INTRODUCTION TO MACHINE LEARNING COMPSCI 4ML3

Lecture 19

Hassan Ashtiani

NAÏVE BAYES CLASSIFIERS

- The Naïve Bayes assumption:
  - Given the label, the coordinates are statistically independent
  - $\underline{P(x|y=k,\Theta)} = \underline{\pi_j P\left(x_j|y=k,\Theta\right)}$
- Choices for $P(x|y=i,\Theta)$
  - Gaussian, categorical, binomial, etc.
- How to find most probable y?

NAÏVE BAYES – INFERENCE

$p(y=0 \mid x, \theta)$ vs $p(y=1 \mid x, \theta)$

$\downarrow$

exercise →

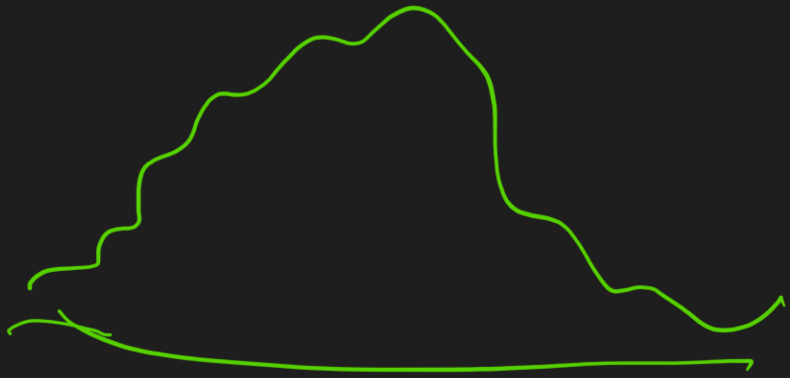
find the classification rule

NAÏVE BAYES CLASSIFIERS

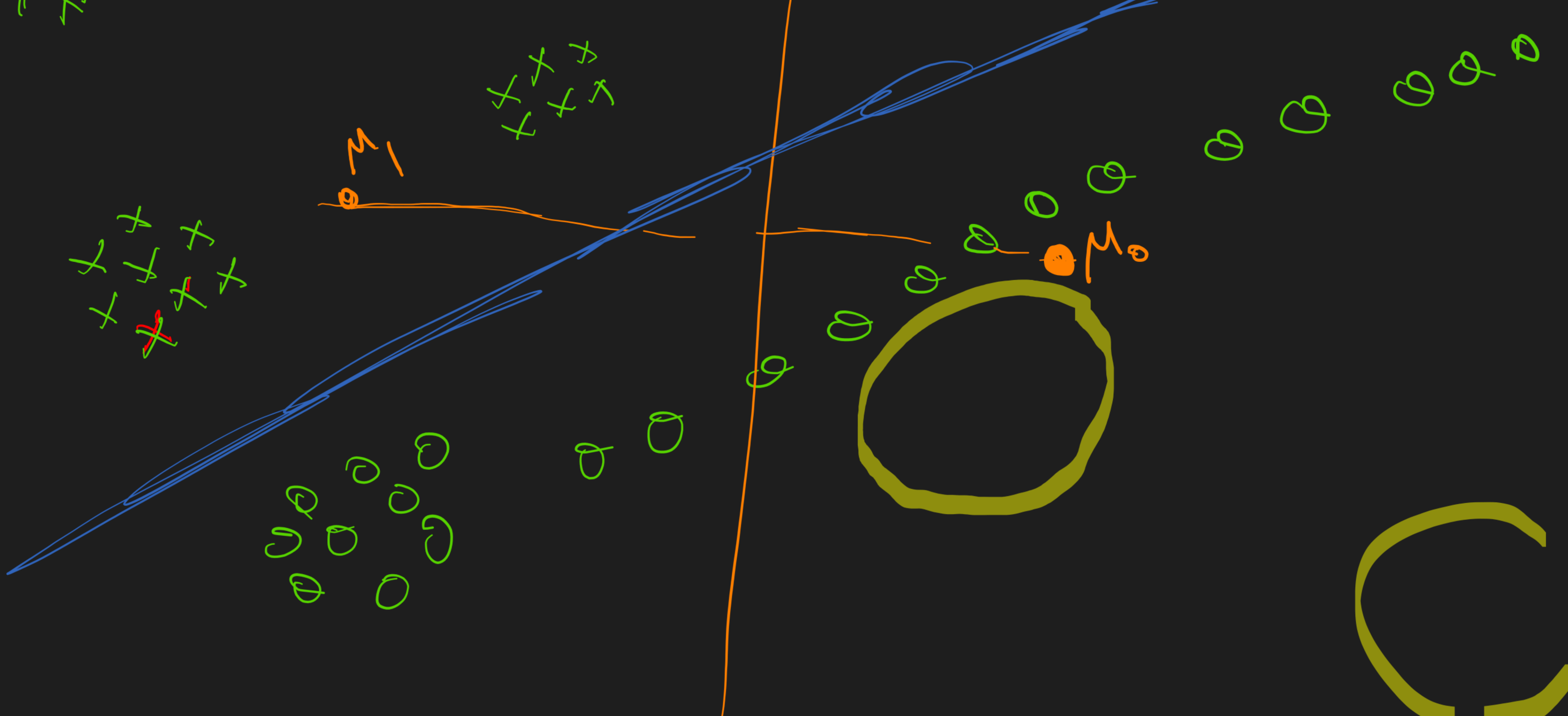
- Only need to estimate the distribution of each coordinate separately given the label.

Learning: learn $p(x_i \mid y)$ separately --- using ML.

  - No curse of dimensionality
  - Fast computation for learning and prediction
  - Assumptions are strong – may be far from reality
    - Independence of features
    - "The generative assumption"



GENERATIVE (GAUSSIAN) ASSUMPTION



GENERATIVE VS DISCRIMINATIVE MODELS

$P(y=0)$

$p(x|y=0)$

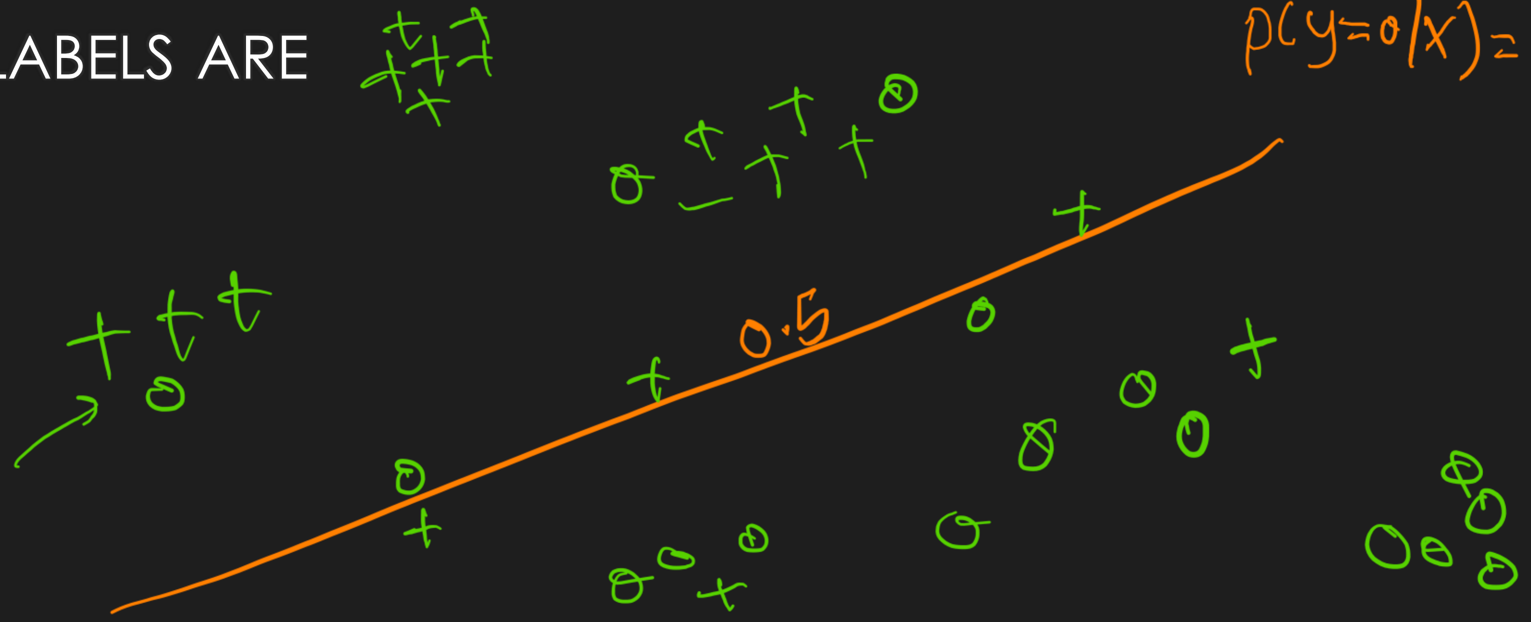
$p(x|y=1)$

- Probabilistic Generative models
  - Try to model $P(x, y)$
  - i.e., learn both $P(x|y)$ and $P(y)$
  - We are learning $P(x)$ as well
- Probabilistic Discriminative models
  - Try to model $P(y|x)$ only
  - No need to learn or model $P(x)$
    - E.g., logistic regression
- Non-probabilistic discriminative Approaches

LOGISTIC REGRESSION

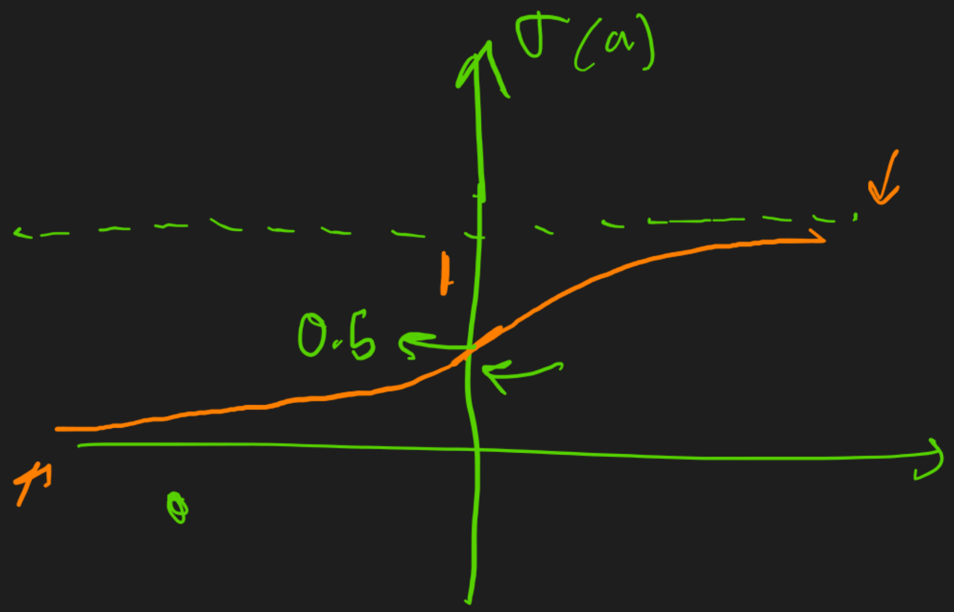
- Assume there is a plane in $\mathbb{R}^d$, parametrized by W
  - W separates the two classes ``nicely''
  - near the boundary, labels are (more) random
  - The more we get away from the boundary, the more deterministic the labels are

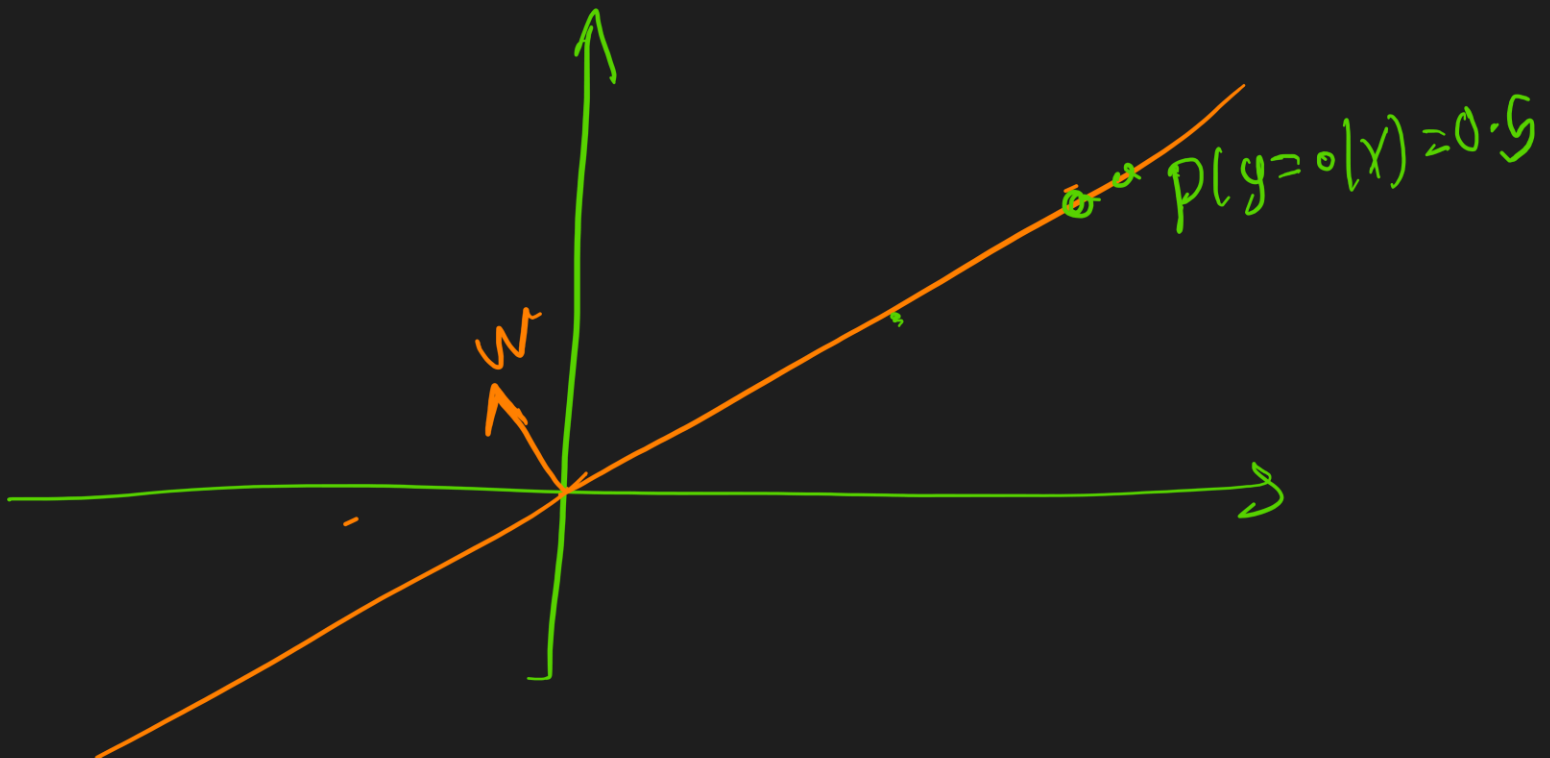
$p(y=0|x)=$



LOGISTIC REGRESSION

- $P(Y = 1|x, W) = \sigma(W^T x)$
- $P(Y = -1|x, W) = 1 - \sigma(W^T x)$
- $P(x|W) = P(x)$
- Where
  - $\sigma(a) = \frac{1}{1+e^{-a}}$





$\sigma(a) = \frac{1}{1+e^{-a}}$

MAXIMUM LIKELIHOOD

- $1 - \sigma(a) = \sigma(-a)$
  - $P(Y = y|x, W) = \sigma(yW^T x)$
- $W^{ML} = \underset{W}{\arg\max} \prod p(x^i, y^i | W)$

$= \underset{W}{\arg\max} \prod \frac{p(x^i, y^i, W)}{p(W)} = \underset{W}{\arg\max} \prod p(y^i | x^i, W) p(x^i)$

$= \underset{W}{\arg\max} \left[\prod p(x^i)\right]\left[\prod p(y^i | x^i, W)\right] = \underset{W}{\arg\max} \sum_{i=1}^{n} \log(\sigma(y^i W^T x^i))$

ANOTHER EQUIVALENT FORM

- For $y \in \{-1,1\}, P(Y = y|x, W) = \sigma(yW^T x)$
- Change to $y \in \{0,1\}$,
  - $P(Y = y|x, W) = \sigma(W^T x)^y (1 - \sigma(W^T x))^{1-y}$

$$= \begin{cases} \sigma(w^T x) & y = 1 \\ 1 - \sigma(w^T x) & y = 0 \end{cases}$$

- $Log\big(P(Y = y|x, W)\big) = y\, LOG\, \sigma(W^T x) + (1 - y) LOG\, (1 - \sigma(W^T x))$
- $\boxed{p} = \sigma(W^T x) \in [0,1]$
  - p IS A PROBABILITY, REPRESENTES THE CONFIDENCE OF THE MODEL
- MAXIMIZE $\sum_{i=1}^{n} (y^i \text{LOG}\, p^i + (1 - y^i) \text{LOG}(1 - p^i))$

$y^i \in \{0, 1\}$

  - Related to the cross entropy loss (more on this later)

$= \text{crossentropy}(p, y^i)$

Optimizing the Likelihood

- No closed form solution
- But still a concave function
    - Computer the gradient
        - Useful fact: $\frac{\partial \sigma(a)}{\partial a} = \sigma(a)\ (1\text{-}\sigma(a))$
    - do Gradient descent
- ....Or use automatic differentiation!

```python
class LogisticRegression(nn.Module):
    def __init__(self, input_size, num_classes):
        super(LogisticRegression, self).__init__()
        self.fc = nn.Linear(input_size, num_classes, bias=True)

    def forward(self, x):
        # Flatten the image
        x= x.view(-1, 28*28)
        return self.fc(x)
```

$W^T x$

missing sigmoid

```python
model = LogisticRegression(input_size, num_classes)
criterion = nn.CrossEntropyLoss()
optimizer = optim.SGD(model.parameters(), lr=learning_rate)
```

W

```python
for epoch in range(num_epochs):
    for i, (images_batch, labels_batch) in enumerate(train_loader):
        optimizer.zero_grad() # Clear the gradients
        outputs = model(images_batch) # Forward pass
        loss = criterion(outputs, labels_batch) # Calculate loss
        loss.backward() # Backward pass
        optimizer.step() # Update weights
```

$$\frac{\partial \, loss(w)}{\partial W}$$