Lecture 8: Classification

Reading: Chapter 4

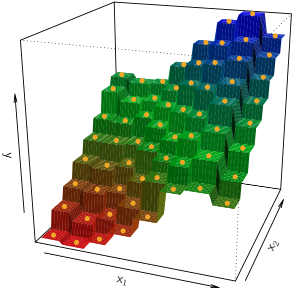
STATS 202: Data mining and analysis

October 9, 2019

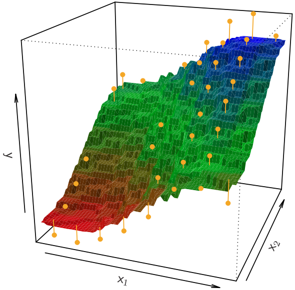
Comparing Linear Regression to K-nearest neighbors

Linear regression: prototypical parametric method.
KNN regression: prototypical nonparametric method.

$$\hat{f}(x) = \frac{1}{K} \sum_{x_i \in N_K(x)} y_i$$



$K = 1$



$K = 9$

Comparing Linear Regression to K-nearest neighbors

Linear regression: prototypical parametric method.

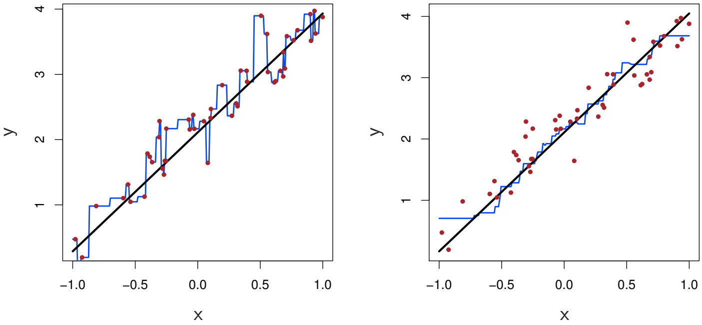
KNN regression: prototypical nonparametric method.

Long story short:

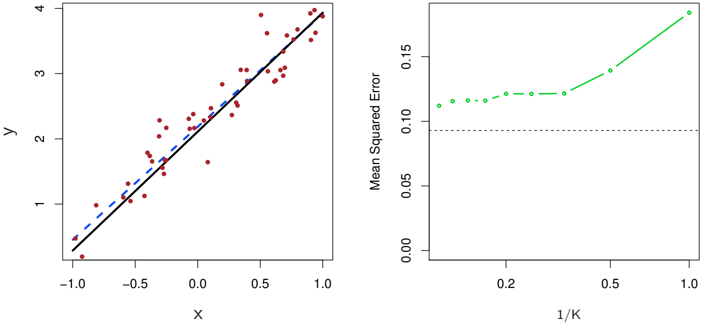
- KNN is only better when the function f is not linear.
- When n is not much larger than p, even if f is nonlinear, Linear Regression can outperform KNN. KNN has smaller bias, but this comes at a price of higher variance.

KNN estimates for a simulation from a linear model

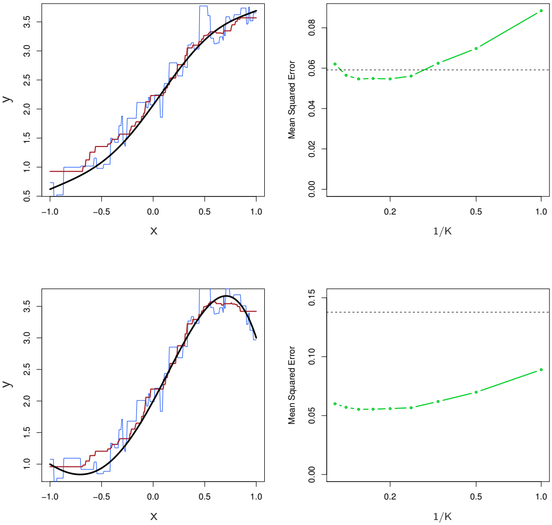
$K = 1$ $\qquad$ $K = 9$



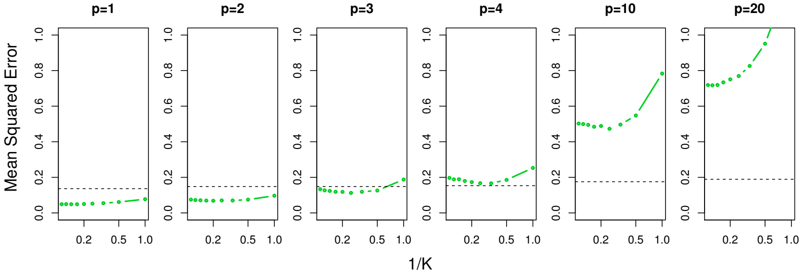
Linear models dominate KNN when true model linear



Increasing deviations from linearity



When the number of predictors is large, Linear Regression can dominate KNN



When p sufficiently large, the nearest neighbors are not especially near, and KNN accuracy can break down. This is known as the *curse of dimensionality*.

Classification problems

Supervised learning with a **qualitative or categorical** response.

Just as common, if not more common than regression:

- *Medical diagnosis:* Given the symptoms a patient shows, predict which of 3 conditions they are attributed to.
- *Online banking:* Determine whether a transaction is fraudulent or not, on the basis of the IP address, client's history, etc.
- *Web searching:* Based on a user's history, location, and the string of a web search, predict which link a person is likely to click.
- *Online advertising:* Predict whether a user will click on an ad or not.

Review: Bayes classifier

Suppose $P(Y \mid X)$ is known. Then, given an input x_0, we predict the response

$$\hat{y}_0 = \text{argmax}_y \; P(Y = y \mid X = x_0).$$

This Bayes classifier minimizes the expected 0-1 loss:

$$E\left[\frac{1}{m}\sum_{i=1}^{m} \mathbf{1}(\hat{y}_i \neq y_i)\right]$$

The minimum expected 0-1 loss (the best we can hope for) is the **Bayes error rate** $1 - E[\text{argmax}_y \; P(Y = y|X)]$. It is the analogon of the irreducible error in regression.

Strategy: estimate $P(Y \mid X)$

If we have a good estimate for the conditional probability $\hat{P}(Y \mid X)$, we can use the classifier:

$$\hat{y}_0 = \text{argmax}_{\,y}\ \hat{P}(Y = y \mid X = x_0).$$

Suppose Y is a binary variable. Could we use a linear model?

$$P(Y = 1|X) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$$

Problems:

- This would allow probabilities <0 and >1.
- Difficult to extend to more than 2 categories.

Logistic regression

We model the joint probability as:

$$P(Y = 1 \mid X) = \frac{e^{\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p}}{1 + e^{\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p}},$$

$$P(Y = 0 \mid X) = \frac{1}{1 + e^{\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p}}.$$

This is the same as using a linear model for the log odds:

$$\log \left[\frac{P(Y = 1 \mid X)}{P(Y = 0 \mid X)} \right] = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p.$$

Fitting logistic regression

The training data is a list of pairs $(y_1, x_1), (y_2, x_2), \ldots, (y_n, x_n)$. In the linear model

$$\log \left[\frac{P(Y = 1 \mid X)}{P(Y = 0 \mid X)} \right] = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p,$$

we don't observe the left hand side.

We cannot use a least squares fit.

Fitting logistic regression

Solution:

The likelihood is the probability of the training data, for a fixed set of coefficients $\beta_0, \ldots, \beta_p$:

$$\prod_{i=1}^{n} P(Y = y_i \mid X = x_i)$$

$$= \underbrace{\prod_{i;y_i=1} \frac{e^{\beta_0+\beta_1 x_{i1}+\cdots+\beta_p x_{ip}}}{1+e^{\beta_0+\beta_1 x_{i1}+\cdots+\beta_p x_{ip}}}}_{\text{Probability of responses} = 1} \underbrace{\prod_{j;y_j=0} \frac{1}{1+e^{\beta_0+\beta_1 x_{j1}+\cdots+\beta_p x_{jp}}}}_{\text{Probability of responses} = 0}$$

- Choose estimates $\hat{\beta}_0, \ldots, \hat{\beta}_p$ which maximize the likelihood.
- Solved with numerical methods (e.g. Newton's algorithm).

Logistic regression in R

```
> glm.fit=glm(Direction~Lag1+Lag2+Lag3+Lag4+Lag5+Volume,
    data=Smarket,family=binomial)
> summary(glm.fit)

Call:
glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5
    + Volume, family = binomial, data = Smarket)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
 -1.45   -1.20    1.07    1.15    1.33

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.12600    0.24074   -0.52     0.60
Lag1        -0.07307    0.05017   -1.46     0.15
Lag2        -0.04230    0.05009   -0.84     0.40
Lag3         0.01109    0.04994    0.22     0.82
Lag4         0.00936    0.04997    0.19     0.85
Lag5         0.01031    0.04951    0.21     0.83
Volume       0.13544    0.15836    0.86     0.39
```

Logistic regression in R

- We can estimate the Standard Error of each coefficient.
- The z-statistic is the equivalent of the t-statistic in linear regression:

$$z = \frac{\hat{\beta}_j}{\mathsf{SE}(\hat{\beta}_j)}.$$

- The p-values are test of the null hypothesis $\beta_j = 0$ (Wald's test).
- Other possible hypothesis tests: likelihood ratio test (chi-square distribution) is useful for testing whether groups of variables have coefficients equal to 0.

Example: Predicting credit card `default`

Predictors:

- `student`: 1 if student, 0 otherwise.
- `balance`: credit card balance.
- `income`: person's income.

In this dataset, there is *confounding*, but little collinearity.

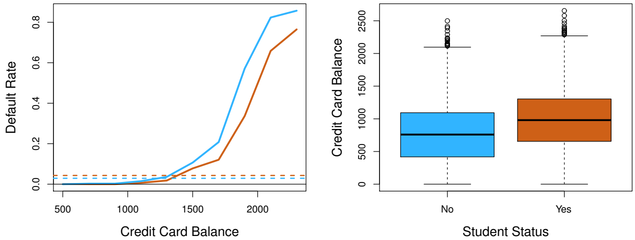
- Students tend to have higher balances. So, `balance` is explained by `student`, but not very well.
- People with a high `balance` are more likely to default.
- Among people with a given `balance`, students are less likely to default.

Example: Predicting credit card `default`

Predictors: `student` (yes/no), (credit card) `balance`, `income`

In this dataset, there is *confounding*, but little collinearity.

- Students tend to have higher balances. So, `balance` is explained by `student`, but not very well.
- People with a high `balance` are more likely to default.
- Among people with a given `balance`, students are less likely to default.



Example: Predicting credit card `default`

Logistic regression using only `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

Logistic regression using only `student`:

	Coefficient	Std. error	Z-statistic	P-value
`Intercept`	−3.5041	0.0707	−49.55	<0.0001
`student[Yes]`	0.4049	0.1150	3.52	0.0004

Logistic regression using all 3 predictors:

	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	0.0082	0.37	0.7115
`student[Yes]`	−0.6468	0.2362	−2.74	0.0062

Extending logistic regression to more than 2 categories

Multinomial logistic regression:

Suppose Y takes values in $\{1, 2, \ldots, K\}$, then we use a linear model for the log odds against a baseline category (e.g. category 1):

$$\log\left[\frac{P(Y=2 \mid X)}{P(Y=1 \mid X)}\right] = \beta_{0,2} + \beta_{1,2}X_1 + \cdots + \beta_{p,2}X_p,$$

$$\cdots$$

$$\log\left[\frac{P(Y=K \mid X)}{P(Y=1 \mid X)}\right] = \beta_{0,K} + \beta_{1,K}X_1 + \cdots + \beta_{p,K}X_p.$$

Some issues with logistic regression

- The coefficients become unstable when there is collinearity. Furthermore, this affects the convergence of the fitting algorithm.
- When the classes are linearly separated (you can pass a hyperplane between them), the coefficients become unstable. This is always the case when $p \geq n-1$.

Linear Discriminant Analysis (LDA)

Strategy: Instead of estimating $P(Y \mid X)$, we will estimate:

1. $P(X \mid Y)$: Given the response, what is the distribution of the inputs.
2. $P(Y)$: How likely are each of the categories.

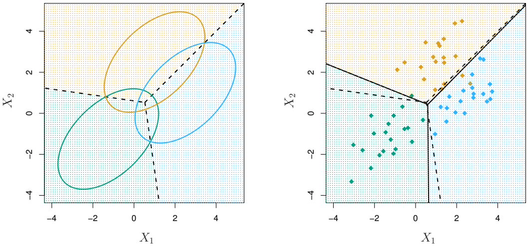
Given estimates $\hat{P}(X \mid Y)$ and $\hat{P}(Y)$, we use *Bayes rule* to obtain the estimate:

$$\begin{aligned}\hat{P}(Y = k \mid X = x) &= \frac{\hat{P}(X = x \mid Y = k)\hat{P}(Y = k)}{\hat{P}(X = x)} \\ &= \frac{\hat{P}(X = x \mid Y = k)\hat{P}(Y = k)}{\sum_j \hat{P}(X = x \mid Y = j)\hat{P}(Y = j)}\end{aligned}$$

Linear Discriminant Analysis (LDA)

Strategy: Instead of estimating $P(Y \mid X)$, we compute estimates:

1. $\hat{P}(X = x \mid Y = k) = \hat{f}_k(x)$, where each $\hat{f}_k(x)$ is a *Multivariate Normal Distribution* density:



2. $\hat{P}(Y = k) = \hat{\pi}_k$, the fraction of training samples of class k.

Next time

- Linear Discriminant Analysis (LDA):
  - How do we estimate the parameters of the MVN distribution $\hat{f}_k$ for each class k?
  - What do LDA predictions look like.
- How to evaluate a classification method?
- Examples: comparing KNN, logistic regression and LDA.