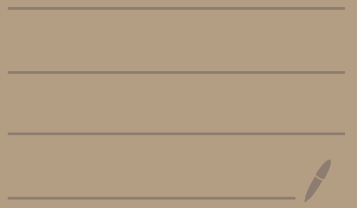
# STAT 457



# Week 1: K Nearest Neighbors and Bias/Variance tradeoff

## Core Regression Model

- $Y = f^*(X) + \varepsilon$
  - $X$ is feature vector
  - $Y$ is numeric response
  - $f^*$ some relationship
  - $\varepsilon$ is random noise

## Identity: $E[Y \mid X = x] = f^*(x)$

The regression function is the conditional mean.

## Training Data

- builds estimate for $f^*$
- Used to predict new inputs

Training → Estimator → Prediction at $x_0$

## K Nearest Neighbours (KNN)

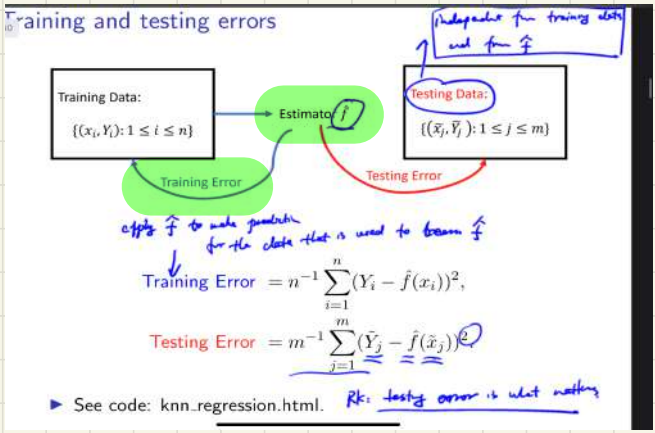
- $N_K(x_0)$ - The K number of closest training points at $x_0$
  - Euclidean, assume no ties
  - Prediction based on average nearby points

## Training error vs Testing Error

- The training error is evaluated on training data
  - Always Optimistic
- Testing error is far more accurate
- For KNN: Training error goes down as K goes down
  - If K=1 (1 group of cluster), training error =0
  - Testing error U Shaped

Small K → overfitting → high test error

Large K → underfitting → high test error




If K is chosen from training error. The training data is used twice. once to fit the model, then once to evaluate.

## Cross-Validation (Breaking loop)

- Split training data into m folds
  - For each fold train on m-1
  - Test on remaining fold
  - Average error

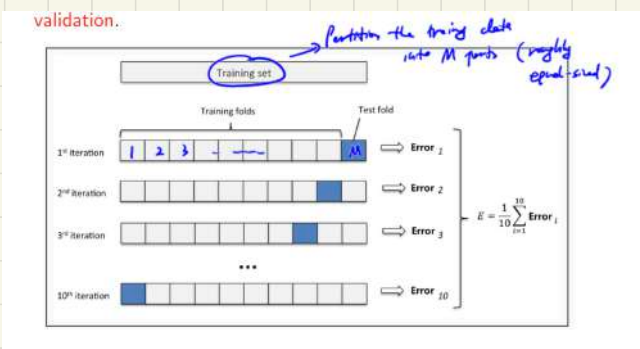
### Important Identity

$$E[Y \mid X = x] = f^*(x)$$

$$Y = f^*(x) + \varepsilon$$

Expectation is linear

True regression function is average output at a given input.




Average out errors for m folds

Once you calculate all possible $\alpha$, cross validation candidates

Then choose lowest average CV error.

Then retrain all data.

In Sample Risk (Errin): Error on a special testing set that has same input but new $Y_i$.

The gap is the difference between In Sample risk and training error.

Total Risk: gap + training error

## Quantifying Optimism

- $E[Err_{in}] = E[Err_{train}] + \frac{2}{n} \sum Cov(Y_i, Y_i)$
- Covariance is how much models prediction changes based on training target ($Y_i$'s)
- High covariance means noise is followed very closely. (High optimism)

## Effective number of parameters (w)

- This is to minimize the total risk

$$w = \frac{1}{\sigma^2} \sum_{i=1}^{n} Cov(Y_i, Y_i)$$

- large w -> complex model, gap is large
- small w -> model is simple, gap is small

## Bias Variance Tradeoff

- 3 sources: irreducible error ($\sigma^2$)
  - ↳ natural noise in dataset
  - → Bias: too simple does not capture patterns
  - → variance: too sensitive,

## Emperical Risk Minimization

- low training error does not always mean good
- Simple model F (high training error)
- Rich/complex models (overfitting)

# Week 2: Linear Algebra Review

## Trace

- For square matrix, sum of its diagonal entries

$$Tr(AB) = Tr(BA)$$

## Linear Representation

- Vectors can be linear combinations of each other

$$v_1 = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, \quad v_2 = \begin{bmatrix} 4 \\ 1 \\ 5 \end{bmatrix}, \quad v_3 = \begin{bmatrix} 2 \\ -3 \\ -1 \end{bmatrix}$$

$$v_3 = -2v_1 + v_2$$

- Vector is lincarly indep, if it can't be made up using other vectors.

- Rank is number of lin indep vectors in matrix

$$\text{Col Rank}(A) = \text{Row Rank}(A) = \text{Rank}(A)$$

## Sub spaces

- two vectors are closed under addition and multiplication
- $x + y \in V$, $\alpha x \in V$

- dim of subspace V is largest subset of lin indep vectors

- **Span:** Smallest subspace contans every possible linear combination of those vectors.

- **Basis:** span is equal to subspace. Minimum building blocks to create every vector in that subspace.

# Orthogonality

- two vectors are orthogonal if their inner product $x^T y = 0$. A vector is orthogonal to a subspace where its perpendicular to every vector inside the subspace.

Orthonormal matrix: every vector perpendicular to every other vector and unit magnitude of one. (preserves angle and length)

Property: $U^T U = U U^T = I$

- For 2 by 2 orthonormal matrices, these are ones that either rotate or reflect
- Preserves length $\|Ux\|_2 = \|x\|_2$
- Preserves angle $\langle Ux, Uy \rangle = \langle x, y \rangle$

# SVD

- Breaking down Matrix into $X = U \Sigma V^T$
- U has orthonormal columns, V is an orthonormal matrix, $\Sigma$ is diagonal matrix.

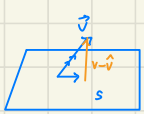
Spectral Decomp: Symmetric Square matrix, $Ax = \pi x$ (A times eigenvector)

Positive Def: $x^T A x$ is always $> 0$

Posibive Semi-Def: $x^T A x$ is always $\geq 0$.

Eigen values also determine definiteness

# Projections




- Vector V, Subspace S

  $v - \hat{v} \perp S$

The residual is perpendicular to subspace, because removes all components of S. Minimizes distance

$$\|v-u\|^2 = \|v-v\|^2 + \|\hat{v} - u\|^2$$

$$\|v-u\| \geq \|v-\hat{v}\|$$

Iterated projection: Drop v1 into $S_2$, then drop result in $S_1$

# Week 2/3 : Optimization

- ERM (Emperical Risk Minimization)
  - ↳ finds prediction rule $f = \{f_\theta : \theta \in \Theta\}$ that minimizes average loss.
- gradient: direction and steepness, vector of first derivatives
- Hessian H: tells you curvature

## Taylor Expansions

- approximate complex functions
- Fastest descent $(-\nabla F)$
- $f(\beta + \delta) \approx f(\beta) + \nabla f(\beta)^T \delta$
- $f(\beta+\delta) \approx f(\beta) + \nabla f(\beta)^T \delta + \frac{1}{2}\delta^T H f(\beta) \delta$
- $g(\delta) = f(\beta) + \nabla f(\beta)^T \delta$

$\nabla f(\beta)^T \delta$ subject to $\|\delta\| = 1$

$\delta^* = -\frac{\nabla f(\beta)}{\|\nabla f(\beta)\|}$

$\delta = -\nabla F(\beta)$

$\delta \propto \nabla f(\beta)$

is proportional soon

## Local minima

$f(\beta) \geq f(\beta_0)$ if $\|\beta - \beta_0\| < \delta$

If you pick any $\beta$ close enough to $\beta_0$. the function value at $\beta$ is not smaller than $\beta_0$

local minima $\neq$ global minima

(maybe)

Necessary condition: $\nabla f(\beta_0) = 0$, slope is 0, cant have immediate decrease

Sufficient condition: $\nabla f(\beta_0) = 0$ and $H f(\beta_0)$ is positive definite, $\beta_0$ is local min

(guarantee)

## Gradient Descent Algorithms

- take small steps down till flat hill
- First Order Algorithm: $x_{n+1} = x_n - \alpha_n \nabla f(x_n)$
  - next guess
  - current guess
  - step size
  - grad at current point
- Stop when $\|\nabla f(x_n)\| \leq \epsilon$

## Newton Raphson

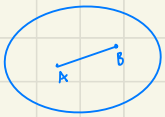
- uses second order
- $x_{n+1} = x_n - H f(x_n)^{-1} \nabla f(x_n)$ ↑ Hessian curvature
- uses slope and curvature, faster but more expensive

# Convex Sets

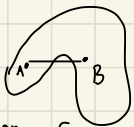
- A set $A \subset \mathbb{R}^d$ is convex if:
  for any two points $x, y$ in the set, every point on the straight line between them is also in the set

$\alpha x + (1-\alpha) y \in A$, for $\alpha \in (0,1)$

$A_1 \cap A_2$ is convex if they are individually convex.



Convex



non Convex

- Convex functions: $f(\alpha x + (1-\alpha) y) \leq \alpha f(x) + (1-\alpha) f(y)$
  - value in middle
  - straight line interpolation
- Convex means: curve stays under line
- Examples: $f(x) = |x|$
  $f(x) = \max(x, 0)$
- Convex set $\{x : f(x) \leq c\}$

## Proving Convexity

1) Hessian is PSD
2) First order tangent line condition $f(y) \geq f(x) + \nabla f(x)^T (y - x)$

if $\nabla f(x_0) = 0$, $x_0$ is global min

# Week 3: Linear Regression

- linear regression $f_\beta(x) = \beta_0 + \beta_1 x_1 + \dots \beta_p x_p$

  $\beta = \{\beta_0, \beta_1, \dots \beta_p\}$ unknown parameters

  - model predicts number using weighted sum

- Loss function (RSS): $RSS(\beta) = \sum^{n} (y_i - \beta_0 - \beta_1 x_{i1} - \dots \beta_p x_{ip})^2$

- Matrix formulation: $X = \begin{bmatrix} 1 & x_1^T \\ 1 & x_2^T \\ \vdots & \vdots \\ 1 & x_N^T \end{bmatrix}$ $(n \times (p+1))$

- response vector: $Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$, Prediction: $X\beta$ = all predicted values

  $RSS(\beta) = \|y - X\beta\|^2$

- RSS is a quadratic function of $\beta$.

  ↳ gradient $\nabla RSS(\beta) = -2X^T y + 2X^T X\beta$

  $H\,RSS(\beta) = 2X^T X$

- Solution is unique when $\text{rank}(x) = p+1$, colums of $x$ are lin indep

- Projection matrix: $P = X(X^T X)^{-1} X^T$

  $\hat{Y} = Py$

## Subset Selection

- Choosing subset of features instead of all of them in lin reg
- reducess error, avoid overfitting

2 ways

1. Search Strategy (try diff subsets of predictions)
2. Scoring rule (find best subset)

- Score = Fit + Penalty

  $RSS(\hat{\beta}_M)$ = residual sum of squares for $m$

- Common Choices include: Mallows $C_p$, AIC, BIC

# KNN

7/9

- non parametric, supervised, no functional form
- K is only key param
- distance based voting/averaging
- No training phase (stores data)

## cross fold

- split into m folds
- train on m-1, test on leftover
- average out errors, choose lowest K

L.R - has effective number of params - p+1

KNN - effective number of params depends on K

$E.P = \frac{n}{K}$

↳ how flexible model is

small K - low bias, high variance (wiggly)

Big K - high bias, low variance (smooth)

Bias = error from simplifying assumptions

Var = sesitivity to training data.

$Ax$ = matrix · Scalar (weight)

$x^T y = 0$ orthogonal

$U^T U = UU^T = I$ → Orthonormal preserves angle/length

PSD ⟹ convex function

P → projection matrix, closest point to subspace

$a^T x + b$ → affine → convex

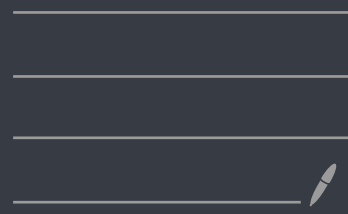
---

Gradient Desc $x_{k+1} = x_k + \alpha \nabla f(x_k)$

Newton Raphson $x_{k+1} = x_k - [\nabla^2 f(x_k)]^{-1} \nabla f(x_k)$

If convex, global min = local min

# Midterm 2



# Quiz Content (7 bullets)

## Linear Regression

- Supervised learning predicts numerical response

model formulation: $Y = X\beta + \varepsilon$

$\varepsilon$ is noise

$Y \in \mathbb{R}^n$: response vector

$X \in \mathbb{R}^{n \times (p+1)}$: design matrix

$\beta \in \mathbb{R}^{p+1}$: coefficients

L.R: Objective:

- minimize: $RSS(\beta) = \|Y - X\beta\|^2$
- Choose Best $\beta$ parameters

Trying to choose Best $\beta$ coefficients so that we minimize RSS

Proof for solving:

$$RSS(\beta) = \|Y - \beta X\|^2$$
$$-2x^T(Y - \beta x)$$
$$x^T(Y - \beta x) = 0$$
$$x^TY - \beta x x^T = 0$$
$$\beta = x^TY\,(xx^T)$$

## Ordinary Least Squares Solution

$$\nabla RSS(\beta) = -2X\,(Y - X\beta)$$
$$X^T(Y - X\beta) = 0$$
$$X^TY - XX^T\beta =$$
$$X^TY = XX^T\beta$$
$$\beta = X^TY\,(XX^T)^{-1}$$

L.R: $\|Y - X\beta\|^2$

R.R: $\|Y - X\beta\|^2 + |\pi^2|\beta$

L.R: $\|Y - X\beta\|^2 + |\pi|\beta$

## Projection Matrix

$y = X\beta$

Sub back in: $y = X(X^TX)^{-1}X^Ty$

$H = X(X^TX)^{-1}X^Ty$

$y = Hy$

Projection matrix projects $y$ onto column space of $x$

Linear regression $\longrightarrow$ Ridge Regression

$y = X\beta + \varepsilon$ $\longrightarrow$ $y = \beta x + \|x - \beta\|^2$

$\beta = (X^TX + \pi I)X^Ty$

as $\pi$ increases, less flexible, less effective param

Harsher penalty, lower variance

Shrinks by $\frac{\delta_j}{\delta_j + \pi}$

For $\pi = 0$, $df = p$

Minimizing RSS:

$\min \|Y - X\beta\|^2$

OLS: $\beta = X^T(X^TX)^{-1}Y$ → if $p > n$, not unique. Should Shrinkage

Projection matrix:

$H = Hy$

where $H = X^TX\,(X^TX)^{-1}$

Projection matrix is Symmetric and independent

$H^2 = H$, $H^1 = H$

$\text{trace}(H) = p$

number of Params

high variance, low bias

# Ridge Regression

OLS: $\min_\beta \|y - X\beta\|^2$

Ridge Adds: L2 Penalty

$\hat{\beta}_\lambda = \arg\min \left( \|y - X\beta\|^2 + \lambda \|\beta\|^2 \right)$

$\lambda \geq 0$

$\|\beta\|^2 = \beta^T \beta$

Penalize large coefficients to reduce variance

Shrinks them toward zero

given $\beta^{OLS} = (3, 1, 5)$ $\lambda = 2$

Show variable selection

$\beta_{i1} = +1\,(+1) = 1$

$\beta_{i2} = (+1)(-1) = -1 \to 0$

$\beta_{i3} = (+1)(+1) = 1$

$S_\lambda$ = matrix that maps $y$ to fitted range

## Closed Form Solution

$\nabla\beta = -2x^T(y - x\beta) + 2\lambda\beta$

$0 = -2x^T(y - x\beta) + 2\lambda\beta$

$x^T x\beta + \lambda\beta = x^T y$

Factor $(x^T x + \lambda I)\beta = x^T y$

$\beta = (x^T y)(x^T x + \lambda I)^{-1}$

For ridge, $y_\lambda = S_\lambda y$

$S_\lambda = x(x^T x + \lambda I)^{-1} x^T$

Not orthogonal projection

Ridge shrinks regularization projection toward 0

Model behaves as if it has less effective parameters. Shrinkage

$df_\lambda = \sum_{j=1}^{p} \frac{d_j}{d_j + \lambda}$

$\arg\min \|y - X\beta\| + \lambda \|\beta^2\|$

$\beta = (x^T y)(x^T x + \lambda I)^{-1}$

$x^T x + \lambda I = \begin{bmatrix} 4 & 0 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 5 & 0 \\ 0 & 2 \end{bmatrix}$

inverse

$\begin{bmatrix} \frac{1}{5} & 0 \\ 0 & \frac{1}{2} \end{bmatrix} \cdot \begin{bmatrix} 8 \\ 2 \end{bmatrix} = \begin{bmatrix} \frac{8}{5} \\ 1 \end{bmatrix}$

## Ridge Main Takeaways

- Shrinks coefficients
- Smaller coefficients → lower variance
- Shrinkage pulls away from truth → bias

Higher $\lambda$ = more Penalty

- lower variance, higher bias

## Two Forms

1.) Penalized: $\min_\beta \|y - X\beta\|^2 + \lambda \|\beta\|^2$ → only Penalize coeff, not intercept, that would only shift up/down

2.) Constrained: $\min_\beta \|y - X\beta\|^2$ s.t $\|\beta\|^2 \leq t$ } more intuitive

only allowed to choose $\beta$ inside ball of radius $\sqrt{t}$

For every $\lambda$, there exists radius $t$

eigenvals of $x^T x$: $\in \frac{d_j}{d_j + \lambda}$

Singular vals of $x$: $\in \frac{\sigma^2}{\sigma^2 + \lambda}$

# Lasso Regression

- stands for least absolute shrinkage selection operator

## Core Definition

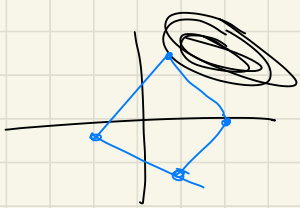
- minimizing RSS with an $l_1$ norm penalty on the coefficients making the objective function is: $\min_{\beta \in \mathbb{R}^p} \|y - X\beta\|^2 + \lambda \|\beta\|_1$

$\lambda$ is a tuning parameter, controls bias/variance trade off. Larger $\lambda$, more shrinkage

## Key Properties

- Variable selection: unlike ridge, some coefficients can be set to 0
  - ↳ Performs automatic feature selection. if $\hat{\beta}_j = 0$, feature does not affect final predictions
- Lasso constraint region is diamond shape. geometry?

- Lasso objective is convex because RSS term and $l_1$ are convex.
  Meaning: local min found is global minimum



- Equivalent Constrained Form:

## Solution under orthogonal design

- Closed form solution exists if design matrix is orthogonal ($X^TX = I$) — Each prediction is uncorrelated and contains completely independent information
  - ↳ Solution for each coefficient given by soft-thresholding

$$\beta_j^{lasso} = sign(\beta_j^{LS})(|\beta_j^{LS}| - \lambda/2)$$

becomes 0 if magnitude of Least squares is smaller than $\lambda/2$, lasso estimate

Ex. given = $X^TX = I$
$\beta_{LS} = (3, -1, 0.5)$
$\lambda = 2$

$\beta^{lasso} = +1(2) = 2$
$\beta^{lasso} = -1(0) = 0$
$= 1(-0.5) = -0.5 < 0 = 0$

- Lasso lacks closed form solution, therefore uses coordinate descent (good for large Data)

Converts High Dimensional → One dimensional problems

- Optimize one coeffecient at a time

# Coordinate Descent Process

1) Compute residual: $r = y - X\beta$

2) Measure correlation with residual: $z_j = \frac{1}{n} x_j^T r$

Big value → feature matters
Small val → feature useless

$\beta_j = \text{sign}(z_j)(|z_j| - \lambda)$

$z_j$ is partial residual correction

$r = y - X\beta$
$z = x^T r$

3) Apply soft thresholding

$$\beta_j = \begin{cases} z_j - \lambda & , z_j > \lambda \\ 0 & |z_j| \leq \lambda \\ z_j + \lambda & z_j < -\lambda \end{cases}$$

where $z_j$ is correlation

$\beta_{OLS} = (\text{sign } \beta_j)(|\beta_j| - \lambda)$
$= + (3 - 1)$
$= 2$
$0.5 - 1$
$= -0.5$
$= 0$
$\} \rightarrow (2, 0)$

# Gradient Descent

- update all parameters at once

$$\beta^{k+1} = \beta^k - \alpha \nabla f$$

next step = last step − learning rate times gradient

$r = y - X\beta$
$z = x^T r$
$\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \end{bmatrix} = [0, 0]$

$r = \begin{bmatrix} 2 \\ 0 \end{bmatrix} - [0, 0]$
$r = [2, 0]$

$\begin{bmatrix} 3 & 1 \\ 2 & 4 \end{bmatrix} \rightarrow \begin{bmatrix} 3 & 2 \\ 1 & 4 \end{bmatrix}$

$z = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \cdot [2, 0]$
$z = [4, 0]$ → soft threshold
$(1, 0)$

# Polynomial Regression - type of linear regression

- Linear regression we assume data can be fit using straight line $y = mx + b$
- When data is non-linear or curves, we use a bending term $x^2, x^3 \ldots$

  Higher Degree = more flexible But can overfit.

  - $f(x) = \beta_0 + \beta_1 x + \beta_2 x^2$

- uses expanded features, that are transformations of $x$
  - ↳ Basis expansion

- Splines are broken down parts of small polynomials together
  - ↳ Breaks $x$ axis into pieces.
- The joining points of these polynomials is Knots
- Cubic Splines: each piece is Knot of degree 3 polynomial
  - ↳ Smooth curvature:

    function is continous on both sides of knot
    first deriv continous
    second deriv continous

- Spline basis functions: pre built building blocks auto matically create piecewise cubic behaviour and enforce smoothness at knots.

Natural Cubic Spline: Forced to become linear at boundaries

→ Piecewise Cubic Polynomials

→ Smoothness at Knots

→ boundary constraint, second deriv=0

How to Find "Solution"

1.) Choose K nots
2.) Build Spline basis function
3.) Fit using OLS: $\hat{\beta} = (x^T x)^{-1} x^T y$

Dimension of Cubic Spline Space

- $K + 4 \rightarrow \underbrace{(4K+1)}_{\text{Params Knots}} - \underbrace{3K}_{\text{3 conditions}}$

- Natural Cubic · $K + \underbrace{2}_{K+4-2}$ } 2 new boundaries, second deriv=0 on both sides

# Smoothing Splines

- Fit data, Penalize curve

$$\min \sum \underbrace{(y_i - f(x_i))^2}_{\text{fit data}} + \lambda \underbrace{\int f''(x)^2 dx}_{\text{smoothness penalty}}$$

- Knots at every data point.
- controls complexity by Penalizing curvature from $\pi$.

- Degrees of freedom for smoothing Splines, same as ridge $\hat{y} = S_\lambda y$, $df = tr(S_\lambda)$

- $\lambda$ controls tradeoff

$\lambda$ large → almost linear → lower variance

$\lambda$ small → high curvature → higher variance

---

## Questions - Practise

Ridge regression: objective function is: $J(\beta) = \sum_{i=1}^{n} (y_i - \beta_0 - \sum \beta_i x_{ij})^2 + \lambda \sum \beta_j^2$

$\lambda = 0$ → same as OLS

$\lambda$ increase means less flexible, effective number of params decrease

Coeff's shrink toward zero

4) $f(\alpha, \beta) = (1 - \alpha - \beta)^2 + |\alpha| + \beta^2$

using coordinate Descent:

fix $\alpha$, minimize over $\beta$

$f(\alpha) = (1 - \alpha - \beta)^2 + |\alpha| + \text{const}$

minimize: $(1 - \alpha - \beta)^2 + \beta^2$

$c = 1 - \alpha$

$(c - \beta)^2 + \beta^2$

Expanding, we get $\beta = \frac{1-\alpha}{2}$

Plug back into f: $1 - \alpha - \beta = 1 - \alpha \frac{1-\alpha}{2}$

⋮

long expansion: $\min (\alpha - 1)^2 < 1 \cdot |\alpha|$

$1 - 1 = 0$

$\alpha^* = 0 \quad \therefore (0, 1/2)$

3.) 10 knots, 11 intervals

$g(x) = \beta_i x + \alpha_i$ means

1 slope param $\beta_i$

1 intercept param $\alpha_i$

$2 \times 11 = 22$

$22 - 10$ knots

Dimension = 12

m knots, 2 extra conditions on front and last, meaning

$dim = m$

# Regression Trees

- Building predictive models by recursively partitioning feature space into smaller regions
- Top-down greedy approach

- Partition feature space into non overlapping sub regions. $R1, R2 \ldots R_K$
- Tries to choose smart splitting rule to give greatest reduction in RSS
- greedy nature is from making best split at that moment.

- predicted value $\hat{c}_K$ is average of response variables $(y_i)$

- growing tree until every leaf is small leads to overfitting (high variance)
  - ↳ Cost Complexity pruning

Complexity pruning: instead of minimizing RSS, minimize penalized version
$PRSS(T) = RSS(T) + \alpha|T|$, where $t$ is number of leaf nodes and alpha is tuning parameter

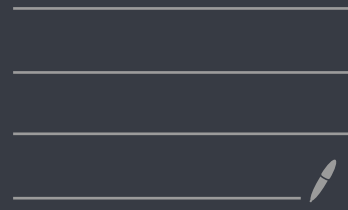
- $\alpha$ Large values → smaller trees
- $\alpha$ small values → Larger trees

- Effective number of Params = number of leaf nodes.

## Key Properties

- Very interpretable and easier to explain
- automatic feature handling
- invariance
- high variance
- Predictive power

# Midterm 3



# PCA - Principal component analysis

- finds new principal components that capture as much variance as possible
- Reduces dimensionality, clean directions with most signal
- all principal components are mutually perpendicular
- works best for linear models / distance based

## Computation Process for PCA

- Center the data (subtract mean from features)
- compute covariance matrix: $\longrightarrow \Sigma = \frac{1}{n} X^T X$
  - $n$ is number of observations (rows)
- Find eigenvectors and eigenvalues of covariance matrix
  - eigenvectors → Principal Components
  - eigenvalues → how important that PC is (variance)

Explained variance: $\frac{\lambda_i}{\sum_j \lambda_j}$

- Sort by importance, then project data onto top K PCs: $\longrightarrow Z = XV_K$

$V_K$ is top K eigenvectors

EX

given $\lambda_1 = 5$, $\lambda_2 = 2$, $\lambda_3 = 1$

Find amount of variance per PC
Find number of components needed to keep atleast 85% of var

1) PC1: $\frac{5}{8} = 62.5\%$
   PC2: $\frac{2}{8} = 25\%$
   PC3: $\frac{1}{8} = 12.5\%$

   $62.5 + 25 = 87.5\%$
   Keep PC1 and PC2

To find eigenvalues $\det(A - \lambda I) = 0$

$(A - \lambda I)v = 0$

Variance lossed = sum of unused eigenvalues

# Need to Memorize

- overfitting = higher variance/low bias
- underfitting = high bias, low variance

## Linear Regression

- Model: $Y = X\beta + \epsilon$
- Prediction $Y = X(X^TX)^{-1}X^Ty$
- Hat matrix $H = X(X^TX)^{-1}X^T$
  - Hat matrix projects $y$ onto column space of $X$
- Effective number of param = $\text{trace}(H) = p$
  - ↳ Equal to number of $p$ features
- 2 main properties of Hat matrix. $H = H^T$, $H^2 = H$
- affine function: $\beta_0 + \beta_1 z_1 + \ldots \beta_p z_p$ often better in practice

- Objective: $\min \|y - X\beta\|^2$
- Closed form solution: $\beta = (X^TX)^{-1}X^Ty$