

1 Middle, and Diversity around Middle

Here's the great secret about optimization and data mining

- under the hood, they share a lot of the same structures
- data mining divides up the world
- optimizers comment where to jump between the divisions.

So here we want to talk about the data and operations relevant to columns and rows of data.

	{Clndrs	Volume	Hpx	Lbs-	Acc+	Model	origin	Mpg+}
t1= {cells=	{4	97	52	2130	24.6	82	2	40}}
t2= {cells=	{4	97	54	2254	23.5	72	2	20}}
t3= {cells=	{4	97	78	2188	15.8	80	2	30}}
t4= {cells=	{4	151	90	2950	17.3	82	1	30}}
t5= {cells=	{6	200	?	2875	17	74	1	20}}
t6= {cells=	{6	146	97	2815	14.5	77	3	20}}
t7= {cells=	{8	267	125	3605	15	79	1	20}}
t9= {cells=	{8	307	130	4098	14	72	1	10}}

Things to watch for are

- nominal, ratio, SYM, NUM
- mid, mean, median, mode
- div, entropy, standard deviation, IQR
- parametric (normal, Gaussian)
- bayes classifier, low frequency variables

2 Kinds of of Attributes (aka Columns, Variables, Features)

	Nominal	Ordinal	Interval	Ratio
used in this class	y	n	n	y
aka	SYMbolic			NUMeric
e.g.	eye color, genotype	small,medium	temperature	weight,length
compare $\neq, =$,	Y	Y	Y	Y
order, sort $<$,	N	Y	Y	Y
$>$				
+ or -	N	N	Y	Y
* or /	N	N	N	Y
centrality(mid)	mode(most common symbol)	mode		mean $(\sum_i x_i)/n$ median (50th percentile)
diversity(div)	entropy (effort to recreate signal) $e = -\sum_i (p_i \times \log_2(p_i))$	entropy		standard deviation (distances from the mean) $\sigma = \sqrt{\frac{\sum_i (x_i - \mu)^2}{n-1}}$ IQR: inter-quartile range(75th-25th) percentile

2.1 Class SYM

Here are some classes to compute the important parts of the above:

```

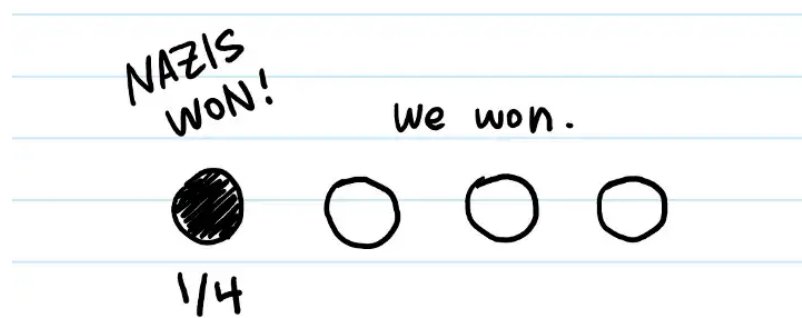
SYM = obj "SYM"
function SYM.new(i) --> SYM; constructor
  i.n = 0
  i.has = {}
  i.most, i.mode = 0, nil end

function SYM.add(i,x) --> nil; update counts of things seen so far
  if x ~= "?" then
    i.n = i.n + 1
    i.has[x] = 1 + (i.has[x] or 0)
    if i.has[x] > i.most then
      i.most, i.mode = i.has[x], x end end end

function SYM.mid(i,x) return i.mode end --> n; return the mode

function SYM.div(i,x) --> n; return the entropy, calculated via Shannon entropy
  local function fun(p) return p*math.log(p,2) end
  local e=0; for _,n in pairs(i.has) do e = e + fun(n/i.n) end
  return -e end

```



By the way, to understand SYM.div (entropy), think of it as

- the effort required by binary chop to find clumps of a signal hiding in a stream of noise
- e.g. in a vector of size 4,
 - nazis have a "1" near one end
 - and England are all the other bits
 - This means that 1/4% of the time we need to do binary chops to find nazis (i.e. $p_{\text{nazis}} = .25$)
 - and 75% if the time we need to binary chops to find England (i.e. $p_{\text{england}} = .75$)
 - Each chop will cost us $\log_2(p_i)$ so the total effort is $e = -\sum_i (p_i \times \log_2(p_i))$
 - * By convention, we add a minus sign at the front (else all entropies will be negative).

(Actually, formally entropy has other definition: - The entropy of a discrete random variable is a lower bound on the expected number of bits required to transfer the result of the random variable. - Also, entropy of continuous distributions is defined, but we do not use that in this subject.)

2.2 Class NUM

```
NUM = obj"NUM"
function NUM.new(i) --> NUM;  constructor;
  i.n, i.mu, i.m2 = 0, 0, 0
  i.lo, i.hi = math.huge, -math.huge end

function NUM.add(i,n) --> NUM
  if n ~= "?" then
    i.n = i.n + 1
    local d = n - i.mu
    i.mu = i.mu + d/i.n
    i.m2 = i.m2 + d*(n - i.mu)
    i.lo = math.min(n, i.lo)
    i.hi = math.max(n, i.hi) end end

function NUM.mid(i,x) return i.mu end --> n; return mean

function NUM.div(i,x) --> n; return standard deviation using Welford's algorithm http://t
  return (i.m2 < 0 or i.n < 2) and 0 or (i.m2/(i.n-1))^0.5 end
```

2.2.1 So-called "Normal" Curves

If we are talking standard deviation, then we had better talk about normal curves.

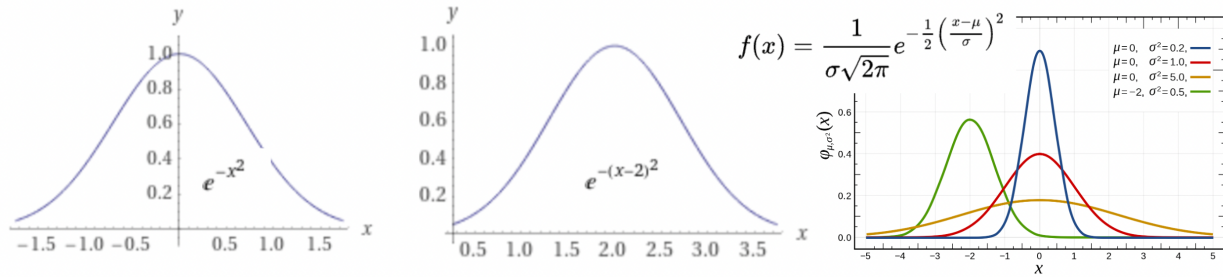
The French mathematician Abraham de Moivre¹ notes that probabilities associated with discretely generated random variables (such as are obtained by flipping a coin or rolling a die) can be approximated by the area under the graph of an exponential function.

This function was generalized by Laplace² into the first central limit theorem, which proved that probabilities for almost all independent and identically distributed random variables converge rapidly (with sample size) to the area under an exponential function—that is, to a normal distribution.

This function was extended, extensively by Gauss. Now its a curve with an area under the curve of one. As standard deviation shrinks, the curve spikes upwards.

¹Hald, Anders (1990), "De Moivre and the Doctrine of Chances, 1718, 1738, and 1756", History of Probability and Statistics and Their Applications before 1750, Wiley Series in Probability and Statistics, Wiley Interscience, p. 397.

²Laplace, Pierre-Simon (1812). Théorie analytique des probabilités [Analytical theory of probabilities]. Paris, Ve. Courcier.

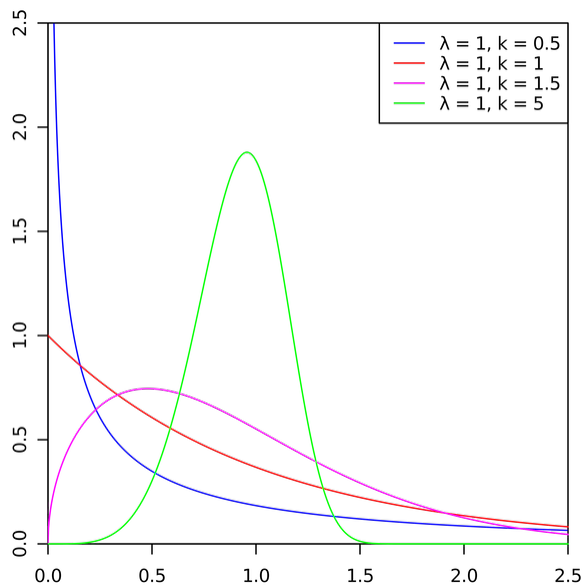


To sample from a normal curve from a Gaussian with mean μ and diversity sd

$$\mu + sd * \text{sqrt}(-2 * \log(\text{random})) * \cos(2 * \pi * \text{random})$$

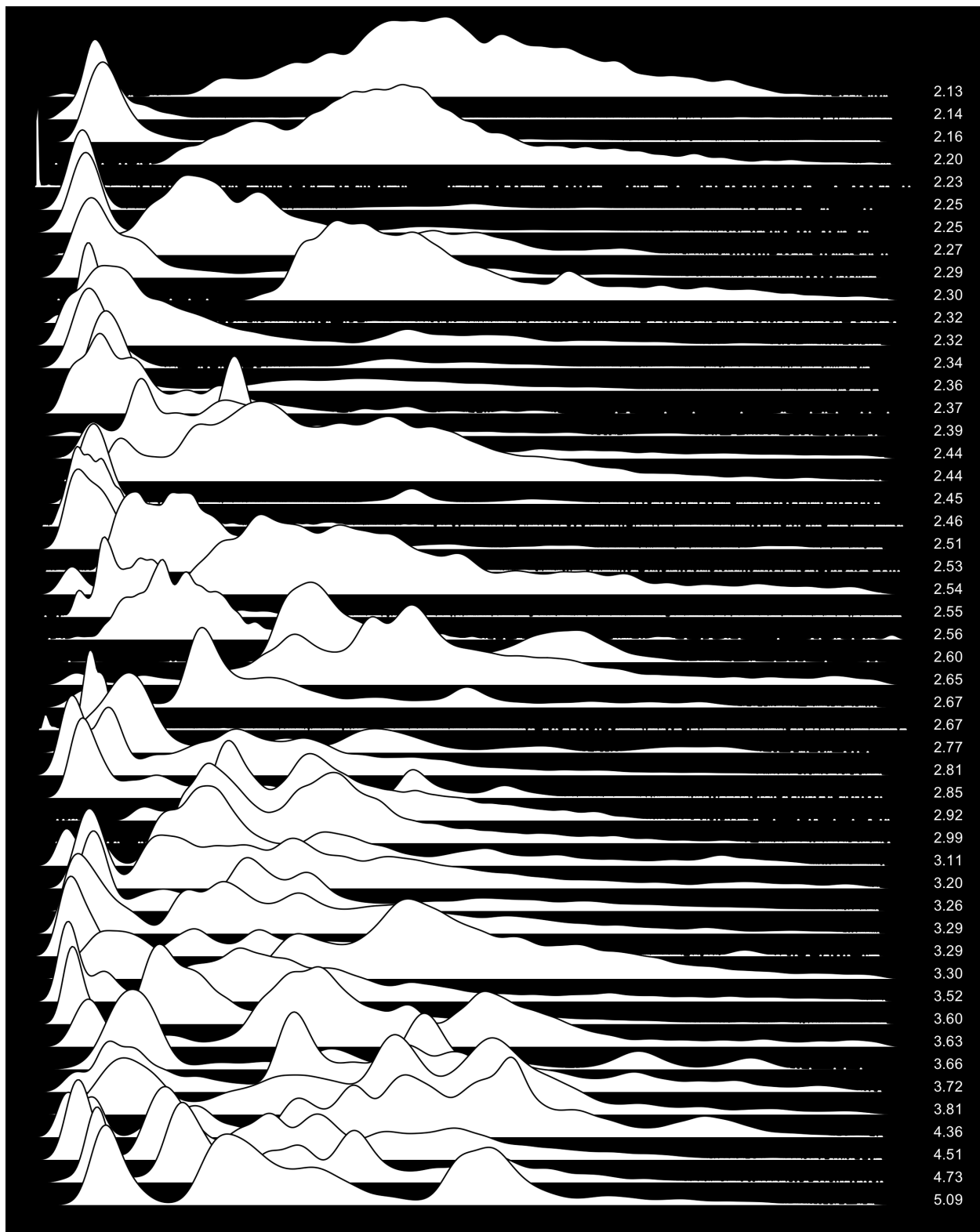
Beware: Not all things are normal Gaussians. If you want to get fancy, you can use Weibull functions to make a variety of shapes (just by adjusting λ, k):

$$f(x; \lambda, k) = \begin{cases} \frac{k}{\lambda} \left(\frac{x}{\lambda}\right)^{k-1} e^{-(x/\lambda)^k}, & x \geq 0, \\ 0, & x < 0, \end{cases}$$



Or you could forget all about parametric assumptions. Many things get improved by going beyond the Gaussian guess ³: Not everything is best represented by a smooth curve with one peak that is symmetrical around that peak:

³James Dougherty, Ron Kohavi, Mehran Sahami: Supervised and Unsupervised Discretization of Continuous Features. ICML 1995: 194-202



All that said, Gaussians take up far less space and are very easy to update. So all engineers should know their gaussians.

3 Bayes Classifier

A Bayes classifier is a simple statistical-based learning scheme.

Advantages:

- Tiny memory footprint
- Fast training, fast learning
- Simplicity
- Often works surprisingly well

Assumptions

- Learning is done best via statistical modeling
- Attributes are
 - equally important
 - statistically independent (given the class value)
 - Which means that knowledge about the value of a particular attribute doesn't tell us anything about the value of another attribute (if the class is known)

Although based on assumptions that are almost never correct, this scheme works well in practice ⁴.

⁴Pedro Domingos and Michael Pazzani. 1997. On the Optimality of the Simple Bayesian Classifier under Zero-One Loss. Mach. Learn. 29, 2-3 (November 1997), 103-130

Table 1. Classification accuracies and sample standard deviations, averaged over 20 random training/test splits. “Bayes” is the Bayesian classifier with discretization and “Gauss” is the Bayesian classifier with Gaussian distributions. Superscripts denote confidence levels for the difference in accuracy between the Bayesian classifier and the corresponding algorithm, using a one-tailed paired t test: 1 is 99.5%, 2 is 99%, 3 is 97.5%, 4 is 95%, 5 is 90%, and 6 is below 90%.

Data Set	Bayes	Gauss	C4.5	PEBLs	CN2	Def.
Audiology	73.0±6.1	73.0±6.1 ⁶	72.5±5.8 ⁶	75.8±5.4 ³	71.0±5.1 ⁵	21.3
Annealing	95.3±1.2	84.3±3.8 ¹	90.5±2.2 ¹	98.8±0.8 ¹	81.2±5.4 ¹	76.4
Breast cancer	71.6±4.7	71.3±4.3 ⁶	70.1±6.8 ⁵	65.6±4.7 ¹	67.9±7.1 ¹	67.6
Credit	84.5±1.8	78.9±2.5 ¹	85.9±2.1 ³	82.2±1.9 ¹	82.0±2.2 ¹	57.4
Chess endgames	88.0±1.4	88.0±1.4 ⁶	99.2±0.1 ¹	96.9±0.7 ¹	98.1±1.0 ¹	52.0
Diabetes	74.5±2.4	75.2±2.1 ⁶	73.5±3.4 ⁵	71.1±2.4 ¹	73.8±2.7 ⁶	66.0
Echocardiogram	69.1±5.4	73.4±4.9 ¹	64.7±6.3 ¹	61.7±6.4 ¹	68.2±7.2 ⁶	67.8
Glass	61.9±6.2	50.6±8.2 ¹	63.9±8.7 ⁶	62.0±7.4 ⁶	63.8±5.5 ⁶	31.7
Heart disease	81.9±3.4	84.1±2.8 ¹	77.5±4.3 ¹	78.9±4.0 ¹	79.7±2.9 ³	55.0
Hepatitis	85.3±3.7	85.2±4.0 ⁶	79.2±4.3 ¹	79.0±5.1 ¹	80.3±4.2 ¹	78.1
Horse colic	80.7±3.7	79.3±3.7 ¹	85.1±3.8 ¹	75.7±5.0 ¹	82.5±4.2 ²	63.6
Hypothyroid	97.5±0.3	97.9±0.4 ¹	99.1±0.2 ¹	95.9±0.7 ¹	98.8±0.4 ¹	95.3
Iris	93.2±3.5	93.9±1.9 ⁶	92.6±2.7 ⁶	93.5±3.0 ⁶	93.3±3.6 ⁶	26.5
Labor	91.3±4.9	88.7±10.6 ⁶	78.1±7.9 ¹	89.7±5.0 ⁶	82.1±6.9 ¹	65.0
Lung cancer	46.8±13.3	46.8±13.3 ⁶	40.9±16.3 ⁵	42.3±17.3 ⁶	38.6±13.5 ³	26.8
Liver disease	63.0±3.3	54.8±5.5 ¹	65.9±4.4 ¹	61.3±4.3 ⁶	65.0±3.8 ³	58.1
LED	62.9±6.5	62.9±6.5 ⁶	61.2±8.4 ⁶	55.3±6.1 ¹	58.6±8.1 ²	8.0
Lymphography	81.6±5.9	81.1±4.8 ⁶	75.0±4.2 ¹	82.9±5.6 ⁶	78.8±4.9 ³	57.3
Post-operative	64.7±6.8	67.2±5.0 ³	70.0±5.2 ¹	59.2±8.0 ²	60.8±8.2 ⁴	71.2
Promoters	87.9±7.0	87.9±7.0 ⁶	74.3±7.8 ¹	91.7±5.9 ³	75.9±8.8 ¹	43.1
Primary tumor	44.2±5.5	44.2±5.5 ⁶	35.9±5.8 ¹	30.9±4.7 ¹	39.8±5.2 ¹	24.6
Solar flare	68.5±3.0	68.2±3.7 ⁶	70.6±2.9 ¹	67.6±3.5 ⁶	70.4±3.0 ²	25.2
Sonar	69.4±7.6	63.0±8.3 ¹	69.1±7.4 ⁶	73.8±7.4 ¹	66.2±7.5 ⁵	50.8
Soybean	100.0±0.0	100.0±0.0 ⁶	95.0±9.0 ³	100.0±0.0 ⁶	96.9±5.9 ³	30.0
Splice junctions	95.4±0.6	95.4±0.6 ⁶	93.4±0.8 ¹	94.3±0.5 ¹	81.5±5.5 ¹	52.4
Voting records	91.2±1.7	91.2±1.7 ⁶	96.3±1.3 ¹	94.9±1.2 ¹	95.8±1.6 ¹	60.5
Wine	96.4±2.2	97.8±1.2 ³	92.4±5.6 ¹	97.2±1.8 ⁶	90.8±4.7 ¹	36.4
Zoology	94.4±4.1	94.1±3.8 ⁶	89.6±4.7 ¹	94.6±4.3 ⁶	90.6±5.0 ¹	39.4

4 Example

outlook	temperature	humidity	windy	play
rainy	cool	normal	TRUE	no
rainy	mild	high	TRUE	no
sunny	hot	high	FALSE	no
sunny	hot	high	TRUE	no
sunny	mild	high	FALSE	no
overcast	cool	normal	TRUE	yes
overcast	hot	high	FALSE	yes
overcast	hot	normal	FALSE	yes
overcast	mild	high	TRUE	yes
rainy	cool	normal	FALSE	yes
rainy	mild	high	FALSE	yes
rainy	mild	normal	FALSE	yes
sunny	cool	normal	FALSE	yes
sunny	mild	normal	TRUE	yes%%

This data can be summarized as follows:

Outlook			Temperature			Humidity		
=====			=====			=====		
	Yes	No		Yes	No		Yes	No
Sunny	2	3	Hot	2	2	High	3	4
Overcast	4	0	Mild	4	2	Normal	6	1
Rainy	3	2	Cool	3	1			
-----			-----			-----		
Sunny	2/9	3/5	Hot	2/9	2/5	High	3/9	4/5
Overcast	4/9	0/5	Mild	4/9	2/5	Normal	6/9	1/5
Rainy	3/9	2/5	Cool	3/9	1/5			

Windy			Play	
=====			=====	
	Yes	No	Yes	No
False	6	2	9	5
True	3	3		
-----			-----	
False	6/9	2/5	9/14	5/14
True	3/9	3/5		

So, what happens on a new day:

Outlook	Temp.	Humidity	Windy	Play
Sunny	Cool	High	True	?%%

First find the likelihood of the two classes

- For “yes” = $2/9 * 3/9 * 3/9 * 3/9 * 9/14 = 0.0053$
- For “no” = $3/5 * 1/5 * 4/5 * 3/5 * 5/14 = 0.0206$

Conversion into a probability by normalization:

- $P(\text{“yes”}) = 0.0053 / (0.0053 + 0.0206) = 0.205$
- $P(\text{“no”}) = 0.0206 / (0.0053 + 0.0206) = 0.795$

So, we aren’t playing golf today.

5 Bayes' rule

More generally, the above is just an application of Bayes' Theorem.

Probability of event H given evidence E :

$$P(H|E) = \frac{P(H) \times \prod_x P(E_x|H)}{P(E)}$$

A *priori* probability of H is $P(H)$.

- Probability of event before evidence has been seen

A *posteriori* probability of H is $P(H|E)$

- Probability of event after evidence has been seen

Classification learning: what's the probability of the class given an instance?

- Evidence E = instance
- Event H = class value for instance

Naive Bayes assumption: evidence can be split into independent parts (i.e. attributes of instance!)

$$P(H|E) = \frac{(P(E_1|H) * P(E_2|H) * \dots * P(E_n|H)) \times P(H)}{P(E)}$$

We used this above. Here's our evidence:

Outlook	Temp.	Humidity	Windy	Play
Sunny	Cool	High	True	?

Here's the probability for "yes":

$$\begin{aligned} P(\text{yes} | E) &= P(\text{Outlook} = \text{Sunny} | \text{yes}) * \\ &\quad P(\text{Temperature} = \text{Cool} | \text{yes}) * \\ &\quad P(\text{Humidity} = \text{High} | \text{yes}) * P(\text{yes}) \\ &\quad P[\text{Windy} = \text{True} | \text{yes}] * P(\text{yes}) / P(E) \\ &= (2/9 * 3/9 * 3/9 * 3/9) * 9/14 / P(E) \end{aligned}$$

Return the classification with highest probability

6 Pragmatics

6.1 Probability of the evidence $P[E]$

- Constant across all possible classifications;
- So, when comparing N classifications, it cancels out

6.2 Numerical errors

From multiplication of lots of small numbers

- Safest and slowest: Use a language with arbitrary precision arithmetic e.g. LISP
- Not so safe: use a language with big num support; e.g. Python
- Safest: don't multiply the numbers, add the logs

6.3 The “low-frequencies problem”

What if an attribute value doesn't occur with every class value (e.g. “Humidity = high” for class “yes”)?

- Probability will be zero!
- $\Pr[\text{Humidity} = \text{High} \mid \text{yes}] = 0$
- A posteriori probability will also be zero! $P[\text{yes} \mid E] = 0$ (No matter how likely the other values are!)

So use an estimators for low frequency attribute ranges

- Add a little “ m ” to the count for every attribute value-class combination + The Laplace estimator + Result: probabilities will never be zero!

And use an estimator for low frequency classes

- Add a little “ k ” to class counts + The M-estimate

Magic numbers: $m=2$, $k=1$

```
function NUM:like(x,_, nom,denom)
  local mu, sd = self:mid(), (self:div() + 1E-30)
  nom = 2.718^(-.5*(x - mu)^2/(sd^2))
  denom = (sd*2.5 + 1E-30)
  return nom/denom end

function SYM:like(x, prior)
  return ((self.has[x] or 0) + the.m*prior)/(self.n +the.m) end

function ROW:like(data,n,nHypotheses, prior,out,v,inc)
  prior = (#data.rows + the.k) / (n + the.k * nHypotheses)
  out = math.log(prior)
  for _,col in pairs(data.cols.x) do
    v= self.cells[col.at]
    if v ~= "?" then
      inc = col:like(v,prior)
      out = out + math.log(inc) end end
  return math.exp(1)^out end
```

7 Handling numerics

The above assumes that the attributes are discrete. The usual approximation is to assume a “Gaussian” (i.e. a “normal” or “bell-shaped” curve) for the numerics.

The probability density function for the normal distribution is defined by the mean and standardDev (standard deviation)

- m = mean
- s = standard deviation
- x = the value we are trying to score (will be maximum if x is the mean)

Code:

```
function norm(x,m,s) {  
  pi= 3.1415926535  
  e = 2.7182818284  
  a = 1/sqrt(2*pi*s^2)  
  b = (x-m)^2/(2*s^2)  
  return a * e ^ (-1*b)  
}
```

For example:

outlook	temperature	humidity	windy	play
sunny	85	85	FALSE	no
sunny	80	90	TRUE	no
overcast	83	86	FALSE	yes
rainy	70	96	FALSE	yes
rainy	68	80	FALSE	yes
rainy	65	70	TRUE	no
overcast	64	65	TRUE	yes
sunny	72	95	FALSE	no
sunny	69	70	FALSE	yes
rainy	75	80	FALSE	yes
sunny	75	70	TRUE	yes
overcast	72	90	TRUE	yes
overcast	81	75	FALSE	yes
rainy	71	91	TRUE	no

This generates the following statistics:

Outlook			Temperature			Humidity		
	Yes	No		Yes	No		Yes	No
Sunny	2	3		83	85		86	85
Overcast	4	0		70	80		96	90
Rainy	3	2		68	65		80	70
-----			-----			-----		
Sunny	2/9	3/5	mean	73	74.6	mean	79.1	86.2
Overcast	4/9	0/5	std dev	6.2	7.9	std dev	10.2	9.7
Rainy	3/9	2/5						

Windy			Play	
	Yes	No	Yes	No
False	6	2	9	5
True	3	3		
-----			-----	
False	6/9	2/5	9/14	5/14
True	3/9	3/5		

Example density value:

- $f(\text{temperature}=66|\text{yes}) = \text{norm}(66, 73, 6.2) = 0.0340$

Classifying a new day:

Outlook	Temp.	Humidity	Windy	Play
Sunny	66	90	true	??%

- Likelihood of “yes” = $2/9 * 0.0340 * 0.0221 * 3/9 * 9/14 = 0.000036$
- Likelihood of “no” = $3/5 * 0.0291 * 0.0380 * 3/5 * 5/14 = 0.000136$
- So:
 - $P(\text{“yes”}) = 0.000036 / (0.000036 + 0.000136) = 20.9\%$
 - $P(\text{“no”}) = 0.000136 / (0.000036 + 0.000136) = 79.1\%$

Note: missing values during training: not included in calculation of mean and standard deviation

BTW, an alternative to the above is apply some discretization policy to the data. Such discretization is good practice since it can dramatically improve the performance of a Naive Bayes classifier (see ⁵):

⁵James Dougherty, Ron Kohavi, Mehran Sahami: Supervised and Unsupervised Discretization of Continuous Features. ICML 1995: 194-202