Clasificadores Naïve Bayes

Edgar Acuna
Departamento de Matematicas
Universidad de Puerto Rico en Mayaguez

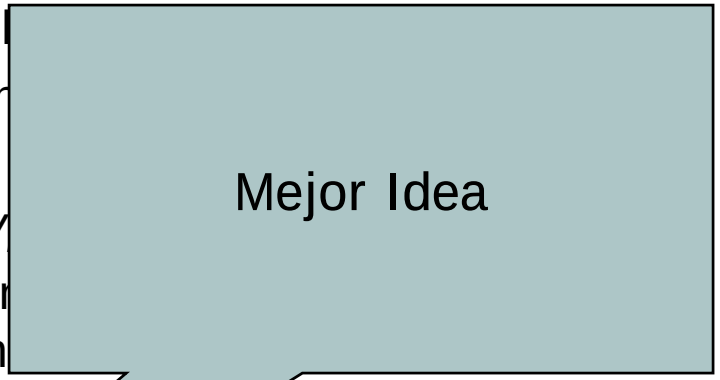
Construccion de un clasificador No Bayesiano estimando f(x/C)

- Asumir que se quiere predecir la variable Y que asume G valores distintos y que estos valores son v_1, v_2, … v_G.
- Asumir que hay m atributos de entrada llamados X_1, X_2, … X_m
- Dividir el conjunto de datos en G subconjuntos de datos llamados DS_1, DS_2, … DS_G.
- Definir DS_i = Registros en los cuales $Y=v_i$
- Para cada grupo DS_i , usamos estimacion de densidad para estimar el modelo M_i *que modela* la distribucion de las variables de entrada entre los registros $Y=v_i$.
- M_i estima la funcion de probabilidad conjunta por clase P(X_1, X_2, … X_m | $Y=v_i$)
- Idea 1. Suponga que se quiere predecir la clase Y a la cual pertenece el vector ($X_1 = u_1$, $X_2 = u_2$, …. $X_m = u_m$). Se puede tomar como la clase $Y=v_i$ a aquella para la cual P(X_1, X_2, … X_m | $Y=v_i$) sea la mayor posible. Esto es,

$$Y^{\text{predict}} = \underset{v}{\operatorname{argmax}}\, P(X_1 = u_1 \cdots X_m = u_m \mid Y = v)$$

Construccion de un clasificador Bayesiano estimando f(x/c)

- Asumir que se quiere predecir la variable Y que asume ny valores distintos y que estos valore son v_1, v_2, … v_{ny}.
- Asumir que hay m atributos de entrada ll
- Dividir el conjunto de datos en n_Y subcor
  DS_1, DS_2, … DS_{ny}.
- Definir DS_i = Registros en los cuales $Y=v$
- Para cada grupo DS_i , usamos estimacior
  el modelo M_i *que modela* la distribucion
  entre los registros $Y=v_i$.
- M_i estima la funcion de probabilidad conju or clase P(X_1, X_2, … X_m | $Y=v_i$
- Idea 2: Para predecir la clase a cual pertenece el nuevo vector de entradas ($X_1 = u_1$, $X_2 = u_2$, …. $X_m = u_m$) es mejor hallar la clase Y=vi para la cual la probabilidad posterior P($Y=v_i$ | X_1, X_2, … X_m) sea la mayor posible.



Terminologia

- MLE (Estimador Maximo Verosimil):

$$Y^{\text{predict}} = \underset{v}{\operatorname{argmax}}\, P(X_1 = u_1 \cdots X_m = u_m \mid Y = v)$$

- MAP (Estimador Maximo a Posteriori):

$$Y^{\text{predict}} = \underset{v}{\operatorname{argmax}}\, P(Y = v \mid X_1 = u_1 \cdots X_m = u_m)$$

Obteniendo la probabilidad posterior

$$P(Y = v \mid X_1 = u_1 \cdots X_m = u_m)$$

$$= \frac{P(X_1 = u_1 \cdots X_m = u_m \mid Y = v) P(Y = v)}{P(X_1 = u_1 \cdots X_m = u_m)}$$

$$= \frac{P(X_1 = u_1 \cdots X_m = u_m \mid Y = v) P(Y = v)}{\sum_{j=1}^{G} P(X_1 = u_1 \cdots X_m = u_m \mid Y = v_j) P(Y = v_j)}$$

$$P(X_1 = u_1, ..., X_n = u_n) = \sum_{v=1}^{G} P(X_1 = u_1, ... X_n = u_n, Y = v)$$

$$\sum_{v=1}^{G} P(X_1 = u_1, ..., X_n = u_n / Y = v) P(Y = v)$$

Estimacion de un Clasificador Bayesiano

1. Estimar la distribucion de las predictoras en cada clase. Es decir, estimar P(X_1, X_2, ... X_m / $Y=v_i$).

2. Estimar P($Y=v_i$). Como la fraccion de registros con $Y=v_i$.

3. Para una nueva prediccion:

$$Y^{\text{predict}} = \operatorname{argmax}_v P(Y = v \mid X_1 = u_1 \cdots X_m = u_m)$$
$$= \operatorname{argmax}_v P(X_1 = u_1 \cdots X_m = u_m \mid Y = v) P(Y = v)$$

Estimacion de un clasificador Bayesiano

1. Estimar la distribucion de las predictoras en cada clase. Es decir, estimar P(X_1, X_2, ... X_m / $Y=v_i$).

3. Estimar P($Y=v_i$). Como la fracci[illegible]de registros con $Y=v_i$.

4. Para una nueva prediccion:

$$Y^{\text{predict}} = \operatorname*{argmax}_v P(Y = v$$

$$= \operatorname*{argmax}_v P(X_1 = u_1 \cdots X_r$$

Podemos usar nuestro favorito estimador de densidad.

Tenemos dos opciones:

•Estimador de densidad conjunta (kernel, k-nn)
• Estinador de densidad Naïve

Clasificador Naïve Bayes

En el caso del clasificador naive se supone que las variables predictoras son independientes en cada una de las clases . Esto es,

$$P(X_1 = u_1 \cdots X_m = u_m \mid Y = v) = P(X_1 = u_1 / Y = v) \ldots . P(X_m = u_m / Y = v).$$

Luego,

$$Y^{\text{predict}} = \underset{v}{\operatorname{argmax}} P(X_1 = u_1 \cdots X_m = u_m \mid Y = v) P(Y = v)$$

Se convierte en:

$$Y^{\text{predict}} = \underset{v}{\operatorname{argmax}} \; P(Y = v) \prod_{j=1}^{m} P(X_j = u_j \mid Y = v)$$

Clasificador Naïve Bayes

$$Y^{\text{predict}} = \underset{v}{\operatorname{argmax}} P(Y = v) \prod_{j=1}^{m} P(X_j = u_j \mid Y = v)$$

Si hay muchos atributos de entrada este producto puede producir underflow, asi que es mejor usar logaritmos.

$$Y^{\text{predict}} = \underset{v}{\operatorname{argmax}} \left(\log P(Y = v) + \sum_{j=1}^{m} \log P(X_j = u_j \mid Y = v) \right)$$

Si X_j es discreta entonces
$P(X_j=u_j/Y=v)$ = (#de records con $X_j=u_j$ en la clase v)/# de records en la clase v

Clasificador Naïve Bayes

El clasificador Naïve Bayes puede ser aplicado tambien cuando hay predictoras continuas, hay dos alternativas

a) Aplicar previamente un metodo de discretizacion tal como: Usando intervalos de igual ancho, usando intervalos con igual frecuencia, ChiMerge,1R, Discretizacion usando el metodo de la entropia. Todos ellos estan disponible en la libreria dprep (ver disc.mentr, disc.ew, disc.ef y chiMerge) .

b) Asumiendo uan distribucion para cada predictora, por lo general Gausiana, con media y varianza estimada de los datos. La libreria e1071 de R contiene una funcion **naiveBayes** que calcula el clasificador naïve Bayes.

Clasificador Naïve Bayes (cont)

En este caso

$$P[X_j = a_j / C_i] = \frac{1}{s_j\sqrt{2\pi}} \exp[\frac{-(a_i - \bar{x}_j)^2}{2s_j^2}]$$

Donde, $\bar{x}_j$ y s_j son la media y la varianza de los valores de la variable X_j en la clase C_i .

La libreria e1071 de R contiene una funcion **naiveBayes** que calcula el clasificador naïve Bayes, tanto para datos discretos como continuos.

Tambien esta disponible en Xlminer y en analysis services de Microsoft SQL 2008.

Ejemplo 1.(atributos discretos solamente)

X1	X2	X3	Y
0	0	1	0
0	1	0	0
1	1	0	0
0	0	1	1
1	1	1	1
0	0	1	1
1	1	0	1

Ejemplo 1. (Cont.)

$P(Y=0)=3/7$ $\quad$ $P(Y=1)=4/7$

A que clase sera asignada el registro (X1=0,X2=0,X3=1)?

$$P(X_1=0,X_2=0,X_3=1/Y=0)=P(X_1=0/Y=0)P(X_2=0/Y=0)$$
$$P(X_3=1/Y=0)=(2/3)(1/3)(1/3)=2/27$$

$$P(X_1=0,X_2=0,X_3=1/Y=1)=P(X_1=0/Y=1)P(X_2=0/Y=1)$$
$$P(X_3=1/Y=1)=(2/4)(2/4)(3/4)=3/16$$

Como (3/7)(2/27)<(4/7)(3/16) entonces (X1=0,X2=0, x3=1) sera asignado a la clase 1. Si el objeto esta asignado a la clase 0 entonces el NB comete un error.

Ejemplo 2. (atributos discretos y continuos)

X1	X2	X3	X4	Y
0	0	1	3.15	0
0	1	0	8.17	0
1	1	0	5.72	0
0	0	1	7.16	1
1	1	1	9.32	1
0	0	1	12.81	1
1	1	0	15.48	1

Ejemplo 2. (cont.)

- #Metodo 1. Discretizando la columna 4
- dnaiveeje2=disc.ew(naiveeje2,c(4:5))

```
> dnaiveeje2
     col1 col2 col3 col4 col5
[1,]    0    0    1    1    0
[2,]    0    1    0    1    0
[3,]    1    1    0    1    0
[4,]    0    0    1    1    1
[5,]    1    1    1    2    1
[6,]    0    0    1    2    1
[7,]    1    1    0    2    1
```

Ejemplo 2. (cont.)

- #Metodo 2. Sin discretizar la columna 4
- Media y desviacion estandar de la col4 en cada clase
- > mean(naiveeje2[naiveeje2[,5]==0,4])
- [1] 5.68
- > mean(naiveeje2[naiveeje2[,5]==1,4])
- [1] 11.1925
- > sd(naiveeje2[naiveeje2[,5]==0,4])
- [1] 2.510239
- > sd(naiveeje2[naiveeje2[,5]==1,4])
- [1] 3.686293

> # a que clase sera asignado el vector(0,0,1,4.25)?
Hay que calcular P[X1=0/Y=0]P[X2=0/Y=0]P[x3=1/Y=0]f(x4=425/Y=0][Y=0]
y compararla con
P[X1=0/Y=1]P[X2=0/Y=1]P[x3=1/Y=1]f(x4=425/Y=1][Y=1]
> (2/27)*pnorm(4.25,5.68,2.5102)*3/7
[1] 0.009030122
> (3/16)*pnorm(4.25,11.1925,3.6862)*4/7
[1] 0.003195506
Luego el vector sera asigando a la clase 0.

Ejemplo 2. (cont.)

- #Sin discretizar la columna 4
- Media y desviacion estandar de la col4 en cada clase
- > mean(naiveeje2[naiveeje2[,5]==0,4])
- [1] 5.68
- > mean(naiveeje2[naiveeje2[,5]==1,4])
- [1] 11.1925
- > sd(naiveeje2[naiveeje2[,5]==0,4])
- [1] 2.510239
- > sd(naiveeje2[naiveeje2[,5]==1,4])
- [1] 3.686293

> # a que clase sera asigando el vector(0,0,1,4.25)?
Hay que calcular P[X1=0/Y=0]P[X2=0/Y=0]P[x3=1/Y=0]f(x4=425/Y=0][Y=0]
y compararla con
P[X1=0/Y=1]P[X2=0/Y=1]P[x3=1/Y=1]f(x4=425/Y=1][Y=1]
> (2/27)*pnorm(4.25,5.68,2.5102)*3/7
[1] 0.009030122
> (3/16)*pnorm(4.25,11.1925,3.6862)*4/7
[1] 0.003195506
Luego el vector sera asigando a la clase 0.

Ejemplo 2. (cont.)

- #usando naivebayes de e1071
- naivebayes21=as.data.frame(naiveeje2)

```
> a=naiveBayes(col5~.,data=naivebayes21)
> pred=predict(a,naivebayes21[,-5],type="raw")
```

- pred1=max.col(pred)
- > pred1
- [1] 1 1 1 2 2 2 2

```
> table(pred1,naivebayes21[,5])
Pred1   0 1
        1 3 0
        2 0 4
Error=0
```

Naive Bayes para Bupa

Sin discretizar

```
> a=naiveBayes(V7~.,data=bupa)
> pred=predict(a,bupa[,-7],type="raw")
> pred1=max.col(pred)
> table(pred1,bupa[,7])
pred1   1   2
    1 112 119
    2  33  81
> error=152/345
[1] 0.4405797
```

Discretizando con el metodo de la entropia

```
>dbupa=disc.mentr(bupa,1:7)
>for (i in 1:7)
+dbupa[,i]=as.factor(dbupa[,i])
>b=naiveBayes(V7~.,data=dbupa)
>pred=predict(b,dbupa[,-7])
>error=sum(pred!=bupa[,7])/dim(dbupa)[1]
>error
[1] 0.3681159
```

Naïve Bayes para Bupa (cont.)

Discretizando por el metodo ChiMerge

```
> chibupa=chiMerge(bupa,1:6)
> for (i in 1:7)
+ chibupa[,i]=as.factor(chibupa[,i])
> b=naiveBayes(V7~.,data=chibupa)
> pred=predict(b,chibupa[,-7])
> error=sum(pred!=chibupa[,7])/dim(chibupa)[1]
> error
[1] 0.1420290
```

Discretizando usando intervalos de igual ancho

```
> dbupa=disc.ew(bupa,1:6)
> for (i in 1:7)
+ dbupa[,i]=as.factor(dbupa[,i])
> b=naiveBayes(V7~.,data=dbupa)
> pred=predict(b,dbupa[,-7])
> error=sum(pred!=dbupa[,7])/dim(dbupa)[1]
> error
[1] 0.2608696
```

Naïve Bayes para Diabetes

Sin Descretizar

```
> a=naiveBayes(V9~.,data=diabetes)
> pred=predict(a,diabetes[,-9],type="raw")
> pred1=max.col(pred)
> table(pred1,diabetes[,9])

pred1   1   2
    1 421 104
    2  79 164
> error=(79+104)/768
[1] 0.2382813
```

Discretizando por el metodo de la entropia

```
> ddiabetes=disc.mentr(diabetes,1:9)
> for (i in 1:9)
+ ddiabetes[,i]=as.factor(ddiabetes[,i])
> b=naiveBayes(V9~.,data=ddiabetes)
> pred=predict(b,ddiabetes[,-9])
> error=sum(pred!=ddiabetes[,9])/dim(ddiabetes)[1]
> error
[1] 0.2161458
```

Naïve Bayes para Diabetes (cont.)

Discretizando por el metodo ChiMerge

```
> ddiabetes=chiMerge(diabetes,1:8)
> for (i in 1:9)
+ ddiabetes[,i]=as.factor(ddiabetes[,i])
> b=naiveBayes(V9~.,data=ddiabetes)
> pred=predict(b,ddiabetes[,-9])
> error=sum(pred!=ddiabetes[,9])/dim(ddiabetes)[1]
> error
[1] 0.09895833
```

Discretizando por el metodo de intervalos de igual ancho

```
>ddiab=disc.ew(diabetes,1:8)
> ddiabetes=disc.ew(diabetes,1:8)
> for (i in 1:9)
+ ddiabetes[,i]=as.factor(ddiabetes[,i])
> b=naiveBayes(V9~.,data=ddiabetes)
> pred=predict(b,ddiabetes[,-9])
> error=sum(pred!=ddiabetes[,9])/dim(ddiabetes)[1]
> error
[1] 0.2083333
```

The Auto-mpg dataset

Donor: Quinlan,R. (1993)

Number of Instances: 398 minus 6 missing=392 (training: 196 test: 196):

Number of Attributes: 9 including the class attribute7. Attribute Information:

1. mpg: continuous (discretizado bad<=25,good>25)
2. cylinders: multi-valued discrete
3. displacement: continuous (discretizado low<=200, high>200)
4. horsepower: continuous ((discretizado low<=90, high>90)
5. weight: continuous (discretizado low<=3000, high>3000)
6. acceleration: continuous (discretizado low<=15, high>15)
7. model year: multi-valued discrete (discretizado 70-74,75-77,78-82
8. origin: multi-valued discrete
9. car name: string (unique for each instance)

Note: horsepower has 6 missing values

The auto-mpg dataset

18.0 8 307.0 130.0 3504. 12.0 70 1 "chevrolet chevelle malibu"

15.0 8 350.0 165.0 3693. 11.5 70 1 "buick skylark 320"

18.0 8 318.0 150.0 3436. 11.0 70 1 "plymouth satellite"

16.0 8 304.0 150.0 3433. 12.0 70 1 "amc rebel sst"

17.0 8 302.0 140.0 3449. 10.5 70 1 "ford torino"

……..

27.0 4 140.0 86.00 2790. 15.6 82 1 "ford mustang gl"

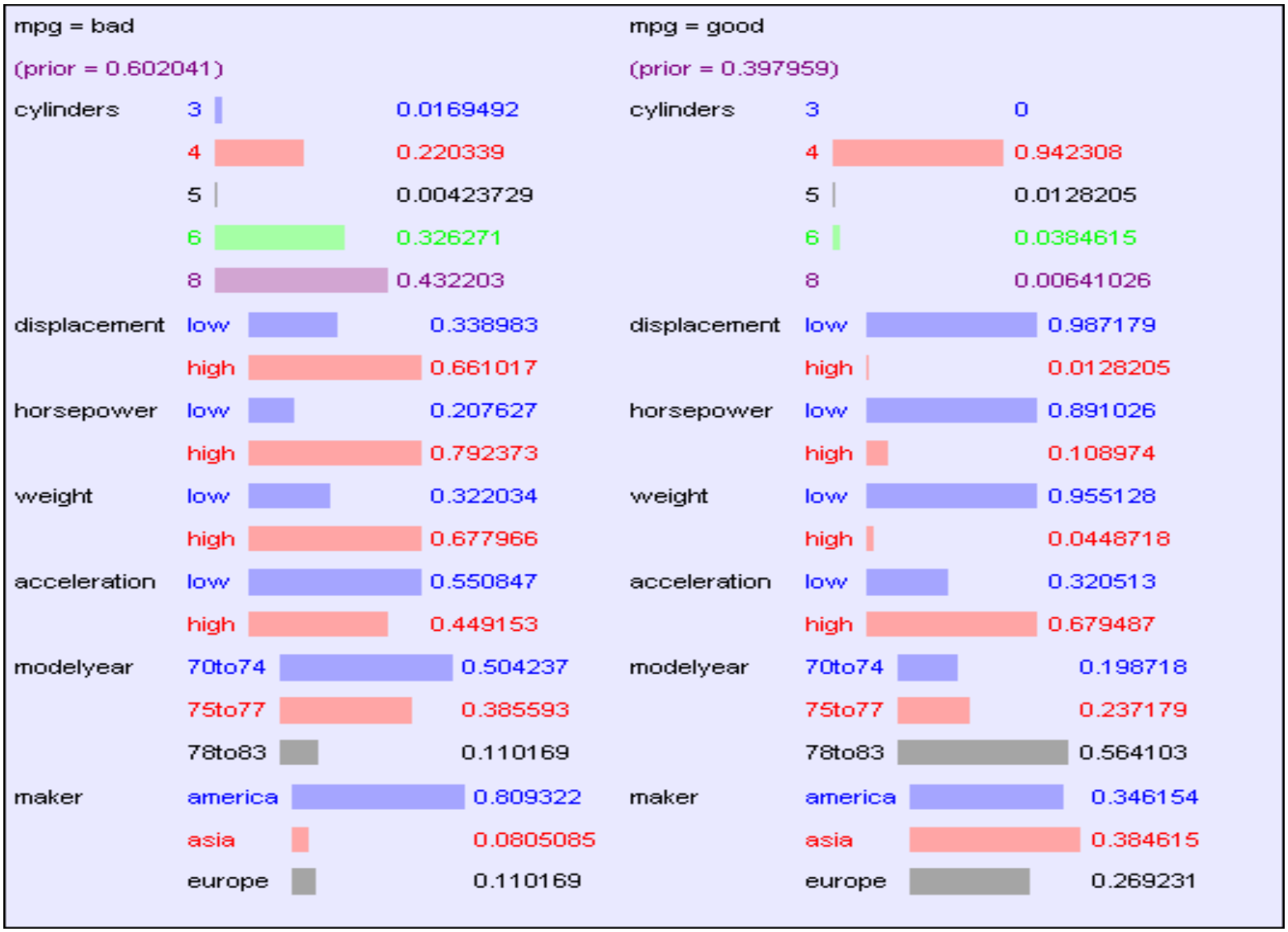
44.0 4 97.00 52.00 2130. 24.6 82 2 "vw pickup"

32.0 4 135.0 84.00 2295. 11.6 82 1 "dodge rampage"

28.0 4 120.0 79.00 2625. 18.6 82 1 "ford ranger"

31.0 4 119.0 82.00 2720. 19.4 82 1 "chevy s-10"

Resultados del clasificador NB para “MPG”: 392 records



Resultados del clasificador NB para: “mpg”

```
> #sin discretizar
> b=naiveBayes(mpg~.,data=autompg)
> pred=predict(b,autompg[,-1],type="raw")
> pred1=max.col(pred)
> table(pred1,autompg[,1])
pred1   1   2
    1 180   8
    2  56 148
> error=64/345
> error
[1] 0.1855072
>#Discretizando manualmente
> b=naiveBayes(mpg~.,data=autompg2)
> pred=predict(b,autompg2[,-1])
> table(pred,autompg2[,1])
pred   1   2
   1 182   7
   2  54 149
> 61/392
[1] 0.1556122
```

Clasificadores Naive Bayes (cont.)

- Las probabilidades cero afectan al clasificador Naïve Bayes. Una funcion de probabilidad a priori de Dirichlet es usada para resolver el problema.
- El proceso de discretizacion tambien parecer afectar el rendimiento del clasificador.
- Naïve Bayes es bastante barato. No tiene problemas para trabajar con 10,000 atributos.
- Naïve Bayes es un caso particular de Redes bayesianas