Statistical modeling

- "Opposite" of 1R: use all the attributes
- Two assumptions: Attributes are
  - *equally important*
  - *statistically independent* (given the class value)
    - This 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!

Probabilities for the weather data

Outlook			Temperature			Humidity			Windy			Play	
	Yes	*No*		*Yes*	*No*		*Yes*	*No*		*Yes*	*No*	*Yes*	*No*
Sunny	2	3	Hot	2	2	High	3	4	False	6	2	9	5
Overcast	4	0	Mild	4	2	Normal	6	1	True	3	3		
Rainy	3	2	Cool	3	1								
Sunny	2/9	3/5	Hot	2/9	2/5	High	3/9	4/5	False	6/9	2/5	9/14	5/14
Overcast	4/9	0/5	Mild	4/9	2/5	Normal	6/9	1/5	True	3/9	3/5		
Rainy	3/9	2/5	Cool	3/9	1/5								

- A new day:

Outlook	Temp.	Humidity	Windy	Play
Sunny	Cool	High	True	?

Likelihood of the two classes

For “yes” = $2/9 \times 3/9 \times 3/9 \times 3/9 \times 9/14 = 0.0053$

For “no” = $3/5 \times 1/5 \times 4/5 \times 3/5 \times 5/14 = 0.0206$

Conversion into a probability by normalization:

P(“yes”) = 0.0053 / (0.0053 + 0.0206) = 0.205

P(“no”) = 0.0206 / (0.0053 + 0.0206) = 0.795

Bayes's rule

- Probability of event *H* given evidence *E*:

$$\Pr[H \mid E] = \frac{\Pr[E \mid H]\Pr[H]}{\Pr[E]}$$

- *A priori* probability of *H*: $\Pr[H]$
  - Probability of event *before* evidence has been seen
- *A posteriori* probability of *H*: $\Pr[H \mid E]$
  - Probability of event *after* evidence has been seen

Naïve Bayes for classification

- Classification learning: what's the probability of the class given an instance?
  - Evidence E = instance
  - Event H = class value for instance
- Naïve Bayes assumption: evidence can be split into independent parts (i.e. attributes of instance!)

$$\Pr[H \mid E] = \frac{\Pr[E_1 \mid H]\Pr[E_1 \mid H]\ldots\Pr[E_n \mid H]\Pr[H]}{\Pr[E]}$$

The weather data example

Outlook	Temp.	Humidity	Windy	Play
Sunny	Cool	High	True	?

Evidence E

Probability for class "yes"

$$\Pr[yes \mid E] = \Pr[Outlook = Sunny \mid yes] \times$$

$$\Pr[Temperature = Cool \mid yes] \times$$

$$\Pr[Humdity = High \mid yes] \times$$

$$\Pr[Windy = True \mid yes] \times \frac{\Pr[yes]}{\Pr[E]}$$

$$= \frac{2/9 \times 3/9 \times 3/9 \times 3/9 \times 9/14}{\Pr[E]}$$

The “zero-frequency problem”

- What if an attribute value doesn’t occur with every class value (e.g. “Humidity = high” for class “yes”)?
  - Probability will be zero! $\Pr[Humdity = High \mid yes] = 0$
  - *A posteriori* probability will also be zero! $\Pr[yes \mid E] = 0$

    (No matter how likely the other values are!)
- Remedy: add 1 to the count for every attribute value-class combination (*Laplace estimator)*
- Result: probabilities will never be zero! (also: stabilizes probability estimates)

Modified probability estimates

- In some cases adding a constant different from 1 might be more appropriate
- Example: attribute *outlook* for class *yes*

$$\frac{2+\mu/3}{9+\mu} \qquad \frac{4+\mu/3}{9+\mu} \qquad \frac{3+\mu/3}{9+\mu}$$

Sunny *Overcast* *Rainy*

- Weights don't need to be equal (if they sum to 1)

$$\frac{2+\mu p_1}{9+\mu} \qquad \frac{4+\mu p_2}{9+\mu} \qquad \frac{3+\mu p_3}{9+\mu}$$

Missing values

- Training: instance is not included in frequency count for attribute value-class combination
- Classification: attribute will be omitted from calculation
- Example:

Outlook	Temp.	Humidity	Windy	Play
?	Cool	High	True	?

Likelihood of “yes” = $3/9 \times 3/9 \times 3/9 \times 9/14 = 0.0238$

Likelihood of “no” = $1/5 \times 4/5 \times 3/5 \times 5/14 = 0.0343$

P(“yes”) = 0.0238 / (0.0238 + 0.0343) = 41%

P(“no”) = 0.0343 / (0.0238 + 0.0343) = 59%

Dealing with numeric attributes

- Usual assumption: attributes have a *normal* or *Gaussian* probability distribution (given the class)
- The *probability density function* for the normal distribution is defined by two parameters:
  - The *sample mean* μ: $$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$$
  - The *standard deviation* σ: $$\sigma = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \mu)^2$$
  - The density function *f(x)*: $$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Statistics for the weather data

Outlook			Temperature			Humidity			Windy			Play	
	Yes	*No*		*Yes*	*No*		*Yes*	*No*		*Yes*	*No*	*Yes*	*No*
Sunny	2	3		83	85		86	85	False	6	2	9	5
Overcast	4	0		70	80		96	90	True	3	3		
Rainy	3	2		68	65		80	70					
				…	…		…	…					
Sunny	2/9	3/5	*mean*	73	74.6	*mean*	79.1	86.2	False	6/9	2/5	9/14	5/14
Overcast	4/9	0/5	*std dev*	6.2	7.9	*std dev*	10.2	9.7	True	3/9	3/5		
Rainy	3/9	2/5											

- Example density value:

$$f(temperature = 66 \mid yes) = \frac{1}{\sqrt{2\pi}\,6.2} e^{-\frac{(66-73)^2}{2*6.2^2}} = 0.0340$$

Classifying a new day

- A new day:

Outlook	Temp.	Humidity	Windy	Play
Sunny	66	90	true	?

Likelihood of “yes” = $2/9 \times 0.0340 \times 0.0221 \times 3/9 \times 9/14 = 0.000036$

Likelihood of “no” = $3/5 \times 0.0291 \times 0.0380 \times 3/5 \times 5/14 = 0.000136$

P(“yes”) = 0.000036 / (0.000036 + 0. 000136) = 20.9%

P(“no”) = 0. 000136 / (0.000036 + 0. 000136) = 79.1%

- Missing values during training: not included in calculation of mean and standard deviation

Probability densities

- Relationship between probability and density:

$$\Pr[c - \frac{\varepsilon}{2} < x < c + \frac{\varepsilon}{2}] \approx \varepsilon * f(c)$$

- But: this doesn't change calculation of *a posteriori* probabilities because ε cancels out
- Exact relationship:

$$\Pr[a \le x \le b] = \int_a^b f(t)dt$$

Discussion of Naïve Bayes

- Naïve Bayes works surprisingly well (even if independence assumption is clearly violated)
- Why? Because classification doesn't require accurate probability estimates *as long as maximum probability is assigned to correct class*
- However: adding too many redundant attributes will cause problems (e.g. identical attributes)
- Note also: many numeric attributes are not normally distributed ($\rightarrow$ *kernel density estimators*)

Linear models

- Work most naturally with numeric attributes
- Standard technique for numeric prediction: linear regression
  - Outcome is linear combination of attributes

$$x = w_0 + w_1 a_1 + w_2 a_2 + ... + w_k a_k$$

- Weights are calculated from the training data
- Predicted value for first training instance **a**$^{(1)}$

$$w_0 a_0^{(1)} + w_1 a_1^{(1)} + w_2 a_2^{(1)} + ... + w_k a_k^{(1)} = \sum_{j=0}^{k} w_j a_j^{(1)}$$

Minimizing the squared error

- *k*+1 coefficients are chosen so that the squared error on the training data is minimized
- Squared error: $$\sum_{i=1}^{n}\left(x^{(i)}-\sum_{j=0}^{k}w_j a_j^{(i)}\right)^2$$
- Coefficient can be derived using standard matrix operations
- Can be done if there are more instances than attributes (roughly speaking)
- Minimization of *absolute error* is more difficult!

Classification

- *Any* regression technique can be used for classification
  - Training: perform a regression for each class, setting the output to 1 for training instances that belong to class, and 0 for those that don't
  - Prediction: predict class corresponding to model with largest output value (*membership value*)
- For linear regression this is known as *multi-response linear regression*

Theoretical justification

Observed target value (either 0 or 1)

Model

Instance

The scheme minimizes this

$$E_y\{(f(X)-Y)^2 \mid X=x\}$$

True class probability

$$= E_y\{(f(X)-P(Y=1 \mid X=x)+P(Y=1 \mid X=x)-Y)^2 \mid X=x\}$$

$$=(f(x)-P(Y=1 \mid X=x))^2+2\times(f(x)-P(Y=1 \mid X=x))\times$$
$$E_y\{P(Y=1 \mid X=x)-Y \mid X=x\}+E_y\{(P(Y=1 \mid X=x)-Y)^2 \mid X=x\}$$

$$=(f(x)-P(Y=1 \mid X=x))^2+2\times(f(x)-P(Y=1 \mid X=x))\times$$
$$(P(Y=1 \mid X=x)-E_y\{Y \mid X=x\})+E_y\{(P(Y=1 \mid X=x)-Y)^2 \mid X=x\}$$

$$=\underbrace{(f(x)-P(Y=1 \mid X=x))^2}_{\text{We want to minimize this}}+\underbrace{E_y\{(P(Y=1 \mid X=x)-Y)^2 \mid X=x\}}_{\text{Constant}}$$

Pairwise regression

- Another way of using regression for classification:
  - A regression function for every *pair* of classes, using only instances from these two classes
  - An output of +1 is assigned to one member of the pair, an output of –1 to the other
- Prediction is done by voting
  - Class that receives most votes is predicted
  - Alternative: “don’t know” if there is no agreement
- More likely to be accurate but more expensive

Logistic regression

- Problem: some assumptions violated when linear regression is applied to classification problems
- *Logistic* regression: alternative to linear regression
  - Designed for classification problems
  - Tries to estimate class probabilities directly
    - Does this using the *maximum likelihood* method
  - Uses the following linear model:

$$\log(P/(1-P) = w_0 a_0 + w_1 a_1 + w_2 a_2 + \ldots + w_k a_k$$

Class probability

Discussion of linear models

- Not appropriate if data exhibits non-linear dependencies
- But: can serve as building blocks for more complex schemes (i.e. model trees)
- Example: multi-response linear regression defines a *hyperplane* for any two given classes:

$$(w_0^{(1)} - w_0^{(2)})a_0 + (w_1^{(1)} - w_1^{(2)})a_1 + (w_2^{(1)} - w_2^{(2)})a_2 + \ldots + (w_k^{(1)} - w_k^{(2)})a_k > 0$$

- Obviously the same for pairwise linear regression

Instance-based learning

- Distance function defines what's learned
- Most instance-based schemes use *Euclidean distance*:

$$\sqrt{(a_1^{(1)}-a_1^{(2)})^2+(a_2^{(1)}-a_2^{(2)})^2+...+(a_k^{(1)}-a_k^{(2)})^2}$$

  $\mathbf{a}^{(1)}$ and $\mathbf{a}^{(2)}$: two instances with k attributes
- Taking the square root is not required when comparing distances
- Other popular metric: *city-block metric*
  - Adds differences without squaring them

Normalization and other issues

- Different attributes are measured on different scales ⇒ they need to be *normalized*:

$$a_i = \frac{v_i - \min v_i}{\max v_i - \min v_i}$$

  v_i: the actual value of attribute *i*
- Nominal attributes: distance either 0 or 1
- Common policy for missing values: assumed to be maximally distant (given normalized attributes)

Discussion of 1-NN

- Often very accurate but also slow: simple version scans entire training data to derive a prediction
- Assumes all attributes are equally important
  - Remedy: attribute selection or weights
- Possible remedies against noisy instances:
  - Taking a majority vote over the *k* nearest neighbors
  - Removing noisy instances from dataset (difficult!)
- Statisticians have used *k*-NN since early 1950s
  - If $n \rightarrow \infty$ and $k/n \rightarrow 0$, error approaches minimum