Math 583 Exam 3 is on Wednesday, Dec. 6 with emphasis on homeworks 9-11 and quizzes 9-11. You are allowed 9 sheets of notes and a calculator. Any needed tables will be provided. CHECK FORMULAS: YOU ARE RESPONSIBLE FOR ANY ERRORS ON THIS HANDOUT! Final: Wednesday, December 13 2:45-4:45.

113) 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 $i = 1, ..., m$.

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 $(\hat{\boldsymbol{\mu}}_j, \hat{\boldsymbol{\Sigma}}_j) = (\overline{\boldsymbol{x}}_j, \boldsymbol{S}_j)$.

114) 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 107).

115) 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
```

116) Let $\hat{\boldsymbol{\Sigma}}_{pool}$ be a pooled dispersion estimator such as $\boldsymbol{S}_{pool}$. 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. LDA can be useful if the population dispersion matrices are equal: $\boldsymbol{\Sigma}_j \equiv \boldsymbol{\Sigma}$ for $j = 1, ..., G$. LDA can be useful if some of the $n_j < 10p$. In high dimensions, replace $\boldsymbol{S}_{pool}$ by a regularized estimator such as $\hat{\boldsymbol{\Sigma}}_{pool} = \boldsymbol{I}_p$ or $\hat{\boldsymbol{\Sigma}}_{pool} = diag(\boldsymbol{S}_{pool})$.

117) 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 $= 1 - m_A/n$.

118) Get apparent error rates for `LDA` with the following commands.

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

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)
```

119) 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.

120) leave one out 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 $1 - m_C/n$.

121) leave out a validation set V (data splitting): Suppose the training data has n cases. Randomly select a subset V 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 V that are correctly classified. Then the "validation set" 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 $0.5n$.

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

multiple testing

123) Suppose there are m tests for $g \in I = \{1, ..., m\}$ with $H_{0g} : \mu_{gA} = \mu_{gB}$ versus $H_{1g} : \mu_{gA} \neq \mu_{gB}$. The m tests result in m estimated p-values (pvalues) $\hat{p}_1, ..., \hat{p}_m$. Let $\hat{R}$ give a set of indices i for hyotheses H_{0i}, that are rejected. The empty set $\hat{R} = \emptyset$ is possible. Often $\alpha = 0.1, 0.05$, or 0.01. Let $\hat{p}_{(1)}, ..., \hat{p}_{(m)}$ be the order statistics of the $\hat{p}_i$.

124) Let $I_0 = \{i \in I : H_{0i} \text{ is true } \}$. We call *false positive* the indices $i \in \hat{R} \cap I_0$ and *true positive* the indices $i \in \hat{R} \cap I_0^c$. Let $FP =$ number of false positives and $TP =$ number of true positives. The false discovery proportion $FDP = \dfrac{FP}{FP + TP}$ with $0/0 = 0$. The false discovery rate $FDR = E\left[\dfrac{FP}{FP + TP} I(FP + TP \geq 1)\right]$.

125) The Bonferroni correction uses $\hat{R}_B = \{i \in I : \hat{p}_i \leq \alpha/m\}$. Often $\alpha = 0.1, 0.05$, or 0.01.

126) The Benjamani-Hochberg procedure uses $\hat{R}_{BH} = \emptyset$ if $\{k : \hat{p}_{(k)} \leq \alpha k/m\} = \emptyset$. Otherwise, let $\hat{k} = \max\{k \in I : \hat{p}_{(k)} \leq \alpha k/m\}$, and $\hat{R}_{BH} = \{i \in I : \hat{p}_i \leq \alpha \hat{k}/m\} = \{i_1, ..., i_{\hat{k}} \text{ corresponding to } \hat{p}_{(1)}, ..., \hat{p}_{(\hat{k})}\}$.

Note that the Bonferroni correction uses α/m while 126) uses $\hat{k}\alpha/m$.

ch. 10

127) The **multivariate linear regression mreg1 model**

$$\boldsymbol{y}_i = \boldsymbol{B}^T \boldsymbol{x}_i + \boldsymbol{\epsilon}_i$$

for $i = 1, ..., n$ has $m \geq 2$ response variables $Y_1, ..., Y_m$ and p predictor variables $x_1, x_2, ..., x_p$ where $x_1 \equiv 1$ is the trivial predictor. The ith case is $(\boldsymbol{x}_i^T, \boldsymbol{y}_i^T) = (1, x_{i2}, ..., x_{ip}, Y_{i1}, ..., Y_{im})$ where the 1 could be omitted. The model is written in matrix form as $\boldsymbol{Z} = \boldsymbol{X}\boldsymbol{B} + \boldsymbol{E}$ where the matrices are defined below. The model has $E(\boldsymbol{\epsilon}_k) = \boldsymbol{0}$ and $\text{Cov}(\boldsymbol{\epsilon}_k) = \boldsymbol{\Sigma}_{\boldsymbol{\epsilon}} = (\sigma_{ij})$ for $k = 1, ..., n$. Also $E(\boldsymbol{e}_i) = \boldsymbol{0}$ while $\text{Cov}(\boldsymbol{e}_i, \boldsymbol{e}_j) = \sigma_{ij}\boldsymbol{I}_n$ for $i, j = 1, ..., m$ where $\boldsymbol{I}_n$ is the $n \times n$ identity matrix and $\boldsymbol{e}_i$ is defined below. Then the $p \times m$ coefficient matrix $\boldsymbol{B} = \begin{bmatrix} \boldsymbol{\beta}_1 & \boldsymbol{\beta}_2 & \dots & \boldsymbol{\beta}_m \end{bmatrix}$ and the $m \times m$ covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{\epsilon}}$ are to be estimated, and $E(\boldsymbol{Z}) = \boldsymbol{X}\boldsymbol{B}$ while $E(Y_{ij}) = \boldsymbol{x}_i^T \boldsymbol{\beta}_j$. The $\boldsymbol{\epsilon}_i$ are assumed to be iid. The data matrix $\boldsymbol{W} = [\boldsymbol{X} \;\; \boldsymbol{Y}]$ except usually the first column $\boldsymbol{1}$ of $\boldsymbol{X}$ is omitted. The $n \times m$ matrix

$$\boldsymbol{Z} = \begin{bmatrix} Y_{1,1} & Y_{1,2} & \dots & Y_{1,m} \\ Y_{2,1} & Y_{2,2} & \dots & Y_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ Y_{n,1} & Y_{n,2} & \dots & Y_{n,m} \end{bmatrix} = \begin{bmatrix} \boldsymbol{Y}_1 & \boldsymbol{Y}_2 & \dots & \boldsymbol{Y}_m \end{bmatrix} = \begin{bmatrix} \boldsymbol{y}_1^T \\ \vdots \\ \boldsymbol{y}_n^T \end{bmatrix}.$$

The $n \times p$ design matrix of predictor variables is

$$\boldsymbol{X} = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,p} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n,1} & x_{n,2} & \cdots & x_{n,p} \end{bmatrix} = \begin{bmatrix} \boldsymbol{v}_1 & \boldsymbol{v}_2 & \cdots & \boldsymbol{v}_p \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_1^T \\ \vdots \\ \boldsymbol{x}_n^T \end{bmatrix}$$

where $\boldsymbol{v}_1 = \mathbf{1}$. The $p \times m$ matrix

$$\boldsymbol{B} = \begin{bmatrix} \beta_{1,1} & \beta_{1,2} & \cdots & \beta_{1,m} \\ \beta_{2,1} & \beta_{2,2} & \cdots & \beta_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ \beta_{p,1} & \beta_{p,2} & \cdots & \beta_{p,m} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\beta}_1 & \boldsymbol{\beta}_2 & \cdots & \boldsymbol{\beta}_m \end{bmatrix}.$$

The $n \times m$ matrix

$$\boldsymbol{E} = \begin{bmatrix} \epsilon_{1,1} & \epsilon_{1,2} & \cdots & \epsilon_{1,m} \\ \epsilon_{2,1} & \epsilon_{2,2} & \cdots & \epsilon_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ \epsilon_{n,1} & \epsilon_{n,2} & \cdots & \epsilon_{n,m} \end{bmatrix} = \begin{bmatrix} \boldsymbol{e}_1 & \boldsymbol{e}_2 & \cdots & \boldsymbol{e}_m \end{bmatrix} = \begin{bmatrix} \boldsymbol{\epsilon}_1^T \\ \vdots \\ \boldsymbol{\epsilon}_n^T \end{bmatrix}.$$

Considering the ith row of $\boldsymbol{Z}, \boldsymbol{X}$ and $\boldsymbol{E}$ shows that $\boldsymbol{y}_i^T = \boldsymbol{x}_i^T \boldsymbol{B} + \boldsymbol{\epsilon}_i^T$.

128) Each response variable in a mreg1 model follows a multiple linear regression model $\boldsymbol{Y}_j = \boldsymbol{X}\boldsymbol{\beta}_j + \boldsymbol{e}_j$ for $j = 1, ..., m$ where it is assumed that $E(\boldsymbol{e}_j) = \mathbf{0}$ and $\text{Cov}(\boldsymbol{e}_j) = \sigma_{jj}\boldsymbol{I}_n$. Hence the errors corresponding to the jth response are uncorrelated with variance $\sigma_j^2 = \sigma_{jj}$. Notice that the **same design matrix $\boldsymbol{X}$** of predictors is used for each of the m models, but the jth response variable vector $\boldsymbol{Y}_j$, coefficient vector $\boldsymbol{\beta}_j$ and error vector $\boldsymbol{e}_j$ change and thus depend on j.

Now consider the ith case $(\boldsymbol{x}_i^T, \boldsymbol{y}_i^T)$ which corresponds to the ith row of $\boldsymbol{Z}$ and the ith row of $\boldsymbol{X}$. Then

$$\begin{bmatrix} Y_{i1} = \beta_{11}x_{i1} + \cdots + \beta_{p1}x_{ip} + \epsilon_{i1} = \boldsymbol{x}_i^T\boldsymbol{\beta}_1 + \epsilon_{i1} \\ Y_{i2} = \beta_{12}x_{i1} + \cdots + \beta_{p2}x_{ip} + \epsilon_{i2} = \boldsymbol{x}_i^T\boldsymbol{\beta}_2 + \epsilon_{i2} \\ \vdots \\ Y_{im} = \beta_{1m}x_{i1} + \cdots + \beta_{pm}x_{ip} + \epsilon_{im} = \boldsymbol{x}_i^T\boldsymbol{\beta}_m + \epsilon_{im} \end{bmatrix}.$$

129) The **OLS estimators** for the mreg1 model are

$$\hat{\boldsymbol{B}} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{Z} = \begin{bmatrix} \hat{\boldsymbol{\beta}}_1 & \hat{\boldsymbol{\beta}}_2 & \cdots & \hat{\boldsymbol{\beta}}_m \end{bmatrix}.$$

130) Let $\boldsymbol{X} = (\mathbf{1} \;\; \boldsymbol{X}_1)$. The **mreg2 model** is

$$\boldsymbol{y}_i = \boldsymbol{\alpha} + \boldsymbol{B}_S^T\boldsymbol{x}_i + \boldsymbol{\epsilon}_i$$

for $i = 1, ..., n$ with

$$\boldsymbol{Z} = \boldsymbol{X}\boldsymbol{B} + \boldsymbol{E} = \boldsymbol{X}\begin{bmatrix} \boldsymbol{\alpha}^T \\ \boldsymbol{B}_S \end{bmatrix} + \boldsymbol{E} = \begin{bmatrix} \boldsymbol{\alpha}^T \\ \vdots \\ \boldsymbol{\alpha}^T \end{bmatrix} + \boldsymbol{X}_1\boldsymbol{B}_S + \boldsymbol{E}.$$

Now consider the ith case $(\boldsymbol{x}_i^T, \boldsymbol{y}_i^T)$ which corresponds to the ith row of $\boldsymbol{Z}$ and the ith row of $\boldsymbol{X}$. The nontrivial predictors are in $\boldsymbol{x}_i$ for the mreg2 model. Then

$$\left[\begin{array}{c} Y_{i1} = \alpha_1 + \beta_{11} x_{i1} + \cdots + \beta_{p1} x_{ip} + \epsilon_{i1} = \alpha_1 + \boldsymbol{x}_i^T \boldsymbol{\beta}_1 + \epsilon_{i1} \\ Y_{i2} = \alpha_2 + \beta_{12} x_{i1} + \cdots + \beta_{p2} x_{ip} + \epsilon_{i2} = \alpha_2 + \boldsymbol{x}_i^T \boldsymbol{\beta}_2 + \epsilon_{i2} \\ \vdots \\ Y_{im} = \alpha_m + \beta_{1m} x_{i1} + \cdots + \beta_{pm} x_{ip} + \epsilon_{im} = \alpha_m + \boldsymbol{x}_i^T \boldsymbol{\beta}_m + \epsilon_{im} \end{array} \right].$$

131) For the mreg2 model, the OLS estimators are $\hat{\boldsymbol{B}} = (\boldsymbol{X}^T \boldsymbol{X})^{-1} \boldsymbol{X}^T \boldsymbol{Z}$, $\hat{\boldsymbol{\alpha}} = \overline{\boldsymbol{y}} - \hat{\boldsymbol{B}}_S^T \overline{\boldsymbol{x}}$, and $\hat{\boldsymbol{B}}_S = \hat{\boldsymbol{\Sigma}}_{\boldsymbol{x}}^{-1} \hat{\boldsymbol{\Sigma}}_{\boldsymbol{xy}}$ where $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{xy}} = \frac{1}{n-1} \sum_{i=1}^n (\boldsymbol{x}_i - \overline{\boldsymbol{x}})(\boldsymbol{y}_i - \overline{\boldsymbol{y}})^T$ which has jth column $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{x} Y_j}$ for $j = 1, ..., m$. Note that $\hat{\alpha}_j = \overline{Y}_j - \hat{\boldsymbol{\beta}}_j^T \overline{\boldsymbol{x}}$. Let

$$\boldsymbol{v} = \begin{pmatrix} \boldsymbol{y} \\ \boldsymbol{x} \end{pmatrix}, \quad E(\boldsymbol{v}) = \boldsymbol{\mu}_{\boldsymbol{v}} = \begin{pmatrix} E(\boldsymbol{y}) \\ E(\boldsymbol{x}) \end{pmatrix} = \begin{pmatrix} \boldsymbol{\mu}_{\boldsymbol{y}} \\ \boldsymbol{\mu}_{\boldsymbol{x}} \end{pmatrix}, \ \hat{\boldsymbol{\mu}}_{\boldsymbol{v}} = \overline{\boldsymbol{v}} = \begin{pmatrix} \overline{\boldsymbol{y}} \\ \overline{\boldsymbol{x}} \end{pmatrix},$$

$$\mathrm{Cov}(\boldsymbol{v}) = \boldsymbol{\Sigma}_{\boldsymbol{v}} = \begin{pmatrix} \boldsymbol{\Sigma}_{\boldsymbol{y}} & \boldsymbol{\Sigma}_{\boldsymbol{yx}} \\ \boldsymbol{\Sigma}_{\boldsymbol{xy}} & \boldsymbol{\Sigma}_{\boldsymbol{x}} \end{pmatrix}, \quad \text{and} \quad \hat{\boldsymbol{\Sigma}}_{\boldsymbol{v}} = \boldsymbol{S}_{\boldsymbol{v}} = \begin{pmatrix} \hat{\boldsymbol{\Sigma}}_{\boldsymbol{y}} & \hat{\boldsymbol{\Sigma}}_{\boldsymbol{yx}} \\ \hat{\boldsymbol{\Sigma}}_{\boldsymbol{xy}} & \hat{\boldsymbol{\Sigma}}_{\boldsymbol{x}} \end{pmatrix}.$$

Let the vector of constants be $\boldsymbol{\alpha}^T = (\alpha_1, ..., \alpha_m)$ and the matrix of slope vectors $\boldsymbol{B}_S = \begin{bmatrix} \boldsymbol{\beta}_1 & \boldsymbol{\beta}_2 & \cdots & \boldsymbol{\beta}_m \end{bmatrix}$. For iid case $\boldsymbol{v}_i$, the population least squares coefficient matrix is

$$\boldsymbol{B} = \begin{pmatrix} \boldsymbol{\alpha}^T \\ \boldsymbol{B}_S \end{pmatrix}$$

where $\boldsymbol{\alpha} = \boldsymbol{\mu}_{\boldsymbol{y}} - \boldsymbol{B}_S^T \boldsymbol{\mu}_{\boldsymbol{x}}$ and $\boldsymbol{B}_S = \boldsymbol{\Sigma}_{\boldsymbol{x}}^{-1} \boldsymbol{\Sigma}_{\boldsymbol{xy}}$ provided the relevant matrices exist.

The OLS mreg2 estimator can be calculated by computing the sample mean and sample covariance matrix $(\overline{\boldsymbol{v}}, \boldsymbol{S}_{\boldsymbol{v}}) = (\overline{\boldsymbol{v}}, \hat{\boldsymbol{\Sigma}}_{\boldsymbol{v}})$ of the $\boldsymbol{v}_i$, and then plug in the results into the formulas for $\hat{\boldsymbol{\alpha}}$ and $\hat{\boldsymbol{B}}_S$.

More on classification

132) For two groups with $Z = 1$ or $Z = -1$, 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 $\hat{C}(\boldsymbol{x}) = sign(ESP)$.

133) Suppose the two groups of training data in 132) 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 $Z_i \ SP_i = Z_i(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}_i) > 0$ 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.

134) *Wide data* = ultra-high dimensional 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."

135) 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 (*): $Z_i \ SP_i = Z_i(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}_i) \geq M$ 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 $\hat{\boldsymbol{\beta}}_M = \sum_{i \in S} \hat{\alpha}_i \boldsymbol{x}_i$.

136) 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 $Z_i \ SP_i = Z_i(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}_i) \geq 1 - \epsilon_i$ for $i = 1, ..., n$ and $\sum_{i=1}^n \epsilon_i \leq B$. Equivalently $\min_{\beta_0,\boldsymbol{\beta}} \sum_{i=1}^n [1 - Z_i(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}_i)]_+ + \lambda \|\boldsymbol{\beta}\|_2^2$, 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* $[1 - Z_i(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}_i)]_+$ 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).

137) 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.

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

139) 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.

140) A *truth table = confusion matrix*.

	truth		total
predict	-1	1	
-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.

141) A SVM uses a kernel function $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$. The SVM in 137) 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 $\hat{\alpha}_i$.

142) Let $Y = 1$ if $Z = 1$ and $Y = 0$ if $Z = -1$. Then $Y|\boldsymbol{x} \sim$ binomial$(m = 1, \rho(\boldsymbol{x}))$ where $\rho(\boldsymbol{x}) = \rho(SP) = P(Y = 1|\boldsymbol{x})$ 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.

143) 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 $\hat{f}_d(\boldsymbol{x}) = \max(\hat{f}_1(\boldsymbol{x}), ..., \hat{f}_G(\boldsymbol{x}))$. (These are ESPs.)

144) The two classifiers in 143) can be applied to other binary classifiers, and the labels $Y \in \{a, b\}$ can be used. For example $a = 0$ and $b = 1$.

145) 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.

146) 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 $\hat{Y} = \hat{Y}|\boldsymbol{x}$.

147) 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.

148) 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.

149) 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* $\hat{f}_{bag}^*(\boldsymbol{x}) = \dfrac{1}{B}\sum_{i=1}^{B} \hat{f}_i^*(\boldsymbol{x})$. 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.

150) 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 $= \dfrac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_{io})^2$ for regression and $\dfrac{1}{n}\sum_{i=1}^{n} I(Y_i \neq \hat{Y}_{io})$ for classification. The OOB MSE is "virtually" equivalent to the leave one out CV estimate for sufficiently large B.

151) 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.

152) 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.

153) 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 $\dfrac{1}{\mathrm{B}}\sum_{\mathrm{i}=1}^{\mathrm{B}} \hat{\mathrm{P}}_{\mathrm{i}}^*(\mathrm{Y} = 1|\boldsymbol{x}) \geq 0.5$ and $\hat{Y}|\boldsymbol{x} = 0$ if $\dfrac{1}{\mathrm{B}}\sum_{\mathrm{i}=1}^{\mathrm{B}} \hat{\mathrm{P}}_{\mathrm{i}}^*(\mathrm{Y} = 1|\boldsymbol{x}) < 0.5$.