Classification 2

Lecture 8

Connor Dowd

April 22nd, 2021

Today's Class

1. Review
   - KNN
   - Binomial Classification
   - ROC Curve
2. Multinomial Regression
3. HW3 solns/discussion
4. HW4

Quick Review

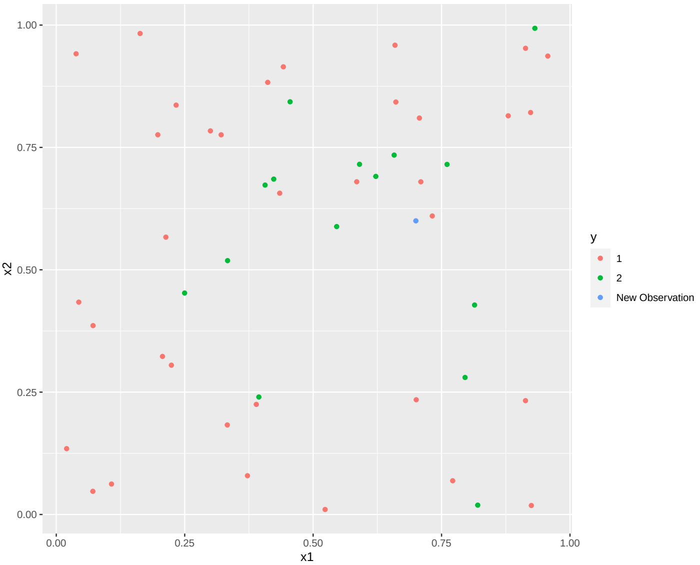
KNN Basics

Basic Idea: Estimate $P[y|x]$ locally using the labels of similar observations in the training data.

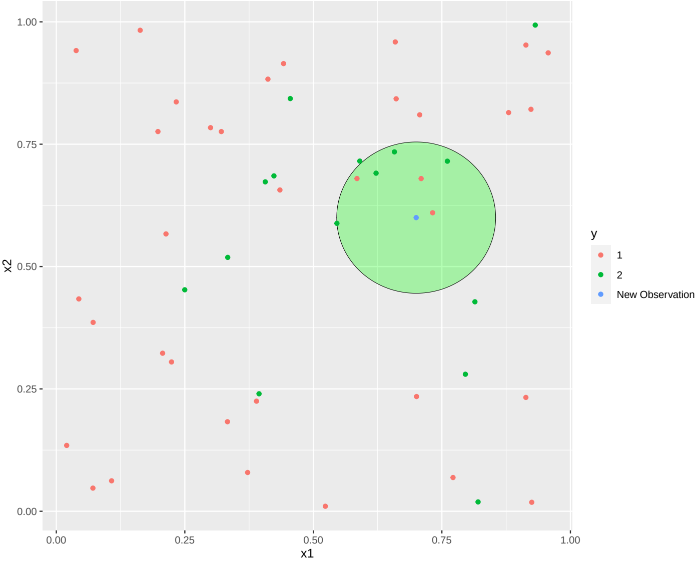
KNN: What is the most common class near x^{new}?

1. Take the K nearest neighbors $x_{i,1}, ..., x_{i,K}$ of x^{new} in the training data
   - Nearness is (usually) Euclidean distance: $\sqrt{\sum_{j=1}^{p}(x_j^{new} - x_{i,k,j})^2}$
2. Estimate $P[y = j|x] = \frac{1}{n}\sum_{i=1}^{K} 1(y_i = j)$
3. Select the class j with the highest probability.

KNN Example Data

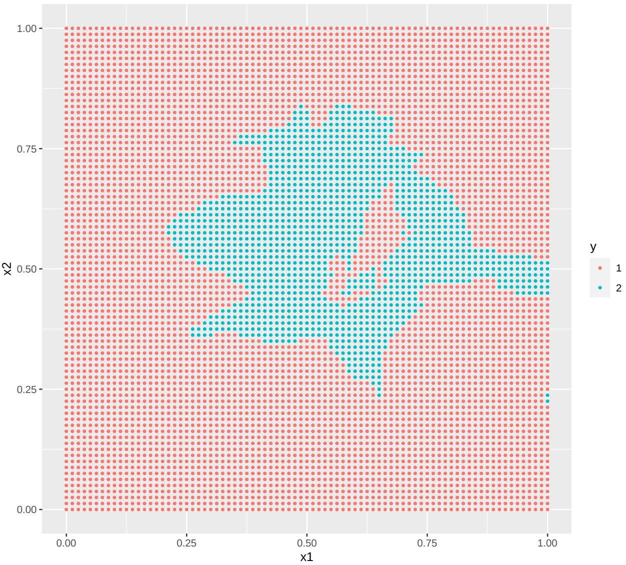


KNN Example $K = 7$



The relative ‘vote counts’ are a very crude estimate of probability.

KNN Example



Binary Classification

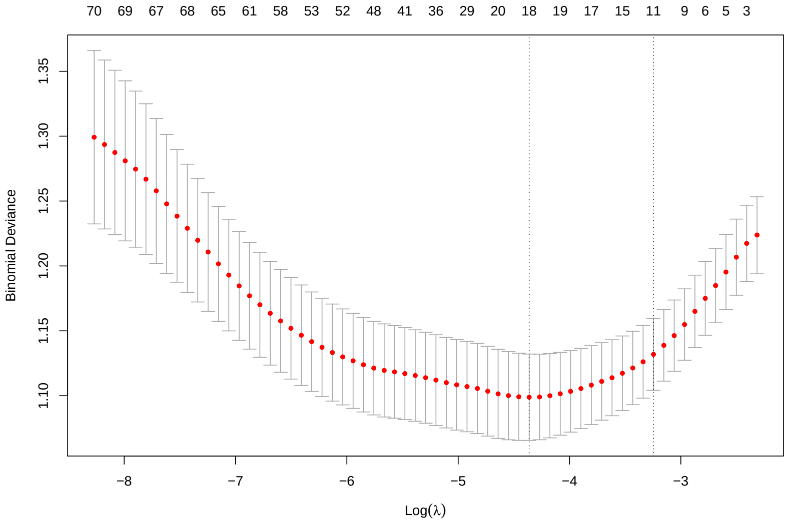
Many problems can be reduced to binary classification (as above).

KNNs are a useful non-parametric classification tool.

Logits are a useful parametric classification tool.
- Remember Spam?

- Logits yield parametric decision boundaries. Easy to interpret.
- Logits are global methods. Use *all* the training data to inform predictions.
  - The probability estimates are more useful and more stable.
- Logits can do variable selection.

Credit Classification Example



Sensitivity and Specificity

But we may also want to think about sensitivity and specificity.

- Sensitivity: proportion of true $y = 1$ classified as such.
- Specificity: proportion of true $y = 0$ classified as such.

A rule is sensitive if it mostly gets the 1s right. A rule is specific if it mostly gets the 0s right.

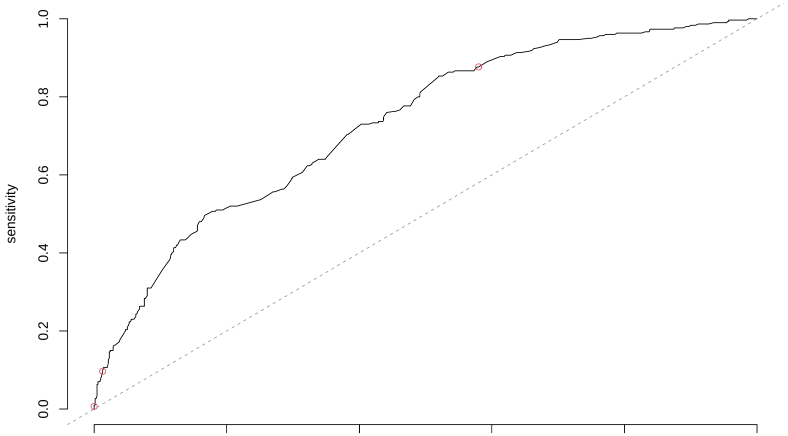
Error Types

		Decision		
		Fail to Reject	*Reject*	*Sum*
Truth	*Noise*	Real non-Discovery (TN)	False Discovery (FD)	N_0
	Signal	Missed Discovery. (FN)	Real Discovery (TD)	N_1
	Sum	p-R	R	

ROC Curve

We can plot the ROC curve for different choices of threshold.

```
## Warning in plot.xy(xy.coords(x, y), type = type, ...): '
## parameter
```

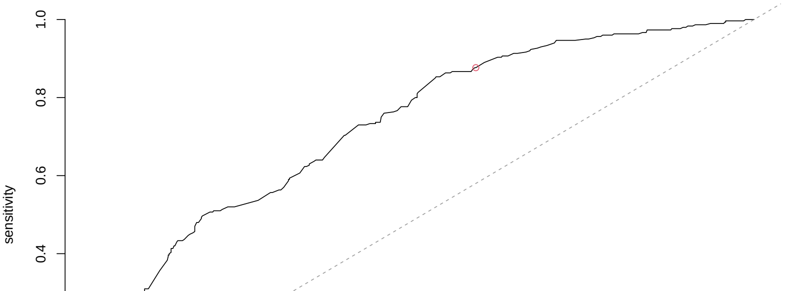


AUC

The AUC is a common metric for choosing between classifiers, which selects the classifier that maximizes the Area Under the Curve – specifically the area under the ROC curve.

```
roc(p=pred,y=default,bty="n")
```

```
## Warning in plot.xy(xy.coords(x, y), type = type, ...): '
## parameter
```

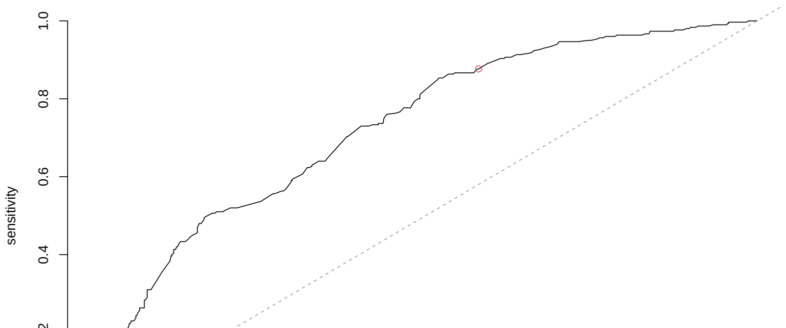


ROC & Decisions

We face a tradeoff with differential costs? Find where the slope of the ROC equates those costs.

```
roc(p=pred,y=default,bty="n")
```

```
## Warning in plot.xy(xy.coords(x, y), type = type, ...): '
## parameter
```



Multinomial

Multiple Options

What if we have more than one class possible?

E.g. Not just $\{0, 1\}$ but $\{0, 1, 2, ..., M\}$

KNN Still works

We can still just ask which category is most likely in the group nearest to a point. Though our code may need to change.

(Recommended optional long Q in HW4)

Multinomial Logit

We can also switch to a multinomial logit. At its simplest, this combines M models of $P[Y = 1|x]$, $P[Y = 2|x]$,...,$P[Y = M|x]$, in a manageable manner.

We can estimate each of those models as a simple logit – to get their coefficients.

Then we make sure that our probabilities sum to 1.

$$\sum_{j=1}^{M} P[Y = j|x] = 1$$

Multinomial Logit

Our full model then is something different:

$$P[Y_i = k|\mathbf{x}_i] = p_{ik} = \frac{e^{\mathbf{x}_i'\beta_k}}{\sum_{j=1}^{M} e^{\mathbf{x}_i'\beta_j}}$$

NOTICE there are different coefficients for each class K

Multinomial Logit

This will change our likelihood slightly. Denoting by k_i the class of y_i, we get:

$$L(\beta_1, ..., \beta_M) \propto \prod_{i=1}^{n} p_{ik_i}$$

Which implies the deviance is

$$Dev(\beta_1, ..., \beta_M) \propto -2 \sum_{i=1}^{n} log(p_{ik_i})$$

Multinomial

Now we can try to minimize the deviance (standard multinomial) or do some penalized version of this (make sure you sum across all the coefficients!)

Glass Example

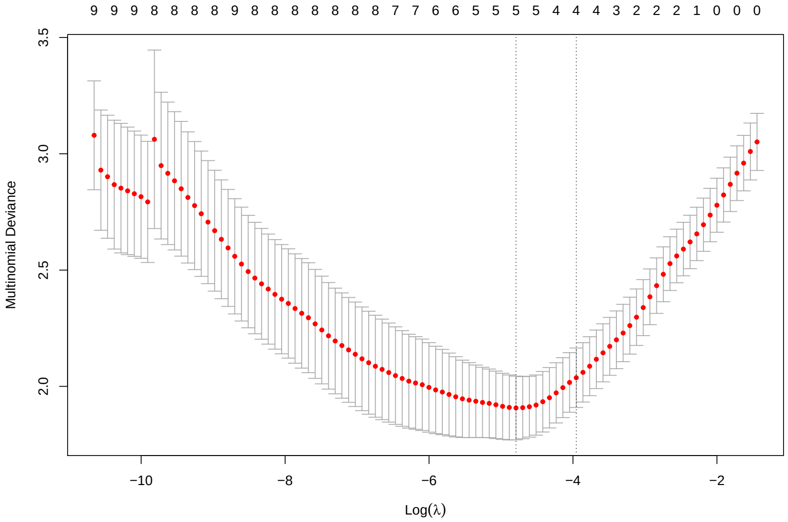
We have shards of broken glass, and we want to determine what of a few common types of glass it is.

- Headlight
- Vehicle Window
- Standard Window (floating or not)
- Container (e.g. glass bottle)
- Tableware (e.g. a glass)

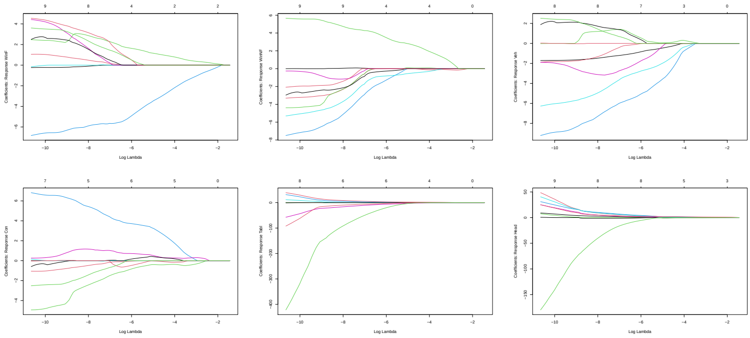
This is where a multinomial regression may be useful.

Glass Example

```
glassfit <- cv.glmnet(xfgl, gtype, family="multinomial")
```



Multinomial



Wrap up

Things to do

HW 3 is due next Wednesday night.

See you Tuesday

Rehash

- ROC curves are powerful tools
- Multinomial regressions are very possible

Bye!