From sample exam

$t > t_{n-K-1,\alpha/2}$; n is samples, K is predictors, α is confidence

$$t = \frac{\beta_i - \beta^{null}(0)}{SE(\beta_i)}$$

when $\nu \to \infty$, $t_\nu \to Z$, Z is normal

$$CI = \beta \pm (t_{n-K-1,\alpha/2}) \cdot SE$$

null is n-2 DOF

	True class 1	class 2
predicted c1		
c2		

$$RSE = \sqrt{\frac{1}{n-2} RSS}$$ est. of ε

$$TSS = \sum (y_i - \bar{y})^2$$

$$RSS = \sum (y_i - \hat{y}_i)^2$$

$$R^2 = \frac{TSS - RSS}{TSS}$$

$$F_{p,n-p-1,\alpha} = \frac{(TSS - RSS)/p}{RSS/(n-p-1)}$$ p is predictors

TSS - RSS = SSR (regression sum of squares)

$$SSR = \sum (\hat{y}_i - \bar{y})^2$$

Linear Regression

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{f}(x_i))^2 = \frac{RSS}{n}$$

- $Y = \beta_0 + \beta_1 X + \varepsilon$
- $\varepsilon_1 = y_1 - \hat{\beta}_0 - \hat{\beta}_1 x_1$

$$SE(\beta_1) = \frac{\sigma^2}{\sum_{i=1}^{n} (x_i - \bar{x})^2}$$

$$SE(\beta_0) = \sigma^2 \left[\frac{1}{n} + \frac{\bar{x}^2}{\sum (x_i - \bar{x})^2} \right]$$

$\sigma^2 = Var(\varepsilon)$

- Least squares!

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n} (x_i - \bar{x})^2} = \frac{\text{Sample covariance}}{\text{Sample variance}^2}$$

- $$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2} \sqrt{\sum (y_i - \bar{y})^2}}$$

Lesson 3
Classification

$$p_K(x) = Pr(Y = K \mid X = x)$$

Maximum Likelihood:

$$\ell(\beta_0, \beta) = \prod_{i: y_i = 1} p(x_i) \prod_{i: y_i = 0} (1 - p(x_i))$$

Choose $\beta_0 \cdot \beta_1$ to maximize

$$p_{Y|X}(y|x) = Pr(Y = y \mid X = x) = [p(x)]^y [1 - p(x)]^{1-y}, \; y = 0, 1$$

$$\log_{(ln)} \left(\frac{p(x)}{1 - p(x)} \right) = \beta_0 + \beta_1 X$$

Logit: "odds"

$$Z = \frac{\hat{\beta}_i - \beta^{(null)}}{SE(\hat{\beta}_i)}$$ - standard normal

Logistic Regression:

$$p(x) = \frac{e^{\beta_0 + \beta_1 X}}{1 + e^{\beta_0 + \beta_1 X}}$$

Bayes' Theorem:

$$Pr(Y = K \mid X = x) = \frac{Pr(X = x \mid Y = K) \cdot Pr(Y = K)}{Pr(X = x)}$$

LDA or

$$pr(Y = K \mid X = x) = \frac{\pi_K f_K(x)}{\sum_{l=1}^{K} \pi_l f_l(x)}$$ ← density, prior

False positive: Type 1
False negative: Type 2
Sensitivity: TP/P (recall)
Specificity: TN/N
precision: TP/TP+FP
Neg predictive value: TN/TN+FN

$$F_1 \text{ score} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

$$F_\beta = \frac{\beta^2 + 1}{\frac{\beta^2}{\text{recall}} + \frac{1}{\text{precision}}}$$

Naive Bayes: LDA but w/

$$f_K(x) = \prod_{j=1}^{p} f_{jK}(x_j)$$

QDA

$$\delta_K(x) = -\frac{1}{2}(x - \mu_K)^T \Sigma_K^{-1} (x - \mu_K) + \log \pi_K$$

LDA w/ p=1 Gaussian Density:

$$f_K(x) = \frac{1}{\sqrt{2\pi}\sigma_K} e^{-\frac{1}{2}\left(\frac{x - \mu_K}{\sigma_K}\right)^2}$$

or maximize

$$\delta_K(x) = x \cdot \frac{\mu_K}{\sigma^2} - \frac{\mu_K^2}{2\sigma^2} + \log(\pi_K)$$

$$\hat{\mu}_K = \frac{1}{\pi_K} \sum_{i: y_i = K} x_i \; ; \; \sigma^2 = \left(\frac{1}{n-K}\right) \sum_{K=1}^{K} \sum_{i: y_i = K} (x_i - \hat{\mu}_K)^2 = \sum_{K=1}^{K} \frac{n_K - 1}{n - K} \cdot \hat{\sigma}_K^2$$

$$\hat{Pr}(Y = K \mid X = x) = \frac{e^{\hat{\delta}_K(x)}}{\sum_{l=1}^{K} e^{\hat{\delta}_l(x)}}$$

Lesson 4
Resampling

$$LOOCV = \frac{1}{n} \sum_{i=1}^{n} MSE_i$$

$$KFCV: \sum_{K=1}^{K} \frac{n_K}{n} MSE_K$$

Bootstrapping: $\bar{S}^* = \sum_b S(Z^{*b})/B$

$$\hat{Var}[S(Z)] = \frac{1}{B-1} \sum_{b=1}^{B} (S(Z^{*b}) - \bar{S}^*)^2$$

Z^* is a bootstrap dataset

$$SE_B(\hat{\alpha}) = \sqrt{\frac{1}{B-1} \sum_{r=1}^{B} (\hat{\alpha}^{*r} - \bar{\hat{\alpha}}^*)^2}$$

$$\hat{Err}_{boot} = \frac{1}{B} \frac{1}{N} \sum_{b=1}^{B} \sum_{i=1}^{N} L(y_i, \hat{f}^{*b}(x_i))$$

Lesson 5: Model Selection + Regularization

- $C_p = \frac{1}{n}(RSS + 2d\hat{\sigma}^2)$; $\hat{\sigma}^2$ is $\hat{var}(\varepsilon)$, d is # parameters
- σ (aka $var(\varepsilon)$) = RSE - hard to know
- AIC = $-2 \log L + 2 \cdot d$ (max like) or $AIC = \frac{1}{n\hat{\sigma}^2}(RSS + 2d\hat{\sigma}^2)$
- BIC = $\frac{1}{n}(RSS + \log(n) d\hat{\sigma}^2)$
- Adjusted $R^2 = 1 - \frac{RSS/(n-d-1)}{TSS/(n-1)}$

Ridge Regression:

$$RSS = \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2$$

RR estimates should minimize: $RSS + \lambda \sum_{j=1}^{p} \beta_j^2$

First standardize via:

$$\tilde{x}_{ij} = \frac{x_{ij} - \bar{x}_j}{\sqrt{\frac{1}{n} \sum_{i=1}^{n} (x_{ij} - \bar{x}_j)^2}}$$

The Lasso: $RSS + \lambda \sum_{j=1}^{p} |\beta_j|$

Elastic net for redundant variables, $L_1 + L_2$.

$$RSS + \lambda \left[\tfrac{1}{2}(1-\alpha) \|\beta\|_2^2 + \alpha \|\beta\|_1 \right]$$

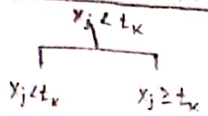
Dimension Reduction

$$Z_m = \sum_{j=1}^{p} \phi_{mj} X_j \; ; \; y_i = \theta_0 + \sum_{m=1}^{M} \theta_m z_{im} + \varepsilon$$

$$\beta_j = \sum_{m=1}^{M} \theta_m \phi_{jm} \; ; \; \theta \to \beta, \; Z \to X, \; Z \text{ via } \phi$$

Choosing ϕ w/ PCR/PCA.
PLS considers Y too

Lesson 6: Tree Based Methods



- Find boxes that minimize

$$RSS = \sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2 \; (+\, \alpha |T|)$$

pruning

- cost-complexity pruning: grow a large tree, then prune it back, via cross-validation.
- Bagging: bootstrap (sample w/ replacement) our data repeatedly, getting B data sets. We average all predictions to obtain $\hat{f}_{bag}(x)$ (or maj vote).
- **Boosting:** Many trees, each fit to the residuals from previous trees. λ is shrinkage parameter. B is # of trees - overfit!
  1) set $\hat{f}(x) = 0$ + $r_i = y_i$ for all i
  2) For $b = 1, 2 \ldots B$: a) Fit tree $\hat{f}^b$ w/ d splits to training data (X, r)
     b) Update $\hat{f}$ by adding in shrunken version of new tree:
     $\hat{f}(x) \leftarrow \hat{f}(x) + \lambda \hat{f}^b(x)$ c) Update residuals $r_i - \lambda \hat{f}^b(x_i)$
     d) Final Model: $\hat{f}(x) = \sum_{b=1}^{B} \lambda \hat{f}^b(x)$.

Gini index: $G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$ (classification) "purity"

A measure of total variance across k classes.

or

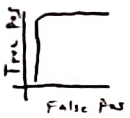
Cross-entropy: $D = -\sum_{k=1}^{K} \hat{p}_{mk} \log(\hat{p}_{mk})$

- Random Forests: **Decorrelated bagging.** Only a random selection of m predictors is considered per split. $m = \sqrt{p}$.

Stacking: Multiple models input to a meta model.

Adaptive meta learner: Implement different models on each section of the feature space, as is appropriate.

- Bayes Error Rate: $1 - E(\max_j \Pr(Y = j \mid X))$



- SMOTE: up + down sampling to account for imbalanced data.
- MSE = $Var(\hat{f}(x_0)) + Bias(f(x_0)) + Var(\varepsilon)$
- KNN: $p_j(x_0) = \frac{1}{K} \sum_{i \in N_0} I(y_i = j)$

DSCI 552 Final

Lesson 7: SVM's

A 2-D hyperplane: $\beta_0 + \beta_1 X_1 + \beta_2 X_2 = 0$

$X: \overbrace{\quad}^{n}$ {p is $X_{n,p}$.

Hence, $\beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p \begin{cases} >0 \text{ if } y_i = 1 \\ <0 \text{ if } y_i = -1 \end{cases}$

or

$y_i(\beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p) > 0$

Thus, we can classify a test observation based on the sign of $\beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p$

- Magnitude suggests confidence

MMH is the solution to:

1) maximize M over $\beta_0, \beta_1, \ldots \beta_p, M$ | 2) Subject to: $\sum_{j=1}^{p} \beta_j^2 = 1$

3) $y_i(\beta_0 + \beta_1 x_{i1} + \ldots + \beta_p x_{ip}) \geq M \quad \forall i = 1, \ldots, n$

- M is the margin

SVC: Soft margin. Append above to

$\geq M(1 - \varepsilon_i)$; $\varepsilon_i \geq 0$; $\sum_{i=1}^{n} \underbrace{\varepsilon_i}_{\text{slack var}} \leq \underbrace{C}_{\text{nonnegative tuning parameter}}$

SVM: Using polynomial functions of the predictors. Append above to.

ε_i: 0, correct; >0, margin; >1, hyp[e]r

$y_i\left(\beta_0 + \sum_{j=1}^{p} \beta_{j1} x_{ij} + \sum_{j=1}^{1} \beta_{j2} x_{ij}^2\right) \geq M(1 - \varepsilon_i)$

- computationally infeasible, thus:

Kernels: $f(x) = \beta_0 + \sum_{i=1}^{n} \alpha_i \langle x, x_i \rangle$; where

$\langle x_i, x_{i'} \rangle = \sum_{j=1}^{p} x_{ij} x_{i'j}$ Alpha is nonzero only for the support vectors, + $\sum_{i}^{p} \alpha_i = 0$.

Now, replace $\langle x, x_i \rangle$ with $K(x, x_i)$ (Kernel).

often $K(x_i, x_{i'}) = \left(1 + \sum_{j=1}^{p} x_{ij} x_{i'j}\right)^d$, but not always.

Thus now, $f(x) = \beta_0 + \sum_{i \in S} \alpha_i K(x, x_i)$

- Multi-Class: One vs One (small K) • One Vs All.
- Multi-Label (religion + politics, etc):
  - Hamming score: fraction of wrong labels to total # of labels.
- RBF: $K(x_i, x_{i'}) = \exp(-\gamma \sum (x_{ij} - x_{i'j})^2)$
- SVMs better than LR for $p \gg n$

Lesson 8: Unsupervised Learning

K-means clustering: Set K, then.

minimize $\left\{ \sum_{k=1}^{K} W(C_k) \right.$, "within cluster variation"

which is average squared distance between all points in a cluster.

- Here's how: 1) Randomly assign each point.
  2) Iterate: a) compute centroid, b) Assign to closest

Alternative: Metoid is centroid observation.

Hierarchical Clustering: Dendogram

1) Measure distane between all pairs.
   Each point is a cluster.
2) For $i = n, n-1, \ldots, 2$:
   a) Measure all pairwise inter-cluster dissimilarities + identify pair of clusters most similar. Fuse them.
   b) Repeat

Types of linkage: a) Complete: compute distance between all points in 2 clusters, record the largest
b) Single: record the smallest
c) Average: Average of all dissimilarities
d) centroid: Can result in undesirable inversions

- Correlation-based distance: Similar if features are correlated, even if euclid. far.
- Between cluster variation: are groups spread apart? Overfits, find scree plot elbow
- Within cluster variation: optimize for small.
- Calinski-Harabasz index: Ideal local maximum for small W, large B.
  $CH(k) = \frac{B(k)/(k-1)}{W(k)/(n-k)}$ - maximize via k.
- Gap statistic: How much $W(k)$ drops @ each k.
  $G(k) = \log W_u(k) - \log W(k)$
- Silhouette analysis: $S_i = \frac{b_i - a_i}{\max(a_i, b_i)}$
  a_i is average intracluster distance
  b_i is average distance to nearest cluster

principal component analysis

$Z_1 = \phi_{11} X_1 + \phi_{21} X_2 + \ldots + \phi_{p1} X_p$ - find largest variance.

where $\sum^{p} \phi_{j1}^2 = 1$ is "normalized." (loading vector)

- proportion of variance explained: PVE
  - What's lost by minimizing dimensions?
  - Visualize a scree plot:

- Fisher's LDA: Supervised PCA

Lesson 9: Semi-Supervised Learning

- Transductive just creates labels, Inductive creates labels and a classifier.
- Self Training: inductive, classifier-based
  - A classifier is built on labeled data, used on unlabeled data, and the most confident are added in.
  - Refinement: reduce the weight of unlabeled data. "Yarowsky"
- Co-training: Build 2 classifiers on two different "views" of the data, if they agree, we add it.
- Cluster and label approach:
  - Cluster, classifier on labeled, assign unlabeled to each cluster label.
- Active Supervised Learning
  - Send intelligent queries about unlabeled data to an oracle, who labels them. Update model.
  - What's "intelligent"?
    - Uncertain • Expected model change
    - Variance reduction
    - Query by committee: many models predict label, disagreements to oracle.

Lesson 10: Neural Networks

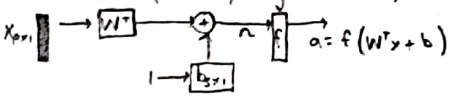
perceptron: $f(x) = \begin{cases} 1, & \beta^T x + \beta_0 \geq 0 \\ -1, & \beta^T x + \beta_0 < 0 \end{cases}$

Update rule: $\beta(i+1) = \beta(i) + 0.5\, e(i)\, x(i)$

$e(i) = y(i) - f(\beta^T(i)\, x(i) + \beta_0(i))$ - error

$\beta_0(i+1) = \beta_0(i) + 0.5\, e(i)$

- Multi-class perceptron: (f is stepwise function)
  - $W = {}_p\left[\overbrace{w_1 \ldots S \ldots w_S}\right.$ • Each class gets a hyperplane.
  - $W(i+1) = W(i) + \alpha\, y(i)\, e^T(i)$
    - α is learning rate or step size
    - n: $W^T(i)\, x(i) + b(i)$ "net weight"
    - a: $f(n) = f(W^T x + b) = \text{sign}(W^T x + b)$



- Layers of perceptrons are for when the train set is not separable, new feature space.
- Sigmoid Function: $f(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{e^x+1}$
- Tanh: $\frac{e^x - e^{-x}}{e^x + e^{-x}} = \frac{e^{2x}-1}{e^{2x}+1}$
- ReLu:  • M: # of layers
- J: <u>objective function</u>: something to be minimized by calculating weights often "expected sum of square errors"

$$E\left\{\sum_{i}^{S^M} (y_i - a_i^M)^2\right\} = E\{e^T e\}$$

- Backpropagation Update Rule:

S^m (first) $= -2F'^{(m)}(n^{(m)})(y-a)$

S^m (subsequent): $F'^{(m)}(n^{(m)})\, W^{(m+1)} S^{(m+1)}$

where: - $F'^{(m)}(n^{(m)}) = \frac{\partial f^m(n_j^m)}{\partial n_j^m}$

(- note: replace $-2(y-a)$ w/ $\nabla_a J$ w/ $J \neq e^T e$)

- $W^{(m)}(k+1) = W^{(m)}(k) - \alpha\, a^{(m-1)} S^{(m)T}$ (k from 1 to N)
- $b^{(m)T}(k+1) = b^{(m)T}(k) - \alpha\, S^{(m)T}$

<u>Regularization</u>

In above, $W^{(m)}(k)(1 - \eta\alpha)$ ← forgetting factor

- η is decay rate

Empirical: Noisy input, noise to weights, rotate.

Softmax: Fixes sigmoid gradient problem

$p = e^a / (1+e^a)$, also cross entropy.

X-entropy: $-y_2 \log a_2' - y_1 \log a_1'$ ← minimize

$p = a_2'$, $1-p = a_1'$, $y_1 = 1$, $y_2 = 0$

Lesson 11: Hidden Markov Models

T: length of observation sequence
N: number of states in the model
M: number of observation symbols
Q: states of Markov process ($0 \to q_{N-1}$)
V: set of possible observations ($0 \to M-1$)
A: State transition probabilities
B: observation probability matrix
π: initial state distribution
O: observation sequence

A is $N \times N$, B is $N \times M$

process: 1) Find p for each possible X
- O^N of them, For len(O) = 4,

eg: $\pi_{x_0} b_{x_0}(O_0)\, a_{x_0,x_1}\, b_{x_1}(O_1)\, a_{x_1,x_2}\, b_{x_2}(O_2)\, a_{x_2,x_3}\, b_{x_3}(O_3)$

- The sum of all O^N of them is the probability of O. Choose the highest for Dynamic Programming.
- For Expectation Maximization, instead of highest total score, choose the likeliest for each position.

<u>3 types</u>

1) Given $\lambda = (A, B, \pi) + O$, find $P(O|\lambda)$
2) Given $\lambda = (A, B, \pi) + O$, find optimal state sequence (hidden).
3) Given O, N, M, find λ that maximizes probability of O.

Scanned with CamScanner