The assumption of **Linear Discriminant Analysis** (LDA) is not that the joint density function f(x) is linear in x. Rather, LDA assumes that the conditional probability distribution of the independent variables X given the category Y=k is multivariate normal, with a class-specific mean vector and a covariance matrix that is common to all K classes. The discriminant functions, which are derived from these densities, are linear in x. This is where the "linear" in Linear Discriminant Analysis comes from. The discriminant function, not the density function, is linear in x.

Bootstrap dataset (size n) is drawing from the original dataset n time with replacement.

P-value Logistic Regression: each coefficient represents the change in the log-odds of the dependent variable for a one-unit change in the predictor variable. The p-value for each coefficient tests the null hypothesis that the coefficient is equal to zero (no effect). A low p-value (< 0.05) indicates that you can reject the null hypothesis. In the context given:

LINEAR REGRESSION: Adding a variable transformation: AgeSquared = (Age)^2

It is possible for this new variable to improve the result. By default linear regression assumes a functional form that provides a linear mapping of independent variables to the dependent variable. If one suspects that the relationship between the independent variables and the dependent variable is more complex than a linear relationship, one can add new features which are a non-linear transformation of one or multiple of the existing features. Since these non-linear relations were not considered before with the original set of features, these new variables can improve the model.

*Response variable Y is continuous *R^2 is for train, OSR2 is for test.

Categorical Variables in linear regression: **can only use dummy variable**. NO one-hot encoding

CART can use either dummy or on-hot encoding.

A CART model is more easily interpretable than a linear regression or logistic regression model
* Simple rules to determine a prediction * Provides transparency to decision process
* Graphical display **CART selects the significant variables for us.**

CART can deliver nonlinear predictions
* Difficult to capture "high but not too high" with linear regression (or logistic regression)
* CART creates its "partitions" based on simple rules (e.g.high GPA and moderate experience)

*** How can we handle categorical features?**: *Splitting requires considering all possible partitions of k categories into two groups * The amount of computation is naively exponential in k but there are a few non-trivial tricks to improve this * Implemented in rpart, but not in sklearn * sklearn requires manually creating dummy variables

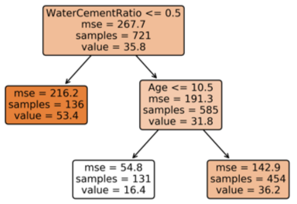
***What if our data has some missing values?: ***This is a general problem in all of statistical learning* It turns out that CART handles this problem relatively gracefully via the more advanced concept of surrogate splits

LOSS: Loss = L$_{FP}$ · # False Positive + L$_{FN}$ · # False Negative

In CART, if we want to modify the loss table then we need to **re-run the model.**

In logistic regression, we can tradeoff between false positive and false negative errors by changing the threshold but still **using the same model.**

Setting the "Technical Parameters": * In linear and logistic regression, the model coefficients are computed to best fit the training data; ***We** could take the same approach with CART; ***Set** the min_samples_leaf parameter to 1: split data all the way down to a single observation in a bucket – ensures that each bucket would have no impurity; ***Set** the cp parameter to 0.0: retain any split as long as it helps improve predictions;
*** This** approach will overfit the training data; ***Instead**, try to optimize for the prediction error, using k-fold cross validation to estimate the prediction error with the training set



IMPURITY ***Ccp_alpha*** is the pruning parameter. We prune the CART tree by removing splits that do not sufficiently improve the model fit. For regression trees, a split must reduce the total impurity by at least *ccp_alpha* to remain. If we would like to produce a **tree with more splits, we would want to prune less**. To produce a tree with more splits we must set **lower** *ccp_alpha* value.

BASELINE/NULL

IMPURITY COST VALUE OF THE CURRENT TREE:

Solution: We calculate the impurity of bucket m for CART regression trees using: $Q_m(T) = \frac{1}{N_m}\sum_{i:x_i\in R_m}(y_i - \hat{y_m})^2$. We compute the total impurity cost of the tree by: $C_{imp}(T) = \sum_{m=1}^{|T|} N_m Q_m(T)$. Using these equations and the values from the CART tree we get: $C_{imp}(T) = \sum_{m=1}^{|T|} N_m MSE_m = 136 * 216.2 + 131 * 54.8 + 454 * 142.9 = 101,458.6$

Solution: We will use our answers from the previous two parts: $R^2 = 1 - \frac{SSE}{TSS} = 1 - \frac{C_{imp}(T)}{C_{imp}(T_{baseline})} = 1 - \frac{101,458.6}{193,011.7} = 0.474$

CART: Adding a variable transformation: AgeSquared = (Age)^2

Unlike the case in the linear regression model, **this feature would NOT improve the CART regression tree model.** As Age is always a positive integer variable and *f(x)=x²* is an increasing function in positive domain, *AgeSquared* would have the same split as Age. For example, a split Age ≤ 10 is equivalent to *AgeSquared ≤ 100.* So, the performance of our model remains same.

Linear Regression: (multiple) * Predicts a continuous response variable – the dependent variable
* Prediction is based on a set of independent variables

- The (true) regression coefficients $\beta = (\beta_0, \beta_1, \ldots, \beta_p)$ are unknown to us
- How do we estimate the regression coefficients?
- Minimize prediction error, as measured by the **residual sum of squares (RSS):**

$$\text{RSS}(\beta) := \sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_{i1} - \ldots - \beta_p x_{ip})^2$$

- Recall the 2-norm of an n-vector z is defined by:

$$\|z\|_2 = \sqrt{z_1^2 + z_2^2 + \ldots + z_n^2}$$

- Then it is easy to see that:

$$\text{RSS}(\beta) = \sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_{i1} - \cdots - \beta_p x_{ip})^2 = \|\mathbf{y} - \mathbf{X}\beta\|_2^2$$

- Also, it holds that (slightly less obvious):

$$\nabla\text{RSS}(\beta) = -2\mathbf{X}^T(\mathbf{y} - \mathbf{X}\beta)$$

R SQUARED: R^2 is the coefficient of determination. R^2 is a measure of the overall quality of the regression model R^2 is a number between 0.0 and 1.0. A higher R^2 means the regression model is a better fit to the (training) data.

- Using the representation $\text{RSS}(\beta) = \|\mathbf{y} - \mathbf{X}\beta\|_2^2$ and assuming that $\text{rank}(\mathbf{X}) = p + 1 < n$, then one may use calculus/linear algebra to show that the solution of $\min_{\beta\in\mathbb{R}^p} \text{RSS}(\beta)$ is given by:

$$\begin{aligned} \nabla\text{RSS}(\hat{\beta}) = 0 \quad &\Leftrightarrow \\ -2\mathbf{X}^T(\mathbf{y} - \mathbf{X}\hat{\beta}) = 0 \quad &\Leftrightarrow \\ 2\mathbf{X}^T\mathbf{X}\hat{\beta} = 2\mathbf{X}^T\mathbf{y} \quad &\Leftrightarrow \\ \hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \end{aligned}$$

- **Prediction for the ith observation:**

$$\hat{y}_i := \hat{\beta}_0 + \hat{\beta}_1 x_{i1} + \ldots + \hat{\beta}_p x_{ip}$$

- **Residuals:** $e_i = y_i - \hat{y}_i$
- **RSS with respect to the estimated coefficients:**

$$\text{RSS} = \text{SSE} = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

- SSE is the sum of squared errors (both conventions often used)

R^2 compares two models:
* 1)the regression model (the one determined by minimizing the RSS (residual sum of squares error), and 2) the "baseline" model. Think of the baseline model as a model you might have built using this data but without any real mathematical thinking.
* Baseline model predicts simplistically using only the mean/average of the sample outcomes

$$R^2 = 1 - \frac{\text{Sum of squared residuals of regression model}}{\text{Sum of squared residuals of baseline model}} = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} = 1 - \frac{SSE}{SST}$$

$$SST = \sum_{i=1}^{n}(y_i - \bar{y})^2$$

OSR2 is an assessment of the real-world performance of the model we have built
*It should only be computed once, at the end of your analysis, as a final metric
*If OSR2 is significantly smaller than R2 (on the training data), this is an indicator of potential overfitting
Overfitting is more likely when:
*The number of parameters to be estimated is large *Data is limited *Care must be taken to make sure that the model we estimate does not suffer from overfitting
* We will see how to address this issue throughout the course, including today's lecture
* Overfitting is related to the "bias-variance tradeoff
* Adjusted R^2 is a simplistic way to account for the risk of overfitting for a linear reg. model
* It is an adjustment to the training set R2 to account for "degrees of freedom".

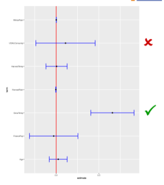
$$\text{Adjusted } R^2 = 1 - \frac{\frac{1}{n-p-1}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\frac{1}{n-1}\sum_{i=1}^{n}(y_i - \bar{y})^2} = 1 - \frac{SSE}{SST}\cdot\frac{n-1}{n-p-1}$$

Bias and Variance of Learning Methods
*Bias refers to the error that is introduced by modeling a complicated relationship with a simple one *Less flexible method have more bias. * Variance refers to the amount that our estimated function changes when you slightly change the dataset * More flexibility usually comes at the cost of higher variance * The bias-variance tradeoff is a common theme in this course that we will continue discussing

Testing the Significance of Regression Coefficients

$H_0 : \beta_j = 0$ vs. $H_a : \beta_j \neq 0$

- Hypothesis test is equivalent to looking at confidence intervals
- Reject null hypothesis as significance level α if and only if (1-α)% confidence interval does not contain 0



- 1.) The observed data (x_i, y_i) $i = 1, \ldots, n$ satisfies

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} + \epsilon_i$$

where $\beta = (\beta_0, \beta_1, \ldots, \beta_p)$ are the true but unknown regression coefficients and the ϵ_i are noise terms

- 2.) $\epsilon_1, \epsilon_2, \ldots, \epsilon_n$ are independent and identically distributed **normal** random variables with mean 0 and variance σ^2
- 3.) If the features $x_1, x_2, \ldots, x_n$ are also regarded as random variables, then they are independent of $\epsilon_1, \epsilon_2, \ldots, \epsilon_n$

- Under the previous set of assumptions, it is possible to prove mathematically that:
- 1.) $\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$ is an unbiased estimator of the true vector of coefficients β:

$$\mathbb{E}\left[\hat{\beta} \mid \mathbf{X}\right] = \beta$$

- 2.) The covariance matrix of $\hat{\beta}$ given $\mathbf{X}$ is:

$$\text{cov}(\hat{\beta} \mid \mathbf{X}) = \sigma^2(\mathbf{X}^T\mathbf{X})^{-1}$$

- 3.) $\hat{\beta}$ is a normally distributed random vector given $\mathbf{X}$

- Question: What's the problem?
- Answer: we usually don't know σ^2 and must estimate that from the data in order to construct the matrix $\text{cov}(\hat{\beta} \mid \mathbf{X}) = \sigma^2(\mathbf{X}^T\mathbf{X})^{-1}$
- Letting $e = \mathbf{y} - \mathbf{X}\hat{\beta}$ denote the vector of training set residuals, then use the estimate:

$$\hat{\sigma}^2 = \frac{\|e\|_2^2}{n-p-1} = \frac{1}{n-p-1}\sum_{i=1}^{n} e_i^2$$

- Given the formula $\text{cov}(\hat{\beta} \mid \mathbf{X}) = \sigma^2(\mathbf{X}^T\mathbf{X})^{-1}$, we can read off the diagonal entries of this matrix to get the standard errors for each coefficient
- Given that $\hat{\beta}$ is normally distributed, we can now easily construct confidence intervals in the usual way, i.e., for some z-score (such as $z^* = 1.96$):

$$\hat{\beta}_j \pm z^*\sqrt{\text{cov}(\hat{\beta} \mid \mathbf{X})_{jj}}$$

OUT-OF-SAMPLE
It is important to understand the assumptions that lead to the results of your analysis (e.g., which variables you retain in your model)
*Ultimately though – regardless of whether you believe or doubt that the assumptions hold for your dataset – it is critical to validate your final model on an out of sample testing se
Multicollinearity * Occurs when two or more predictors are highly correlated
* Makes the estimated coefficients very sensitive to noise in the training data
* can produce very inaccurate estimates hurting interpretability and predictive performance
* **Tell-tale signs:** * Some of the estimated coefficients have the "wrong" sign.
* Some of the coefficients are not significantly different from zero.
* **Multicollinearity** can usually be fixed by deleting one or more independent variable
Multicollinearity can exist without evidence of large correlations in the correlation table.
* **Rule of thumb**: * VIF > 10: definitely a problem
* VIF > 5: could be a problem
*VIF <= 5: probably okay * **VIF: Greater than 10**, remove. It is a sign of multicollinearity.
Regression vs. Classification
*Two important classes of supervised learning problem *Regression involves predicting a continuous response variable *The value of a household *The logarithm of the auction price of a vintage wine *Classification involves predicting a binary yes/no outcome *Did the user click on the ad or not? *Overarching themes but different methods for each

What is VIF?

- Consider regressing each predictor variable X_j on all of the others:

$$X_j = \alpha_0 + \alpha_1 X_1 + \ldots + \alpha_{j-1}X_{j-1} + \alpha_{j+1}X_{j+1} + \ldots + \alpha_p X_p$$

- If the R² for the above (call it R_j^2) is equal to 1, then there exists a perfect linear relationship between X_j and all other independent variables (at least according to the training data)
- So, define: $\text{VIF}_j = \frac{1}{1 - R_j^2}$

Interpreting the Coefficients Using Odds, cont.

$$\text{Odds}(Y = 1) = \frac{\Pr(Y = 1)}{1 - \Pr(Y = 1)}$$

- The logistic regression model is naturally expressed using odds:

$$\log\left(\text{Odds}(Y = 1|X)\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p$$

Logistic Regression
* Logistic Regression is a classification method
* The dependent variable is modeled as a Bernoulli random variable (think success or failure): Y = 0 or Y = 1 * Y = 1 if "success" – default on loan * Y = 0 if "failure" – not default on loan
* We seek to predict the probability of a success outcome of the dependent variable Y:
* Predict probability that the loan defaults, namely Pr(Y=1)
* Probability of success is a particular function of the independent variables $X_1, X_2, \ldots, X_p$.
* logistic regression model gives us predicted probability values that don't always represent the true probability of the corresponding event.

Interpreting Positive Coefficients
*Inquiries in last 6 months has positive coefficient (Beta=0.1171) * Higher value of the variable will result in a higher value of the probability * Lower value of the variable will result in a lower value of the probability
* Consider the probability of default for applicants with different number of inquiries
Interpreting Negative Coefficients
* log(Annual Income) has negative coefficient (Beta = -0.8537)
* Higher variable values yield lower probability estimates
* Lower variable values yield higher probability estimates
* Consider estimated default risk for applicants with different annual incomes but all other variables as in the earlier slide
Change in Outputted Probability is NOT Linear
* A change in a variable does not yield a linear change in the probability outputted by the model.
* This makes it challenging to communicate the precise meaning of a model coefficient (unlike the situation in linear regression).
Bayes Optimal Classifier

Given the classifier $\hat{h}$ trained on the data, we would naturally like to know its error rate: $R(\hat{h}) = \Pr(Y = \hat{h}(X))$
* In practice, we estimate the error rate using test data * In theory, note that the Bayes optimal classifier gives us a lower bound: $R(\hat{h}) \geq R(h^*)$ * the error rate of the Bayes optimal classifier is the irreducible error that can never do better * the difference $R(\hat{h}) - R(h^*)$ is the reducible error in a classification problem.
Two sources of reducible error: * Approximation error (a.k.a. bias), which relates to how well the learning method approximates the Bayes optimal classifier h^* * Estimation error (a.k.a. variance), which relates to the error due to the fact that the training data only provides partial information about the distribution $P_{X,Y}$
Consider a binary classification problem where ■$L_{FP} > 0$ and $L_{FN} > 0$ ■ $L_{TP} = L_{TN} = 0$
■ Given p = Pr(Y = 1 | X = x), derive a threshold q so that it is optimal to predict that Y = 1 if and only if p >= q

$$EL[\hat{Y} = 1|X] = L_{FP} * p_{Y=1}$$
$$EL[\hat{Y} = 0|X] = L_{FN} * p_{Y=0} = L_{FN} * (1-p)$$
$$L_{FP} * p_{Y=1} = L_{FN} * (1-p) => p = \frac{L_{FP}}{L_{FP}+L_{FN}}$$

Interpreting the Coefficients Using Odds

File Preview

■ We can describe probabilities using **odds:**

$$\text{Odds}(Y=1) = \frac{\Pr(Y=1)}{1-\Pr(Y=1)}$$

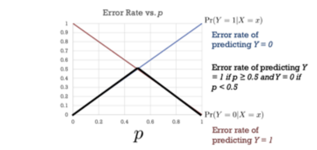
■ An event with probability close to 0 has odds close to 0
■ An event with probability 0.5 has odds equal to 1
■ An event with probability close to 1 has very high odds
■ **Odds** is a notion that is very popular in gambling

Interpreting Logistic Regression Coefficients Using Odds

■ Using odds, we can make simple statements to interpret the logistic regression model's coefficients
■ **Step 1:** Look up the coefficient *b* in the regression summary output
  ■ The coefficient for monthly installment (b_1) is 0.0010
■ **Step 2:** Calculate e^b
  ■ $e^{0.0010} = 1.001$
■ **Step 3:** Interpret using odds
  ■ Holding all other variables constant, each \$1 increase in the installment size yields a 1.001 times higher estimated odds of default
  ■ What about a \$1 decrease?

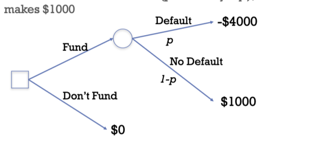
Proof by Picture

■ Let $X = x$ be fixed and consider the range of possible values of $p = f(x) = \Pr(Y = 1|X = x)$



Recall the Simple Lending Decision Tree

■ If borrower defaults (probability *p*), the lender loses \$4000
■ If borrower does not default (probability *1-p*), the lender makes \$1000



Identifying Loans that are Bad Risks

■ A loan is a **good risk** if we expect to make money from it (corresponds to a low probability of default)
■ A loan is a **bad risk** if we expect to lose money from it (corresponds to a high probability of default)
■ From the decision tree, our expected profit from funding the loan is *1000(1-p) – 4000p*
■ Using arithmetic, the break-even point is *p = 0.2*
  ■ A loan with default probability below 0.2 is a good risk
  ■ A loan with default probability above 0.2 is a bad risk
■ Equivalently this threshold minimizes expected loss with L_{FN} = 4000 and L_{FP} = 1000
■ We should only fund a loan that is a good risk

Predictions with a Loss Table

Model

Reality		Predict Y = 0	Predict Y = 1
	Y = 0	L_{TN}	L_{FP}
	Y = 1	L_{FN}	L_{TP}

■ Answer: make a prediction by comparing the expected loss of each option
■ Expected Loss (EL) of Predicting Y = 0 is:

$$EL(0|x) = L_{TN} \cdot \Pr(Y=0|x) + L_{FN} \cdot \Pr(Y=1|x) = L_{TN} \cdot (1-f(x)) + L_{FN} \cdot f(x)$$

Predictions with a Loss Table, cont.

Model

Reality		Predict Y = 0	Predict Y = 1
	Y = 0	L_{TN}	L_{FP}
	Y = 1	L_{FN}	L_{TP}

■ Answer: make a prediction by comparing the expected loss of each option
■ Expected Loss (EL) of Predicting Y = 1 is:

$$EL(1|x) = L_{FP} \cdot \Pr(Y=0|x) + L_{TP} \cdot \Pr(Y=1|x) = L_{FP} \cdot (1-f(x)) + L_{TP} \cdot f(x)$$

Predictions with a Loss Table, cont.

Model

Reality		Predict Y = 0	Predict Y = 1
	Y = 0	L_{TN}	L_{FP}
	Y = 1	L_{FN}	L_{TP}

■ Given perfect knowledge of $f(x) = \Pr(Y=1|X=x)$ the optimal strategy is to choose the option with smaller expected loss (ties broken arbitrarily)

$$h^*(x) = \begin{cases} 1 \text{ if } EL(1|x) \leq EL(0|x) \\ 0 \text{ if } EL(1|x) > EL(0|x) \end{cases}$$

■ h^* defined above is called the "Bayes decision rule"

to minimize L_{FN}, we maximize TPR
in the case LFN=100 (high) LFP=1

Bayes Decision Rule and "Expected Risk"

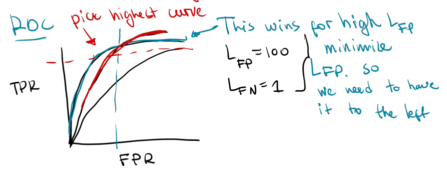
■ Given any classifier h, its "expected risk" $R(h)$ is

$$R(h) = L_{TN} \cdot \Pr(Y=0, h(X)=0) + L_{FP} \cdot \Pr(Y=0, h(X)=1) + L_{FN} \cdot \Pr(Y=1, h(X)=0) + L_{TP} \cdot \Pr(Y=1, h(X)=1)$$

■ It can be shown that h^* is optimal in the sense that

$$R(h^*) = \min_h R(h)$$

■ (Here the minimum is over all possible classifiers h.)
■ This is a complete generalization of the discussion concerning the "Bayes optimal classifier" in the previous lecture



Accuracy

■ **Accuracy:** the proportion of loans that we correctly classified
  ■ Loans that did not default classified as Good Risk
  ■ Loans that did default classified as Bad Risk

$$Accuracy = \frac{Number\ Correct}{Number\ in\ Total} = \frac{1865+178}{1865+178+534+278} = 0.716$$

Model

Reality		Good Risk Pr(Y = 1) < 0.2	Bad Risk Pr(Y = 1) > 0.2
	No Default (Y = 0)	1865	534
	Default (Y = 1)	278	178

True Positive Rate (TPR)

■ **True Positive Rate:** the proportion of defaulting loans (Y_i = 1) that we correctly identified as bad risks (Pr(Y_i = 1) > 0.2)
■ Synonyms you may hear: **sensitivity** or **recall**

$$True\ Positive\ Rate\ (TPR) = \frac{True\ Positives}{All\ Positives} = \frac{178}{178+278} = 0.390$$

Model

Reality		Good Risk Pr(Y = 1) < 0.2	Bad Risk Pr(Y = 1) > 0.2
	No Default (Y = 0)	1865	534
	Default (Y = 1)	278	178

False Positive Rate (FPR)

■ **False Positive Rate (FPR):** the proportion of non-defaulting loans (Y_i = 0) incorrectly identified as bad risks (Pr(Y_i = 1) > 0.2)
■ Synonyms you may hear: **1-specificity** or **fall-out**

$$False\ Positive\ Rate\ (FPR) = \frac{False\ Positives}{All\ Negatives} = \frac{534}{534+1865} = 0.223$$

Model

Reality		Good Risk Pr(Y = 1) < 0.2	Bad Risk Pr(Y = 1) > 0.2
	No Default (Y = 0)	1865	534
	Default (Y = 1)	278	178

$$TPR = \frac{TP}{Actual\ Positive} = \frac{TP}{TP+FN}$$
$$FNR = \frac{FN}{Actual\ Positive} = \frac{FN}{TP+FN}$$
$$TNR = \frac{TN}{Actual\ Negative} = \frac{TN}{TN+FP}$$
$$FPR = \frac{FP}{Actual\ Negative} = \frac{FP}{TN+FP}$$

Interpreting the AUC
* Interpretation: given a randomly selected positive observation (customer who churned) and a randomly selected negative observation (customer who did not churn), AUC is the likelihood that the model would correctly differentiate which is which.
* That is, it is the likelihood that the model would assign a higher churn probability to the customer who churned * AUC measures the model's discriminative ability
* LDA is popular for multiclass classification – when there are more than 2 response categories * LDA is more "stable" in certain situations

Validation Set Approach :Step 1: Split into 3 sets – training, validation, and test; **Step 2:** Build a sequence of models using the training data (for example, using different cp values in CART**); Step 3**: Evaluate each model's performance (e.g. R2, accuracy, TPR, FPR, ...) on the validation set; **Step 4:** Pick the best model and use the test set to estimate future real-world performance of the model;

k-fold Cross-Validation method: For max_features = 1, 2, ..., 18: ***** Do cross-validation with the current value of max_features * Record the averaged cross-validated R2 (over the 5 folds) using the current max_features value ***** Then set the max_features value to be that value that yielded the highest R2 value ***** Looking at R2 is equivalent to looking at RMSE)

Interpreting Random Forest Models: * Random forests consist of hundreds of large CART trees, so interpreting models is much more difficult than for the models *Some interpretability of the relative importance of variables is possible through variable importance measures * For numeric/regression outcomes, we look at the improvement in the sum of squared errors from splitting with the variable, averaged across all trees *Provides an ordering of variables by importance (even though the actual values cannot be readily interpreted)

BOOSTING: The key idea of boosting is to intelligently combine a bunch of "weak" models *Each model is a very shallow tree (for example, 1, 2, or 3 splits only)* But the models are combined adaptively – each model is designed to perform well on parts of the data that the previous models do not perform as well on

■ How do we take this analysis and "back out" a prediction for our original dependent variable Y?

$$R_1 = Y - \hat{f}_1(X)$$
$$R_2 = R_1 - \hat{f}_2(X)$$
$$R_3 = R_2 - \hat{f}_3(X)$$
$$R_4 = R_3 - \hat{f}_4(X)$$

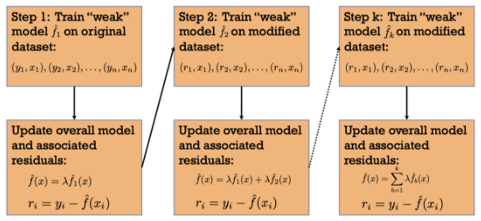
Hence, a prediction for Y is given by:

$$\hat{Y} = \sum_{i=1}^{4} \hat{f}_i(X)$$

■ Boosting produces trees $\hat{f}_1, \hat{f}_2, \ldots, \hat{f}_B$ *sequentially*
■ Tree $\hat{f}_k$ is designed in such a way that it performs well on the parts of the data that trees $\hat{f}_1, \hat{f}_2, \ldots, \hat{f}_{k-1}$ perform weakest on
■ The final model is a rescaled average of the trees:
  ■ λ is called the **shrinkage parameter**

$$\hat{f}(x) = \lambda \sum_{b=1}^{B} \hat{f}_b(x)$$

Note that we initialize the boosting procedure with the null model and proceed for a total of B rounds *Variable importance measures also available for Boosting *Boosting is inherently sequential and is a slow learning process


Step 1: Train "weak" model $\hat{f}_1$ on original dataset:
$(y_1,x_1),(y_2,x_2),\ldots,(y_n,x_n)$
Update overall model and associated residuals:
$\hat{f}(x)=\lambda\hat{f}_1(x)$
$r_i = y_i - \hat{f}(x_i)$
Step 2: Train "weak" model $\hat{f}_2$ on modified dataset:
$(r_1,x_1),(r_2,x_2),\ldots,(r_n,x_n)$
Update overall model and associated residuals:
$\hat{f}(x)=\lambda\hat{f}_1(x)+\lambda\hat{f}_2(x)$
$r_i = y_i - \hat{f}(x_i)$
Step k: Train "weak" model $\hat{f}_k$ on modified dataset:
$(r_1,x_1),(r_2,x_2),\ldots,(r_n,x_n)$
Update overall model and associated residuals:
$\hat{f}(x)=\sum_{b=1}^{k}\lambda\hat{f}_b(x)$
$r_i = y_i - \hat{f}(x_i)$

AdaBoost *It is not clear how to extend the concept of "residuals" for classification problems * Again, trees are added to the model in a sequential way whereby the next tree performs well on parts of the data that are poorly explained by the current model * AdaBoost uses reweighted versions of the training data to achieve this.
Boosting vs. Random Forests *How does boosting compare to random forests?* Contrast the adaptive design of trees in boosting with random forests' completely independent construction of trees*forests' trees are strong (i.e., deep), so each tree fits the data wel and they are averaged to reduce variance *Boosting's trees are weak (i.e., shallow), so each tree has low variance, but they are added together in such a way that the overall model fits the data progressively better

Comparison of Models Seen so Far

	Linear Regression	Logistic Regression	LDA	CART	Random Forests	Boosting
Captures non-linear structure				✓	✓	✓
Simple and interpretable	(✓)?	(✓)??		✓		
Best out-of-sample prediction quality					✓	✓
Can be used for binary classification problems		✓	✓	✓	✓	✓
Can be used for multiclass classification problems			✓	✓	✓	(✓)?
Can be used to predict continuous outcomes	✓			✓	✓	✓