Why probabilities?

Why are we bring here a Bayesian framework?

Recall Classification framework:

**Given:** Training data: \((x_1, y_1), \ldots, (x_n, y_n)\) \(x_i \in \mathbb{R}^d\) and \(y_i \in \mathcal{Y}\).

**Task:** Learn a classification function: \(f: \mathbb{R}^d \rightarrow \mathcal{Y}\).

Learn a mapping from \(x\) to \(y\).

We would like to find this mapping \(f(x) = y\) through \(p(y|x)\)!
Discriminative Algorithms

- **Discriminative Algorithms:**
  - Idea: model $p(y|x)$, conditional distribution of $y$ given $x$.
  - In Discriminative Algorithms: find a decision boundary that separates positive from negative example.
  - To predict a new example, check on which side of the decision boundary it falls.
  - Model $p(y|x)$ directly.
Generative Algorithms

- Generative Algorithms adopt a different approach:
  - Idea: Build a model for what positive examples look like. Build a different model for what negative example look like.
  - To predict a new example, match it with each of the models and see which which match is best.
  - Model $p(x|y)$ and $p(y)$!
  - Use Bayes rule to obtain $p(y|x) = \frac{p(x|y)p(y)}{p(x)}$. 
Generative Algorithms

- **Generative Algorithms** adopt a different approach:
  - Idea: Build a model for what positive examples look like. Build a different model for what negative example look like.
  - To predict a new example, match it with each of the models and see which match is best.
  - Model $p(x|y)$ and $p(y)$!
  - Use Bayes rule to obtain $p(y|x) = \frac{p(x|y)p(y)}{p(x)}$.
  - To make a prediction:

$$\arg\max_y p(y|x) = \arg\max_y \frac{p(x|y)p(y)}{p(x)}$$

$$\arg\max_y p(y|x) \approx \arg\max_y p(x|y)p(y)$$
Naive Bayes Classifier

- Probabilistic model.
- Highly practical method.
- Application domains to natural language text documents.
- Naive because of the strong independence assumption it makes (not realistic).
- Simple model.
- Strong method can be comparable to decision trees and neural networks in some cases.
Setting

- A training data \((x_i, y_i)\), \(x_i\) is a feature vector and \(y_i\) is a discrete label.
- \(d\) features, and \(n\) examples.
- Example: consider document classification, each example is a document, each feature represents the presence or absence of a particular word in the document.
- We have a training set.
- A new example with feature values \(x_{new} = (a_1, a_2, \cdots, a_d)\).
- We want to predict the label \(y_{new}\) of the new example.
Setting

\[ y_{new} = \arg\max_{y \in Y} \ p(y | a_1, a_2, \ldots, a_d) \]
Setting

\[ y_{new} = \arg\max_{y \in Y} \ p(y|a_1, a_2, \cdots, a_d) \]

Use Bayes rule to obtain:

\[ y_{new} = \arg\max_{y \in Y} \ \frac{p(a_1, a_2, \cdots, a_d|y) \ast p(y)}{p(a_1, a_2, \cdots, a_d)} \]
Setting

\[ y_{new} = \arg\max_{y \in Y} \ p(y|a_1, a_2, \ldots, a_d) \]

Use Bayes rule to obtain:

\[ y_{new} = \arg\max_{y \in Y} \ \frac{p(a_1, a_2, \ldots, a_d|y) \ast p(y)}{p(a_1, a_2, \ldots, a_d)} \]

\[ y_{new} = \arg\max_{y \in Y} \ p(a_1, a_2, \ldots, a_d|y) \ast p(y) \]
**Setting**

\[ y_{new} = \arg\max_{y \in Y} \ p(y|a_1, a_2, \cdots, a_d) \]

Use Bayes rule to obtain:

\[ y_{new} = \arg\max_{y \in Y} \ \frac{p(a_1, a_2, \cdots, a_d|y) * p(y)}{p(a_1, a_2, \cdots, a_d)} \]

\[ y_{new} = \arg\max_{y \in Y} \ p(a_1, a_2, \cdots, a_d|y) * p(y) \]

**Can we estimate these two terms from the training data?**
Setting

\[ y_{new} = \arg\max_{y \in Y} \ p(y|a_1, a_2, \ldots, a_d) \]

Use Bayes rule to obtain:

\[ y_{new} = \arg\max_{y \in Y} \ \frac{p(a_1, a_2, \ldots, a_d|y) \cdot p(y)}{p(a_1, a_2, \ldots, a_d)} \]

\[ y_{new} = \arg\max_{y \in Y} \ p(a_1, a_2, \ldots, a_d|y) \cdot p(y) \]

Can we estimate these two terms from the training data?
1. \( p(y) \) can be easy to estimate: count the frequency with which each label \( y \) occurs in the training data.
Setting

\[ y_{\text{new}} = \arg\max_{y \in Y} \quad p(y|a_1, a_2, \cdots, a_d) \]

Use Bayes rule to obtain:

\[ y_{\text{new}} = \arg\max_{y \in Y} \quad \frac{p(a_1, a_2, \cdots, a_d|y) * p(y)}{p(a_1, a_2, \cdots, a_d)} \]

\[ y_{\text{new}} = \arg\max_{y \in Y} \quad p(a_1, a_2, \cdots, a_d|y) * p(y) \]

Can we estimate these two terms from the training data?

1. \( p(y) \) can be easy to estimate: count the frequency with which each label \( y \).
2. \( p(a_1, a_2, \cdots, a_d|y) \) is not easy to estimate unless we have a very very large sample. (We need to see every example many times to get reliable estimates)
Naive Bayes Classifier

Makes a simplifying assumption that the feature values are conditionally independent given the label. Given the label of the example, the probability of observing the conjunction \(a_1, a_2, \cdots, a_d\) is the product of the probabilities for the individual features:

\[
p(a_1, a_2, \cdots, a_d | y) = \prod_j p(a_j | y)
\]
Naive Bayes Classifier

Makes a simplifying assumption that the feature values are conditionally independent given the label. Given the label of the example, the probability of observing the conjunction \(a_1, a_2, \cdots, a_d\) is the product of the probabilities for the individual features:

\[
p(a_1, a_2, \cdots, a_d | y) = \prod_j p(a_j | y)
\]

Naive Bayes Classifier:

\[
y_{new} = \arg\max_{y \in Y} p(y) \prod_j p(a_j | y)
\]
Naive Bayes Classifier

Makes a simplifying assumption that the feature values are conditionally independent given the label. Given the label of the example, the probability of observing the conjunction $a_1, a_2, \cdots, a_d$ is the product of the probabilities for the individual features:

$$p(a_1, a_2, \cdots, a_d | y) = \prod_j p(a_j | y)$$

Naive Bayes Classifier:

$$y_{new} = \arg\max_{y \in Y} p(y) \prod_j p(a_j | y)$$

Can we estimate these two terms from the training data?
Naive Bayes Classifier

Makes a simplifying assumption that the feature values are conditionally independent given the label. Given the label of the example, the probability of observing the conjunction $a_1, a_2, \cdots, a_d$ is the product of the probabilities for the individual features:

$$p(a_1, a_2, \cdots, a_d | y) = \prod_j p(a_j | y)$$

Naive Bayes Classifier:

$$y_{new} = \arg\max_{y \in Y} p(y) \prod_j p(a_j | y)$$

Can we estimate these two terms from the training data? Yes!
Algorithm

**Learning:** Based on the frequency counts in the dataset:
1. Estimate all $p(y)$, $\forall y \in Y$.
2. Estimate all $p(a_j|y)$ $\forall y \in Y$, $\forall a_i$.

**Classification:** For a new example, use:

$$y_{new} = \arg\max_{y \in Y} p(y) \prod_j p(a_j|y)$$

Note: No model per se or hyperplane, just count the frequencies of various data combinations within the training examples.
<table>
<thead>
<tr>
<th>Highest Degree</th>
<th>Work Experience</th>
<th>Favorite Language</th>
<th>Needs Work Visa</th>
<th>Hire</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bachelors</td>
<td>Mobile Dev</td>
<td>Objective-C</td>
<td>TRUE</td>
<td>yes</td>
</tr>
<tr>
<td>Masters</td>
<td>Web Dev</td>
<td>Java</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>Masters</td>
<td>Mobile Dev</td>
<td>Java</td>
<td>TRUE</td>
<td>yes</td>
</tr>
<tr>
<td>PhD</td>
<td>Mobile Dev</td>
<td>Objective-C</td>
<td>TRUE</td>
<td>yes</td>
</tr>
<tr>
<td>PhD</td>
<td>Web Dev</td>
<td>Objective-C</td>
<td>TRUE</td>
<td>no</td>
</tr>
<tr>
<td>Bachelors</td>
<td>UX Design</td>
<td>Objective-C</td>
<td>TRUE</td>
<td>no</td>
</tr>
<tr>
<td>Bachelors</td>
<td>Mobile Dev</td>
<td>Java</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>PhD</td>
<td>Web Dev</td>
<td>Objective-C</td>
<td>FALSE</td>
<td>no</td>
</tr>
<tr>
<td>Bachelors</td>
<td>UX Design</td>
<td>Java</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>Masters</td>
<td>UX Design</td>
<td>Objective-C</td>
<td>TRUE</td>
<td>no</td>
</tr>
<tr>
<td>Masters</td>
<td>UX Design</td>
<td>Java</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>PhD</td>
<td>Mobile Dev</td>
<td>Java</td>
<td>FALSE</td>
<td>no</td>
</tr>
<tr>
<td>Masters</td>
<td>Mobile Dev</td>
<td>Java</td>
<td>TRUE</td>
<td>yes</td>
</tr>
<tr>
<td>Bachelors</td>
<td>Web Dev</td>
<td>Objective-C</td>
<td>FALSE</td>
<td>no</td>
</tr>
</tbody>
</table>

Can we predict the class of the new example?
Example

\[ y_{\text{new}} = \arg\max_{y \in \{\text{yes, no}\}} p(y) \cdot p(\text{Masters}|y) \cdot p(\text{UX Design}|y) \cdot p(\text{Java}|y) \cdot p(\text{TRUE}|y) \]

\[ p(\text{yes}) = \frac{8}{14} = 0.572 \]
\[ p(\text{no}) = \frac{6}{14} = 0.428 \]

Conditional probabilities:

\[ p(\text{masters}|\text{yes}) = \frac{4}{8} \quad p(\text{masters}|\text{no}) = \frac{1}{6} \]
\[ p(\text{UX Design}|\text{yes}) = \frac{2}{8} \quad p(\text{UX Design}|\text{no}) = \frac{2}{6} \]
\[ p(\text{Java}|\text{yes}) = \frac{6}{8} \quad p(\text{Java}|\text{no}) = \frac{1}{6} \]
\[ p(\text{TRUE}|\text{yes}) = \frac{4}{8} \quad p(\text{TRUE}|\text{no}) = \frac{3}{6} \]

\[ p(\text{yes}) \cdot p(\text{Masters}|\text{yes}) \cdot p(\text{UX Design}|\text{yes}) \cdot p(\text{Java}|\text{yes}) \cdot p(\text{TRUE}|\text{yes}) = 0.026 \]
\[ p(\text{no}) \cdot p(\text{Masters}|\text{no}) \cdot p(\text{UX Design}|\text{no}) \cdot p(\text{Java}|\text{no}) \cdot p(\text{TRUE}|\text{no}) = 0.002 \]

\[ y_{\text{new}} = \text{yes} \]
Text Classification

- Given a document (corpus), define an attribute for each word position in the document.
- The value of the attribute is the English word in that position.
- To reduce the number of probabilities that needs to be estimated, besides NB independence assumption, we assume that: The probability of a given word \( w_k \) occurrence is independent of the word position within the text. That is:

\[
p(x_1 = w_k | c_j), p(x_2 = w_k | c_j), \ldots
\]

estimated by:

\[
p(w_k | c_j)
\]