STA 4990 Introduction to Statistical Learning

Spring 2023

Thursday March 2

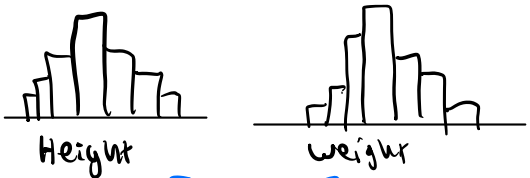
LDA, QDA, Naive Bayes all use Bayes Theorem

$$\mathbb{P}(Y=k \mid X=x) = \frac{\pi_k f_k(x)}{\sum_{l=1}^{K} \pi_l f_l(x)}$$

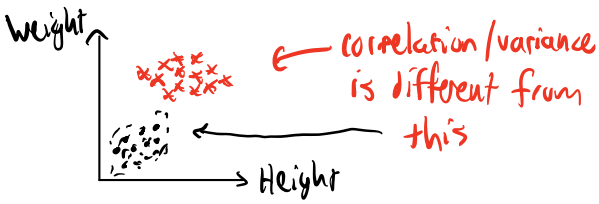
They differ in the assumptions on $f_k(x) = "\mathbb{P}(X=x \mid Y=k)"$

Ex 1

y	Height	Weight
⋮	⋮	⋮



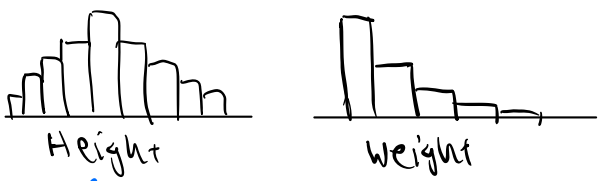
X_1, X_2 are univariate normal ✓



← correlation/variance is different from this

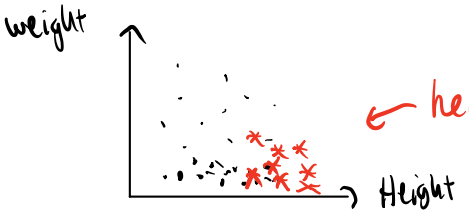
Ex 2

y	Height	Weight
⋮	⋮	⋮



X_1 is normal

X_2 is NOT



← height and weight look relatively uncorrelated

Naive Bayes

- LDA / QDA assume X is normally distributed: $X \sim MVN(\mu, \Sigma)$
  - Σ: covariance matrix, describes variances and correlations of all variables
- Naive Bayes assumes each $X_1, \dots, X_p$ is independent of one another

$$f_k(x_1, x_2, \dots, x_p) = f_{k1}(x_1) \, f_{k2}(x_2) \cdots f_{kp}(x_p)$$

($Y = k$; coordinate of X)

$\Rightarrow$ Highly flexible, since each $f_{kj}(x_j)$ can have a different distribution (ex $X_1 \sim N(\mu, \sigma^2)$, $X_2 \sim Exp(\lambda)$)

$\Rightarrow$ More appropriate when some X_j are categorical / binary

$\Rightarrow$ Independence assumption is rarely true!

(Despite this, its flexibility still makes it a good model when n is very large, even if $X_1, \dots, X_p$ are not independent).

Naive Bayes Syntax

```
library(caret)
library(modelr)
library(ISLR)

head(Default)
```

```
##   default student   balance    income
## 1      No      No  729.5265 44361.625
## 2      No     Yes  817.1804 12106.135
## 3      No      No 1073.5492 31767.139
## 4      No      No  529.2506 35704.494
## 5      No      No  785.6559 38463.496
## 6      No     Yes  919.5885  7491.559
```

```
fit_nbayes <- train(default ~ student + balance + income,
                    data = Default,
                    method = "naive_bayes")

Default <- Default %>%
  spread_predictions(fit_nbayes)

library(yardstick)
conf_mat(data=Default, truth=default, estimate=fit_nbayes)
```

```
##           Truth
## Prediction   No  Yes
##        No  9648  276
##        Yes   19   57
```

Comparing Classification Methods Thus Far

	Model Form	Notes
Logistic Regression	$P(Y=k\|X=x) = \frac{e^{\beta_0+\beta_1 x_1+\cdots+\beta_p x_p}}{1+e^{\beta_0+\beta_1 x_1+\cdots+\beta_p x_p}}$	- No assumption on dist. of X - Assumes prob. has form on left - Linear decision boundary
LDA (all use Bayes Thm $P(Y=k\|X=x) = \frac{\pi_k f_k(x)}{\sum_{\ell=1}^{K}\pi_\ell f_\ell(x)}$)	$X\|Y=k \sim MVN(\mu_k, \Sigma)$	- Assumes X is normal (strong assumption) → variances/correlations are same regardless of Y - Linear Decision boundary
QDA	$X\|Y=k \sim MVN(\mu_k, \Sigma_k)$	- Assumes X is normal → variances/correlations are different depending on Y - Quadratic decision boundary
Naive Bayes	$f_k(x_1,\cdots,x_p) = f_{k1}(x_1)\cdots f_{kp}(x_p)$	- Assumes $X_1,\cdots,X_p$ are independent (strong assumption) - Highly flexible - Any dist.'s can be used for $f_{kj}(x_j)$
KNN	$P(Y=k\|X=x) = \frac{\#\text{ of } Y=k \text{ in } N(x)}{\#\text{ in } N(x)}$ $N(x)$ is a "neighborhood" around x	- Nonparametric (others are all parametric) - Highly flexible (good when n large) - Has problems when p is large

Ex 1 Above

X's are normal (LDA+QDA good). Different variance/correlation depending on • or * => use QDA

Ex 2 Above

X_2 is not normal (LDA + QDA bad)

X_1, X_2 are uncorrelated $\Rightarrow$ Naive Bayes is good

Synthetic Examples ($K=2$)

Scenario 1:

$n=20$, $p=2$, X_1, X_2 are uncorrelated and normally distributed (same variance)

Scenario 2:

Same as 1, but $\text{corr}(X_1, X_2) = -0.5$

Scenario 3:

As before, but X_1 and X_2 are t-distributed (heavy tailed)

Hypothesized Ranking (1 - 10) (worst, Best) (my ranking in pink)

Logit	LDA	QDA	NB	KNN
8	10	7	9	5
8	10	8	3	6
9	6	5	6	7

Synthetic Experiments

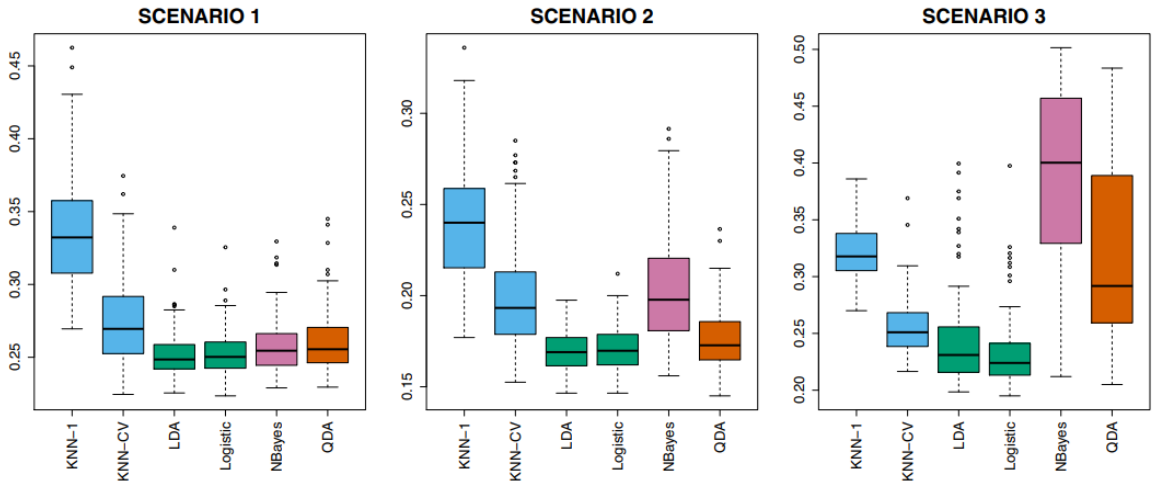


FIGURE 4.11. *Boxplots of the test error rates for each of the linear scenarios described in the main text.*

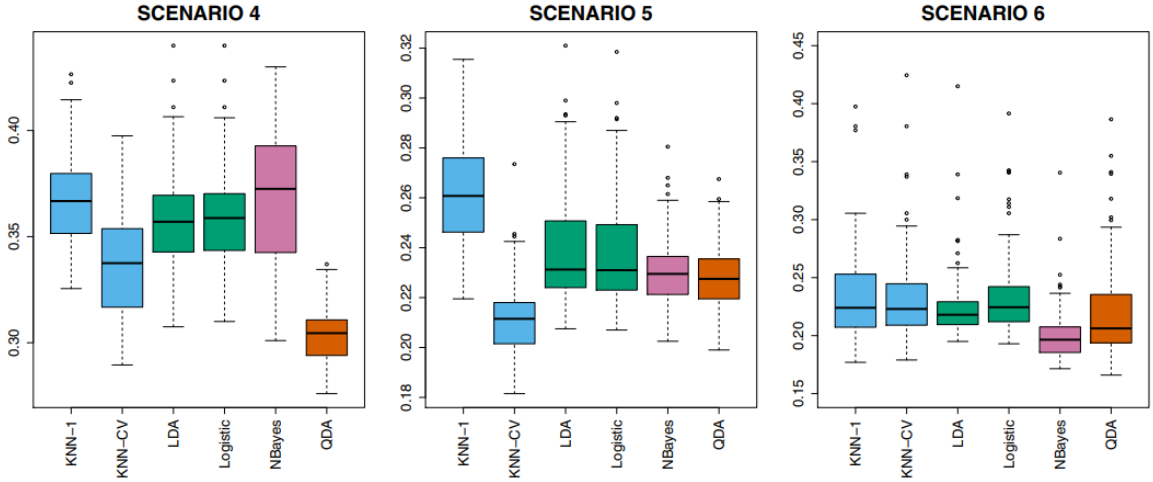


FIGURE 4.12. *Boxplots of the test error rates for each of the non-linear scenarios described in the main text.*

Prostate

- `lcavol`: log-cancer volume
- `lweight`: log prostate weight
- `age`: age in years
- `lbph`: log benign prostatic hyperplasia
- `svi`: seminal vesicle invasion
- `lcp`: log of capsular penetration
- `gleason`: Gleason score
- `pgg45`: percent of Gleascores 4/5
- `lpsa`: Outcome. Log of PSA (*prostate specific antigen*)
- `train`: TRUE or FALSE (*added variable*)

Turn into classification problem: categorize if `lpsa` greater than 3 (high PSA)

```
library(dplyr)
library(ggplot2)
library(genridge)

prostate$PSAhigh <- as.factor(prostate$lpsa > 3)
prostate$svi <- as.factor(prostate$svi)
prostate <- prostate %>% select(-lpsa)


head(prostate)
```

```
##       lcavol  lweight age      lbph svi       lcp gleason pgg45 train PSAhigh
## 1 -0.5798185 2.769459  50 -1.386294   0 -1.386294       6     0  TRUE   FALSE
## 2 -0.9942523 3.319626  58 -1.386294   0 -1.386294       6     0  TRUE   FALSE
## 3 -0.5108256 2.691243  74 -1.386294   0 -1.386294       7    20  TRUE   FALSE
## 4 -1.2039728 3.282789  58 -1.386294   0 -1.386294       6     0  TRUE   FALSE
## 5  0.7514161 3.432373  62 -1.386294   0 -1.386294       6     0  TRUE   FALSE
## 6 -1.0498221 3.228826  50 -1.386294   0 -1.386294       6     0  TRUE   FALSE
```

```
prostate_train <- prostate %>% filter(train == TRUE) %>% select(-train)
prostate_test <- prostate %>% filter(train == FALSE) %>% select(-train)

library(GGally)
prostate_train %>%
  ggpairs(aes(col=PSAhigh))  # GGally
```

train or test set


```
library(caret)
fit_logit <- train(
  PSAhigh ~ lcavol + age + svi,
  data = prostate_train,
  method = "glm",
  family = "binomial",  # passed to GLM
)

summary(fit_logit$finalModel)
```

```
##
## Call:
## NULL
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.1199  -0.4991  -0.2442   0.3866   2.2641
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.70649    4.07772  -0.418  0.67559
## lcavol       1.56796    0.55772   2.811  0.00493 **
## age         -0.03370    0.06412  -0.526  0.59913
## svi1         1.96124    0.97744   2.007  0.04480 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
```

```
##
##     Null deviance: 83.324  on 66  degrees of freedom
## Residual deviance: 44.536  on 63  degrees of freedom
## AIC: 52.536
##
## Number of Fisher Scoring iterations: 6
```

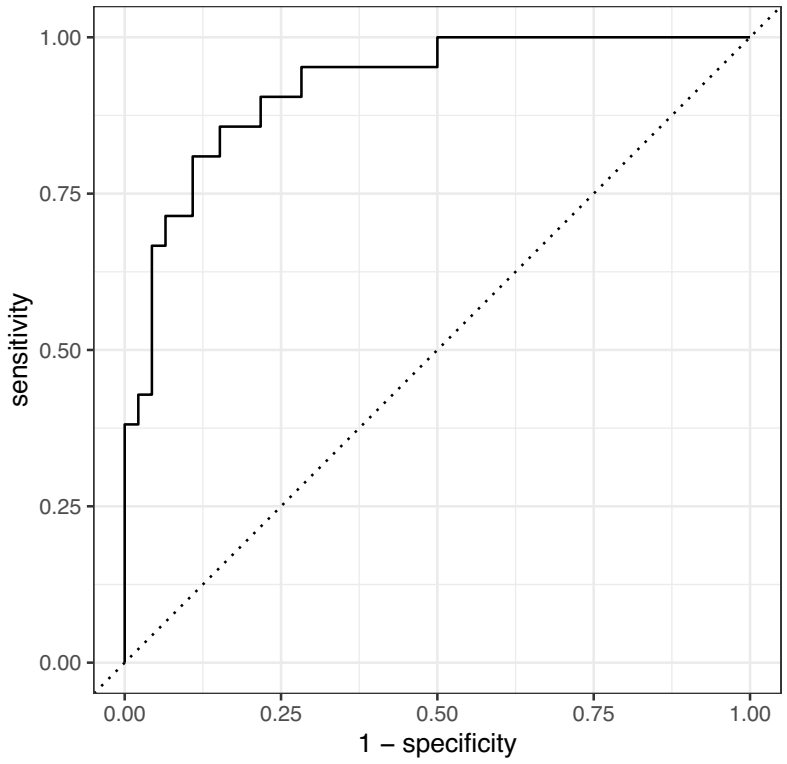
```
# spread predictions onto original "prostate" dataset
prostate <- prostate %>%
  mutate(logit_yhat = predict(fit_logit, newdata = ., type="raw")) %>%
  mutate(logit_prob = predict(fit_logit, newdata = ., type="prob")$`TRUE`)
```

Evaluate on Training Set

```
library(yardstick)
prostate %>%
  filter(train == TRUE) %>%
  conf_mat(PSAhigh, logit_yhat)
```

```
##           Truth
## Prediction FALSE TRUE
##      FALSE    43    6
##      TRUE      3   15
```

```
prostate %>%
  filter(train == TRUE) %>%
  roc_curve(truth = PSAhigh,
            estimate = logit_prob,
            event_level = "second") %>%
  autoplot()
```




```
# without setting event_level
prostate %>%
  filter(train == TRUE) %>%
  roc_auc(PSAhigh, logit_prob)
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary        0.0797
```

```
# with setting event_level
prostate %>%
  filter(train == TRUE) %>%
  roc_auc(PSAhigh, logit_prob, event_level = "second")
```

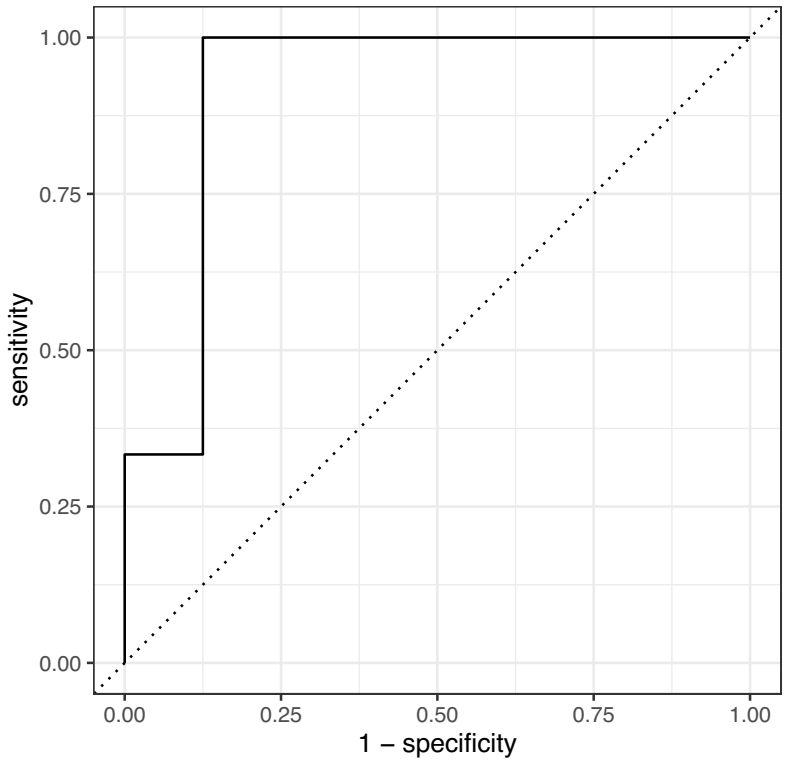
```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.920
```

Test Set (Logistic)

```
prostate %>%
  filter(train == FALSE) %>%
  conf_mat(PSAhigh, logit_yhat)
```

```
##            Truth
## Prediction FALSE TRUE
##      FALSE    21    1
##      TRUE      3    5
```

```
prostate %>%
  filter(train == FALSE) %>%
  roc_curve(truth = PSAhigh,
            estimate = logit_prob,
            event_level = "second") %>%
  autoplot()
```




```
prostate %>%
  filter(train == FALSE) %>%
  roc_auc(PSAhigh, logit_prob, event_level = "second")
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.917
```

Fit Other Models

```
fit_knn <- train(
  PSAhigh ~ lcavol + age + svi,
  data = prostate_train,
  method = "knn",
  tuneGrid=expand.grid(
    k=c(10)),
)

fit_lda <- train(
  PSAhigh ~ lcavol + age + svi,
  data = prostate_train,
  method = "lda",
)

fit_qda <- train(
  PSAhigh ~ lcavol + age + svi,
  data = prostate_train,
  method = "qda",
)

fit_nbayes <- train(
  PSAhigh ~ lcavol + age + svi,
  data = prostate_train,
  method = "naive_bayes",
)
```

Spread Predictions

```
# spread predictions onto original "prostate" dataset
prostate <- prostate %>%
  mutate(knn_yhat = predict(fit_knn, newdata = ., type="raw")) %>%
  mutate(knn_prob = predict(fit_knn, newdata = ., type="prob")$`TRUE`)

prostate <- prostate %>%
  mutate(lda_yhat = predict(fit_lda, newdata = ., type="raw")) %>%
  mutate(lda_prob = predict(fit_lda, newdata = ., type="prob")$`TRUE`)

prostate <- prostate %>%
  mutate(qda_yhat = predict(fit_qda, newdata = ., type="raw")) %>%
  mutate(qda_prob = predict(fit_qda, newdata = ., type="prob")$`TRUE`)

prostate <- prostate %>%
  mutate(nbayes_yhat = predict(fit_nbayes, newdata = ., type="raw")) %>%
  mutate(nbayes_prob = predict(fit_nbayes, newdata = ., type="prob")$`TRUE`)
```

Training Set Comparisons

```
library(tidyr)

print("Training Set")
```

```
## [1] "Training Set"
```

```
prostate %>%
  filter(train == TRUE) %>%
  pivot_longer(c(logit_prob,
                 knn_prob,
                 lda_prob,
                 qda_prob,
                 nbayes_prob)) %>%
  group_by(name) %>%
  roc_auc(PSAhigh, value, event_level = "second")
```

```
## # A tibble: 5 x 4
##   name        .metric .estimator .estimate
##   <chr>       <chr>   <chr>          <dbl>
## 1 knn_prob    roc_auc binary         0.856
## 2 lda_prob    roc_auc binary         0.922
## 3 logit_prob  roc_auc binary         0.920
## 4 nbayes_prob roc_auc binary         0.917
## 5 qda_prob    roc_auc binary         0.937
```

```
yhat_metrics <- metric_set(accuracy,
                           sens, # True Positive Rate
                           spec  # True Negative Rate
                           )

prostate %>%
  filter(train == TRUE) %>%
  pivot_longer(c(logit_yhat,
                 knn_yhat,
                 lda_yhat,
                 qda_yhat,
                 nbayes_yhat)) %>%
  group_by(name) %>%
  yhat_metrics(PSAhigh, estimate = value)
```

```
## # A tibble: 15 x 4
##    name        .metric  .estimator .estimate
##    <chr>       <chr>    <chr>          <dbl>
##  1 knn_yhat    accuracy binary         0.776
##  2 lda_yhat    accuracy binary         0.851
##  3 logit_yhat  accuracy binary         0.866
##  4 nbayes_yhat accuracy binary         0.851
##  5 qda_yhat    accuracy binary         0.866
##  6 knn_yhat    sens     binary         0.978
##  7 lda_yhat    sens     binary         0.957
##  8 logit_yhat  sens     binary         0.935
##  9 nbayes_yhat sens     binary         0.957
## 10 qda_yhat    sens     binary         0.957
## 11 knn_yhat    spec     binary         0.333
## 12 lda_yhat    spec     binary         0.619
## 13 logit_yhat  spec     binary         0.714
## 14 nbayes_yhat spec     binary         0.619
## 15 qda_yhat    spec     binary         0.667
```

Test Set Comparisons

```
print("Test Set")
```

```
## [1] "Test Set"
```

```
prostate %>%
  filter(train == FALSE) %>%
  pivot_longer(c(logit_prob,
                 knn_prob,
                 lda_prob,
                 qda_prob,
                 nbayes_prob)) %>%
  group_by(name) %>%
  roc_auc(PSAhigh, value, event_level = "second")
```

```
## # A tibble: 5 x 4
##   name        .metric .estimator .estimate
##   <chr>       <chr>   <chr>          <dbl>
## 1 knn_prob    roc_auc binary         0.722
## 2 lda_prob    roc_auc binary         0.931
## 3 logit_prob  roc_auc binary         0.917
## 4 nbayes_prob roc_auc binary         0.931
## 5 qda_prob    roc_auc binary         0.924
```

```
prostate %>%
  filter(train == FALSE) %>%
  pivot_longer(c(logit_yhat,
                 knn_yhat,
                 lda_yhat,
                 qda_yhat,
                 nbayes_yhat)) %>%
  group_by(name) %>%
  yhat_metrics(PSAhigh, estimate = value)
```

```
## # A tibble: 15 x 4
##    name        .metric  .estimator .estimate
##    <chr>       <chr>    <chr>          <dbl>
##  1 knn_yhat    accuracy binary         0.8
##  2 lda_yhat    accuracy binary         0.867
##  3 logit_yhat  accuracy binary         0.867
##  4 nbayes_yhat accuracy binary         0.867
##  5 qda_yhat    accuracy binary         0.833
##  6 knn_yhat    sens     binary         0.917
##  7 lda_yhat    sens     binary         0.917
##  8 logit_yhat  sens     binary         0.875
##  9 nbayes_yhat sens     binary         0.917
## 10 qda_yhat    sens     binary         0.875
## 11 knn_yhat    spec     binary         0.333
## 12 lda_yhat    spec     binary         0.667
## 13 logit_yhat  spec     binary         0.833
## 14 nbayes_yhat spec     binary         0.667
## 15 qda_yhat    spec     binary         0.667
```