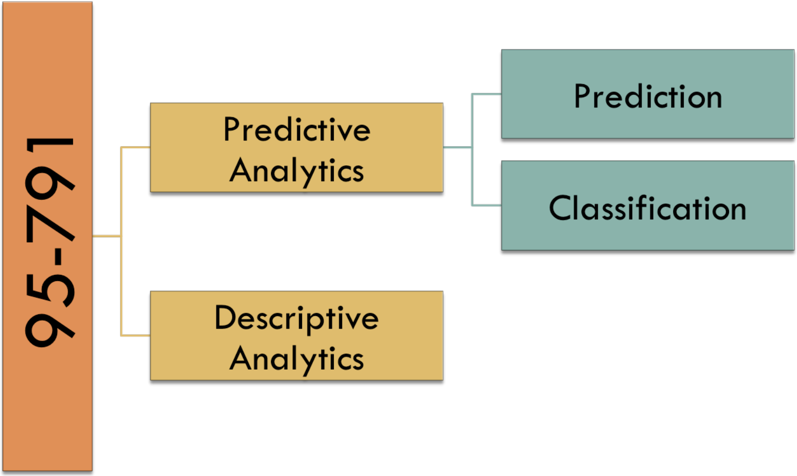
Lecture 3: Model Selection and Classification

Part I: Variable selection, Bootstrap method
Part II: Introduction to Classification

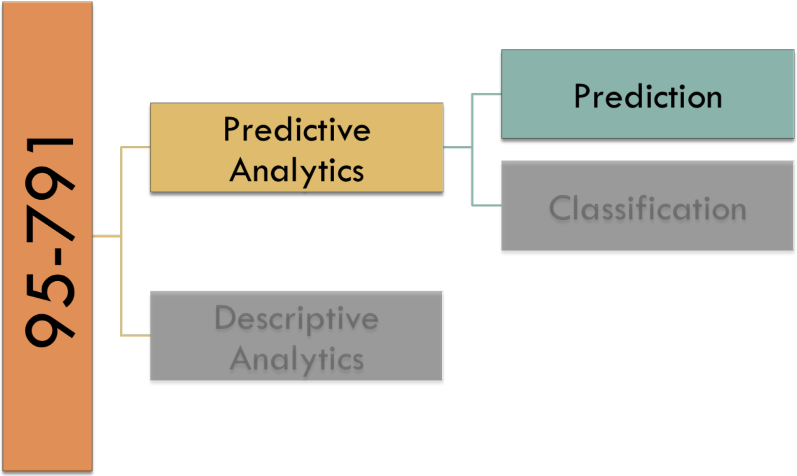
Prof. Alexandra Chouldechova
95-791: Data Mining

November 6, 2017

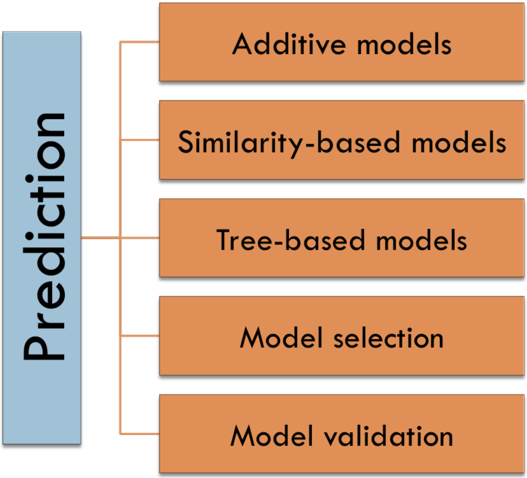
Course Roadmap



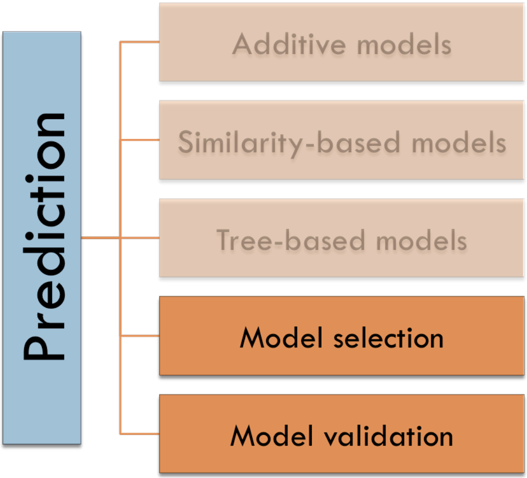
Course Roadmap



Prediction topics



Prediction topics



Agenda for Part I

- **Reminder of last class**

- **Stepwise/Criteria-based methods**
  - Best subset selection
  - Stepwise model selection
  - AIC, BIC

- **Regularized Regression**

K-fold Cross-validation (CV)

- The most widely used approach for estimating Test Error
- Error estimates can be used to select the best model, and also reflect the test error of the final chosen model
- Main idea: K-fold Cross-validation
  1. Randomly split the data into K equal-sized parts ("folds")
  2. Give each part the chance to be the Validation set, treating the other $K-1$ parts (combined) as the Training set
  3. Average the test error over all of the folds
  4. Pick the simplest model among those with the lowest CV-error
  5. Final model: Refit model on the entire data set.
- Most common choices of K: 5, 10, n
  - The case $K = n$ is also known as Leave-one-out Cross-validation (LOOCV)

Cross-validation standard error

- The K-fold CV estimate of prediction error is

$$\mathrm{CV}_{(K)} = \frac{1}{K}\sum_{k=1}^{K}\mathrm{MSE}_k$$

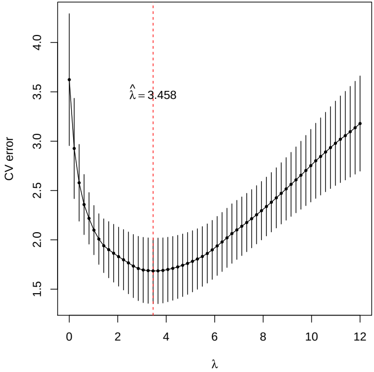
  where MSE_k is the error calculated on Fold k.

- It is also useful to calculate the standard error of the CV estimate
- The typical way of doing this: Calculate the sample standard deviation of $\{\mathrm{MSE}_1 \ldots, \mathrm{MSE}_K\}$, then divide by $\sqrt{K}$:[1]

$$\mathrm{SE}(\mathrm{CV}_{(K)}) = \frac{1}{\sqrt{K}}\mathrm{SD}(\mathrm{MSE}_1 \ldots, \mathrm{MSE}_K)$$

[1] This calculation isn't quite right, but it's a widely accepted approach for calculating standard errors for CV error estimates.

The 1-standard error rule

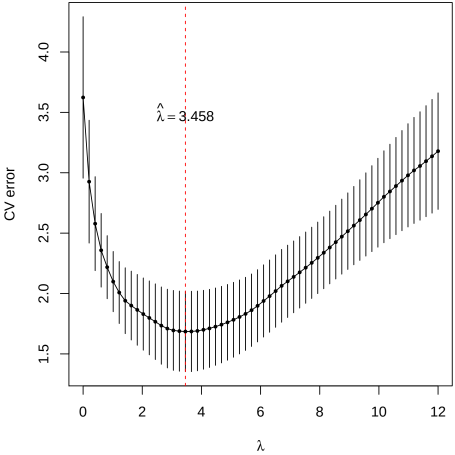


10-fold CV error curve as the tuning parameter λ varies

This plot shows CV error estimates and 1-standard-error bars for a bunch of different choices of a tuning parameter λ.[2]

[2] You can think of λ as the smoothing spline penalty. Large $\lambda \Rightarrow$ simpler model.

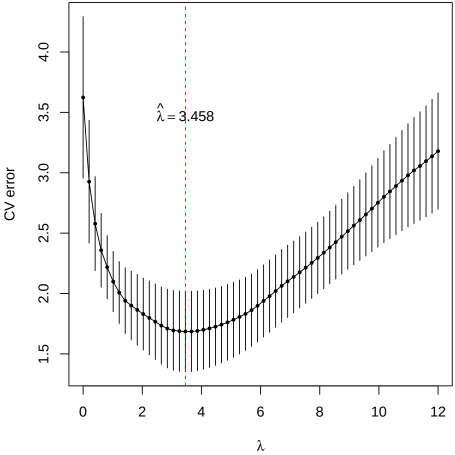
The 1-standard error rule



10-fold CV error curve as the tuning parameter λ varies

- $\lambda = 3.458$ gives us the model with the smallest estimated CV error.
- But we can see from the wide error bars that our prediction error estimates have high uncertainty.

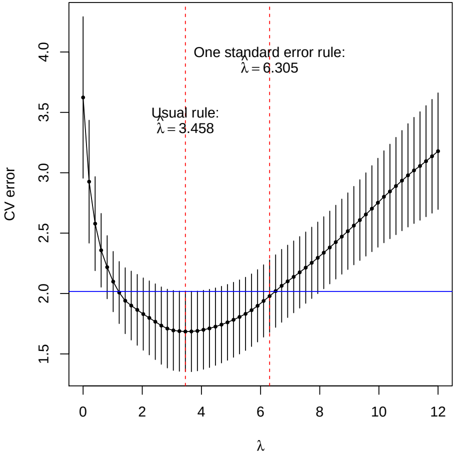
The 1-standard error rule



10-fold CV error curve as the tuning parameter λ varies

- The 1-standard error rule tells us to pick the *simplest model* whose CV error falls inside the 1-SE error bars of the lowest CV error model.

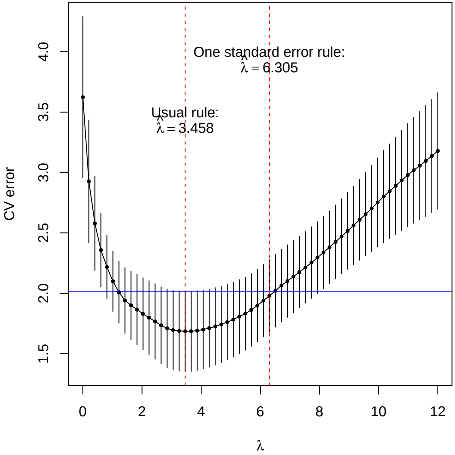
The 1-standard error rule



10-fold CV error curve as the tuning parameter λ varies

- The 1-standard error rule tells us to pick the *simplest model* whose CV error falls inside the 1-SE error bars of the lowest CV error model.

The 1-standard error rule

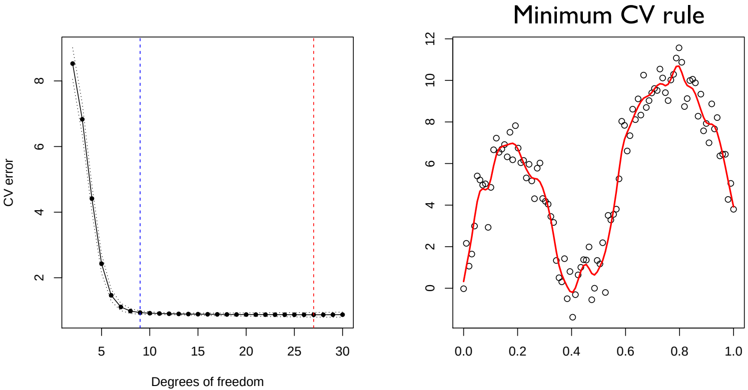


10-fold CV error curve as the tuning parameter λ varies

- Basic idea: We can't be certain that the $\lambda = 6.305$ model actually has higher prediction error than the $\lambda = 3.458$ model, so let's err on the side of caution and go with the *simpler* $\lambda = 6.305$ model.

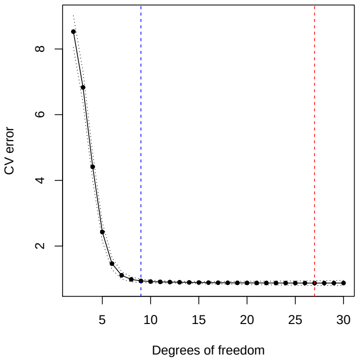
Smoothing spline example

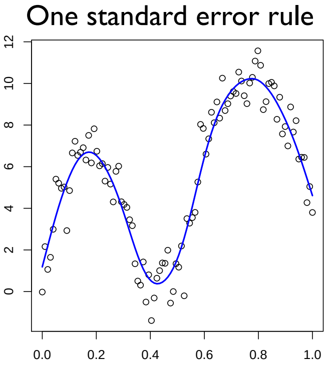
These plots show the results of applying 5-fold cross-validation to select the effective degrees of freedom for a smoothing spline fit to the points in the right panel.



- Even at very large degrees of freedom, the smoothing spline is nicely behaved and has low CV error
- The minimum CV error rule selects a model with 27 degrees of freedom

Smoothing spline example





- The one standard error rule selects a model with 9 degrees of freedom

Summary: Cross-validation

- We started with the question: *How do we pick the best model?*
- One answer: *Pick the model with the lowest prediction error.*
- The Validation set approach and K-fold Cross-validation are two resampling-based methods for estimating the *prediction error* of a model
- K-fold Cross-validation gives much more *stable* and *accurate* estimates of prediction error
- Once we get CV error estimates for the models we're considering, we can either:
  - Pick the model that has the minimum CV error; or
  - Use 1-SE rule and pick the *simplest* model whose error is within 1 standard error of the minimum CV error.
- From this we get: Our chosen model $\hat{f}$, *and* an estimate of its prediction error

Model Selection: Variable Selection in Regression

- Setup: Response Y, predictors $X_1, \ldots, X_p$
- We'd like to use a linear model

$$Y = \beta_0 + \beta_1 X_1 + \ldots \beta_p X_p + \epsilon$$

- We want to identify a small subset of the predictors that we believe to be relevant for predicting Y: i.e., want a *simpler* model

$$Y = \beta_0 + \sum_{j \in \mathcal{S}} \beta_j X_j + \epsilon$$

  for some small subset of predictors $\mathcal{S} \subset \{1, \ldots, p\}$
- E.g., Y is life expectancy, $X_1, \ldots, X_{20,000}$ are $p = 20,000$ genetic measurements.
- Question: Can we identify a small subset of biomarkers (predictors) that accurately predict Y?
- We care about variable selection a lot when getting input measurements is *difficult* or *expensive*

Model Selection: Subset Selection

One approach: Best Subset Selection

1. Let $\mathcal{M}_0$ denote the null model: The intercept-only model.

2. For $k = 1, 2, \ldots, p$
   (a) Fit all $\binom{p}{k}$ models that contain exactly k predictors
   (b) Among these, pick the best model: The one having the *smallest* RSS, or (equivalently) the *largest* R^2

3. Select the single best model from $\mathcal{M}_0, \mathcal{M}_1, \ldots \mathcal{M}_p$ using e.g.,: Cross-validated prediction error, C_p (AIC), BIC, adjusted R^2, etc.

- In Step **2.**, we find the best model of each size
- In Step **3.**, we put the models on *equal footing*, by looking at prediction error or explicitly adjusting for model complexity

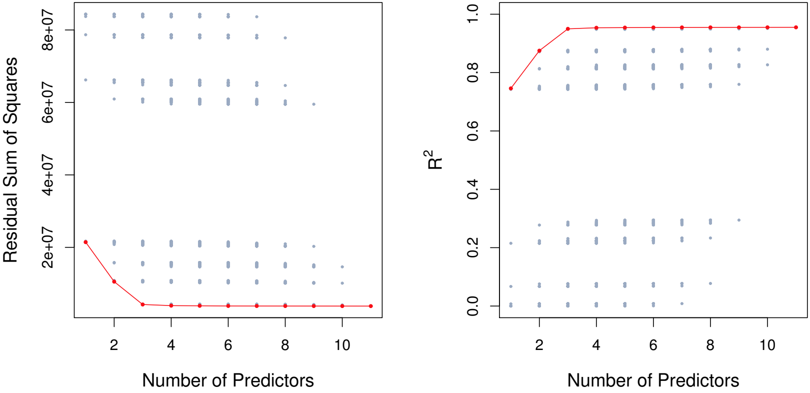
Example: Credit data set

400 observations of 12 variables

	X	Income	Limit	Rating	Cards	Age	Education	Gender	Student	Mar
1	1	14.891	3606	283	2	34	11	Male	No	Yes
2	2	106.025	6645	483	3	82	15	Female	Yes	Yes
3	3	104.593	7075	514	4	71	11	Male	No	No
4	4	148.924	9504	681	3	36	11	Female	No	No
5	5	55.882	4897	357	2	68	16	Male	No	Yes
6	6	80.180	8047	569	4	77	10	Male	No	No
7	7	20.996	3388	259	2	37	12	Female	No	No
8	8	71.408	7114	512	2	87	9	Male	No	No
9	9	15.125	3300	266	5	66	13	Female	No	No

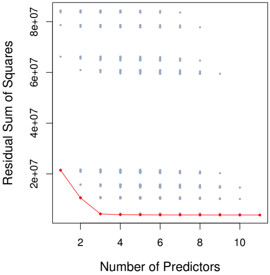
In the `Credit` data example, we wish to figure out which of the $p = 11$ predictors are useful in predicting `Balance`, the individual's average credit card debt.

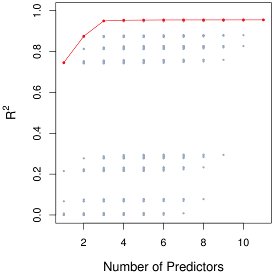
Example: Credit data set



- At $x = k$, there are $\binom{p=11}{k}$ grey points, each corresponding to the RSS from one of the $\binom{11}{k}$ possible models with k predictors.
- Red curves indicate minimum RSS (maximum R^2) attained by the best model of each size.

Which model do we pick?





- *Criterion-based selection:* Choose the final model $\mathcal{M}_k$ according to whichever model on the red curve has the *smallest* AIC (same as C_p here) or BIC.

Method	Criterion
AIC (C_p)	$\frac{1}{n}(RSS + 2k\hat{\sigma}^2)$
BIC	$\frac{1}{n}(RSS + \log(n)k\hat{\sigma}^2)$

AIC and BIC

Method	Criterion
AIC (C_p)	$\frac{1}{n}(RSS + 2k\hat{\sigma}^2)$
BIC	$\frac{1}{n}(RSS + \log(n)k\hat{\sigma}^2)$

$\hat{\sigma}^2$ is an estimate of the variance of the error terms ϵ. It is typically estimated using the *saturated model*, the model with all p predictors included.

- Step 2. of the Best Subset Selection procedure gives us a list of models $\mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_p$, where $\mathcal{M}_k$ has the *lowest* RSS of any model with exactly k predictors
- AIC and BIC are two criteria for choosing the best model among these $p+1$ candidate models

AIC and BIC

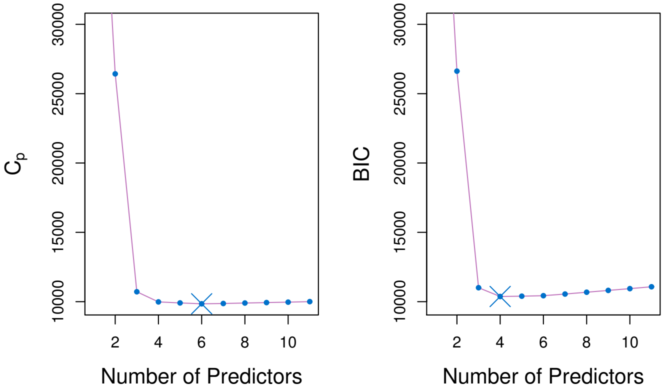
Method	Criterion
AIC (C_p)	$\frac{1}{n}(RSS + 2k\hat{\sigma}^2)$
BIC	$\frac{1}{n}(RSS + \log(n)k\hat{\sigma}^2)$

- Ignoring the $\frac{1}{n}$ factors, both criteria have the form:

$$\underbrace{RSS}_{\text{model error}} + \underbrace{\gamma\hat{\sigma}^2}_{\text{penalty factor}} \times \underbrace{k}_{\text{model df}}$$

- The best model is one that minimizes this trade-off between model error (RSS) and model complexity
- For $n \geq 8$, $\log(n) > 2$, and so the BIC penalty factor is larger than the AIC penalty factor $\Longrightarrow$ BIC selects *smaller* models
- Unlike in Cross-validation, we're not directly trying to pick the model with the best predictive accuracy

Back to the Credit data



- Figure shows AIC (C_p) and BIC curves for the `Credit` data
- AIC: Model $\mathcal{M}_6$ is the best model
- BIC: Model $\mathcal{M}_4$ is the best model

Best subset selection

- To apply the Best Subset Selection approach, we need to fit:

$$1 + \binom{p}{1} + \binom{p}{2} + \cdots + \binom{p}{p-1} + 1 = \sum_{k=0}^{p} \binom{p}{k} = 2^p$$

  regression models

- With $p = 11$, we're already fitting $2^{11} = 2048$ models.

- With $p = 30$, that's $2^{30} = 1,073,741,824$ models

- How will we ever fit all $2^{20,000}$ models to find the best subset of biomarkers for predicting life expectancy?

- Best Subset Selection is computationally prohibitive for even moderate numbers of predictors
  - Actually, recent algorithmic advances in non-convex optimization now enable us to perform BSS on reasonably large data.

- A more computationally efficient approach: Stepwise model selection

Forward Stepwise Selection

- There are three commonly used stepwise methods for model selection
  - Forward stepwise
  - Backward stepwise
  - Forward/Backward stepwise

- Forward stepwise selection begins with model $\mathcal{M}_0$, containing just the intercept, and adds predictors to the model one at a time until all p predictors are in the model

- At each step, the variable that gives the greatest additional improvement to the fit is added to the model

Forward Stepwise Selection

1. Let $\mathcal{M}_0$ denote the intercept-only model
2. For $k = 0, \ldots, p-1$:
   (a) Consider all the models that can be formed by adding one of the remaining $p-k$ predictors to the current model $\mathcal{M}_k$
   (b) Form $\mathcal{M}_{k+1}$ by adding the best new predictor to $\mathcal{M}_k$: the predictor that gives the *smallest* RSS
3. Select the best model among $\mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_p$ using e.g.,: Cross-validated prediction error, C_p (AIC), BIC, adjusted R^2, etc.

Forward Stepwise vs. Best Subset Selection

- With Forward Stepwise Selection, we consider just $p - k$ models at each step, not $\binom{p}{k}$
- This is more computationally efficient, but is *not* guaranteed to give us the best model of size k at each step.

# Variables	Best subset	Forward stepwise
One	`rating`	`rating`
Two	`rating`, `income`	`rating`, `income`
Three	`rating`, `income`, `student`	`rating`, `income`, `student`
Four	`cards`, `income` `student`, `limit`	`rating`, `income`, `student`, `limit`

The first four selected models for best subset selection and forward stepwise selection on the `Credit` *data set. The first three models are identical but the fourth models differ.*

Regularized Regression

- The subset selection methods all try to find the best choice of predictors to use, and then fit the model with least squares
- There's an alternative (better) class of methods that fit a model using all p predictors, and constrain or regularize the coefficient estimates, "shrinking" them towards zero.
- These methods are referred to as Shrinkage Methods or Regularized Regression methods.
- Ridge Regression and the Lasso are two popular regularized regression methods

The Lasso

- The Lasso method estimates coefficients $\hat{\beta}_\lambda$ by minimizing the ℓ_1 penalized RSS:

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

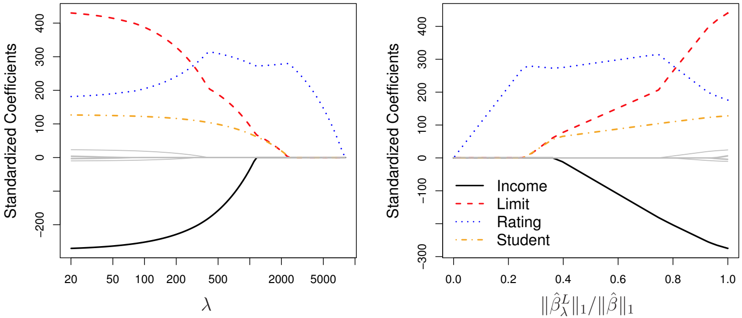
- The quantity $\sum_{j=1}^{p}|\beta_j|$ is typically denoted $\|\beta\|_1$, which is called the ℓ_1 ("ell 1") norm

- For this reason, the Lasso is also called ℓ_1-regularized regression

$$\underbrace{\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)}_{RSS} + \lambda \underbrace{\sum_{j=1}^{p}|\beta_j|}_{\text{penalty}}$$

- We saw this RSS + model complexity penalty structure before when we talked about *smoothing splines*
- The Lasso measures model complexity according to $|\beta_j|$
- The tuning parameter λ allows us to control the overall *complexity* of the model
- $\lambda = 0$ takes us back to least squares
- $\lambda = \infty$ gives us $\hat{\beta}_\lambda = \mathbf{0}$
- The Lasso has the amazing property that for intermediate values of λ, $\hat{\beta}_\lambda$ will have entries *shrunk* towards 0, and some will actually be estimated exactly as 0. i.e., Lasso automatically performs variable selection!

The Lasso Regularization Plot: Credit Data



- x-axis indicates the value of the penalty parameter (or some similar quantity)
- The curves trace out $\hat{\beta}_{\lambda,j}$ for each predictor X_j
- For large λ, most coefficient are 0
- As λ decreases, more coefficients get estimated as non-zero, and the coefficient estimates $\hat{\beta}_{\lambda,j}$ can grow

More about the Lasso

- Lasso: "Least Absolute Shrinkage and Selection Operator"
- The tuning parameter λ allows us to trade off between bias and variance:
  - Large λ: Many $\hat{\beta}_j$ estimated at exactly 0 or are pulled a lot toward 0 (Low Variance, High Bias)
  - Small λ: $\hat{\beta}_j$ close to the least squares solution, but still shrunk toward 0 (Higher Variance, Lower Bias)
- Lasso can be used even if $n < p$, and can still give good results!
- To choose λ: Use Cross-validation!
  1. Choose a sequence of λ values
  2. Calculate the K-fold CV error at each λ
  3. Use the minimum CV error or 1-SE rule to pick $\hat{\lambda}$, and then refit the Lasso on the entire data with $\lambda = \hat{\lambda}$

Coding tip: In **R**, the `glmnet` package makes fitting the Lasso and running Cross-validation super fast and easy.

Lasso example: HIV drug resistance data

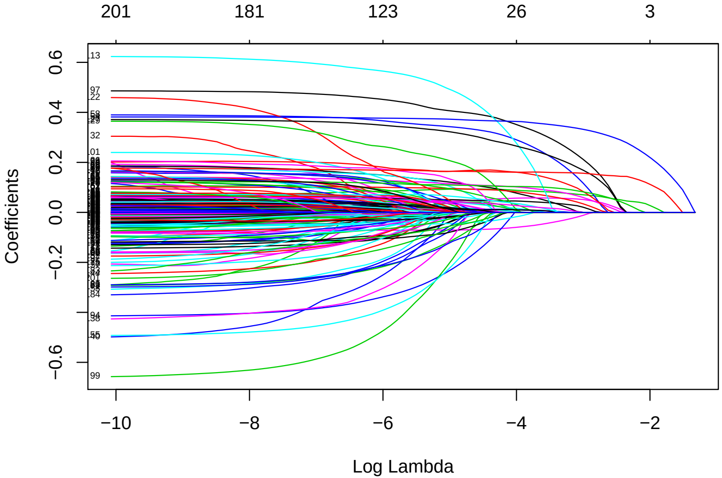
- Data on the presence/absence of $p = 202$ genetic mutations for $n = 1005$ patients

- Response Y is a numeric measure of resistance to a particular HIV drug

- We want to identify which mutations are associated with drug resistance

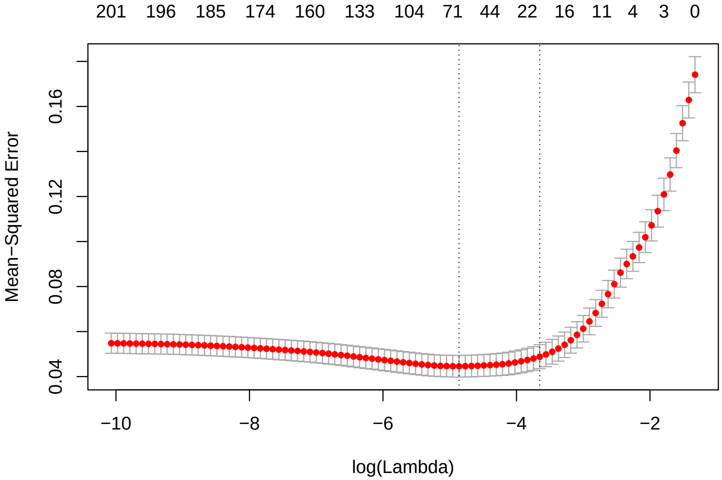
- Approach: Use Cross-validation to fit the Lasso, and look at which coefficients are non-zero

Regularization plot: HIV drug resistance data



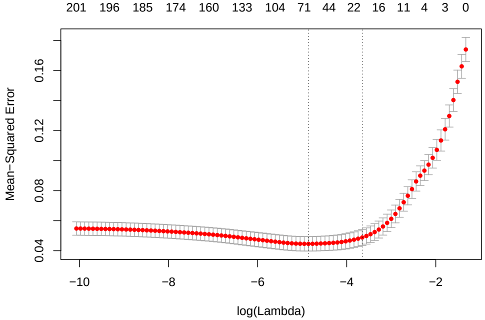
Yikes!

10-fold Cross-validation plot: HIV drug resistance data



- Using the 1-SE rule, we choose the model with $\log(\lambda) = -3.65$

10-fold Cross-validation plot: HIV drug resistance data

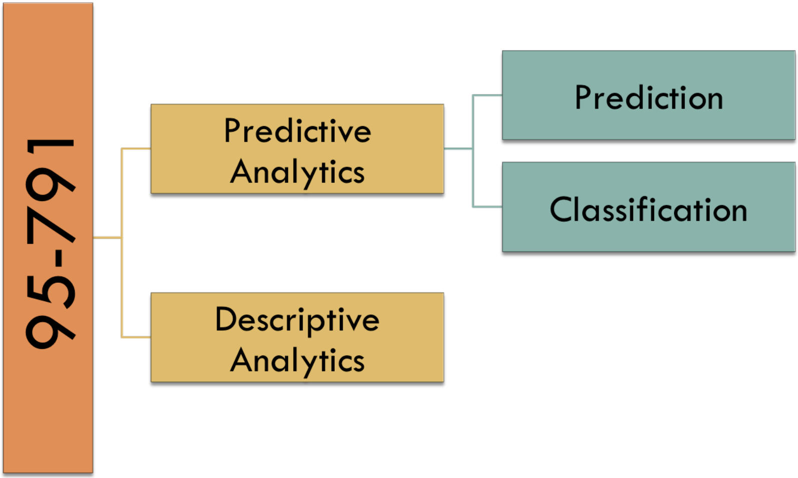


- Using the 1-SE rule, we choose the model with $\log(\lambda) = -3.65$
- At this choice of λ, just 20 of the 202 predictors have non-zero coefficients
- Of these 20 genetic mutations, 15 have been experimentally verified (in biological scientific studies) as being associated with HIV drug resistance

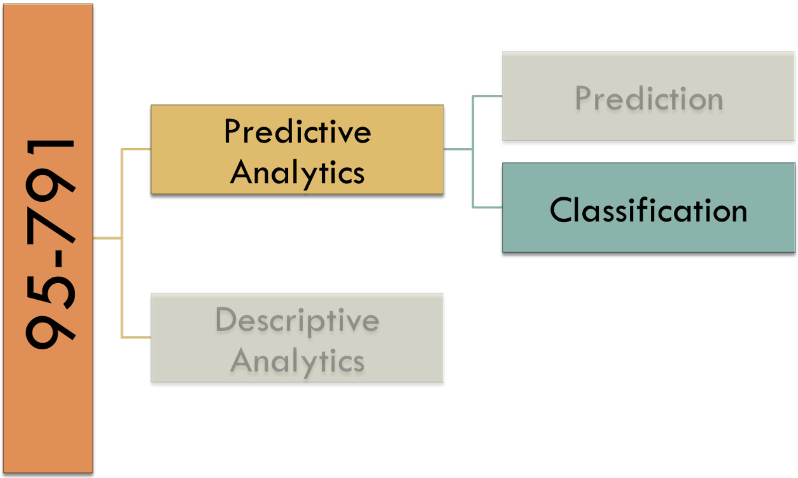
End of Part I

10 minute break

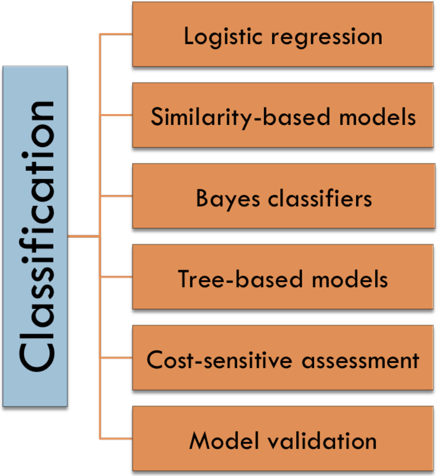
Course Roadmap



Course Roadmap



Prediction topics



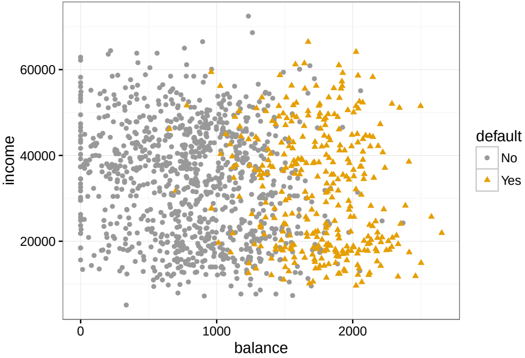
Prediction topics

- Classification
  - Logistic regression
  - Similarity-based models
  - Bayes classifiers
  - Tree-based models
  - Cost-sensitive assessment
  - Model validation

Agenda

- **What is Classification?**
- **Logistic Regression**
- **Nearest-Neighbours methods**
- **A brief introduction to Bayes Methods**

Some motivation: Default data



default	Has the customer defaulted on their debt? (values: Yes, No)
student	Is the customer a student? (values: Yes, No)
balance	Average credit card balance remaining after monthly payment
income	Income of customer

Goal: Use student, balance and income information to predict whether a customer is likely to default on their debts.

Classification

While introductory classes in statistics tend to focus on Prediction, most real world tasks are actually Classification problems.

- An online banking service needs to determine whether a particular transaction is fraudulent or not.
- A firm running a phone marketing campaign needs to identify which individuals would be most likely to adopt their product if contacted
- A law firm has obtained all the incoming and outgoing emails from a large corporation accused of discriminatory hiring and promotion practices. They must identify which documents are *responsive* (relevant to the case), and which are irrelevant.
- A judge deciding whether to release a defendant on bail may want to know the likelihood that the defendant will show up for their trial.
- A client using satellite imaging wants to classify patches of images according to whether they're water, rural, or urban.

What is the classification task?

- For the purpose of this discussion, we'll assume that we're doing binary classification: Our response Y has two possible values $\{0, 1\}$
- E.g., in the `Default` classification task,

$$Y = \begin{cases} 0 & \text{if } \texttt{No} \\ 1 & \text{if } \texttt{Yes} \end{cases}$$

- Our goal is to predict Y using the input variables `student`, `balance` and `income`

Is there an "ideal" classifier?

- When we talked about Prediction, our goal was to minimize the (test) Mean-Squared-Error, and we saw that the best we could ever do at modeling an outcome Y from inputs $X = (X_1, X_2, \ldots, X_p)$, is to use the regression function

$$f(x) = \mathbb{E}(Y \mid X = x)$$

- In Classification, our goal is now to minimize the (test) error rate:

$$\frac{1}{n} \sum_{i=1}^{n} I(y_i \neq \hat{y}_i)$$

- It turns out that the ideal classifier assigns a test observation with predictor values x_0 to the class j that has the *largest* value of:

$$\mathrm{P}(Y = j \mid X = x_0)$$

  i.e., the class that has the highest probability (is the most likely) given the observed predictor vector x_0.

- This classification rule is called the Bayes classifier

Ideal predictor vs. Ideal classifier

- The Bayes classifier in classification plays the same role as the regression function in prediction: They are they best we could ever hope to do, but are unknown functions that we seek to estimate from the data
- It is interesting to note that when Y is binary, $\mathbb{E}(Y \mid X = x) = \mathrm{P}(Y = 1 \mid X = x)$, so our target becomes

$$p(x) = \mathbb{E}(Y \mid X = x) = \mathrm{P}(Y = 1 \mid X = x)$$

  so by accurately estimating the *regression function* we would be able to mimic the Bayes classifier.
- We can think of our intermediate goal as one of constructing good estimators of the unknown true conditional probability function $p(x)$.[3]
- Once we have an estimator $\hat{p}(x)$, we can classify an observation by predicting `default = Yes` if $\hat{p}(x) > \alpha$ for some choice of α
  - We'll see in a future class why we may want to use $\alpha \neq 0.5$

[3]While there exist classification methods that *do not* proceed by first estimating $p(x)$, most methods do.

Is there an "ideal" classifier? Part II

- We just argued that we want to construct $\hat{p}(x)$ that are good estimators of the true conditional probability function

$$p(x) = \mathbb{E}(Y \mid X = x) = \mathrm{P}(Y = 1 \mid X = x)$$

- If we could do this, we could use $\hat{p}(x)$ to classify inputs according to the rule:

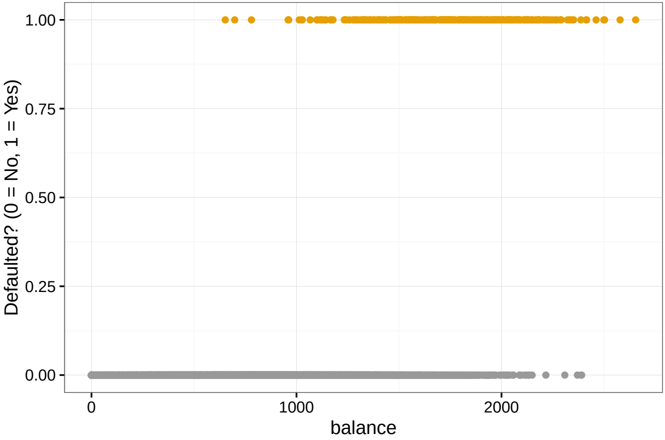
$$\hat{Y} = \begin{cases} 1 & \text{if } \hat{p}(x) > \alpha \\ 0 & \text{if } \hat{p}(x) <= \alpha \end{cases}$$

  $\alpha = 0.5$ is a popular choice, and is what the Bayes classifier uses.

- Now... suppose we define $\tilde{p}(x) = \frac{1}{10}\hat{p}(x)$
- The rule $\hat{p}(x) > \alpha$ is the same as $\tilde{p}(x) > \frac{1}{10}\alpha$, so both $\hat{p}(x)$ and $\tilde{p}(x)$ classify the same way
- Thus we can have estimators $\hat{p}(x)$ that result in good classification rules, but which aren't actually good estimators of $p(x)$

The Default data set

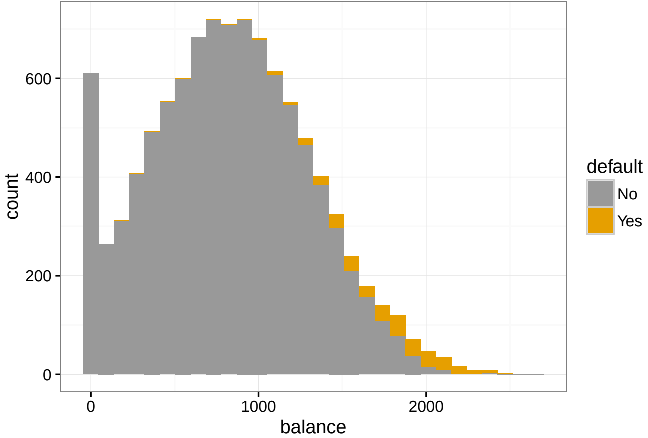
Let's look at a scatterplot of `default` (0 = `No`, 1 = `Yes`) vs. `balance`



This is really hard to make sense of. Since the outcome Y is binary, we don't get a good sense of the trends in the data from looking at scatterplots like this.

The Default data set

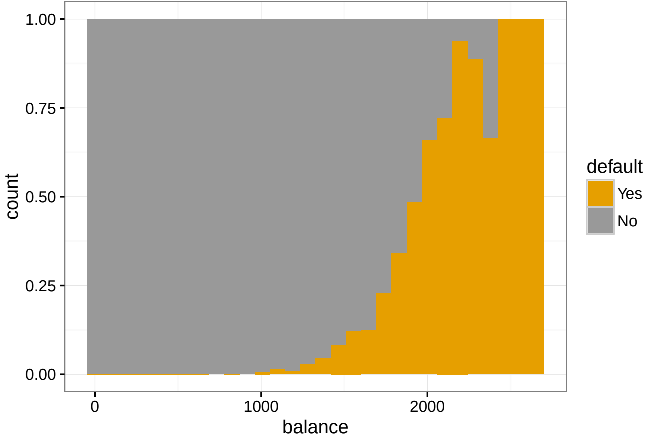
How about a histogram, where the bars are partitioned by `default` status?



That's a little better. It clearly shows that most of the individuals in our data set *did not* default. And it looks like the greater the average monthly balance an individual carries, the more likely they are to default.

The Default data set

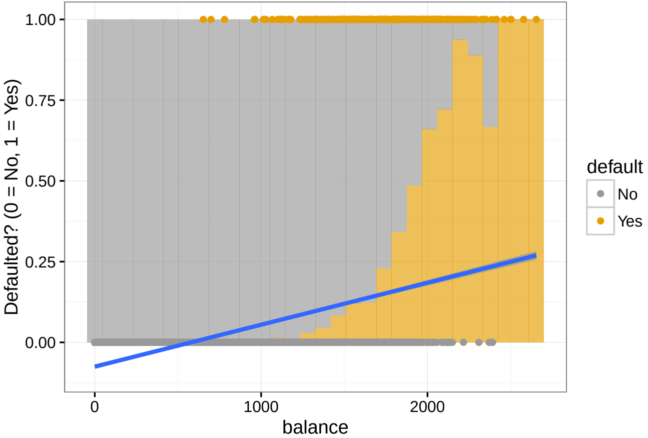
Here's what's called a conditional density plot. This plot is formed by first binning the x-axis variable (`balance`), and then calculating the probability of default within each bin.



Now we can clearly see what $\mathrm{P}(\texttt{default} = \texttt{Yes} \mid \texttt{balance})$ looks like

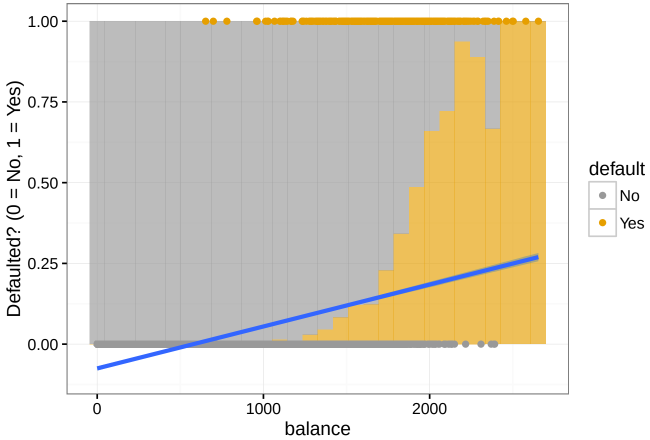
Does linear regression work?

We can run the `lm` command with any numeric choice of outcome variable Y, so what happens if we simply run linear regression here?



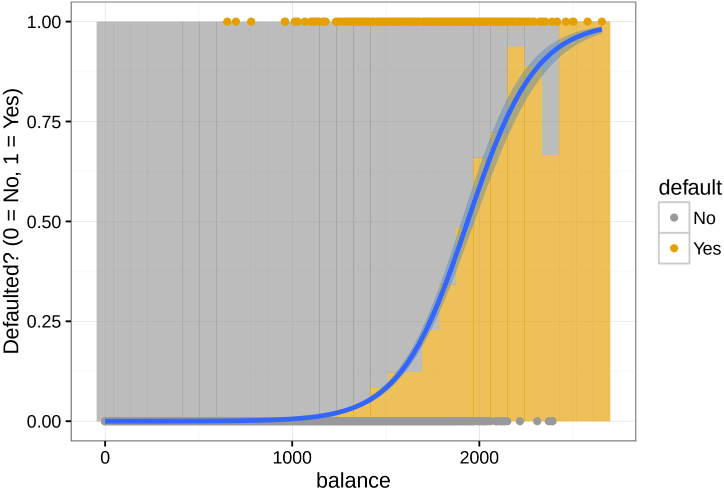
`lm(default ~ balance, data = Default)`

An issue with linear regression



- For balance < 500, the probability estimates are negative!
- Linear regression appears to do a poor job of estimating $p(x)$
- However, it's actually going to perform OK as a classifier. We'll see this when we talk about LDA.

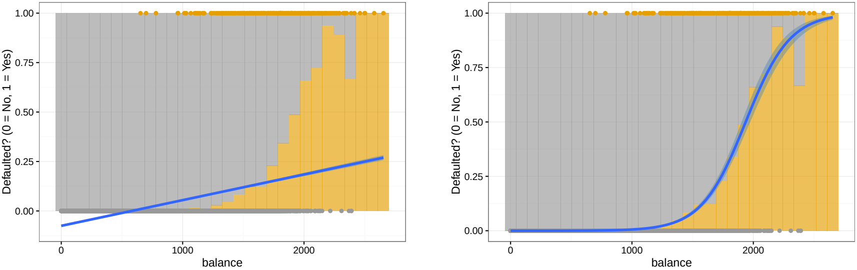
Logistic regression




```
glm(default ~ balance, family = binomial(), data = Default)
```

Much better!

Linear regression vs. Logistic regression



Linear regression | Logistic regression

- Logistic regression does a much better job of estimating the conditional probability of default
- However, the Linear regression fit may still produce good classifications.

How does logistic regression work?

- Logistic regression is an example of a Generalized Linear Model (GLM)
- Remember we have a variable Y that's 0 (if `default = No`), or 1 (if `default = Yes`)
- Instead of modeling

$$Y = \beta_0 + \beta_1 \texttt{balance} + \epsilon$$

  Logistic regression models
- The quantity on the LHS of the equation is called the log-odds of default.
- This turns out to be "right scale" on which to view things

How does logistic regression work?

- Logistic regression is an example of a Generalized Linear Model (GLM)
- Remember we have a variable Y that's 0 (if `default = No`), or 1 (if `default = Yes`)
- Instead of modeling

$$Y = \beta_0 + \beta_1 \texttt{balance} + \epsilon$$

  Logistic regression models

- The quantity on the LHS of the equation is called the log-odds of default.
- This turns out to be "right scale" on which to view things

How does logistic regression work?

- Logistic regression is an example of a Generalized Linear Model (GLM)
- Remember we have a variable Y that's 0 (if `default = No`), or 1 (if `default = Yes`)
- Instead of modeling

$$Y = \beta_0 + \beta_1 \texttt{balance} + \epsilon$$

  Logistic regression models

$$\log\left(\frac{\mathrm{P}(Y = 1 \mid \texttt{balance})}{1 - \mathrm{P}(Y = 1 \mid \texttt{balance})}\right) = \beta_0 + \beta_1 \texttt{balance}$$

- The quantity on the LHS of the equation is called the log-odds of default.
- This turns out to be "right scale" on which to view things

How does logistic regression work?

- Logistic regression is an example of a Generalized Linear Model (GLM)
- Remember we have a variable Y that's 0 (if `default = No`), or 1 (if `default = Yes`)
- Instead of modeling

$$Y = \beta_0 + \beta_1 \texttt{balance} + \epsilon$$

Logistic regression models

$$\log\left(\frac{\mathrm{P}(\texttt{default} = \texttt{Yes} \mid \texttt{balance})}{1 - \mathrm{P}(\texttt{default} = \texttt{Yes} \mid \texttt{balance})}\right) = \beta_0 + \beta_1 \texttt{balance}$$

- The quantity on the LHS of the equation is called the log-odds of default.
- This turns out to be "right scale" on which to view things

What is logistic regression doing?

$$\log\left(\frac{\mathrm{P}(\texttt{default} = \texttt{Yes} \mid \texttt{balance})}{1 - \mathrm{P}(\texttt{default} = \texttt{Yes} \mid \texttt{balance})}\right) = \beta_0 + \beta_1 \texttt{balance}$$

- Logistic regression provides us with coefficient estimates $\hat{\beta}_0$ and $\hat{\beta}_1$
- By rearranging the equation $r = \log(p/(1-p))$ to get $p = e^r/(1+e^r)$, we can get an estimate of the conditional probability of default:

$$\hat{\mathrm{P}}(\texttt{default} = \texttt{Yes} \mid \texttt{balance}) = \frac{e^{\hat{\beta}_0 + \hat{\beta}_1 \texttt{balance}}}{1 + e^{\hat{\beta}_0 + \hat{\beta}_1 \texttt{balance}}}$$

- i.e., Logistic regression gives us a way of estimating the probability that an individual will `default` given their average monthly `balance`.

Making predictions with logistic regression

Using the command

```
glm(default ~ balance, data = Default, family = binomial())
```

we get

	Coefficient	Std. Error	Z-statistic	P-value
`Intercept`	-10.6513	0.3612	-29.5	< 0.0001
`balance`	0.0055	0.0002	24.9	< 0.0001

- Thus we can predict that the default probability for an individual with an average `balance` of \$$1{,}000$ is

$$\hat{p}(1000) = \frac{e^{\hat{\beta}_0 + \hat{\beta}_1 \texttt{balance}}}{1 + e^{\hat{\beta}_0 + \hat{\beta}_1 \texttt{balance}}} = \frac{e^{-10.6513 + 0.0055 \times 1000}}{1 + e^{-10.6513 + 0.0055 \times 1000}} = 0.00576$$

  this is 0.576%.

- For an individual with a balance of \$$2{,}000$, the estimated probability of default winds up being 0.586, or 58.6%!

Interpreting Logistic regression coefficients

- We like to use linear regression for prediction because even if the model is likely to be wrong, it's typically interpretable

- Logistic regression is an example of an interpretable classifier

- Recall that the logistic model says

$$\mathrm{logOdds}(Y = 1) = \beta_0 + \beta_1 X_1$$

- This says that for every unit increase in X_1, the log-odds that $Y = 1$ increase by β_1

Interpreting Logistic regression coefficients: Odds ratios

- The log-odds scale is a bit awkward for interpretation, so we typically like to *exponentiate* both sides of the model and interpret that equation instead:

$$\exp(\text{logOdds}(Y = 1 \mid X_1 = x)) = \frac{\mathrm{P}(Y = 1 \mid X_1 = x)}{1 - \mathrm{P}(Y = 1 \mid X_1 = x)} = e^{\beta_0 + \beta_1 x}$$

- What happens if we increment X_1 by one unit?

$$\underbrace{e^{\beta_0 + \beta_1 (x+1)}}_{\text{odds that } Y = 1 \text{ when } X_1 = x+1} = \underbrace{e^{\beta_0 + \beta_1 x}}_{\text{odds that } Y = 1 \text{ when } X_1 = x} e^{\beta_1}$$

- This says that for every unit increase in X_1, the odds that $Y = 1$ increase by a factor of e^{β_1}
- So if X_1 goes up by 3 units, the odds that $Y = 1$ increase by $e^{3\beta_1}$.
- The exponentiated coefficient e^{β_1} is often called the odds ratio (OR)

Interpreting the coefficient of balance

	Coefficient	Std. Error	Z-statistic	P-value
`Intercept`	-10.6513	0.3612	-29.5	< 0.0001
`balance`	0.0055	0.0002	24.9	< 0.0001

- In this example, $e^{\hat{\beta}_1} = e^{0.0055} = 1.0055$
- Thus we can interpret $\hat{\beta}_1$ as saying that for every additional \$$1$ of `balance`, the individual's odds of defaulting on their loans go up by a factor of 1.0055 (i.e., odds of default increase by 0.55%)
- For every additional \$$100$ of `balance`, the odds of defaulting increase by a factor of $e^{100\times 0.0055} = 1.733$ (i.e., odds of default increase by 73.3%)

Logistic regression with multiple predictors

- Notation: $p(X) = \mathrm{P}(Y = 1 \mid X)$, where X can be one or more predictors

- When we have p predictors $X_1, \ldots, X_p$, the logistic model becomes

$$p(X) = \frac{e^{\beta_0 + \beta_1 X_1 + \cdots \beta_p X_p}}{1 + e^{\beta_0 + \beta_1 X_1 + \cdots \beta_p X_p}}$$

- Now we get the interpretation: All other predictors in the model held fixed, for every unit increase in X_j the log-odds that $Y = 1$ go up by β_j

```
glm(default ~ student, data = Default, family = binomial())
```

	Coefficient	Std. Error	Z-statistic	P-value
`Intercept`	-10.6513	0.3612	-29.5	< 0.0001
`studentYes`	0.4049	0.1150	3.52	0.0004

- `student` is an indicator of whether the individual is a student.
- The logistic regression tells us that the odds that a *student* defaults are $e^{0.4049} = 1.50$ times higher than for a non-student
- So if we're trying to figure out whose credit card applications to approve, this suggests that we should **not** extend credit to students
- *But wait...*

Confounding

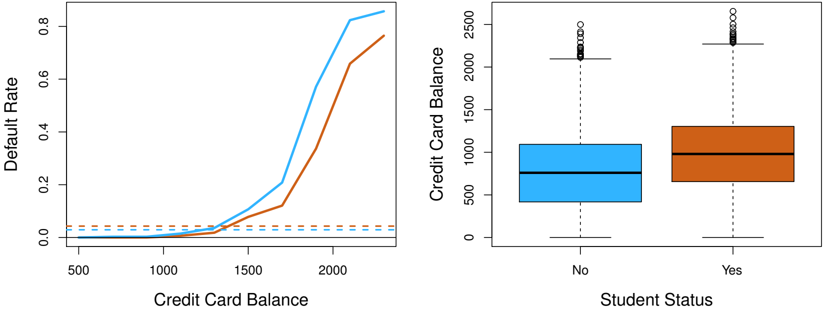


Figure: 4.3 from ISL. Confounding in the `Default` data Left: Default rates for students (orange) and non-students (blue). Solid lines display the default rate as a function of `balance`. Horizontal broken lines display the overall default rates within each group. Right: Boxplots of `balance` for students and non-students

- The *overall* default rate is higher for students than non-students
- *But* if we compare students to a non-students *with the same* `balance`, students have a *lower* default rate

Default data: Logistic regression

	Coefficient	Std. Error	Z-statistic	P-value
`Intercept`	-10.8690	0.4923	-22.08	<0.0001
`balance`	0.0057	0.0002	24.74	< 0.0001
`income`	0.0030	0.0000	0.37	0.7115
`studentYes`	-0.6468	0.2363	-2.74	0.0062

- This says that if we compare a student to a non-student with the same `balance` and `income`, the odds that the *student* defaults are $e^{-0.6468} = 0.523$ times those of the *non-student*.

- $1/0.523 = 1.9$, so we can equivalently say: the odds of the *student* defaulting are 1.9 times lower than the odds of the *non-student* defaulting

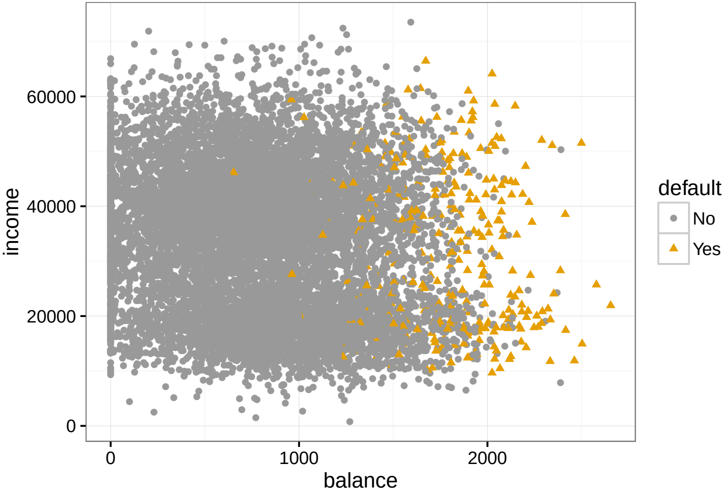
Logistic regression as a classifier

- We now know how to *interpret* logistic regression
- But what does it look like as a classifier?
- i.e., What does the decision rule

$$\hat{Y} = \begin{cases} 1 & \text{if } \frac{e^{\hat{\beta}_0 + \hat{\beta}_1 \texttt{balance} + \hat{\beta}_2 \texttt{income}}}{1 + e^{\hat{\beta}_0 + \hat{\beta}_1 \texttt{balance} + \hat{\beta}_2 \texttt{income}}} > \alpha \\ 0 & \text{otherwise} \end{cases}$$

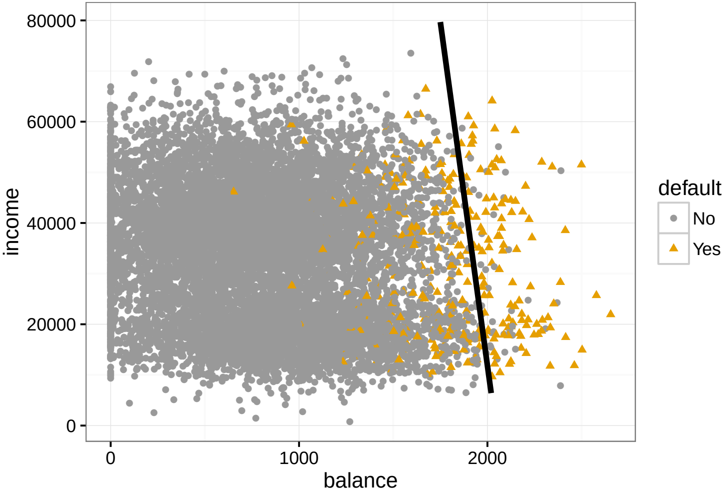
actually look like for various choices of the cutoff α?

Logistic regression: Decision boundary



This is what the data actually looks like (previous scatterplot *undersampled* the `default = No` group.)

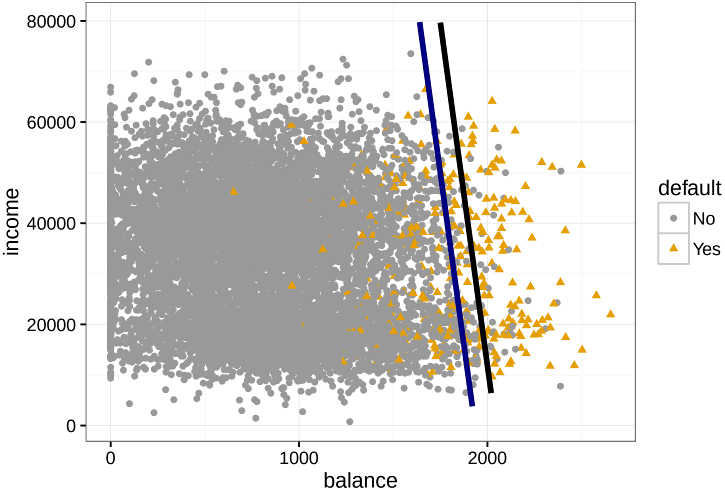
Logistic regression: Decision boundary



Black line shows decision boundary for the rule: $\hat{Y} = 1$ if $\hat{p}(x) > 0.5$

Points to the right of the boundary get classified as `default = Yes`

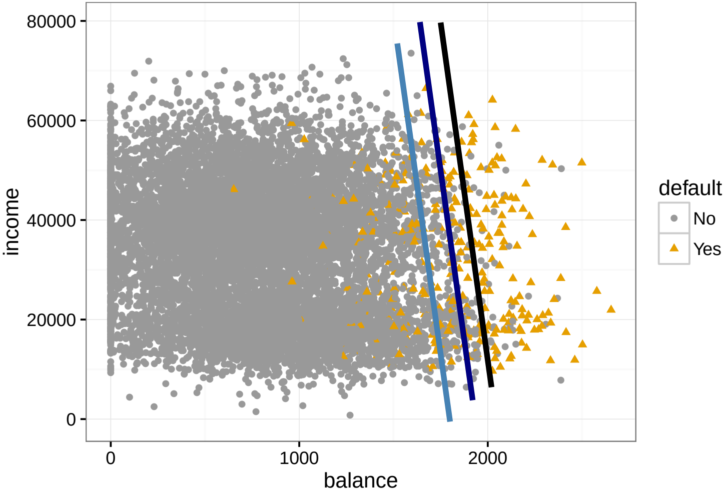
Logistic regression: Decision boundary



Navy line shows decision boundary for the rule: $\hat{Y} = 1$ if $\hat{p}(x) > 0.35$

Points to the right of the boundary get classified as `default = Yes`

Logistic regression: Decision boundary



Blue line shows decision boundary for the rule: $\hat{Y} = 1$ if $\hat{p}(x) > 0.2$

Points to the right of the boundary get classified as `default = Yes`

More general decision boundaries

- We see that the decision boundary provided by logistic regression turns out to be linear:
  - Points on one side of the boundary get classified as $\hat{Y} = 1$
  - Points on the other side get classified as $\hat{Y} = 0$

- When we have $p > 2$ covariates, instead of getting a line, we get a (hyper)-plane

- Just as we don't think that linear models are accurate representations of numeric outcomes, we may not believe that a linear decision boundary is the best way to classify points

- Let's look at one method that results in non-linear decision boundaries

k-Nearest Neighbours Classifier

- Setup: Data (x_i, y_i), $x_i \in \mathbb{R}^p$ vector of inputs, $y_i \in \{0, 1\}$.
- k-Nearest Neighbours (k-NN) classification is an example of a non-parametric "lazy learning" (memory-based) method
- Unlike the methods we've seen before[4], which estimate parameters in some model, k-NN looks at the training data each time it is queried to make a classification[5]
- Let $\mathcal{N}_k(x)$ denote the k training points that are *closest* to x
- If we want to classify a new individual with covariates $X = x$, we simply classify it to the majority class of the points in $\mathcal{N}_k(x)$
- As an estimator of the conditional probability, we can think of k-NN as

$$\hat{p}_{\mathrm{kNN}}(x) = \frac{1}{k} \sum_{x_i \in \mathcal{N}_k(x)} y_i$$

[4]With the exception of local regression, which also has these properties.

[5]k-NN also works for prediction, though we didn't discuss it at the time

k-Nearest Neighbours Classifier: 3-NN example

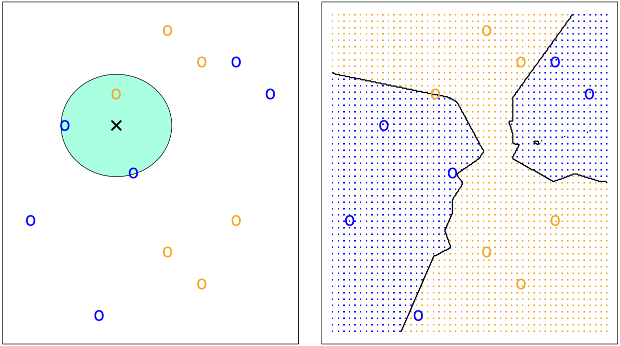


Figure: 2.14 from ISL. Axes represent $p = 2$ predictors.

- *Left*: We have two classes: $Y \in \{$orange, blue$\}$. 3-NN is used to classify the $\times$ as a blue point. 2 of $\times$'s 3 nearest neighbours are blue, one is orange, so blue wins.
- *Right*: The decision boundary. Any point in the orange shaded region gets classified by 3-NN as orange. Any point in the blue shaded region area gets classified by 3-NN as blue.

k-Nearest Neighbours: Simulated data

- When we discussed prediction, we relied a lot on simulation experiment to see how well our estimates $\hat{f}(x)$ approximated the true regression function $f(x)$
- The next few slides demonstrate how well k-NN works on a simulated data set where the true Bayes classifier decision boundary is non-linear

k-Nearest Neighbours: Simulated data

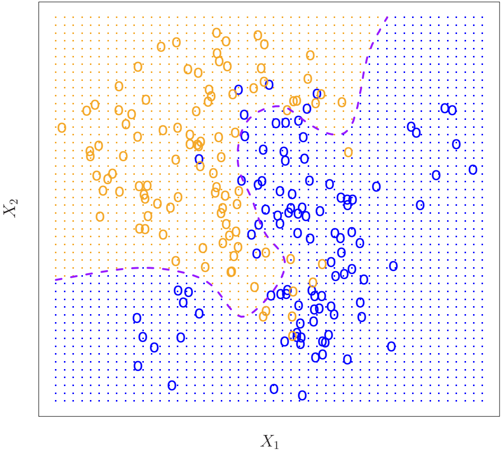


Figure 2.15 from ISL. Dashed purple line is the Bayes classifier “optimal” decision boundary.

k-Nearest Neighbours: Simulated data

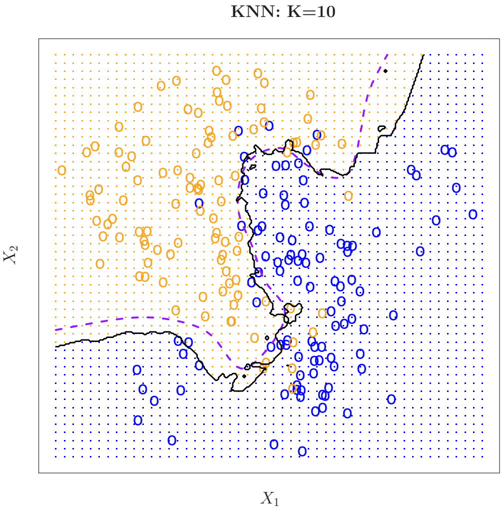


Figure 2.15 from ISL. Dashed purple line is the Bayes classifier "optimal" decision boundary. Solid black line is 10-NN decision boundary.

k-Nearest Neighbours: Simulated data

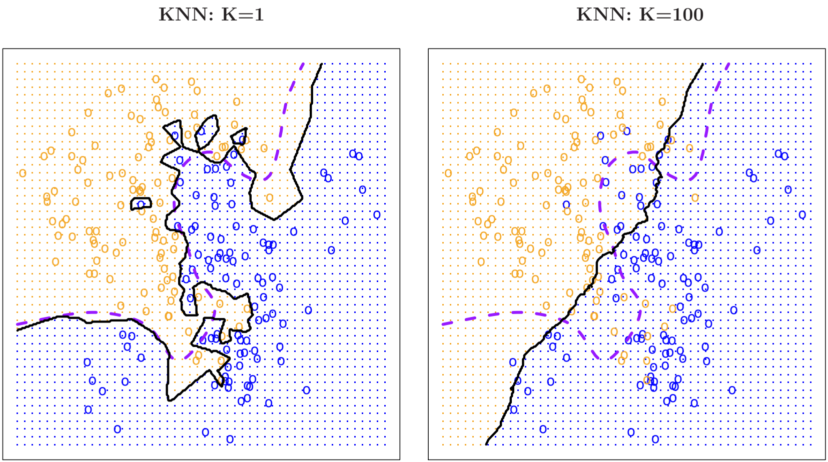


Figure 2.15 from ISL. Dashed purple line is the Bayes classifier “optimal” decision boundary. Solid black lines are 1-NN and 100-NN classifiers.

Can we get non-linear boundaries from Logistic regression?

- When we discussed Linear models, we saw that we could easily extend them to Additive models and Regularized regression models
- The same is true of Logistic regression
- To fit Additive Logistic models:
  Specify `family = binomial()` in a `gam()` command
- To fit ℓ-1 regularized Logistic regression:
  Specify `family = "binomial"` in a `glmnet` command
- Indeed, Google famously uses *massive* regularized logistic regressions as a core component of their AdClick prediction system. This system is trained on many *billions* of observations, with potentially *millions* of predictors

Multi-class classification

- The examples we have considered thus far have all been cases of binary classification
- Y could take on one of *two* possible values: $\{0, 1\}$, {orange, blue}, etc.
- Going forward, it will be just as straightforward to consider the multi-class case where we have $J \geq 2$ classes
- Our goal is to use inputs $X = x$ to classify Y to one of $\{1, 2, \ldots, J\}$, seeing to minimize the (test) error rate

$$\frac{1}{n} \sum_{i=1}^{n} I(y_i \neq \hat{y}_i)$$

- As we said earlier, the best *any classifier whatsoever* could do is to classify to the group j that has the largest value of

$$\mathrm{P}(Y = j \mid X = x)$$

Multi-class classification

- We can now think of our intermediate goal as getting good estimates of the conditional probability functions

$$\mathrm{P}(Y = j \mid X = x)$$

- There is a multi-class version of Logistic regression, which typically goes by the name Multinomial regression

- We will not discuss this method here, because it's not a particularly popular approach

- For k-NN, there's really no generalization that needs to take place to go from $J = 2$ classes to $J \geq 2$ classes
    - We find that k closest points to x, and classify to the plurality class: the class that appears most often among x's k nearest neighbours

Multi-class classification: Generative models

- Logistic regression is an example of what is called a Discriminative model
- Such models estimate $\mathrm{P}(Y = j \mid X = x)$, but do not try to model the *joint distribution* between the inputs and the response Y
- Since the Bayes classifier itself only uses $\mathrm{P}(Y = j \mid X = x)$, it may seem that to model anything beyond that is *too much work* and seemingly *unnecessary*
- However, there are some nice classification methods that *do* proceed by modeling the joint distribution of $(Y, X_1, \ldots, X_p)$.
- Such methods are called Generative models

Generative models: A toy example

- We will go into more detail on Generative models next class

- To whet your appetite, let's look at a simple toy example

- Canadians and Americans keep insisting that they're "different people", but you can't tell them apart

- Lately, though, you've noticed that Canadian men tend to be somewhat *shorter* than American men

- Sure enough, some internet browsing reveals that:
  - American men are on average 178.2cm tall (St Dev = 4.7cm)
  - Canadian men are on average 173cm tall (St Dev = 3.3cm)
  - Heights are generally Normally distributed

Canadian or American?

- Neil, the next "American sounding" guy you come across, turns out to be 171cm tall.
- **Question**: Is Neil American or Canadian?
- Let H denote a man's height. From your internet search, you found that:

$$H \mid \text{American} \sim \text{Normal}(178.2, \sigma = 4.7)$$
$$H \mid \text{Canadian} \sim \text{Normal}(173, \sigma = 3.3)$$

- But we want: $\text{P}(\text{American} \mid H = 171)$...Hmmm...
- Luckily an 18th century Presbyterian minister tells you that

$$\text{P}(\text{American} \mid H = 171) = \frac{\text{P}(H = 171 \mid \text{American})\text{P}(\text{American})}{\text{P}(H = 171)}$$

$$\mathrm{P}(\text{American} \mid H = 171) = \frac{\mathrm{P}(H = 171 \mid \text{American})\mathrm{P}(\text{American})}{\mathrm{P}(H = 171)}$$

- Since $H \mid \text{American} \sim \text{Normal}(178.2, \sigma = 4.7)$, we can calculate that

$$\mathrm{P}(H = 171 \mid \text{American}) = 0.026$$

- How about $\mathrm{P}(\text{American})$? This is the proportion of "American sounding" males on campus who are actually American. You do more research, and find that this proportion is 0.85.

- Finally, the *Law of Total Probability* tells us that

$$\begin{aligned}\mathrm{P}(H = 171) &= \mathrm{P}(H = 171 \mid A) \cdot \mathrm{P}(A) + \mathrm{P}(H = 171 \mid C) \cdot \mathrm{P}(C) \\ &= 0.026 \cdot 0.85 + 0.10 \cdot 0.15 \\ &= 0.0371\end{aligned}$$

- So according to the minister's rule, $\mathrm{P}(\text{American} \mid H = 171) = 0.026 \times 0.85/0.0371 = 0.60 = 60\%$

- So chances are, Neil is just a short American!

Acknowledgements

All of the lectures notes for this class feature content borrowed with or without modification from the following sources:

- 36-462/36-662 Lecture notes (Prof. Tibshirani, Prof. G'Sell, Prof. Shalizi)
- 95-791 Lecture notes (Prof. Dubrawski)
- *An Introduction to Statistical Learning, with applications in R* (Springer, 2013) with permission from the authors: G. James, D. Witten, T. Hastie and R. Tibshirani
- *Applied Predictive Modeling*, (Springer, 2013), Max Kuhn and Kjell Johnson