# Classification

MLAI: Week 9

Neil D. Lawrence

Department of Computer Science
Sheffield University

25th November 2014

# Review

- Last time: Looked at generalisation and validation.
- Introduced cross validation, hold out validation, reviewed training and test sets.
- This time: Classification.

# Outline

# Classification

- We are given data set containing “inputs”, $\mathbf{X}$, and “targets”, $\mathbf{y}$.
- Each data point consists of an input vector $\mathbf{x}_{i,:}$ and a class label, $y_i$.
- For binary classification assume $y_i$ should be either 1 (yes) or $-1$ (no).
- Input vector can be thought of as features.

# Classification Examples

- Classifying hand written digits from binary images (automatic zip code reading).
- Detecting faces in images (e.g. digital cameras).
- Who a detected face belongs to (e.g. Picasa).
- Classifying type of cancer given gene expression data.
- Categorization of document types (different types of news article on the internet).

# The Perceptron

- Developed in 1957 by Rosenblatt.
- Take a data point at, $\mathbf{x}_i$.
- Predict it belongs to a class, $y_i = 1$ if $\sum_j w_j \mathbf{x}_{i,j} + b > 0$ i.e. $\mathbf{w}^\top \mathbf{x}_i + b > 0$. Otherwise assume $y_i = -1$.

# Perceptron-like Algorithm

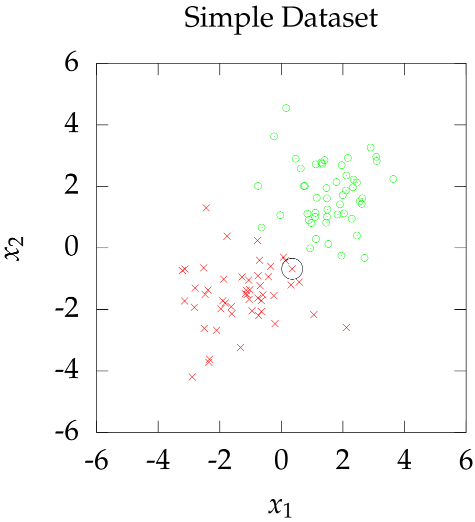
1. Select a random data point $i$.
2. Ensure $i$ is correctly classified by setting $\mathbf{w} = y_i \mathbf{x}_i$.
    - i.e. $\text{sign}\left(\mathbf{w}^\top \mathbf{x}_{i,:}\right) = \text{sign}\left(y_i \mathbf{x}_{i,:}^\top \mathbf{x}_{i,:}\right) = \text{sign}\left(y_i\right) = y_i$

# Perceptron Iteration

1. Select a misclassified point, $i$.
2. Set $\mathbf{w} \leftarrow \mathbf{w} + \eta y_i \mathbf{x}_{i,:}$.
   - If $\eta$ is large enough this will guarantee this point becomes correctly classified.
3. Repeat until there are no misclassified points.

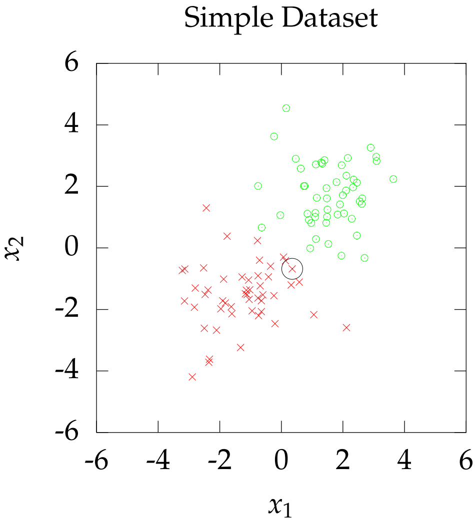
# Perceptron Algorithm

- Iteration 1 data no 29



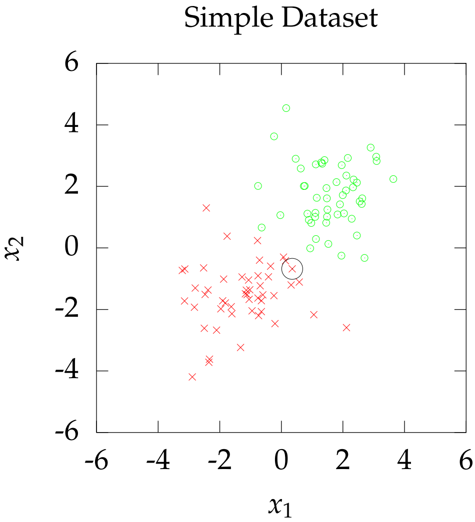
# Perceptron Algorithm

- Iteration 1 data no 29
- $w_1 = 0, w_2 = 0$



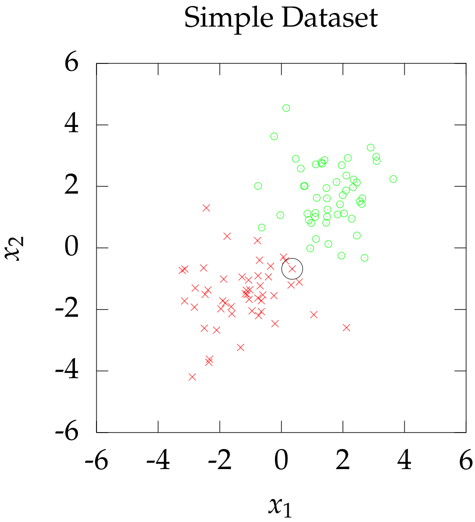
# Perceptron Algorithm

- Iteration 1 data no 29
- $w_1 = 0, w_2 = 0$
- First Iteration



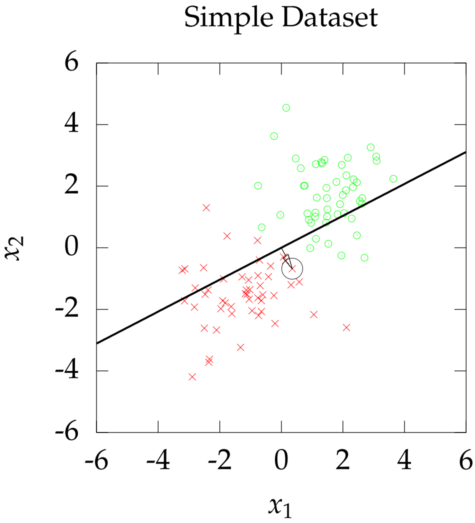
# Perceptron Algorithm

- Iteration 1 data no 29
- $w_1 = 0, w_2 = 0$
- First Iteration
- Set weight vector to data point.



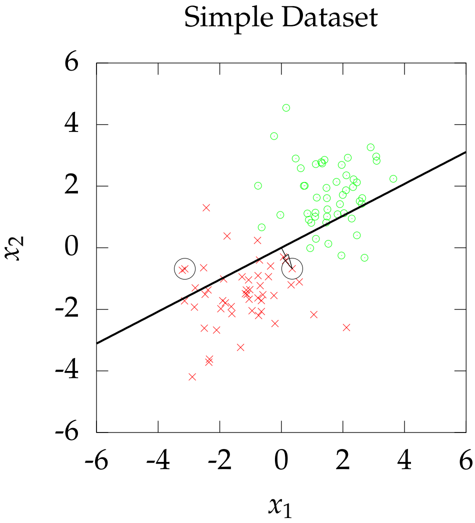
# Perceptron Algorithm

- Iteration 1 data no 29
- $w_1 = 0, w_2 = 0$
- First Iteration
- Set weight vector to data point.
- $\mathbf{w} = y_{29}\mathbf{x}_{29,:}$



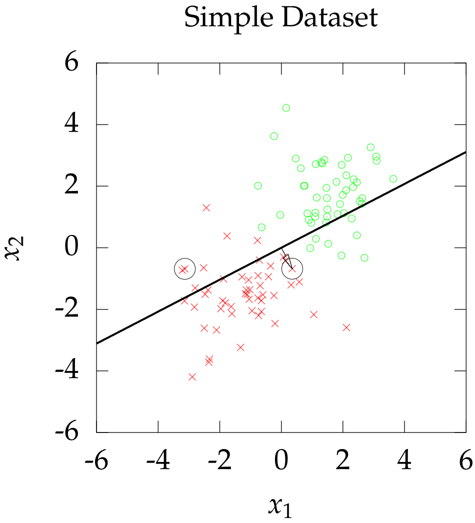
# Perceptron Algorithm

- Iteration 1 data no 29
- $w_1 = 0, w_2 = 0$
- First Iteration
- Set weight vector to data point.
- $\mathbf{w} = y_{29}\mathbf{x}_{29,:}$
- Select new incorrectly classified data point.



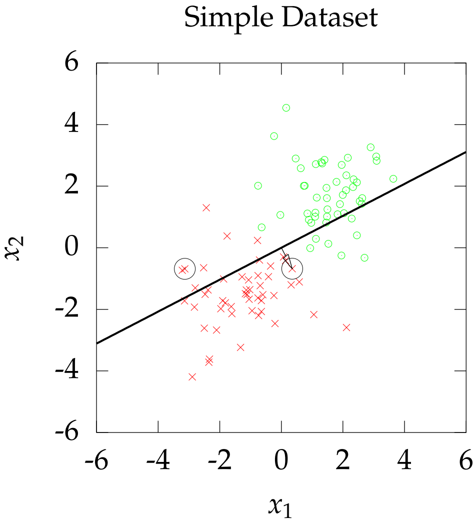
# Perceptron Algorithm

- Iteration 2 data no 16



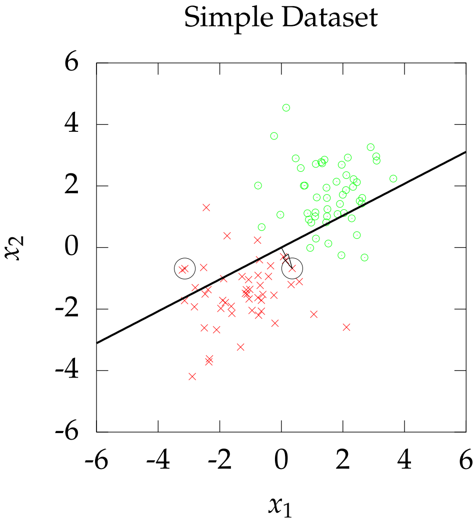
# Perceptron Algorithm

- Iteration 2 data no 16
- $w_1 = 0.3519$,
  $w_2 = -0.6787$



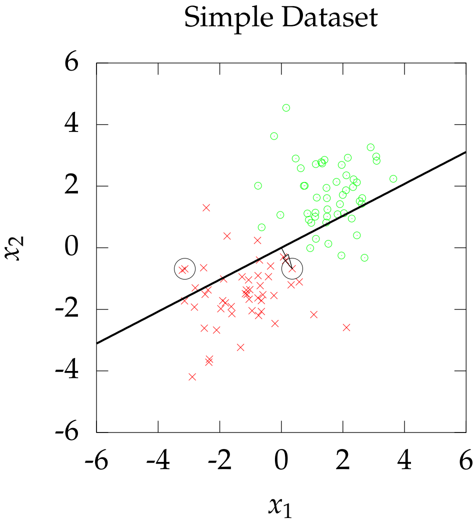
# Perceptron Algorithm

- Iteration 2 data no 16
- $w_1 = 0.3519$,
  $w_2 = -0.6787$
- Incorrect classification



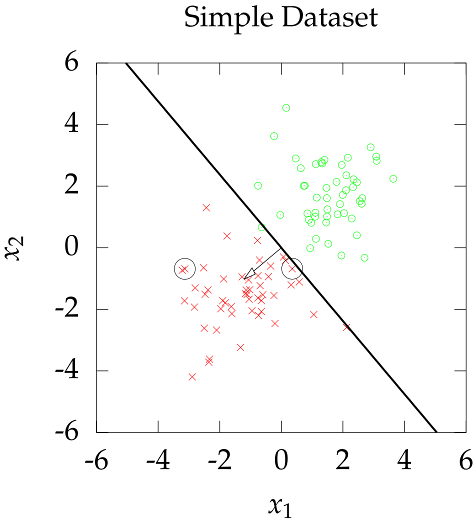
# Perceptron Algorithm

- Iteration 2 data no 16
- $w_1 = 0.3519$,
  $w_2 = -0.6787$
- Incorrect classification
- Adjust weight vector with new data point.



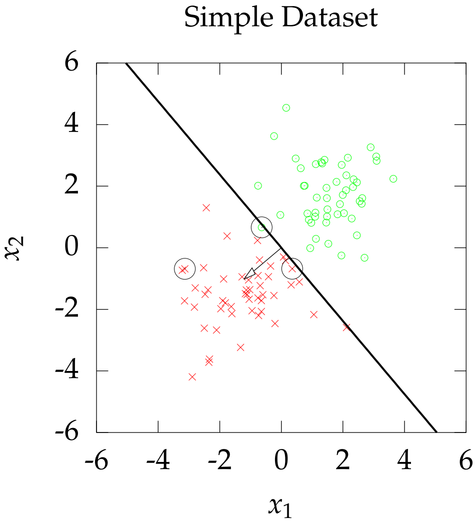
# Perceptron Algorithm

- Iteration 2 data no 16
- $w_1 = 0.3519$,
  $w_2 = -0.6787$
- Incorrect classification
- Adjust weight vector with new data point.
- $\mathbf{w} \leftarrow \mathbf{w} + \eta y_{16} \mathbf{x}_{16,:}$



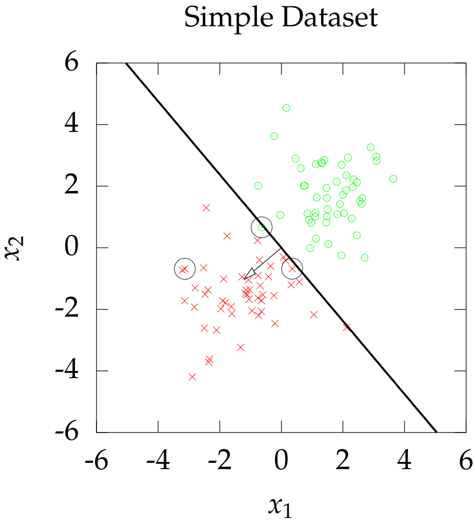
# Perceptron Algorithm

- Iteration 2 data no 16
- $w_1 = 0.3519$,
  $w_2 = -0.6787$
- Incorrect classification
- Adjust weight vector with new data point.
- $\mathbf{w} \leftarrow \mathbf{w} + \eta y_{16} \mathbf{x}_{16,:}$
- Select new incorrectly classified data point.



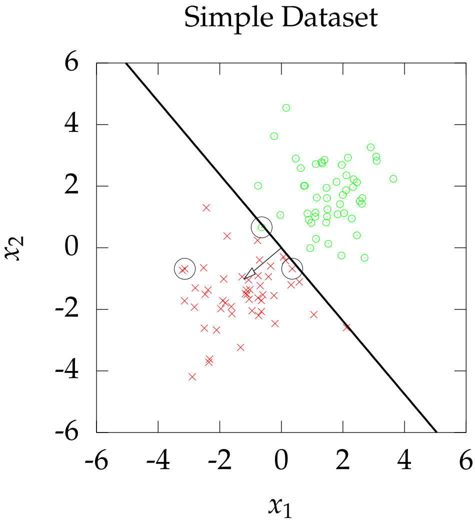
# Perceptron Algorithm

- Iteration 3 data no 58



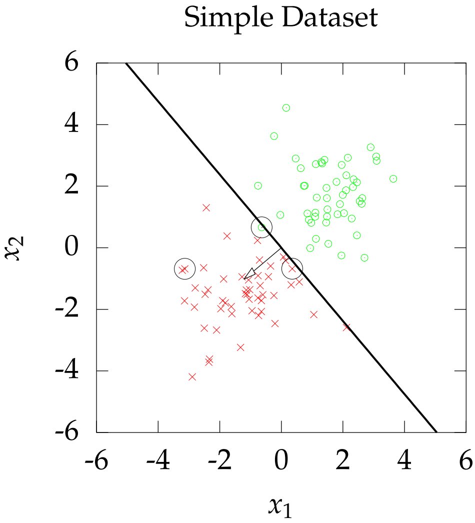
# Perceptron Algorithm

- Iteration 3 data no 58
- $w_1 = -1.2143$,
  $w_2 = -1.0217$



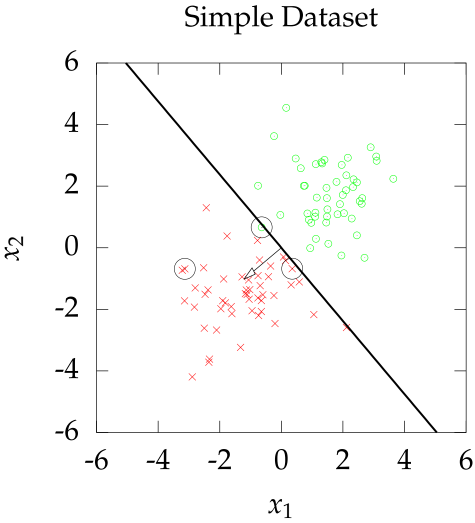
# Perceptron Algorithm

- Iteration 3 data no 58
- $w_1 = -1.2143$,
  $w_2 = -1.0217$
- Incorrect classification



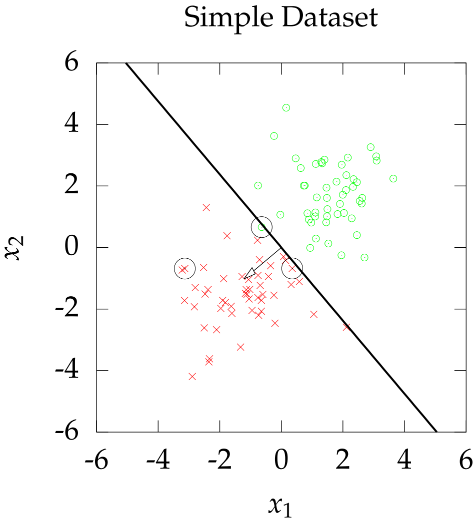
# Perceptron Algorithm

- Iteration 3 data no 58
- $w_1 = -1.2143$,
  $w_2 = -1.0217$
- Incorrect classification
- Adjust weight vector with new data point.



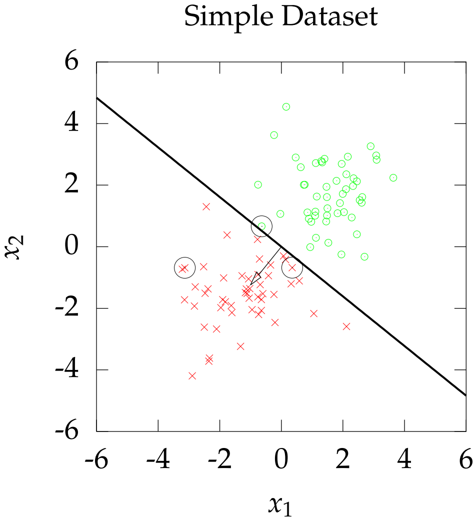
# Perceptron Algorithm

- Iteration 3 data no 58
- $w_1 = -1.2143$,
  $w_2 = -1.0217$
- Incorrect classification
- Adjust weight vector with new data point.
- $\mathbf{w} \leftarrow \mathbf{w} + \eta y_{58} \mathbf{x}_{58,:}$



# Perceptron Algorithm

- Iteration 3 data no 58
- $w_1 = -1.2143$,
  $w_2 = -1.0217$
- Incorrect classification
- Adjust weight vector with new data point.
- $\mathbf{w} \leftarrow \mathbf{w} + \eta y_{58} \mathbf{x}_{58,:}$
- All data correctly classified.



# Outline

# Bayesian Approach

- Likelihood for the regression example has the form

$$p(\mathbf{y}|\mathbf{w}, \sigma^2) = \prod_{i=1}^{n} \mathcal{N}\left(y_i|\mathbf{w}^\top \boldsymbol{\phi}_i, \sigma^2\right).$$

- Suggestion was to maximize this likelihood with respect to $\mathbf{w}$.
- This can be done with gradient based optimization of the log likelihood.
- Alternative approach: integration across $\mathbf{w}$.
- Consider expected value of likelihood under a range of potential $\mathbf{w}$s.
- This is known as the *Bayesian* approach.

# Note on the Term Bayesian

- We will use Bayes' rule to invert probabilities in the Bayesian approach.
  - Bayesian is not named after Bayes' rule (v. common confusion).
  - The term Bayesian refers to the treatment of the parameters as stochastic variables.
  - This approach was proposed by **?** and **?** independently.
  - For early statisticians this was very controversial (Fisher et al).

# Bernoulli Distribution

- Jacob Bernoulli described this distribution in terms of an ‘urn’.

# Bernoulli Distribution

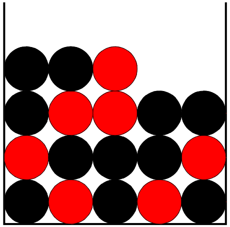
- Jacob Bernoulli described this distribution in terms of an ‘urn’.
- Write as a function

$$P(Y = y) = \pi^y (1 - \pi)^{1-y}$$

# Bernoulli Distribution

- Jacob Bernoulli described this distribution in terms of an 'urn'.
- Write as a function

$$P(Y = y) = \pi^y (1 - \pi)^{1-y}$$

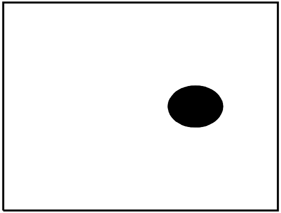


# Bernoulli Distribution Revisited

- Thomas Bayes considered a ball landing uniformly across a table.

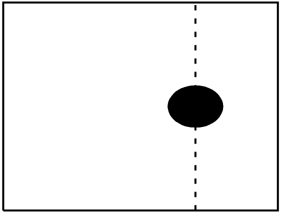
# Bernoulli Distribution Revisited

- Thomas Bayes considered a ball landing uniformly across a table.



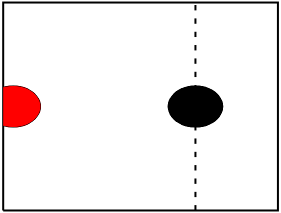
# Bernoulli Distribution Revisited

- Thomas Bayes considered a ball landing uniformly across a table.
- And another ball landing on the left or right (**?**, page 385).



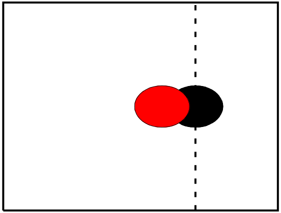
# Bernoulli Distribution Revisited

- Thomas Bayes considered a ball landing uniformly across a table.
- And another ball landing on the left or right (**?**, page 385).



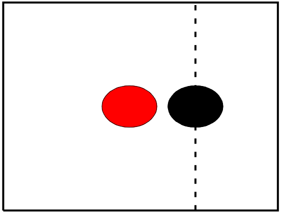
# Bernoulli Distribution Revisited

- Thomas Bayes considered a ball landing uniformly across a table.
- And another ball landing on the left or right (**?**, page 385).
- The position of the first ball gives the parameter $\pi$.



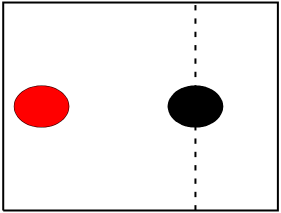
# Bernoulli Distribution Revisited

- Thomas Bayes considered a ball landing uniformly across a table.
- And another ball landing on the left or right (**?**, page 385).
- The position of the first ball gives the parameter $\pi$.
- That 'parameter' is *itself* a random variable.



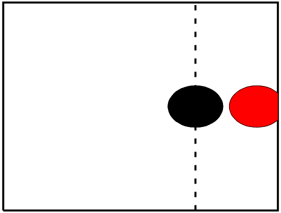
# Bernoulli Distribution Revisited

- Thomas Bayes considered a ball landing uniformly across a table.
- And another ball landing on the left or right (**?**, page 385).
- The position of the first ball gives the parameter $\pi$.
- That 'parameter' is *itself* a random variable.
- This treatment of a parameter, $\pi$, as a random variable that was/is considered controversial.



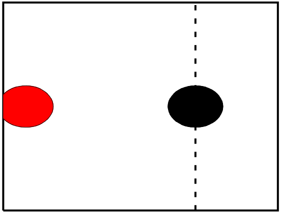
# Bernoulli Distribution Revisited

- Thomas Bayes considered a ball landing uniformly across a table.
- And another ball landing on the left or right (**?**, page 385).
- The position of the first ball gives the parameter $\pi$.
- That 'parameter' is *itself* a random variable.
- This treatment of a parameter, $\pi$, as a random variable that was/is considered controversial.



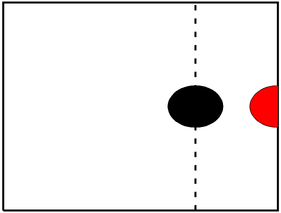
# Bernoulli Distribution Revisited

- Thomas Bayes considered a ball landing uniformly across a table.
- And another ball landing on the left or right (**?**, page 385).
- The position of the first ball gives the parameter $\pi$.
- That 'parameter' is *itself* a random variable.
- This treatment of a parameter, $\pi$, as a random variable that was/is considered controversial.



# Bernoulli Distribution Revisited

- Thomas Bayes considered a ball landing uniformly across a table.
- And another ball landing on the left or right (**?**, page 385).
- The position of the first ball gives the parameter $\pi$.
- That 'parameter' is *itself* a random variable.
- This treatment of a parameter, $\pi$, as a random variable that was/is considered controversial.



# Bayesian Controversy

- Bayesian controversy relates to treating *epistemic* uncertainty as *aleatoric* uncertainty.
- Another analogy:
  - Before a football match the uncertainty about the result is *aleatoric*.
  - If I watch a recorded match *without* knowing the result the uncertainty is *epistemic*.

# Simple Bayesian Inference

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{marginal likelihood}}$$

- Four components:
  1. Prior distribution: represents belief about parameter values before seeing data.
  2. Likelihood: gives relation between parameters and data.
  3. Posterior distribution: represents updated belief about parameters after data is observed.
  4. Marginal likelihood: represents assessment of the quality of the model. Can be compared with other models (likelihood/prior combinations). Ratios of marginal likelihoods are known as Bayes factors.

# Outline

# Naive Bayes

- Recall first lecture: Probabilities over everything.
- Covariances, $\mathbf{x}$, & response $\mathbf{y}$.

# Prediction Reminder

- Idea in Machine Learning: Joint Distribution over *everything*.
- Reformulate joint distribution using *sum* and *product* rules to answer question we want.
- First construct model: $P(y^*, \mathbf{x}_*, \mathbf{y}, \mathbf{X})$
- Then make prediction:

$$P(y^*|\mathbf{x}_*, \mathbf{y}, \mathbf{X})$$

  can be found using product rule of probability.

# Model

1. *Data Conditional Independence* There are parameters of the model, $\boldsymbol{\theta}$, and conditioned on these parameters all data points in the model are independent.

$$P(y^*, \mathbf{x}_*, \mathbf{y}, \mathbf{X}|\boldsymbol{\theta}) = P(y^*, \mathbf{x}^*|\boldsymbol{\theta}) \prod_{i=1}^{n} P(y_i, \mathbf{x}_i|\boldsymbol{\theta})$$

2. *Feature Conditional Independence* The covariates/features of the model are *also* conditionally independent given the label.

$$P(\mathbf{x}_i|y_i, \boldsymbol{\theta}) = \prod_{j=1}^{q} p(x_{i,j}|y_i, \boldsymbol{\theta})$$

where $q$ is the covariate dimensionality.

# Model

- These two assumptions are enough to begin to specify our model.
- We further need a *marginal* distribution over the data labels,
$$p(y_i|\pi) = y_i^{\pi}(1 - y_i)^{(1-\pi)}$$
- Which we can specify as *Bernoulli* because it is the most general form. $\pi$ is the probability of a positive class.
- This equips us to specify the *joint* distribution for a single data point using the product rule.

$$p(y_i, \mathbf{x}_i|\boldsymbol{\theta}) = p(y_i) \prod_{j=1}^{q} p(x_{i,j}|y_i\boldsymbol{\theta})$$

# The Joint Probability of the Training Data

We can now *fit* the *joint probability* to our data $\mathbf{y}$, $\mathbf{X}$.

- Using sum rule and *data conditional independence* we have

$$
\begin{aligned}
P(\mathbf{y}, \mathbf{X}|\boldsymbol{\theta}) &= \sum_{y^*} \sum_{\mathbf{y}^*} P(y^*, \mathbf{x}^*, \mathbf{y}, \mathbf{X}|\boldsymbol{\theta}) \\
&= \prod_{i=1}^{n} P(y_i, \mathbf{x}_i|\boldsymbol{\theta}) \sum_{y^*} \sum_{\mathbf{y}^*} P(y^*, \mathbf{x}^*) \\
&= \prod_{i=1}^{n} P(y_i, \mathbf{x}_i|\boldsymbol{\theta})
\end{aligned}
$$

# The Joint Probability of a Training Point

We now need to specify the joint distribution for a single point.

- Using product rule and *feature conditional independence*.

$$P(y_i, \mathbf{x}_i|\boldsymbol{\theta}) = P(y_i)P(\mathbf{x}_i|y_i, \boldsymbol{\theta}) \qquad = P(y_i)\prod_{i,j} P(x_{i,j}|y_i, \boldsymbol{\theta})$$

GOT TO NHERE!

# Reading

# Outline

# Generalised Linear Models

- Link function

# Logit: Predicting the Log Odds

- ..

# Logit: Interpretation as a Squashing Function

- ..

# Reading

# References I