdot \mathbf{X}_i$$

Question: How are these values useful?

Calculating Probabilities

We need to convert them to probabilities…

- Idea: Multiply and exponentiate

$$\ln \frac{\Pr(Y_i = 1)}{\Pr(Y_i = K)} = \boldsymbol{\beta}_1 \cdot \mathbf{X}_i$$

$$\ln \frac{\Pr(Y_i = 2)}{\Pr(Y_i = K)} = \boldsymbol{\beta}_2 \cdot \mathbf{X}_i$$

$$\cdots\cdots$$

$$\ln \frac{\Pr(Y_i = K-1)}{\Pr(Y_i = K)} = \boldsymbol{\beta}_{K-1} \cdot \mathbf{X}_i$$

First, we get the log-odds ratios

$$\Pr(Y_i = 1) = \Pr(Y_i = K) e^{\boldsymbol{\beta}_1 \cdot \mathbf{X}_i}$$

$$\Pr(Y_i = 2) = \Pr(Y_i = K) e^{\boldsymbol{\beta}_2 \cdot \mathbf{X}_i}$$

$$\cdots\cdots$$

$$\Pr(Y_i = K-1) = \Pr(Y_i = K) e^{\boldsymbol{\beta}_{K-1} \cdot \mathbf{X}_i}$$

$$\Pr(Y_i = 1) = \frac{e^{\boldsymbol{\beta}_1 \cdot \mathbf{X}_i}}{1 + \sum_{k=1}^{K-1} e^{\boldsymbol{\beta}_k \cdot \mathbf{X}_i}}$$

$$\Pr(Y_i = 2) = \frac{e^{\boldsymbol{\beta}_2 \cdot \mathbf{X}_i}}{1 + \sum_{k=1}^{K-1} e^{\boldsymbol{\beta}_k \cdot \mathbf{X}_i}}$$

$$\cdots\cdots$$

$$\Pr(Y_i = K-1) = \frac{e^{\boldsymbol{\beta}_{K-1} \cdot \mathbf{X}_i}}{1 + \sum_{k=1}^{K-1} e^{\boldsymbol{\beta}_k \cdot \mathbf{X}_i}}$$

And, we get them at the end.

Softmax Function

- Reminder: We would like to label each feature vector with a class *k* from a set of *K* classes
- Softmax: A generalization of the Sigmoid function
- Idea: Take a vector **z** = [z_1, z_2, …, z_K] of *K* arbitrary values and map them to a probability distribution
  - Note: Each value in the vector **z** is between **0 and 1** and sum of all values in the vector is **1**.

$$softmax(z_j) = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \quad for\ j = 1, \ldots, K$$

Question: What is the difference between ***softmax*** and ***sigmoid***?

  - Also used in CNN, Discriminant Analysis, Naïve Bayes, Reinforcement Learning etc.

Plan for today

- *Multinomial Logistic Regression*
- ***Regularization***

Regularization

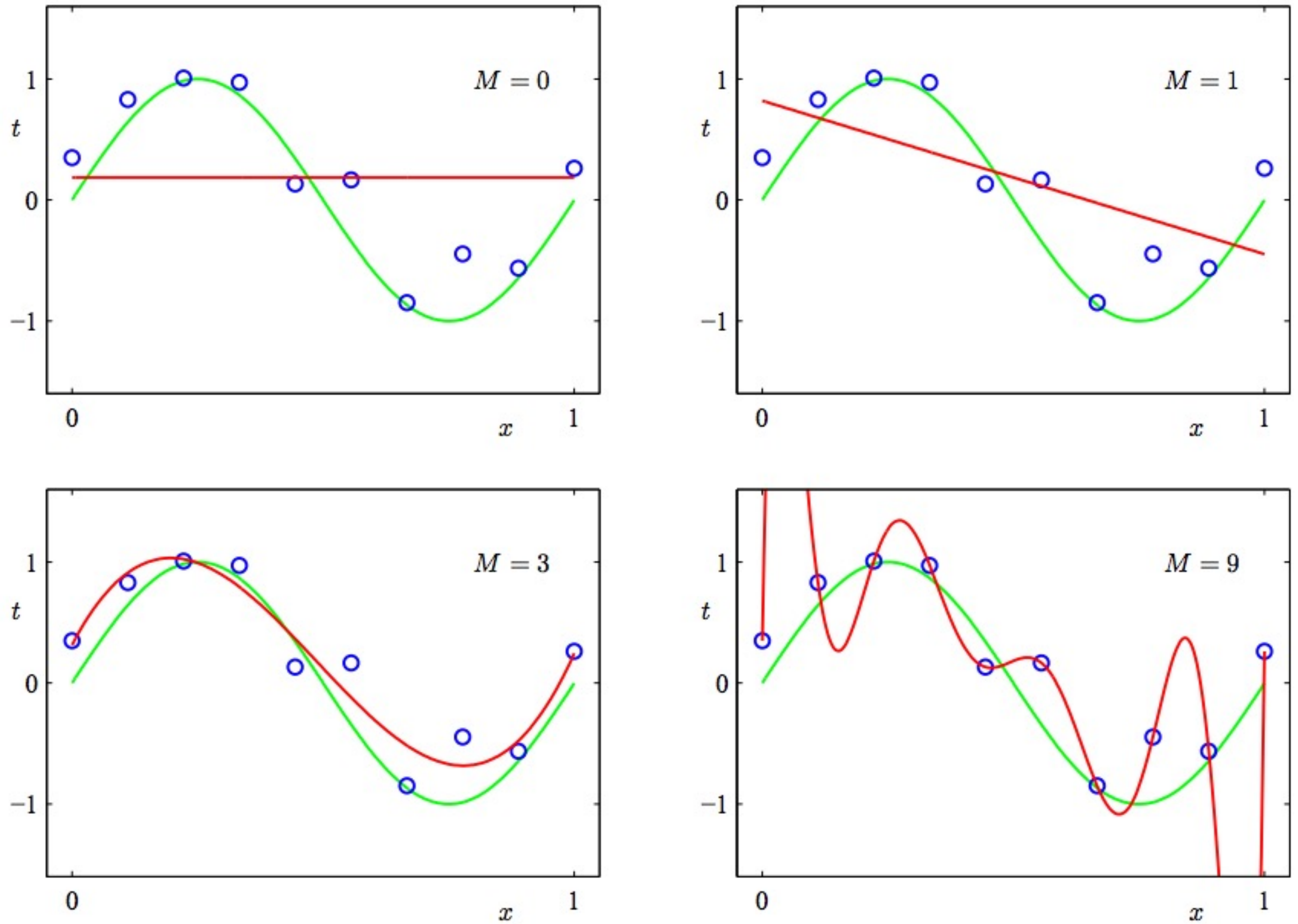
Reminder: Overfitting

- A "**big**" problem for all machine learning algorithms

- The model captures the noise in the training data instead of the underlying structure (that exists in the population)

- Can occur in many models, but especially in:
  - ***Decision Trees*** (e.g. when the tree is too deep)
  - ***KNN*** (e.g. when K is small)
  - ***Perceptron*** (e.g. when sample is not representative)
  - ***Linear Regression*** (e.g. when features are not linear)
  - ***Logistic Regression*** (e.g. when categorical features are unevenly distributed)

Overfitting: Example



- **Example**: Polynomial regression
- Top-left: *Degree 0*
- Top-right: Degree 1
- Bottom-left: Degree 3
- Bottom-right: Degree 9

Result:

- The higher the degree, the more capacity to "***overfit***"
- Error gets lower on training data, but it is higher on new data

Overfitting: More Formally

- Assume that the data is drawn from some fixed, unknown probability distribution

- Every hypothesis has a “true” error **e(h)**, which is the expected error when data is drawn from the distribution

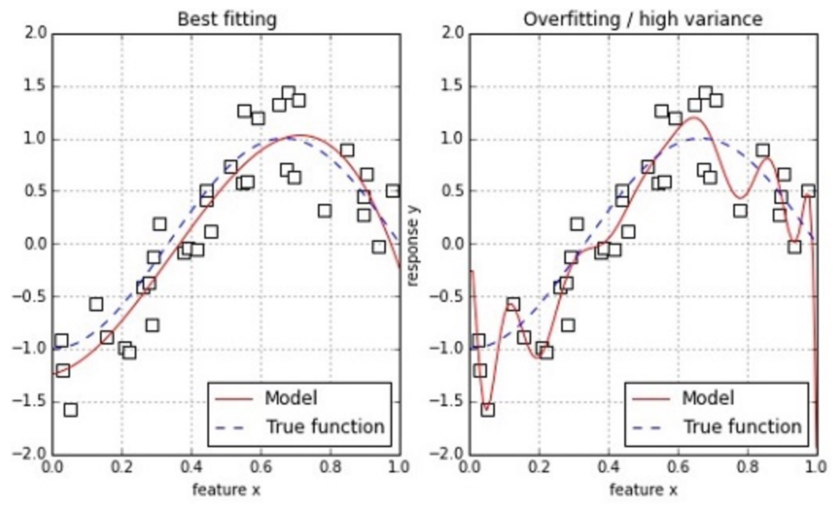
- Because we work on a “sample”, we measure the error on the training set: $\mathbf{e_{Training}(h)}$

- Suppose we compare two hypotheses on the training set, calculate the errors and show statistically significantly that $\mathbf{e_{Training}(h_1) < e_{Training}(h_2)}$, but <u>no</u> statistically significant proof that $\mathbf{e_{Test}(h_1) < e_{Test}(h_2)}$

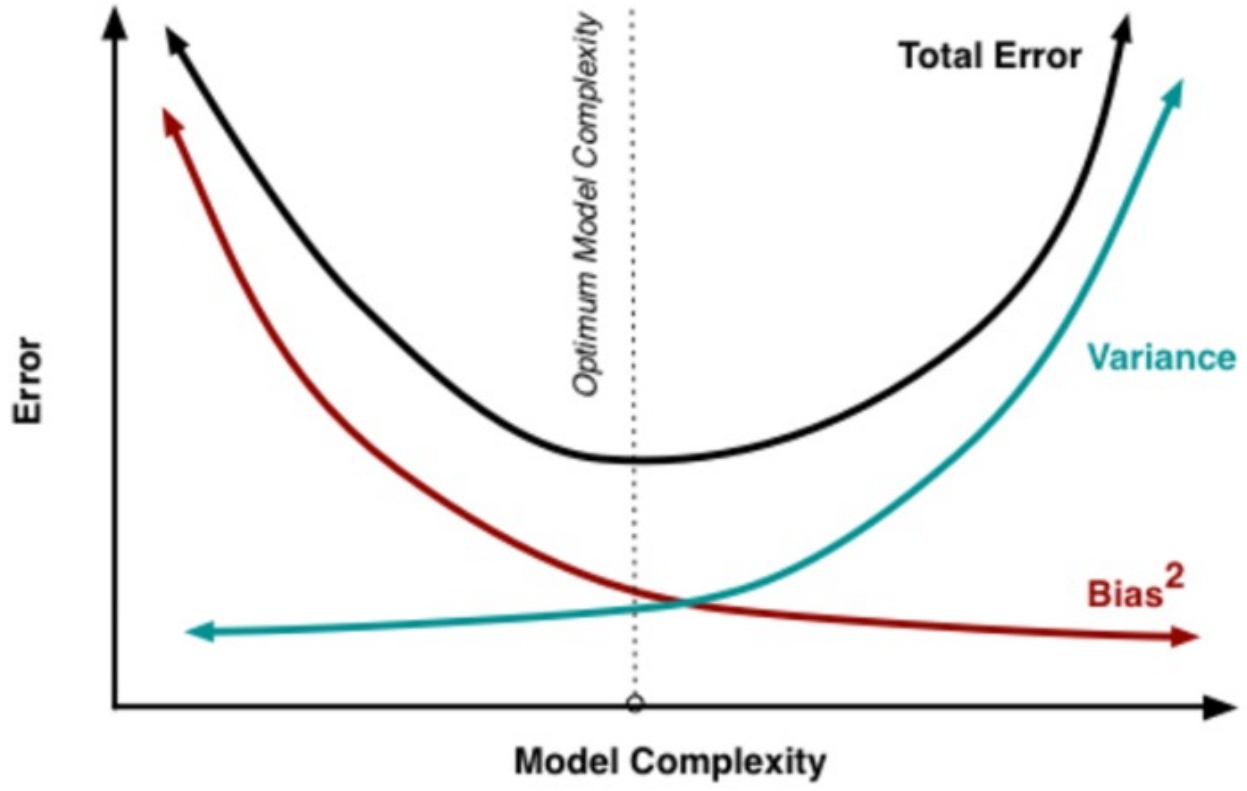
    - We are ”overfitting”! And we need to fix it!

What is model complexity?

- A complex mapping has many **terms** and **parameters**
- And, in some cases, complex models may have more parameters than the data has examples
- Problem: the algorithm starts memorizing everything in the data – not just the signals, but also the random noise, the errors, and all the slightly specific characteristics of the sample



Bias-Variance Tradeoff



- As the model complexity increases:
  - Variance increases
  - Bias decreases
  - There is an ”optimum spot” between bias and variance
  - We can reach that ”optimum level” using regularization

Motivation: Regularization

- (Hypothetical) Question:
    - When does a model 'usually' give its best performance?
    - Answer: When all the variables in a dataset are fully included
    - In other words: When the model is fully saturated

- Example: Stock Prices
    - Suppose we want to predict Tesla's stock price at time ***t+1***
    - ***What do we need?***
        - ***Ideally*** all prices of all stocks in all past periods
        - And all news from all newspapers
        - And all weather reports
        - And all different colors worn by the class in all past periods
    - **Do we really need all of this data?**

Motivation: Regularization

What does regularization do?

- Occam's Razor: Prefer the simplest hypothesis (parsimony)
- Bayesian perspective: Impose a prior distribution on model coefficients

- **What does it mean for a hypothesis to be simple (or parsimonious)?**

1) Small number of features (**model selection**) ← **Simpler** model

2) Small number of 'important' features (**shrinkage**) ← **Sparse** model

Objectives: Regularization

* Idea: Restrict the space of solutions f to an appropriately small hypothesis space

- Helps you to:
  - Engineer appropriate features for a new task
  - Use feature selection techniques to identify and remove irrelevant features
  - Identify when a model is overfitting
  - Add a regularizer to an existing objective in order to combat overfitting
  - Explain why we should not regularize the bias term
  - Convert linearly inseparable dataset to a linearly separable dataset in higher dimensions
  - Describe feature engineering in common application areas

Technical Background

Regularization: Simple Terms

- Lesson: Complicated hypotheses lead to overfitting
- Idea: Change the error function to penalize hypothesis complexity

$$Error(\mathbf{w}) = Error_{D}(\mathbf{w}) + \lambda * Error_{penalty}(\mathbf{w})$$

- This is called ***regularization*** in machine learning and ***shrinkage*** in statistics

- **λ** is called regularization coefficient and controls how much we value **fitting the sample data well** vs. a simple hypothesis that **generalizes well**

Regularization

- Goal: Minimize a quadratic function:

$$\min_{f \in \mathcal{H}} H[f] = \frac{1}{\ell} \sum_{i=1}^{\ell} V(y_i, f(\mathbf{x}_i)) + \lambda \|f\|_K^2$$

- Where ***V( )*** is a loss function, and $\|f\|_K^2$ is a norm defined by function ***K***, ***l*** is the number of training examples, and ***λ*** is the regularization parameter.

- The solution to the equation above is given by $f(\mathbf{x}) = \sum_{i=1}^{\ell} c_i K(\mathbf{x}, \mathbf{x}_i)$

  - Summary: Find a regularization parameter that influences each data point in a 'different' and 'instant' way so that the overall cost is minimized.

Source: Mukherjee, Rifkin, Poggio

UNIVERSITY *of* ROCHESTER

Regularization vs. Generalization

- Question: How does Bayesian statistics come into play?
- Answer: We can only measure the performance of the regularizer on the training sample, not on the population data

$$f(\mathbf{x}) = \sum_{i=1}^{\ell} c_i K(\mathbf{x}, \mathbf{x}_i) \quad \text{is dependent on} \quad I_{exp}[f] \equiv \int V(y, f(\mathbf{x}))dP(\mathbf{x}, y).$$

- where *I( )* represents the generalization error, *V( )* is the cost function, and P(***x***, y) is the probability distribution
- But: Do we know *P(**x**, y)*?
  - Answer: No. We can only the measure the empirical deviation using a sample:

$$I_{emp}[f] \equiv \frac{1}{\ell}\sum_{i=1}^{\ell} V(y_i, f(\mathbf{x}_i))$$

Source: Mukherjee, Rifkin, Poggio

Loss Functions with Regularization Parameter

- General formula:

$$\min_{f \in \mathcal{H}} H[f] = \frac{1}{\ell} \sum_{i=1}^{\ell} V(y_i, f(\mathbf{x}_i)) + \lambda \|f\|_K^2$$

- Loss function V( ) is selected appropriately for context

$$V(y, f(\mathbf{x})) = (y - f(\mathbf{x}))^2$$

Linear regression

$$V(y, f(\mathbf{x})) = |y - f(\mathbf{x})|_\epsilon \qquad |y - f(\mathbf{x})|_\epsilon = \max(0, |y - f(\mathbf{x})| - \epsilon)$$

Support Vector Machine

$$V(y, f(\mathbf{x})) = \Theta(-y f(\mathbf{x}))$$

Classification with misclassification loss

$$V(y, f(\mathbf{x})) = (1 - y f(\mathbf{x}))_+$$

SVM classification with a soft margin

$$V(y, f(\mathbf{x})) = (f(\mathbf{x}) - y)^2$$

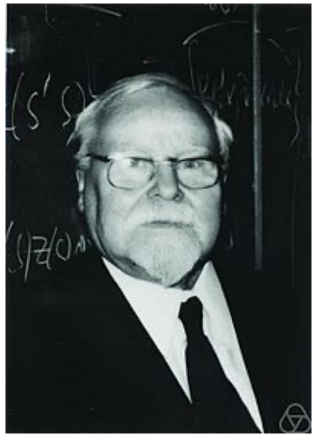
Quadratic classification loss

Source: Mukherjee, Rifkin, Poggio

Tikhonov Regularization

- 'Father' of L2-regularization
  - L2-regularization = Tikhonov regularization
- Idea: Add *L2-norm* of the vector of weights (w) to the loss function in order to prefer solutions with smaller norms
  - Also known as ***Ridge regression***
- Tikhonov Regularization function: $$\min_w \sum_{i=1}^{n} V(\hat{x}_i \cdot w, \hat{y}_i) + \lambda \|w\|_2^2$$
- Generalized form: $$\min_f \sum_{i=1}^{n} V(f(\hat{x}_i), \hat{y}_i) + \lambda \|f\|_{\mathcal{H}}^2$$

H stands for Reproducing Kernel Hilbert space


Andrey Nikolayevich Tikhonov

Question: What are the advantages of adding Tikhonov (L2) Regularization term?

Tikhonov-regularized least squares

- Regularization goal: $\min_w \frac{1}{n}(\hat{X}w - Y)^T(\hat{X}w - Y) + \lambda\|w\|_2^2$
- Gradient: $\nabla_w = \frac{2}{n}\hat{X}^T(\hat{X}w - Y) + 2\lambda w$
- Optimal coefficients first-order condition: $0 = \hat{X}^T(\hat{X}w - Y) + n\lambda w$
- Optimal weights: $w = (\hat{X}^T\hat{X} + \lambda n I)^{-1}(\hat{X}^T Y)$

Reminder: Linear Model

- The equation for linear regression:

$$Y = Intercept + Slope*X + Error$$

- And the expected value for *Error* is zero.

- Goal of regularization:
  - Reduce variance at the cost of introducing some bias, so that:

$$Y = Intercept + Slope*X + Error + Bias$$

Reminder: Classification

- The goal of every classification algorithm:

$$min(\mathbf{J}(f(x), y)$$

We are minimizing the 'distances' between predictions and ground truth…

- Where $\boldsymbol{J}$ is the cost function

- Goal of regularization:
  - Add a regularization term to impose a penalty on the complexity of f

$$min(\mathbf{J}(f(x), y) + \lambda R(f)$$

Regularized Least Squares

Regularization for Linear Models

1. Ridge Regression
2. Lasso Regression
3. ElasticNet Regression

Ridge Regression

- <u>Reminder</u>: Regularization is adding another term to the loss function
- Loss function for (not regularized) linear regression:

  $$\textit{Loss function} = \sum(\hat{Y}_i - Y_i)^2$$

- What if we add a regularization term:

$$\mathbf{J(W)} = \frac{1}{2N}\sum_{i=1}^{N}((W_0 + W_1X_1^{(i)} + \ldots + W_PX_P^{(i)}) - Y_i)^2 + \frac{\lambda}{2N}\sum_{j=1}^{P}W_j^2$$

- The regularization term sums over squared β values and multiplies it by another parameter λ
  - This punishes the loss function for high β coefficients
  - This is also called L2 Regularization

- *As λ -> 0, Ridge coefficient becomes similar to OLS coefficient*
- *As λ -> ∞, Ridge coefficient becomes dominant.*

UNIVERSITY of ROCHESTER

L2 Regularization

- Positive λ will cause the magnitude of the weights to be smaller than in the usual linear solution

Pros and Cons of L2 Regularization

- If λ is chosen well, regularization helps to avoid overfitting
- Choosing λ may be hard…
- If there are irrelevant features in the input (i.e. features that do not affect the output), L_2 will give them small, but non-zero weights
- Ideally, irrelevant input should have weights exactly equal to 0

Lasso Regression

- Also called ***Least Absolute Shrinkage Selection Operator*** (***LASSO***)

- Adds a penalty for non-zero coefficients
  - But, unlike Ridge Regression which penalizes the sum of squared coefficients, Lasso penalizes the sum of their absolute values:

$$\mathbf{J(W)} = \frac{1}{2N}\sum_{i=1}^{N}((W_0 + W_1X_1^{(i)} + \ldots + W_PX_P^{(i)}) - Y_i)^2 + \frac{\lambda}{2N}\sum_{j=1}^{P}|W_j|$$

  - This is called L1 penalty or L1 regularization
  - Result: For high values of of λ, many coefficients are exactly zero, which is never the case in Ridge Regression

Pros and Cons of L1 Regularization

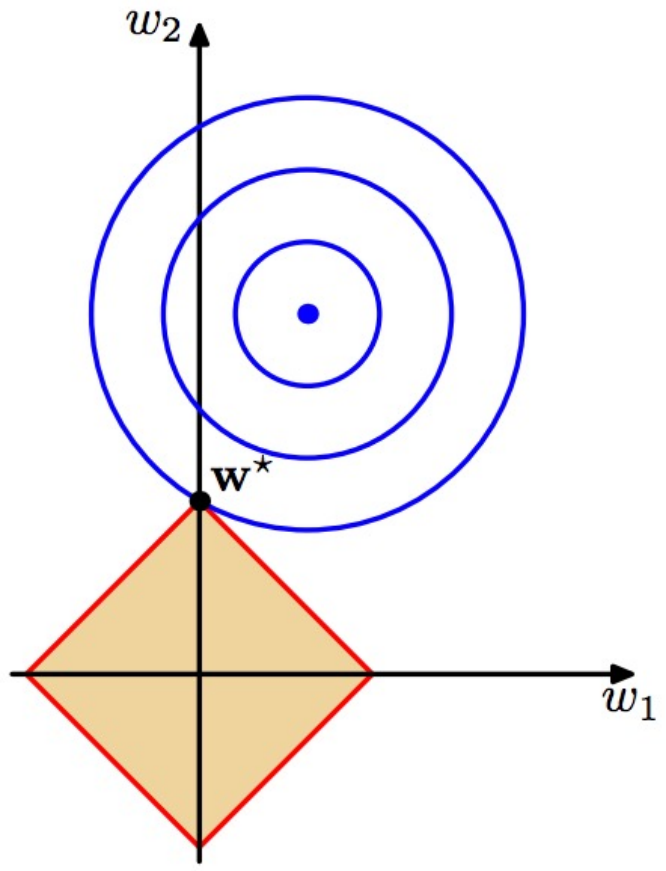
- If there are irrelevant input features, ***Lasso*** is likely to make their weights 0, while ***L2*** is likely to just make all weights small

- ***Lasso*** is biased towards providing sparse solutions in general

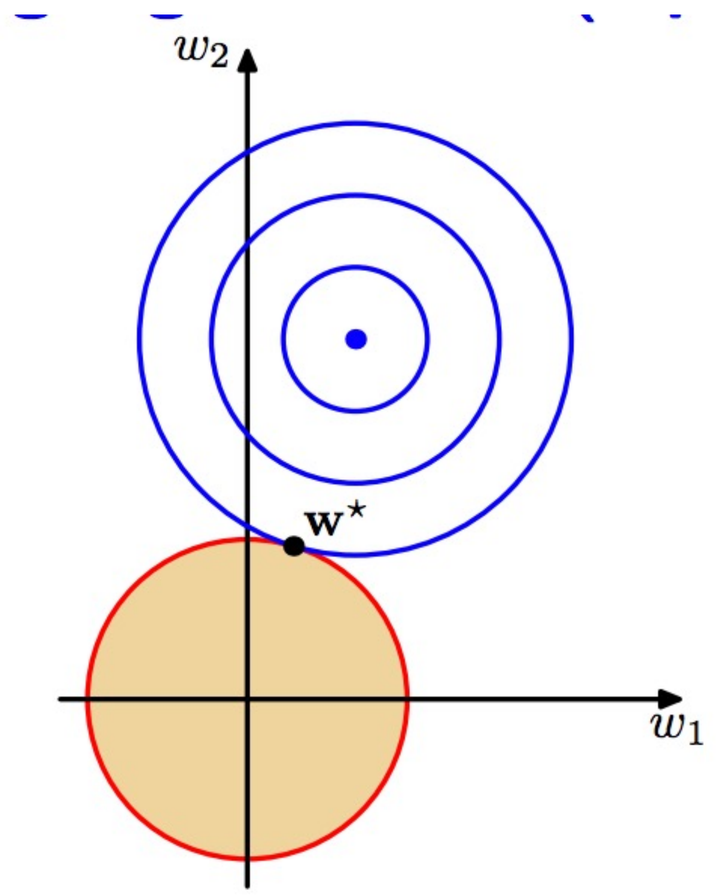
- ***Lasso*** optimization is computationally more expensive than ***L2***

- But, ***L1*** regularization is also very popular…

Visualizing Regularization

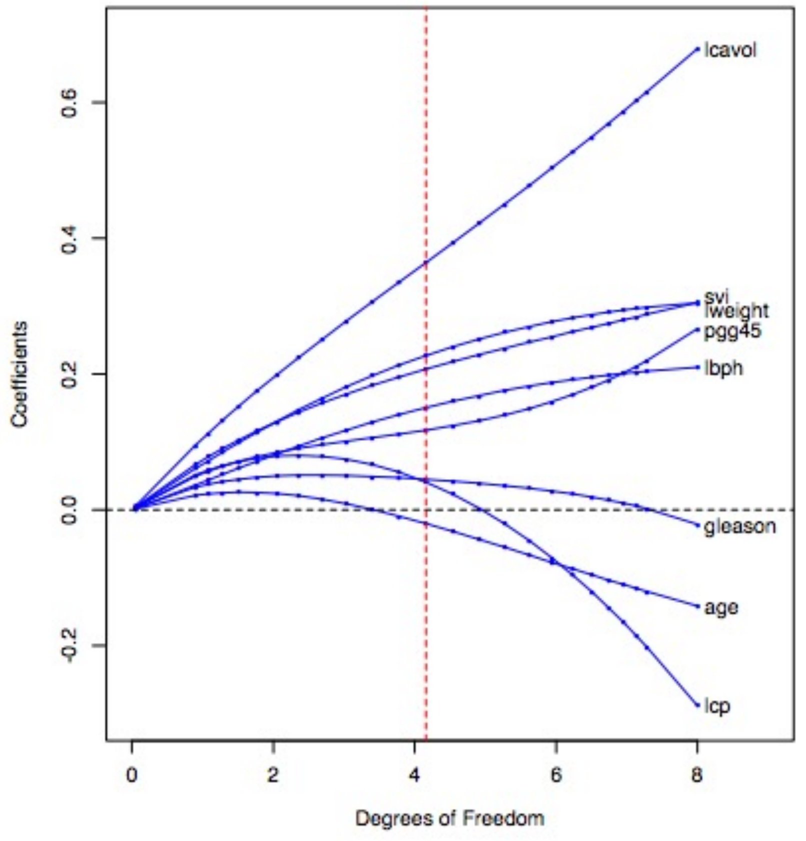


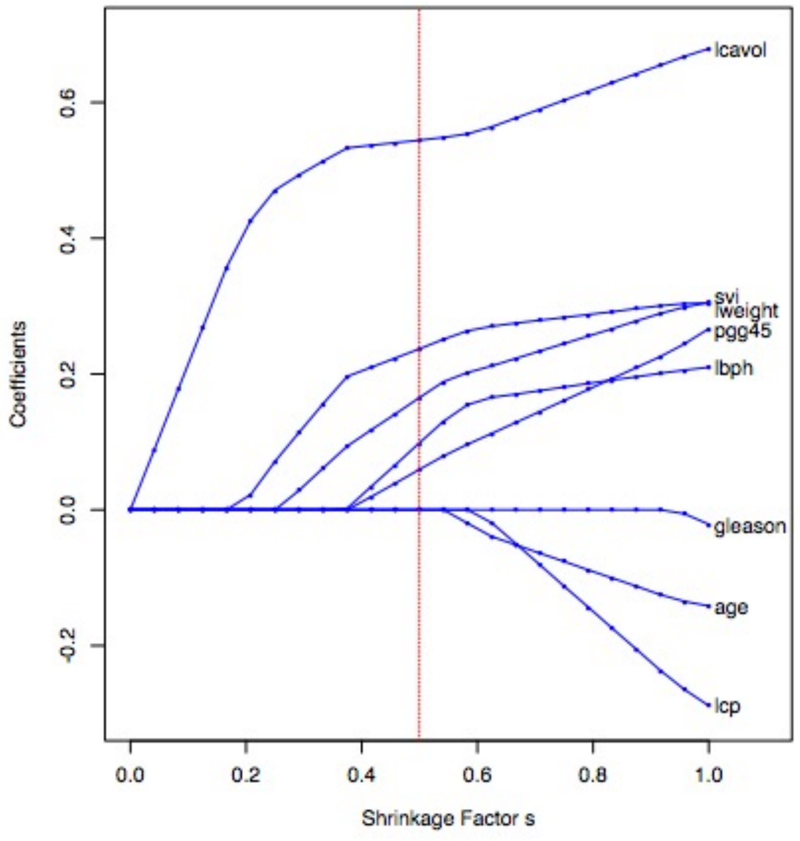
L1 Regularization



L2 Regularization

L1 vs. L2 effect





<u>Question</u>: Which one is L2 regularization?

L2 regularization is on the left!

So, which λ value should we pick?

- Two potential approaches:

1) ***Statistics approach***: Pick a value such that some information criterion, such as **AIC** or **BIC** is the smallest (highest goodness of fit):
   - **AIC**: Akaike Information Criterion
   - **BIC**: Bayesian Information Criterion

2) ***Machine learning approach***: Perform cross-validation and select the value λ that minimizes the cross-validated sum of squared residuals

Three additional methods to pick the best λ

- **Manual tuning**: Finding the right combination of settings for regularization can be done via manual tuning (if you have the expertise)

- **Grid search:** Find two sets of parameters that are believed to contain the best set of parameters and converge toward the best set

Ridge vs. Lasso

- Often neither one is overall better

- ***Lasso*** can set some coefficients to zero
  - Thus, Lasso performs variable/feature selection
  - Ridge does not do that

- Both methods allow to use correlated predictors, but they solve multicollinearity issues differently:
  - ***Ridge*** regression: Coefficients of correlated predictors are similar
  - ***Lasso*** regression: One of the correlated predictors has a larger coefficient, while the rest are (nearly) zeroed

Ridge vs. Lasso

- ***Lasso*** tends to do well if there are a small number of significant parameters and the others are close to zero
  - When only a few predictors actually influence the response

- ***Ridge*** works well if there are many large parameters of about the same value
  - When most predictors impact the response

- In practice, we don't know the parameter values, so it is really hard to decide between two:
  - Answer: *ElasticNet* Regression

ElasticNet Regression

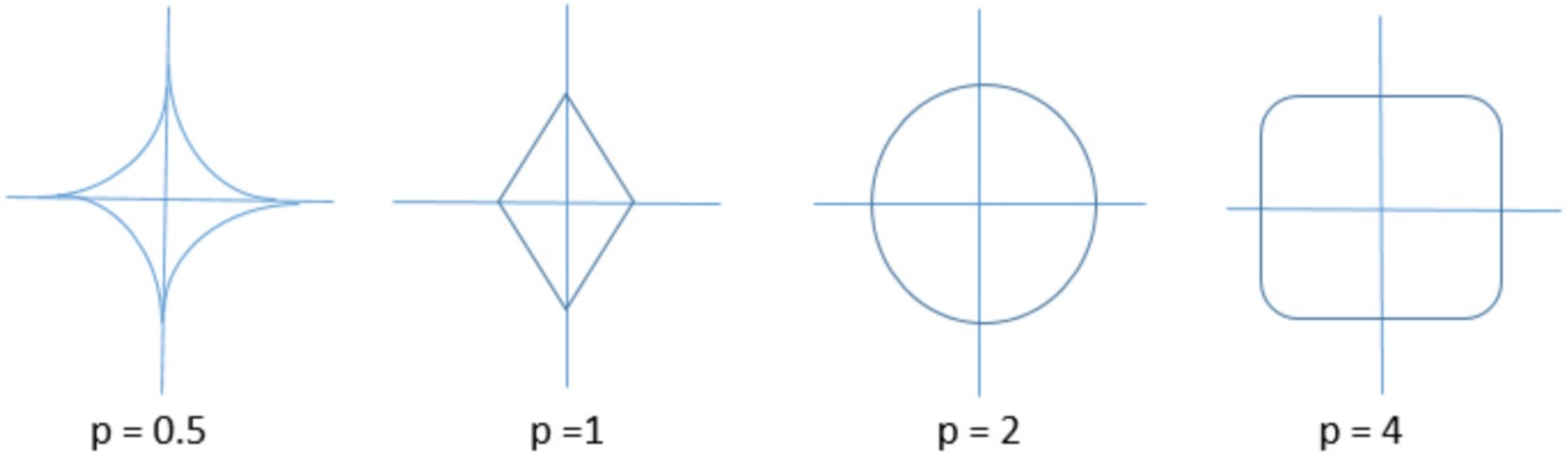
- First emerged as a result of critique on ***Lasso***
  - The variable selection for Lasso can be too dependent on data and thus unstable
  - Solution: Combine the penalties of Ridge and Lasso to get the best of both worlds.
  - The loss function for ElasticNet:

$$\mathbf{J}(\mathbf{W}) = \frac{1}{2N}\sum_{i=1}^{N}((W_0 + W_1X_1^{(i)} + \ldots + W_PX_P^{(i)}) - Y_i)^2 + \frac{\lambda_1}{2N}\sum_{j=1}^{P}|W_j| + \frac{\lambda_2}{2N}\sum_{j=1}^{P}W_j^2$$

  - There are two parameters to tune: λ_1 and λ_2

L^p Regularizers

- No need to use only L1 and L2, or L1+L2
  - ***L^p regularization*** is also possible.



Question: What happens with the parameters here?

Hint: Axes represent your coefficient values

‘Other’ types of regularization

- ***Early stopping***
  - You can avoid overfitting if you are using an iterative optimization method, such as gradient descent
  - Conditions for early stopping: Amount of change in updates (i), number of iterations (ii)
  - Question: What are some disadvantages here?
  - A solution to some disadvantages: Validation-based early stopping

- ***Principal component regression (PCR)***
  - Instead of using the independent variables directly, the principal components of the explanatory variables are used as regressors