Math 583 Exam 3 is on Friday, Dec. 1 and emphasizes homeworks 7-10 and quizzes 7-10. You are allowed 10 sheets of notes and a calculator. Any needed tables will be provided. CHECK FORMULAS: YOU ARE RESPONSIBLE FOR ANY ERRORS ON THIS HANDOUT!

92) Suppose $Y = m(x) + e$ where x is a random variable ($p = 1$). There are several flexible scatterplot smoothers $\hat{m}(x)$ with df $d > 2$ including loess, lowess, cubic spline, smoothing splines, and the additive error GAM $Y = S(x) + e$ with $S = m$. Two more examples are polynomial regression and regression with basis functions $Y = \sum_{i=1}^{J} \beta_i b_i(x) + e$ where $b_1(x) \equiv 1$. Then PI (2.7) tends to work well.

93) Four important regression models with $\boldsymbol{x}$ $p \times 1$ are listed below.

i) Additive error regression: $Y = SP + e$ where $SP = m(\boldsymbol{x})$. Response plots and PIs are like those for MLR.

ii) Poisson regression: $Y|\boldsymbol{x} \sim \text{Poisson}(\exp(SP))$.

iii) Binary regression: $Y|\boldsymbol{x} \sim \text{bin}\left(1, \dfrac{\mathrm{e}^{\mathrm{SP}}}{1+\mathrm{e}^{\mathrm{SP}}}\right)$.

iv) Binomial regression: $Y_i|\boldsymbol{x}_i \sim \text{bin}\left(\mathrm{m_i}, \dfrac{\mathrm{e}^{\mathrm{SP_i}}}{1+\mathrm{e}^{\mathrm{SP_i}}}\right)$.

94) For Poisson regression, estimate $E(Y|\boldsymbol{x})$ with e^{ESP}. The response plot is a plot of ESP versus Y with the estimated mean function e^{ESP} and lowess added as visual aids. The lowess curve should track the exponential curve fairly closely except possible for the largest values of ESP.

95) For binary logistic regression $Y = 0$ or $Y = 1$. Estimate $E(Y\boldsymbol{x}) = P(Y = 1|\boldsymbol{x})$ with $\hat{\rho}(\boldsymbol{x}) = \hat{\rho}(ESP) = \dfrac{\exp(ESP)}{1+\exp(ESP)}$. The response plot is a plot of ESP versus Y with the (estimated mean function) logistic curve $\hat{\rho}(ESP)$ and a step function added as visual aids. The step function heights are the sample proportion of cases with $Y = 1$ in each slice, and the step function should track the logistic curve fairly closely.

96) The OD plot suggest that overdispersion may be present if the vertical scale is more than 10 times that of the horizontal scale and is useful for Poisson regression and for binomial regression with m_i not too small, but not for binary regression.

97) Let the constant be α and let the nontrivial predictors be $\boldsymbol{x}$. In 93) if $SP = \alpha + \boldsymbol{x}^T\boldsymbol{\beta}$, the model tends to be a generalized linear model (GLM). If $SP = \alpha + \sum_{j=1}^{p} S_j(x_j) = AP$ = additive predictor, the model tends to be a generalized additive model (GAM). The GLM is a special case with $S_j(x_j) = x_j\beta_j$. A GAM is flexible while a GLM is inflexible. The estimated additive predictor EAP = ESP = $\hat{\alpha} + \sum_{j=1}^{p} \hat{S}_j(x_j)$. For a GLM $ESP = \hat{\alpha} + \boldsymbol{x}^T\hat{\boldsymbol{\beta}}$ (the same ESP used for MLR).

98) If $n \geq 10p$, AIC is used for forward selection for a GLM. If $n < 10p$, there are lasso-elastic net type criteria for binary and Poisson regression, and maybe for binomial regression. A relaxed lasso GLM fit the GLM to the predictors with nonzero lasso coefficients. Or do MLR lasso and fit the GLM to the predictors with nonzero MLR lasso coefficients.

99) In *supervised classification*, there are G known groups or populations and m test cases. Each case is assigned to exactly one group based on its measurements $\boldsymbol{w}_i$. Assume that for each population there is a probability density function (pdf) $f_j(\boldsymbol{z})$ where $\boldsymbol{z}$ is a

$p \times 1$ vector and $j = 1, ..., G$. Hence if the random vector $\boldsymbol{x}$ comes from population j, then $\boldsymbol{x}$ has pdf $f_j(\boldsymbol{z})$. Assume that there is a random sample of n_j cases $\boldsymbol{x}_{1,j}, ..., \boldsymbol{x}_{n_j,j}$ for each group. The $n = \sum_{j=1}^G n_j$ cases make up the training data. Let $(\overline{\boldsymbol{x}}_j, \boldsymbol{S}_j)$ denote the sample mean and covariance matrix for each group. Let the ith test case $\boldsymbol{w}_i$ be a new $p \times 1$ random vector from one of the G groups, but the group is unknown. *Discriminant analysis = classification* attempts to allocate (classify) the $\boldsymbol{w}_i$ to the correct groups for $i = 1, ..., m$.

100) The *maximum likelihood discriminant rule* allocates case $\boldsymbol{w}$ to group a if $\hat{f}_a(\boldsymbol{w})$ maximizes $\hat{f}_j(\boldsymbol{w})$ for $j = 1, ..., G$. This rule is robust to nonnormality and the assumption of equal population dispersion matrices, but f_j is hard to estimate for $p > 2$.

101) Given the $\hat{f}_j(\boldsymbol{w})$ or a plot of the $\hat{f}_j(\boldsymbol{w})$, determine the maximum likelihood discriminant rule.

For the following rules, assume that costs of correct and incorrect allocation are unknown or equal, and assume that the probabilities $\pi_j = \rho_j(\boldsymbol{w}_i)$ that $\boldsymbol{w}_i$ is in group j are unknown or equal: $\pi_j = 1/G$ for $j = 1, ..., G$. Often it is assumed that the G groups have the same covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{x}}$. Then the pooled covariance matrix estimator is

$$\boldsymbol{S}_{pool} = \frac{1}{n-G} \sum_{j=1}^G (n_j - 1)\boldsymbol{S}_j$$

where $n = \sum_{j=1}^G n_j$. Let $(\hat{\boldsymbol{\mu}}_j, \hat{\boldsymbol{\Sigma}}_j)$ be the estimator of multivariate location and dispersion for the jth group, e.g. the sample mean and sample covariance matrix $(\hat{\boldsymbol{\mu}}_j, \hat{\boldsymbol{\Sigma}}_j) = (\overline{\boldsymbol{x}}_j, \boldsymbol{S}_j)$.

102) Assume the population dispersion matrices are equal: $\boldsymbol{\Sigma}_j \equiv \boldsymbol{\Sigma}$ for $j = 1, ..., G$. Let $\hat{\boldsymbol{\Sigma}}_{pool}$ be an estimator of $\boldsymbol{\Sigma}$. Then the *linear discriminant rule* is allocate $\boldsymbol{w}$ to the group with the largest value of

$$d_j(\boldsymbol{w}) = \hat{\boldsymbol{\mu}}_j^T \hat{\boldsymbol{\Sigma}}_{pool}^{-1} \boldsymbol{w} - \frac{1}{2} \hat{\boldsymbol{\mu}}_j^T \hat{\boldsymbol{\Sigma}}_{pool}^{-1} \hat{\boldsymbol{\mu}}_j = \hat{\alpha}_j + \hat{\boldsymbol{\beta}}_j^T \boldsymbol{w}$$

where $j = 1, ..., G$. *Linear discriminant analysis* (LDA) uses $(\hat{\boldsymbol{\mu}}_j, \hat{\boldsymbol{\Sigma}}_{pool}) = (\overline{\boldsymbol{x}}_j, \boldsymbol{S}_{pool})$. LDA is robust to nonnormality and somewhat robust to the assumption of equal population covariance matrices.

103) The *quadratic discriminant rule* is allocate $\boldsymbol{w}$ to the group with the largest value of

$$Q_j(\boldsymbol{w}) = \frac{-1}{2} \log(|\hat{\boldsymbol{\Sigma}}_j|) - \frac{1}{2}(\boldsymbol{w} - \hat{\boldsymbol{\mu}}_j)^T \hat{\boldsymbol{\Sigma}}_j^{-1} (\boldsymbol{w} - \hat{\boldsymbol{\mu}}_j)$$

where $j = 1, ..., G$. *Quadratic discriminant analysis* (QDA) uses $(\hat{\boldsymbol{\mu}}_j, \hat{\boldsymbol{\Sigma}}_j) = (\overline{\boldsymbol{x}}_j, \boldsymbol{S}_j)$. QDA has some robustness to nonnormality.

104) The K-nearest neighbors (KNN) method identifies the K cases in the training data that are closest to $\boldsymbol{w}$. Suppose m_j of the K cases are from group j. The *KNN rule* allocates $\boldsymbol{w}$ to group a if m_a maximizes m_j for $j = 1, ..., G$. This method is flexible and relatively fast.

105) Assume that $G = 2$ and that there is a group 0 and a group 1. Let $\rho(\boldsymbol{w}) = P(\boldsymbol{w} \in$ group 1). Let $\hat{\rho}(\boldsymbol{w})$ be the logistic regression (LR) estimate of $\rho(\boldsymbol{w})$. Logistic regression

produces an estimated sufficient predictor $ESP = \hat{\alpha} + \hat{\boldsymbol{\beta}}^T \boldsymbol{w}$. Then

$$\hat{\rho}(\boldsymbol{w}) = \frac{e^{ESP}}{1 + e^{ESP}} = \frac{\exp(\hat{\alpha} + \hat{\boldsymbol{\beta}}^T \boldsymbol{w})}{1 + \exp(\hat{\alpha} + \hat{\boldsymbol{\beta}}^T \boldsymbol{w})}.$$

The *logistic regression discriminant rule* allocates $\boldsymbol{w}$ to group 1 if $\hat{\rho}(\boldsymbol{w}) \geq 0.5$ and allocates $\boldsymbol{w}$ to group 0 if $\hat{\rho}(\boldsymbol{w}) < 0.5$. Equivalently, the LR rule allocates $\boldsymbol{w}$ to group 1 if $ESP \geq 0$ and allocates $\boldsymbol{w}$ to group 0 if $ESP < 0$. the response plot is as in point 95).

106) Given LR output, as shown below in symbols and for a real data set, and given $\boldsymbol{x}$ to classify, be able to a) compute ESP, b) classify $\boldsymbol{x}$ in group 0 or group 1, c) compute $\hat{\rho}(\boldsymbol{x})$.

Label	Estimate	Std. Error	Est/SE	p-value
Constant	$\hat{\alpha}$	$se(\hat{\alpha})$	$z_{o,0}$	for Ho: $\alpha = 0$
x_1	$\hat{\beta}_1$	$se(\hat{\beta}_1)$	$z_{o,1} = \hat{\beta}_1/se(\hat{\beta}_1)$	for Ho: $\beta_1 = 0$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
x_p	$\hat{\beta}_p$	$se(\hat{\beta}_p)$	$z_{o,p} = \hat{\beta}_p/se(\hat{\beta}_p)$	for Ho: $\beta_p = 0$

```
Binomial Regression Kernel mean function = Logistic
Response = Status,Terms = (Bottom Left),Trials = Ones
Coefficient Estimates
Label      Estimate   Std. Error  Est/SE  p-value
Constant  -389.806    104.224     -3.740  0.0002
Bottom     2.26423    0.333233     6.795  0.0000
Left       2.83356    0.795601     3.562  0.0004
```

107) Suppose there is training data $\boldsymbol{x}_{ij}$ for $i = 1, ..., n_j$ for group j. Hence it is known that $\boldsymbol{x}_{ij}$ came from group j where there are $G \geq 2$ groups. Use the discriminant analysis method to classify the training data. If m_j of the n_j group j cases are correctly classified, then the *apparent error rate for group* j is $1 - m_j/n_j$. If $m_A = \sum_{j=1}^G m_j$ of the $n = \sum_{j=1}^G n_j$ cases were correctly classified, then the *apparent error rate* AER $= 1 - m_A/n$.

108) Assume that the training data $\boldsymbol{x}_{1,1}, ..., \boldsymbol{x}_{n_G,G}$ is a random sample from the G populations so that $n_j/n \xrightarrow{P} \pi_j$ as $n \to \infty$ for $j = 1, ..., G$. Hence n_j/n is a consistent estimator of π_j. When $K = 1$, AER $= 0$, but the test error rate L_n of KNN method converges in probability to L where $L_B \leq L \leq 2L_B$ and L_B is the test error rate of the Bayes classifier. If $K_n \to \infty$ and $K_n/n \to 0$ as $n \to \infty$, then the KNN method converges to the Bayes classifier in that the KNN test error rate $L_n \xrightarrow{P} L_B$. Then the leave one out cross validation error rate $\hat{L}_n$ is a good estimator of L_n.

109) Get apparent error rates for `LDA` and `QDA` with the following commands.

```
out2  <- lda(x,group)
1-mean(predict(out2,x)$class==group)
```

```
out3  <- qda(x,group)
1-mean(predict(out3,x)$class==group)
```

Get the AERs for the methods that use variables x_1, x_3, and x_7 with the following commands.

```
out <- lda(x[,c(1,3,7)],group)
1-mean(predict(out,x[,c(1,3,7)])$class==group)

out <- qda(x[,c(1,3,7)],group)
1-mean(predict(out,x[,c(1,3,7)])$class==group)
```

Get the AERs for the methods that leave out variables x_1, x_4, and x_5 with the following commands.

```
out <- lda(x[,-c(1,4,5)],group)
1-mean(predict(out,x[,-c(1,4,5)])$class==group)

out <- qda(x[,-c(1,4,5)],group)
1-mean(predict(out,x[,-c(1,4,5)])$class==group)
```

110) Expect the apparent error rate to be too low: the method works better on the training data than on the new test data to be classified.

111) Cross validation (CV): for $i = 1, ..., n$ where the training data has n cases, compute the discriminant rule with case i left out and see if the rule correctly classifies case i. Let m_C be the number of cases correctly classified. Then the CV error rate is $1 - m_C/n$.

112) Suppose the training data has n cases. Randomly select a subset L of n_v cases to be left out when computing the discriminant rule. Hence $n - n_v$ cases are used to compute the discriminant rule. Let m_L be the number of cases from subset L that are correctly classified. Then the "leave a subset out" error rate is $1 - m_L/n_v$. Here n_v should be large enough to get a good rate. Often use n_v between $0.1n$ and $0.5n$.

113) The k-fold CV is similar to that for MLR. Let Let m_k be the number of cases that are correctly classified. Then the k-fold CV error rate is $1 - m_k/n$.

114) Let $\boldsymbol{W}_i$ be the random vector and $\boldsymbol{w}_i$ be the observed random vector. Let $Y = j$ if $\boldsymbol{w}_i$ comes from the jth group for $j = 1, ..., G$. Then $\pi_j = P(Y = j)$ and the *posterior probability* that $Y = k$ or that $\boldsymbol{w}_i$ belongs to group k is $p_k(\boldsymbol{w}_i) = P(Y = k|\boldsymbol{W}_i = \boldsymbol{w}_i) = \dfrac{\pi_k f_k(\boldsymbol{w}_i)}{\sum_{j=1}^G \pi_j f_j(\boldsymbol{w}_i)}$.

The *Bayesian discriminant rule* allocates case $\boldsymbol{w}_i$ to group a if $\hat{p}_a(\boldsymbol{w}_i)$ maximizes

$$\hat{p}_k(\boldsymbol{w}_i) = \frac{\hat{\pi}_k \hat{f}_k(\boldsymbol{w}_i)}{\sum_{j=1}^G \hat{\pi}_j \hat{f}_j(\boldsymbol{w}_i)}$$

for $k = 1, ..., G$.

The (population) *Bayes classifier* allocates case $\boldsymbol{w}_i$ to group a if $p_a(\boldsymbol{w}_i)$ maximizes $p_k(\boldsymbol{w}_i)$ for $k = 1, ..., G$.

115) A *regularized estimator* attempts to use degrees of freedom d such that the estimator is useful and n/d is large. The pooled covariance matrix used in LDA is regularized compared to the G sample covariance matrices $\boldsymbol{S}_i$ used by QDA.

116) Variable selection is the search for a subset of variables that does a good job of classification.

117) Forward selection: suppose $X_1, ..., X_p$ are variables.

Step 1) Choose variable $W_1 = X_1$ that minimizes the AER.

Step 2) Keep W_1 in the model, and add variable W_2 that minimizes the AER. So W_1 and W_2 are in the model at the end of Step 2).

Step k) Have $W_1, ..., W_{k-1}$ in the model. Add variable W_k that minimizes the AER. So $W_1, ..., W_k$ are in the model at the end of Step k).

Step p) $W_1, ..., W_p = X_1, ..., X_p$, so all p variables are in the model.

118) Backward elimination: suppose $X_1, ..., X_p$ are variables.

Step 1) $W_1, ..., W_p = X_1, ..., X_p$, so all p variables are in the model.

Step 2) Delete variable $W_p = X_j$ such that the model with $p-1$ variables $W_1, ..., W_{p-1}$ minimizes the AER.

Step 3) Delete variable $W_{p-1} = X_j$ such that the model with $p-2$ variables $W_1, ..., W_{p-2}$ minimizes the AER.

Step k) $W_1, ..., W_{p-k+2}$ are in the model. Delete variable $W_{p-k+2} = X_j$ such that the model with $p-k+1$ variables $W_1, ..., W_{p-k+1}$ minimizes the AER.

Step p) Have W_1 and W_2 in the model. Delete variable W_2 such that the model with 1 variable W_1 minimizes the AER.

119) Other criterion can be used and `proc stepdisc` in *SAS* does variable selection.

120) In *R*, using LDA, leave one variable out at a time as long as the AER does not increase much, to find a good subset quickly.

121) KNN suffers from the "curse of dimensionality" in that if $\boldsymbol{w}$ is $p \times 1$, the method works worse as p increases. KNN in *R* uses random numbers to break a tie.

122) A regularized correlation or cavariance matrix attempts to reduce the degrees of freedom and to produce a well conditioned matrix.

123) The usual sample correlation matrix $\boldsymbol{R}$ and sample covariance matrix $\boldsymbol{S}$ need $n \geq 10p$ to start working well, and are overfitting if $n < 5p$. The two matrices are singular if $n \leq p$.

124) A common technique is to use $\boldsymbol{S}_d = diag(\boldsymbol{S})$ or $\boldsymbol{R}_d = diag(\boldsymbol{R})$. This technique uses too much regularization.

125) The population and sample correlation are measures of the strength of a **linear relationship** between two random variables, satisfying $-1 \leq \rho_{ij} \leq 1$ and $-1 \leq r_{ij} \leq 1$. Let the $p \times p$ sample standard deviation matrix $\boldsymbol{D} = \text{diag}(\sqrt{\text{S}_{11}}, ..., \sqrt{\text{S}_{\text{pp}}})$. Then $\boldsymbol{S} = \boldsymbol{DRD}$, and $\boldsymbol{R} = \boldsymbol{D}^{-1}\boldsymbol{S}\boldsymbol{D}^{-1}$.

126) Let $\boldsymbol{S}^{-1} = (S^{ij})$ and let the inverse correlation matirix = precision matrix $\boldsymbol{R}^{-1} = \boldsymbol{D}\boldsymbol{S}^{-1}\boldsymbol{D} = (r^{ij})$.

127) For $\delta \geq 0$, a simple way to regularize a $p \times p$ correlation matrix $\boldsymbol{R} = (r_{ij})$ is to

use

$$\boldsymbol{R}_\delta = \frac{1}{1+\delta}(\boldsymbol{R} + \delta \boldsymbol{I}_p) = (t_{ij}) \tag{1}$$

where $t_{ii} = 1$ and

$$t_{ij} = \frac{r_{ij}}{1+\delta}$$

for $i \neq j$. If λ_i is the ith eigenvalue of $\boldsymbol{R}$, then $(\lambda_i + \delta)/(1+\delta)$ is the ith eigenvalue of $\boldsymbol{R}_\delta$ since if $\boldsymbol{R}\ \boldsymbol{x} = \lambda_i\ \boldsymbol{x}$, then

$$\boldsymbol{R}_\delta\ \boldsymbol{x} = \frac{1}{1+\delta}(\boldsymbol{R} + \delta \boldsymbol{I}_p)\ \boldsymbol{x} = \frac{1}{1+\delta}(\lambda_i + \delta)\ \boldsymbol{x}.$$

Note that $\boldsymbol{R}_\delta = \kappa \boldsymbol{R} + (1-\kappa)\boldsymbol{I}_p$ where $\kappa = 1/(1+\delta) \in (0,1]$.

128) The condition number of a symmetric positive definite $p \times p$ matrix $\boldsymbol{A}$ is $cond(\boldsymbol{A}) = \lambda_1(\boldsymbol{A})/\lambda_p(\boldsymbol{A})$ where $\lambda_1(\boldsymbol{A}) \geq \lambda_2(\boldsymbol{A}) \geq \cdots \geq \lambda_p(\boldsymbol{A})$ are the eigenvalues of $\boldsymbol{A}$. Note that $cond(\boldsymbol{A}) \geq 1$. A well conditioned matrix has condition number $cond(\boldsymbol{A}) \leq c$ for some number c such as 50 or 500. Hence $\boldsymbol{R}_\delta$ is nonsingular for $\delta > 0$ and well conditioned if

$$cond(\boldsymbol{R}_\delta) = \frac{\lambda_1 + \delta}{\lambda_p + \delta} \leq c, \quad \text{or} \quad \delta = \max\left(0, \frac{\lambda_1 - \mathrm{c}\lambda_\mathrm{p}}{\mathrm{c} - 1}\right)$$

if $1 < c \leq 500$.

129) The matrix can be further regularized by setting $t_{ij} = 0$ if $|t_{ij}| \leq \tau$ where $\tau \in [0,1)$ should be less than 0.5. Denote the resulting matrix by $\boldsymbol{R}(\delta, \tau)$. We suggest using $\tau = 0.05$. Note that $\boldsymbol{R}_\delta = \boldsymbol{R}(\delta, 0)$. We recommend computing $\boldsymbol{I}_p, \boldsymbol{R}(\delta, 0)$ and $\boldsymbol{R}(\delta, 0.05)$ for $c =$ 50, 100, 200, 300, 400, and 500. Compute $\boldsymbol{R}$ if it is nonsingular. Note that a regularized covariance matrix can be found using $\boldsymbol{S}(\delta,\tau) = \boldsymbol{D}\ \ \boldsymbol{R}(\delta,\tau)\ \ \boldsymbol{D}$. Given $\boldsymbol{R}$, be able to find, for example, $\boldsymbol{R}_{\delta=1}$ and $\boldsymbol{R}(\delta = 1, \tau = 0.3)$.

130) For the simplest version of k-means clustering, there are 4 steps.

i) Partition the n cases into k initial groups and find the means of each group. Alternatively, choose k initial seed points. These are groups of size 1 so the mean is equal to the seed point.

ii) Compute distances between each case and each mean. Assign each case to the cluster whose mean is the nearest.

iii) Recalculate the mean of each cluster.

iv) Go to ii) and repeat until no more reassignments occur.

131) Hierarchical clustering also has several steps. A distance is needed. Single linkage (or nearest neighbor) is the minimum distance between cases in cluster i and cases in cluster j. Complete linkage is the maximum distance between cases in cluster i and cases in cluster j. The average distance between clusters is also sometimes used.

a) Start with m $= n$ clusters. Each case forms a cluster. Compute the distance matrix for the n clusters. Let $d_{U,V}$ be the smallest distance. Combine clusters U and V into a single cluster and set $m = n - 1$.

b) Repeat step a) with the new m. Continue until there is a single cluster.

c) Plot the resulting clusters as a dendrogram. Use the dendrogram to select k reasonable clusters of cases. Vertical distances (the hieght of the fusion) determines the similarity of clusters.

132) A *regression tree* is a flexible method for $Y = m(\boldsymbol{x}) + e$ or for $Y_i = m(\boldsymbol{x}_i) + \sigma_i e_i$. A *classification tree* is a flexible method for classification. Both methods produce graphs called *trees* that look like dendrograms. Each branch has a label like $X_i > 7.56$ or $X_i < 3.45$ where X_i is quantitative or a label like $X_j = a, c$ or $X_j \neq d, g$ if X_j is quantitative with levels $a, b, ..., g, h$. **Unless told otherwise**, go to the left of the branch if the condition is true, and go to the right of the branch if the condition is false. A *split* is a rule for creating new branches. The bottom of the tree has leaves = *terminal nodes* that give $\hat{Y} = \hat{Y}|\boldsymbol{x}$ where $\hat{Y}$ is a number for a regression tree and $\hat{Y}$ is a label for the classification group for a classification tree. The tree is binary so a tree with $d \geq 1$ splits (rules) has $d + 1$ terminal nodes.

133) Know how to find $\hat{Y}$ given a tree and $\boldsymbol{x}$ values. If $\boldsymbol{x} = (X_1, ..., X_p)^T$, often not all of the X_i values are needed to find $\hat{Y} = \hat{Y}|\boldsymbol{x}$.

134) Trees that use recursive partitioning for classification and regression trees use the CART algorithm. In growing the tree the binary CART algorithm recursively splits the data in each node until either the terminal node is homogeneous (the region R_m corresponding to the node has all cases from the same group for classification and $Y \approx$ constant for regression), or until the terminal node has ≤ 5 observations. The region R_m corresponding to the mth terminal node is a hyper-box: a p-dimensional set if $\boldsymbol{x}$ is $p \times 1$. Hence trees suffer from the curse of dimensionality.

135) If $Y = m(\boldsymbol{x}) + e$ where $m(\boldsymbol{x}) = g(\alpha + \boldsymbol{\beta}^T\boldsymbol{x})$ or $m(\boldsymbol{x}) = \alpha + \sum_{j=1}^p S_j(X_j)$ (an additive error GAM), then $ESP = \hat{\alpha} + \hat{\boldsymbol{\beta}}^T\boldsymbol{x}$ or $ESP = \hat{\alpha} + \sum_{j=1}^p \hat{S}_j(X_j)$ reduces the dimension from p to 1, and slicing the ESP is more efficient than partitioning the p-dimensional predictor space into p-dimensional hyper-boxes.

136) The tree divides the p-dimensional predictor space into J distinct and nonoverlapping regions (p-dimensional hyper-boxes) $R_1, ..., R_J$. For each observation that falls in region R_j make the same prediction $\hat{Y}_{R_j}$. For example, $\hat{Y}_{R_j} = \overline{Y}_j$, the sample mean of the training data Y in R_j for regression, and $\hat{Y}_{R_j} = mode_j$ for classification where $mode_j$ is the training data group that occurred most often for the training data Y in R_j (a lot like KNN where the region is a p-dimensional hypersphere that contains K training data Y's). For a regression tree, the *response plot* is a plot of $\hat{Y}$ versus Y. Then there are J dot plots, one for each value of $\hat{Y}_{R_j}$, with n_j values $Y_{1,1}, ..., Y_{1,n_j}$ where the $Y_{i,j}$ are the training data in R_j. These J dot plots scatter about the identity line. The residuals corresponding to R_j are $r_{i,j} = Y_{i,j} - \overline{Y}_j$. The *residual plot* of $\hat{Y}$ versus r consists of J dot plots of the residuals, one for each value of $\hat{Y}_{R_j} = \overline{Y}_j$. These dot plots scatter about the $r = 0$ line.

137) If $Y = m(\boldsymbol{x}) + e$, prediction intervals for Y_f using trees can be made with (2.7) where $d = J$ = number of terminal nodes.

138) If J is too large, the tree overfits. Grow a large tree T_0 with J_0 regions where each terminal node has ≤ 5 observations. Let $T \subseteq T_0$, $\alpha \geq 0$, and $|T|$ = number of terminal nodes of tree T. Each terminal node corresponds to a region R_i. For each value of $\alpha > 0$ there corresponds a subtree $T \subseteq T_0$ such that

$$\sum_{m=1}^{|T|} \sum_{i:\boldsymbol{x}_i \in R_m} (Y_i - \hat{Y}_{R_m})^2 + \alpha|T| = RSS(T) + \alpha|T|$$

is as small as possible. Much like lasso, as α increases, a sequence of nested subtrees is produced: $T_{\alpha_M} \subseteq T_{\alpha_{M-1}} \subseteq \cdots \subseteq T_{\alpha_2} \subseteq T_{\alpha_1} \subseteq T_0$. Then k-fold CV is used to select the final tree $T_{\alpha_j^*}$.

139) Let $\hat{Y}|\boldsymbol{x} = \hat{f}(\boldsymbol{x})$. For a regression tree, draw a sample of size n with replacement from $\boldsymbol{x}_1, ..., \boldsymbol{x}_n$. Fit a tree and find $\hat{f}_1^*(\boldsymbol{x})$. Repeat B times to get $\hat{f}_1^*(\boldsymbol{x}), ..., \hat{f}_B^*(\boldsymbol{x})$. then the *bagging estimator* $\hat{f}_{bag}^*(\boldsymbol{x}) = \frac{1}{B}\sum_{i=1}^B \hat{f}_i^*(\boldsymbol{x})$. For classification take samples of size n_i with replacement from each group. Let $\hat{f}_i^*(\boldsymbol{x}) = j_i(\boldsymbol{x}) \in \{1, ..., G\}$ be the estimated level (group) of Y given $\boldsymbol{x}$. Let m_k = number of $j_i(\boldsymbol{x}) = k$ for $k = 1, ..., G$. Take $\hat{f}_{bag}^*(\boldsymbol{x}) = d$ where $m_d = \max(m_1, ..., m_G)$, so d is the "mode level group." For both regression and classification, the trees are not pruned, so terminate when each terminal node has 5 or fewer observations. Bagging a tree typically gives more accuracy than a single tree.

140) The probability of a case not being selected for the ith bootstrap sample is about $e^{-1} \approx 1/3$. These are called out of bag (OOB) observations. Predict $\hat{Y}$ for each OOB observation. Doing this for all B bootstrap samples produces about $B/3$ predictors $\hat{Y}_i$ for each $\boldsymbol{x}_i$. Let the OOB predictor $\hat{Y}_{io}$ = average $\hat{Y}_i$ for regression and mode level for classification. Then the OOB MSE $= \frac{1}{n}\sum_{i=1}^n (Y_i - \hat{Y}_{io})^2$ for regression and $\frac{1}{n}\sum_{i=1}^n I(Y_i \neq \hat{Y}_{io})$ for classification. The OOB MSE is "virtually" equivalent to the leave one out CV estimate for sufficiently large B.

141) Random forests use the bootstrap, but at each split, a random sample of $m \approx \sqrt{p}$ predictors is used as split candidates. Random forests produce trees that are less correlated than bagged trees, and tend to have better test error than bagging.

142) Boosting has $\hat{f}(\boldsymbol{x}) = \sum_{b=1}^B \lambda \hat{f}_b(\boldsymbol{x})$. First set $\hat{f}(\boldsymbol{x}) \equiv 0$ and $r_i = Y_i$. For $b = 1, ..., B$ fit a tree $\hat{f}_b$ with d splits (often $d = 1$ where the tree is a *stump* or $d = 2$) to the training data $(\boldsymbol{X}, \boldsymbol{r})$. Update the tree and the residuals $\hat{f}(\boldsymbol{x}) \leftarrow \hat{f}(\boldsymbol{x}) + \lambda \hat{f}_b(\boldsymbol{x})$ and $r_i \leftarrow r_i - \lambda \hat{f}_b(\boldsymbol{x})$. Using stumps ($d = 1$) leads to an additive model: $\hat{f}(\boldsymbol{x}) = \sum_{j=1}^p \hat{f}_j(X_j)$ where $\boldsymbol{x} = (X_1, ..., X_p)$. So boosting with $d = 1$ is a competitor of the additive error GAM $\hat{Y} = \hat{\alpha} + \sum_{j=1}^p \hat{S}_j(X_j)$. Typically $\lambda = 0.01$ or 0.001.

143) For a binary classification tree with $Y = 0$ or 1, for a fixed value of $\boldsymbol{x}$, the bootstrap produces B estimates $\hat{P}_i^*(Y = 1|\boldsymbol{x})$ of $P(Y = 1|\boldsymbol{x})$. Let $\hat{Y}_i^* = 1$ if $\hat{P}_i^*(Y = 1|\boldsymbol{x}) \geq 0.5$ and $\hat{Y}_i^* = 0$ if $\hat{P}_i^*(Y = 1|\boldsymbol{x}) < 0.5$. Two common methods to get $\hat{Y}|\boldsymbol{x}$ are a) $\hat{Y}|\boldsymbol{x}$ = mode class of 0 or 1, or b) $\hat{Y}|\boldsymbol{x} = 1$ if $\frac{1}{\mathrm{B}}\sum_{\mathrm{i}=1}^{\mathrm{B}} \hat{\mathrm{P}}_{\mathrm{i}}^*(\mathrm{Y} = 1|\boldsymbol{x}) \geq 0.5$ and $\hat{Y}|\boldsymbol{x} = 0$ if $\frac{1}{\mathrm{B}}\sum_{\mathrm{i}=1}^{\mathrm{B}} \hat{\mathrm{P}}_{\mathrm{i}}^*(\mathrm{Y} = 1|\boldsymbol{x}) < 0.5$.

Projects: i) give something you have done (like an MS thesis) related to this class. ii) Analyze a data set with some of the class methods. Send me the data set and the R code to get the data into R and the R code used to analyze the data set. iii) Bootstrap OLS and forward selection with C_p as in Table 2.2, but use more values of n, p, k, ψ, and error distributions. See some R code for Problem 3.12. iv) Same as iii) but use BIC.

144) For two groups with $Y = 1$ or $Y = -1$, let $SP = \beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}$. Classify $\boldsymbol{x}$ in group 1 if $ESP > 0$ and classify $\boldsymbol{x}$ in group -1 if $ESP < 0$. So the classifier $\hat{C}(\boldsymbol{x}) = sign(ESP)$.

145) Suppose the two groups of training data in 144) are separable by a hyperplane. The estimated *optimal separating hyperplane* ESP has the largest margin on the training data. The hyperplanes parallel to the ESP that form the boundaries of the margin are called fences. The fences pass through as least 2 training data set cases forming the support set S of support vectors. The margin M is the distance between the fences. A separating hyperplane has $SP > 0$ if $\boldsymbol{x} \in$ group 1 and $SP < 0$ if $\boldsymbol{x} \in$ group -1. Hence $Y_i\ SP_i = Y_i(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}_i) > 0$ for $i = 1, ..., n$. Think of the hyperplane $\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}$ as dividing $\mathbb{R}^P$ into two halves. The SVM split tries to make the halves homogeneous.

146) *Wide data* has $p >> n$. If $n \leq p + 1$ then there is a separating hyperplane unless there are "exact predictor ties across the class barrier."

147) The optimal margin classifier $(\hat{\beta}_{0M}, \hat{\boldsymbol{\beta}}_M)$ solves $\max_{\beta_0 \in \mathbb{R}, \boldsymbol{\beta} \in \mathbb{R}^P} M$ subject to (*): $Y_i\ SP_i = Y_i(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}_i) \geq M$ for all $i = 1, ..., n$. Equivalently, solve $\min_{\beta_0, \boldsymbol{\beta}} \|\boldsymbol{\beta}\|_2$ subject to (*). This classifier is called a *hard margin classifier* since no training data cases from either group can pass the fences of the classifier. It turns out that $\hat{\boldsymbol{\beta}}_M = \sum_{i \in S} \hat{\alpha}_i \boldsymbol{x}_i$.

148) A *soft margin classifier* allows training data cases from either group to pass the fences or to be misclassified. Let the $\epsilon_i \geq 0$ be slack variables. This classifier solves $\min_{\beta_0, \boldsymbol{\beta}} \|\boldsymbol{\beta}\|_2$ subject to $Y_i\ SP_i = Y_i(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}_i) \geq 1 - \epsilon_i$ for $i = 1, ..., n$ and $\sum_{i=1}^n \epsilon_i \leq B$. Equivalently $\min_{\beta_0, \boldsymbol{\beta}} \sum_{i=1}^n [1 - Y_i(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}_i)]_+ + \lambda \|\boldsymbol{\beta}\|_2^2$, a criterion similar to that of ridge regression. Here $[w]_+ = w$ if $w \geq 0$ and $[w]_+ = 0$ if $w \leq 0$. The *hinge loss* $[1 - Y_i(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}_i)]_+$ is the cost of $\boldsymbol{x}_i$ being on the wrong side of the margin (which is 0 if $\boldsymbol{x}_i$ is on the correct side of the margin).

149) A *support vector machine* (SVM) that uses $\boldsymbol{x}_i$ minimizes the above loss criterion. For separable training data, $(\hat{\beta}_{0,SVM}, \hat{\boldsymbol{\beta}}_{SVM}) \to (\hat{\beta}_{0,M}, \hat{\boldsymbol{\beta}}_M)$ as $\lambda \to 0$. The SVM also has fences and a support set S of support vectors with $\hat{\boldsymbol{\beta}}_{SVM} = \sum_{i \in S} \hat{\gamma}_i \boldsymbol{x}_i$. The $ESP = \hat{\beta}_{0,SVM} + \hat{\boldsymbol{\beta}}_{SVM}^T \boldsymbol{x} = \hat{\beta}_{0,SVM} + \sum_{i \in S} \hat{\gamma}_i \boldsymbol{x}_i^T \boldsymbol{x}$. The SVM can be computed with $O(n^2 p)$ complexity using the Gram matrix $\boldsymbol{X}\boldsymbol{X}^T$ or with $O(np^2)$ complexity using $\boldsymbol{X}^T\boldsymbol{X}$. Ridge regression could also be computed this way.

150) A lasso-SVM solves $\min_{\beta_0, \boldsymbol{\beta}} \sum_{i=1}^n [1 - Y_i(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}_i)]_+ + \lambda \|\boldsymbol{\beta}\|_1$ and does variable selection. For $Y \in \{-1, 1\}$, a "ridged logistic regression" solves
$\min_{\beta_0, \boldsymbol{\beta}} \sum_{i=1}^n \log[1 + \exp(-Y_i(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}_i))] + \lambda \|\boldsymbol{\beta}\|_2^2$. A "lasso logistic regression" would change the squared norm $\|\boldsymbol{\beta}\|_2^2$ to $\|\boldsymbol{\beta}\|_1$.

151) A ROC curve is used to evaluate binary classifiers, and the overall performance is summarized by the area under the ROC curve (AUC). An ideal ROC curve is close to the top left corner (left and top sides of the rectangle) of the plot. The larger the AUC, the better the classifier. The ROC curve plots the false positive rate versus the true positive rate, so $0 \leq AUC \leq 1$. A classifier with $AUC = 0.5$ does no better than

chance. A ROC from test data or validation data is better than a ROC from training data.

152) A *truth table* = *confusion matrix.*

predict	truth -1	1	total
-1	true negative (TN)	false negative (FN)	N^*
1	false positive (FP)	true positive (TP)	P^*
total	N	P	

The **error rate** is $(FP + FN)/(FP + FN + TN + TP)$. This rate is the AER if training data was used and VER if a validation set was used.

The *false positive rate* $= FP/N = 1-$ *specifity* $\approx$ type I error.
The *true positive rate* $= TP/P = 1-$ *sensitivity* $\approx 1-$ type II error $\approx$ power $\approx$ recall.

153) A SVM uses a kernel function $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$. The SVM in 149) uses a *linear kernel* $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{x}_i^T \boldsymbol{x}_j$. A *polynomial kernel of degree* d is $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = [1 + \boldsymbol{x}_i^T \boldsymbol{x}_j]^d$. A *radial kernel* is $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp[-\gamma \|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2)$. The SVM with a linear kernel is a competitor of LDA and logistic regression. The SVM with a nonlinear kernel is a competitor of QDA and KNN. The SVM uses $f(\boldsymbol{x}) = \hat{\beta}_0 + \sum_{i=1}^n \hat{\alpha}_i K(\boldsymbol{x}, \boldsymbol{x}_i) = \hat{\beta}_0 + \sum_{i \in S} \hat{\alpha}_i K(\boldsymbol{x}, \boldsymbol{x}_i) = ESP$ with $\hat{C}(\boldsymbol{x}) = sign(ESP)$. The $n(n-1)/2$ distinct pairs $(\boldsymbol{x}_i, \boldsymbol{x}_j)$ are needed to estimate $\hat{\beta}_0$ and the $\hat{\alpha}_i$.

154) Let $Z = 1$ if $Y = 1$ and $Z = 0$ if $Y = -1$. Then $Z|\boldsymbol{x} \sim$ binomial$(m = 1, \rho(\boldsymbol{x}))$ where $\rho(\boldsymbol{x}) = \rho(SP) = P(Z = 1|\boldsymbol{x})$ and $\rho(0) = 0.5$. This is a binary regression with ρ unspecified. A response plot is ESP versus Z with lowess added as a visual aid. If $ESP = \hat{\beta}_0 + \hat{\boldsymbol{\beta}}^T \boldsymbol{x}$ and $n_i \geq 20p$, then the bootstrap with n_i cases selected with replacement from each group is likely useful. Use the prediction region method.

155) If there are $G \geq 2$ classes, the *one versus one* or *all pairs* classifier constructs $\binom{G}{2}$ binary classifiers, one for each distinct pair of groups. Classify $\boldsymbol{x}$ with $f_{ij}(\boldsymbol{x})$, and let m_i = number of times $\boldsymbol{x}$ is predicted to be in class i. Then $\hat{Y}(\boldsymbol{x}) = \hat{C}(\boldsymbol{x}) = d$ where $m_d = \max(m_1, ..., m_G)$. The *one versus all* classifier fits G binary classifiers: group $i = 1$ versus the $G - 1$ other classes with -1 with $f_i(\boldsymbol{x})$. Then $\hat{Y}(\boldsymbol{x}) = \hat{C}(\boldsymbol{x}) = d$ where $f_d(\boldsymbol{x}) = \max(f_1(\boldsymbol{x}), ..., f_G(\boldsymbol{x}))$.

156) The two classifiers in 155) can be applied to other binary classifiers, and the labels $Y \in \{a, b\}$ can be used. For example $a = 0$ and $b = 1$.