BAYES CLASSIFIERS + NAÏVE BAYES

1. **Bayes Error**
   → The best ANY ML algorithm can do.

Recall Expected Loss/Error

\[ R, Y \sim P(x, y) \text{ [Unknown]} \] (Using \( P^* \) just to emphasize that this is reality, not from model)

\[
E[\text{Loss}] = \mathbb{E}_{P^*(x, y)}[L(y, \hat{y})]
\]

Bayes Error

\[ \text{min} \text{ possible value of this} \]

\[ = \int_{x \in \mathbb{R}^k} \int_{y \in \text{space of } Y} L(y, \hat{y}) P^*(x, y) \, dx \, dy \]

→ Assume Classification so \( Y = \{1, 2, \ldots, k\} \)

[Replace \( y \) with \( \frac{y}{k} \)]

→ Assume 0-1 Loss i.e. \( L(y, \hat{y}) = \begin{cases} 1 & y \neq \hat{y} \\ 0 & \text{else} \end{cases} \)

\[ \Rightarrow E[\text{Loss}] = \int_{x \in \mathbb{R}^k} \sum_{y=1}^k L(y, \hat{y}) P^*(x, y) \, dx \]

[Mix of PMF & PDF]
\[ E[\text{Loss}] = \int \left[ \sum_{y=1}^{k} L(y, \hat{y}) \ \hat{p}(y|x) \right] p(x) \, dx \]

Looks like \( E_{\hat{p}}[f(x)] \)

\[ = E_{\hat{p}(x)} \left[ \sum_{y=1}^{k} L(y, \hat{y}) \ p^*(y|x) \right] \]

And what happens when I predict \( \hat{y} \) for this \( x \); how likely is the output to be \( y \) (in reality)?

\[ = E_{\hat{p}(x)} \left[ L(1, \hat{y}) \ p^*(1|x) + L(2, \hat{y}) \ p^*(2|x) + \ldots + L(k, \hat{y}) \ p^*(k|x) \right] \]

Either 0 or 1

\[ = E_{\hat{p}(x)} \left[ 1 - p^*(Y=\hat{y}|x) \right] \]

So \( E[\text{Loss}] = E_{\hat{p}(x)} \left[ 1 - p^*(Y=\hat{y}|x) \right] \)

So \( \min_{\hat{y}} E[\text{Loss}] = \max_{Y=1, \ldots, k} p^*(Y=y|x) \)

Bayes Error \( \text{MAP} \) (Maximum-a-posteriori) [Error of the best possible predictor]

\[ \text{MAP is Bayes Optimal!} \]
Bayes Classifier: the classifier that achieves Bayes Error.

Similarly Bayes Regressor.

Note: $\hat{y}_{MAP}$ is Bayes Optimal FOR 0/1-Loss.

Different Loss $\Rightarrow$ Different Bayes Classifiers.

Great theory, how do I implement this?

Problem: Don't know $P^*(y|x=x)$

Solution: Generalized Approach

\[ P(y|x) \text{ from } \text{Bayes } \Rightarrow P(y|x) \Rightarrow \hat{y}_{MAP} \]

Data Rule

Discriminative Approach:

\[ P(y|x) \Rightarrow \text{predict is done} \]

from data
2. Bayes Classifiers are hard to learn: A counting argument

Say \( X = \mathbf{x} \in \{0,1\}^d \) d-binary-features

\( Y = y \in \{1,2,\ldots,k\} \) k classes

Need to estimate: \( P(Y=y) \) & \( P(X=x \mid Y=y) \)

How many parameters? \( (k-1) \cdot (2^d-1) \cdot k \)

If \( d=100 \), \( 2^{100} \approx 10^{30} \), data available

Lots of parameters \( \Rightarrow \) high variance

\( \Rightarrow \) overfitting

Very sparse table

3. Naive Bayes to the Rescue.

Assume \( P(X_1=x_1, \ldots, X_d=x_d \mid Y=y) = \prod_{j=1}^d P(X_j=x_j \mid Y=y) \quad \forall x, y \)

"Features are conditionally independent given class"

If I tell you \( X_2=x_2 \), you don't find out anything about \( X_1=x_1 \) any more than \( Y=y \) already told you

Note: this does not mean \( P(X_1 \mid X_2) = P(X_1) \)

How many parameters in NB?

\[
\frac{P(X_i=x_i \mid Y=c)}{(2^d-1) \cdot k} \cdot \frac{P(Y=c)}{(k-1)}
\]
Decision Rule:

\[ y_{\text{MAP}} = \underset{y}{\arg\max} \frac{P(Y = y \mid X = x)}{P(X = x)} \]

\[ = \underset{y}{\arg\max} \frac{\prod_{j=1}^{d} P(X_j = x_j \mid Y = y) \cdot P(Y = y)}{P(X = x)} \]

(sometimes also written as)

\[ = \underset{y}{\arg\max} \left( \frac{1}{d} \sum_{j=1}^{d} \log P(X_j = x_j \mid Y = y) + \log P(Y = y) \right) \]

(4) MLE/MAP for estimating NB parameters

- Given dataset \( D = \{ (x_1, y_1), \ldots, (x_n, y_n) \} \)

- What are the parameters?

  - \( P(Y) \) — Categorical

    \[ P(Y = c) = \frac{\theta_c}{\sum_{c=1}^{k} \theta_c} \quad \text{(shorthand)} \]

  - \( P(X_j = a \mid Y = c) \)

    \[ = \theta_{ac} \quad \text{(shorthand)} \]

    \[ \sum_{a=1}^{s_y} \theta_{ac} = 1 \]

    \[ \sum_{c=1}^{k} \theta_{ac} = 1 \]

    \[ \text{Each column is categorical} \]
All parameters $\Theta = \{ r_1, \ldots, r_k, \Theta^c, \Theta^a \}$

We know how to estimate categorical parameters:

**MLE**

$$
\hat{\theta} = P(Y=c) = \frac{\text{Count}(Y=c)}{n}
$$

**MAP**

$$
\hat{\theta}^{\text{MAP}} = P(X_j = a | Y = c) = \frac{\text{Count}(X_j = a, Y=c)}{\text{Count}(Y=c)}
$$

---

3) Bag-of-Words model

NB assumes $x_i \perp x_j | Y$

ie $P(x_i | x_j, Y) = P(x_i | Y)$

Bow additionally assumes $P(x_i | Y) = P(x_j | Y)$

that all dimensions are essentially the same. More sharing. Less parameters. Less expressive model class. Less variance. Less overfitting.