Naïve Bayes
Reminders

• Homework 6: PAC Learning / Generative Models
  – Out: Wed, Oct 31
  – Due: Wed, Nov 7 at 11:59pm (1 week)

  TIP: Do the readings!

• Exam Viewing
  – Thu, Nov 1
  – Fri, Nov 2
NAÏVE BAYES
Naïve Bayes Outline

• **Real-world Dataset**
  – Economist vs. Onion articles
  – Document → bag-of-words → binary feature vector

• **Naive Bayes: Model**
  – Generating synthetic "labeled documents"
  – Definition of model
  – Naive Bayes assumption
  – Counting # of parameters with / without NB assumption

• **Naïve Bayes: Learning from Data**
  – Data likelihood
  – MLE for Naive Bayes
  – MAP for Naive Bayes

• **Visualizing Gaussian Naive Bayes**
Fake News Detector

**Today’s Goal:** To define a generative model of emails of two different classes (e.g. real vs. fake news)

---

**The Economist**

*Spain may be heading for its third election in a year*

*Stubborn Socialists are blocking Mariano Rajoy from forming a centre-right government*

---

**The Onion**

*Aiken, SC—Noting that he disappeared for over an hour during a campaign stop meet-and-greet with workers at a Bridgestone tire manufacturing plant, sources confirmed Tuesday that Democratic vice presidential candidate Tim Kaine was finally discovered riding on one of the factory’s conveyor belts. “Shortly after we arrived, Tim managed to get out of our sight, but after an extensive search of the facilities, one of our interns found him moving down the assembly line between several radial tires,” said senior campaign advisor Mike Henry, adding that Kaine could be seen smiling and laughing as...*
Naive Bayes: Model

Whiteboard

– Document $\rightarrow$ bag-of-words $\rightarrow$ binary feature vector
– Generating synthetic "labeled documents"
– Definition of model
– Naive Bayes assumption
– Counting # of parameters with / without NB assumption
Model 1: Bernoulli Naïve Bayes

Flip weighted coin

If HEADS, flip each red coin

Flip weighted coin

If TAILS, flip each blue coin

We can *generate* data in this fashion. Though in practice we never would since our data is *given*.

Instead, this provides an explanation of *how* the data was generated (albeit a terrible one).

<table>
<thead>
<tr>
<th>y</th>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
<th>...</th>
<th>$x_M$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>...</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>...</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>...</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>...</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>...</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>...</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>...</td>
<td>0</td>
</tr>
</tbody>
</table>

Each red coin corresponds to an $x_m$.
What’s wrong with the Naïve Bayes Assumption?

The features might not be independent!!

• Example 1:
  – If a document contains the word “Donald”, it’s extremely likely to contain the word “Trump”
  – These are not independent!

• Example 2:
  – If the petal width is very high, the petal length is also likely to be very high
Naïve Bayes: Learning from Data

Whiteboard

– Data likelihood
– MLE for Naive Bayes
– Example: MLE for Naïve Bayes with Two Features
– MAP for Naive Bayes
NAÏVE BAYES: MODEL DETAILS
Model 1: Bernoulli Naïve Bayes

**Support:** Binary vectors of length $K$

$$x \in \{0, 1\}^K$$

**Generative Story:**

$$Y \sim \text{Bernoulli}(\phi)$$

$$X_k \sim \text{Bernoulli}(\theta_k, y) \ \forall k \in \{1, \ldots, K\}$$

**Model:**

$$p_{\phi, \theta}(x, y) = p_{\phi, \theta}(x_1, \ldots, x_K, y)$$

$$= p_{\phi}(y) \prod_{k=1}^{K} p_{\theta_k}(x_k | y)$$

$$= (\phi)^y (1 - \phi)^{(1-y)} \prod_{k=1}^{K} (\theta_k, y)^{x_k} (1 - \theta_k, y)^{(1-x_k)}$$
# Model 1: Bernoulli Naïve Bayes

**Support:** Binary vectors of length $K$

$x \in \{0, 1\}^K$

**Generative Story:**

- $Y \sim \text{Bernoulli}(\phi)$
- $X_k \sim \text{Bernoulli}(\theta_k, Y) \ \forall k \in \{1, \ldots, K\}$

**Model:**

$$p_{\phi, \theta}(x, y) = (\phi)^y (1 - \phi)^{1-y} \prod_{k=1}^{K} \theta_k^{x_k} (1 - \theta_k)^{1-x_k}$$

**Classification:** Find the class that maximizes the posterior

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

Same as Generic Naïve Bayes
Training: Find the class-conditional MLE parameters

For $P(Y)$, we find the MLE using all the data. For each $P(X_k|Y)$ we condition on the data with the corresponding class.

$$\phi = \frac{\sum_{i=1}^{N} \mathbb{I}(y^{(i)} = 1)}{N}$$

$$\theta_{k,0} = \frac{\sum_{i=1}^{N} \mathbb{I}(y^{(i)} = 0 \land x_{k}^{(i)} = 1)}{\sum_{i=1}^{N} \mathbb{I}(y^{(i)} = 0)}$$

$$\theta_{k,1} = \frac{\sum_{i=1}^{N} \mathbb{I}(y^{(i)} = 1 \land x_{k}^{(i)} = 1)}{\sum_{i=1}^{N} \mathbb{I}(y^{(i)} = 1)}$$

$\forall k \in \{1, \ldots, K\}$
Model 1: Bernoulli Naïve Bayes

**Training:** Find the **class-conditional** MLE parameters

For $P(Y)$, we find the MLE using all the data. For each $P(X_k|Y)$ we condition on the data with the corresponding class.

$$
\phi = \frac{\sum_{i=1}^{N} \mathbb{I}(y^{(i)} = 1)}{N}
$$

$$
\theta_{k,0} = \frac{\sum_{i=1}^{N} \mathbb{I}(y^{(i)} = 0 \land x_{k}^{(i)} = 1)}{\sum_{i=1}^{N} \mathbb{I}(y^{(i)} = 0)}
$$

$$
\theta_{k,1} = \frac{\sum_{i=1}^{N} \mathbb{I}(y^{(i)} = 1 \land x_{k}^{(i)} = 1)}{\sum_{i=1}^{N} \mathbb{I}(y^{(i)} = 1)}
$$

$\forall k \in \{1, \ldots, K\}$
Other NB Models

1. **Bernoulli** Naïve Bayes:
   – for **binary** features

2. **Gaussian** Naïve Bayes:
   – for **continuous** features

3. **Multinomial** Naïve Bayes:
   – for **integer** features

4. **Multi-class** Naïve Bayes:
   – for classification problems with > 2 classes
   – **event model** could be any of Bernoulli, Gaussian, Multinomial, depending on features
Model 2: Gaussian Naïve Bayes

Support: \( \mathbf{x} \in \mathbb{R}^K \)

**Model:** Product of prior and the event model

\[
p(\mathbf{x}, y) = p(x_1, \ldots, x_K, y) = p(y) \prod_{k=1}^{K} p(x_k|y)
\]

Gaussian Naïve Bayes assumes that \( p(x_k|y) \) is given by a Normal distribution.
Model 3: Multinomial Naïve Bayes

Support:
Option 1: Integer vector (word IDs)
\[ \mathbf{x} = [x_1, x_2, \ldots, x_M] \text{ where } x_m \in \{1, \ldots, K\} \text{ a word id.} \]

Generative Story:
\[ \text{for } i \in \{1, \ldots, N\}: \]
\[ y^{(i)} \sim \text{Bernoulli}(\phi) \]
\[ \text{for } j \in \{1, \ldots, M_i\}: \]
\[ x_j^{(i)} \sim \text{Multinomial}(\theta_{y^{(i)}}, 1) \]

Model:
\[ p_{\phi, \theta}(\mathbf{x}, y) = p_{\phi}(y) \prod_{k=1}^{K} p_{\theta_k}(x_k | y) \]
\[ = (\phi)^y (1 - \phi)^{(1-y)} \prod_{j=1}^{M_i} \theta_{y, x_j} \]
Model 5: Multiclass Naïve Bayes

Model:
The only change is that we permit $y$ to range over $C$ classes.

$$p(x, y) = p(x_1, \ldots, x_K, y) = p(y) \prod_{k=1}^{K} p(x_k | y)$$

Now, $y \sim \text{Multinomial}(\phi, 1)$ and we have a separate conditional distribution $p(x_k | y)$ for each of the $C$ classes.
### Generic Naïve Bayes Model

| **Support:** | Depends on the choice of event model, $P(X_k|Y)$ |
|--------------|--------------------------------------------------|
| **Model:**   | Product of prior and the event model             |
|              | $P(X,Y) = P(Y) \prod_{k=1}^{K} P(X_k|Y)$         |
| **Training:**| Find the class-conditional MLE parameters        |
|              | For $P(Y)$, we find the MLE using all the data. For each $P(X_k|Y)$ we condition on the data with the corresponding classification. |
| **Classification:** | Find the class that maximizes the posterior |
|               | $\hat{y} = \operatorname{arg\,max}_y p(y|x)$ |
Naïve Bayes Model

**Classification:**

\[
\hat{y} = \arg\max_y p(y|x) \quad \text{(posterior)}
\]

\[
= \arg\max_y \frac{p(x|y)p(y)}{p(x)} \quad \text{(by Bayes’ rule)}
\]

\[
= \arg\max_y p(x|y)p(y)
\]
Smoothing

1. Add-1 Smoothing
2. Add-\( \lambda \) Smoothing
3. MAP Estimation (Beta Prior)
MLE

What does maximizing likelihood accomplish?

• There is only a finite amount of probability mass (i.e. sum-to-one constraint)

• MLE tries to allocate as much probability mass as possible to the things we have observed…

…at the expense of the things we have not observed
MLE

For Naïve Bayes, suppose we never observe the word “serious” in an Onion article. In this case, what is the MLE of \( p(x_k | y) \)?

\[
\theta_{k,0} = \frac{\sum_{i=1}^{N} \mathbb{1}(y(i) = 0 \land x_k^{(i)} = 1)}{\sum_{i=1}^{N} \mathbb{1}(y(i) = 0)}
\]

Now suppose we observe the word “serious” at test time. What is the posterior probability that the article was an Onion article?

\[
p(y|x) = \frac{p(x|y)p(y)}{p(x)}
\]
1. Add-1 Smoothing

The simplest setting for smoothing simply adds a single pseudo-observation to the data. This converts the true observations $\mathcal{D}$ into a new dataset $\mathcal{D}'$ from which we derive the MLEs.

$$\mathcal{D} = \{(x^{(i)}, y^{(i)})\}_{i=1}^{N} \quad (1)$$
$$\mathcal{D}' = \mathcal{D} \cup \{(\mathbf{0}, 0), (\mathbf{0}, 1), (\mathbf{1}, 0), (\mathbf{1}, 1)\} \quad (2)$$

where $\mathbf{0}$ is the vector of all zeros and $\mathbf{1}$ is the vector of all ones.

This has the effect of pretending that we observed each feature $x_k$ with each class $y$. 
1. Add-1 Smoothing

What if we write the MLEs in terms of the original dataset $\mathcal{D}$?

$$
\phi = \frac{\sum_{i=1}^{N} \mathbb{I}(y^{(i)} = 1)}{N}
$$

$$
\theta_{k,0} = \frac{1 + \sum_{i=1}^{N} \mathbb{I}(y^{(i)} = 0 \land x^{(i)}_{k} = 1)}{2 + \sum_{i=1}^{N} \mathbb{I}(y^{(i)} = 0)}
$$

$$
\theta_{k,1} = \frac{1 + \sum_{i=1}^{N} \mathbb{I}(y^{(i)} = 1 \land x^{(i)}_{k} = 1)}{2 + \sum_{i=1}^{N} \mathbb{I}(y^{(i)} = 1)}
$$

$\forall k \in \{1, \ldots, K\}$
2. Add-\(\lambda\) Smoothing

For the Categorical Distribution

Suppose we have a dataset obtained by repeatedly rolling a \(K\)-sided (weighted) die. Given data \(D = \{x^{(i)}\}_{i=1}^N\) where \(x^{(i)} \in \{1, \ldots, K\}\), we have the following MLE:

\[
\phi_k = \frac{\sum_{i=1}^N \mathbb{I}(x^{(i)} = k)}{N}
\]

With add-\(\lambda\) smoothing, we add pseudo-observations as before to obtain a smoothed estimate:

\[
\phi_k = \frac{\lambda + \sum_{i=1}^N \mathbb{I}(x^{(i)} = k)}{k\lambda + N}
\]
3. MAP Estimation (Beta Prior)

**Generative Story:**
The parameters are drawn once for the entire dataset.

for $k \in \{1, \ldots, K\}$:
  for $y \in \{0, 1\}$:
    $\theta_{k,y} \sim \text{Beta}(\alpha, \beta)$
  for $i \in \{1, \ldots, N\}$:
    $y^{(i)} \sim \text{Bernoulli}(\phi)$
    for $k \in \{1, \ldots, K\}$:
      $x_k^{(i)} \sim \text{Bernoulli}(\theta_{k,y^{(i)}})$

**Training:** Find the **class-conditional** MAP parameters

$\phi = \frac{\sum_{i=1}^{N} \mathbb{I}(y^{(i)} = 1)}{N}$

$\theta_{k,0} = \frac{(\alpha - 1) + \sum_{i=1}^{N} \mathbb{I}(y^{(i)} = 0 \land x_k^{(i)} = 1)}{(\alpha - 1) + (\beta - 1) + \sum_{i=1}^{N} \mathbb{I}(y^{(i)} = 0)}$

$\theta_{k,1} = \frac{(\alpha - 1) + \sum_{i=1}^{N} \mathbb{I}(y^{(i)} = 1 \land x_k^{(i)} = 1)}{(\alpha - 1) + (\beta - 1) + \sum_{i=1}^{N} \mathbb{I}(y^{(i)} = 1)}$

$\forall k \in \{1, \ldots, K\}$
VISUALIZING NAÏVE BAYES
Fisher Iris Dataset

Fisher (1936) used 150 measurements of flowers from 3 different species: Iris setosa (0), Iris virginica (1), Iris versicolor (2) collected by Anderson (1936)

<table>
<thead>
<tr>
<th>Species</th>
<th>Sepal Length</th>
<th>Sepal Width</th>
<th>Petal Length</th>
<th>Petal Width</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>4.3</td>
<td>3.0</td>
<td>1.1</td>
<td>0.1</td>
</tr>
<tr>
<td>0</td>
<td>4.9</td>
<td>3.6</td>
<td>1.4</td>
<td>0.1</td>
</tr>
<tr>
<td>0</td>
<td>5.3</td>
<td>3.7</td>
<td>1.5</td>
<td>0.2</td>
</tr>
<tr>
<td>1</td>
<td>4.9</td>
<td>2.4</td>
<td>3.3</td>
<td>1.0</td>
</tr>
<tr>
<td>1</td>
<td>5.7</td>
<td>2.8</td>
<td>4.1</td>
<td>1.3</td>
</tr>
<tr>
<td>1</td>
<td>6.3</td>
<td>3.3</td>
<td>4.7</td>
<td>1.6</td>
</tr>
<tr>
<td>1</td>
<td>6.7</td>
<td>3.0</td>
<td>5.0</td>
<td>1.7</td>
</tr>
</tbody>
</table>

Full dataset: https://en.wikipedia.org/wiki/Iris_flower_data_set
Slide from William Cohen
Naïve Bayes has a **linear** decision boundary if variance (sigma) is constant across classes.
Iris Data (2 classes)
Iris Data (2 classes)

Classification with Naive Bayes

variance = 1
Iris Data (2 classes)

Classification with Naive Bayes

variance learned for each class
Iris Data (3 classes)
Iris Data (3 classes)

Classification with Naive Bayes

variance = 1
Iris Data (3 classes)

Classification with Naive Bayes

variance learned for each class
One Pocket

Classification with Naive Bayes

variance learned for each class
One Pocket

Naive Bayes Distribution

variance learned for each class
Summary

1. Naïve Bayes provides a framework for **generative modeling**
2. Choose $p(x_m | y)$ appropriate to the data (e.g. Bernoulli for binary features, Gaussian for continuous features)
3. Train by **MLE** or **MAP**
4. Classify by maximizing the posterior
DISCRIMINATIVE AND GENERATIVE CLASSIFIERS
Generative vs. Discriminative

• **Generative Classifiers:**
  – Example: Naïve Bayes
  – Define a joint model of the observations $\mathbf{x}$ and the labels $y$: $p(\mathbf{x}, y)$
  – Learning maximizes (joint) likelihood
  – Use Bayes’ Rule to classify based on the posterior:
    \[
    p(y|\mathbf{x}) = \frac{p(\mathbf{x}|y)p(y)}{p(\mathbf{x})}
    \]

• **Discriminative Classifiers:**
  – Example: Logistic Regression
  – Directly model the conditional: $p(y|\mathbf{x})$
  – Learning maximizes conditional likelihood
Generative vs. Discriminative

Whiteboard

– Contrast: To model $p(x)$ or not to model $p(x)$?
Generative vs. Discriminative

Finite Sample Analysis (Ng & Jordan, 2002)

[Assume that we are learning from a finite training dataset]

If model assumptions are correct: Naive Bayes is a more efficient learner (requires fewer samples) than Logistic Regression

If model assumptions are incorrect: Logistic Regression has lower asymptotic error, and does better than Naïve Bayes
solid: NB  dashed: LR

Slide courtesy of William Cohen
Naïve Bayes makes stronger assumptions about the data but needs fewer examples to estimate the parameters.

Generative vs. Discriminative Learning (Parameter Estimation)

**Naïve Bayes:**
Parameters are decoupled → Closed form solution for MLE

**Logistic Regression:**
Parameters are coupled → No closed form solution – must use iterative optimization techniques instead
Naïve Bayes vs. Logistic Reg.

Learning (MAP Estimation of Parameters)

**Bernoulli Naïve Bayes:**
Parameters are probabilities $\rightarrow$ Beta prior (usually) pushes probabilities away from zero / one extremes

**Logistic Regression:**
Parameters are not probabilities $\rightarrow$ Gaussian prior encourages parameters to be close to zero

(effectively pushes the probabilities away from zero / one extremes)
Naïve Bayes vs. Logistic Reg.

Features

**Naïve Bayes:**
Features $x$ are assumed to be conditionally independent given $y$. (i.e. Naïve Bayes Assumption)

**Logistic Regression:**
No assumptions are made about the form of the features $x$. They can be dependent and correlated in any fashion.
Learning Objectives

Naïve Bayes

You should be able to...

1. Write the generative story for Naive Bayes
2. Create a new Naive Bayes classifier using your favorite probability distribution as the event model
3. Apply the principle of maximum likelihood estimation (MLE) to learn the parameters of Bernoulli Naive Bayes
4. Motivate the need for MAP estimation through the deficiencies of MLE
5. Apply the principle of maximum a posteriori (MAP) estimation to learn the parameters of Bernoulli Naive Bayes
6. Select a suitable prior for a model parameter
7. Describe the tradeoffs of generative vs. discriminative models
8. Implement Bernoulli Naives Bayes
9. Employ the method of Lagrange multipliers to find the MLE parameters of Multinomial Naive Bayes
10. Describe how the variance affects whether a Gaussian Naive Bayes model will have a linear or nonlinear decision boundary
PROBABILISTIC LEARNING
**Probabilistic Learning**

**Function Approximation**
Previously, we assumed that our output was generated using a deterministic target function:

\[ x^{(i)} \sim p^* (\cdot) \]
\[ y^{(i)} = c^* (x^{(i)}) \]

Our goal was to learn a hypothesis \( h(x) \) that best approximates \( c^*(x) \).

**Probabilistic Learning**
Today, we assume that our output is sampled from a conditional probability distribution:

\[ x^{(i)} \sim p^* (\cdot) \]
\[ y^{(i)} \sim p^* (\cdot | x^{(i)}) \]

Our goal is to learn a probability distribution \( p(y|x) \) that best approximates \( p^*(y|x) \).
Robotic Farming

<table>
<thead>
<tr>
<th></th>
<th>Deterministic</th>
<th>Probabilistic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classification</td>
<td>Is this a picture of a wheat kernel?</td>
<td>Is this plant drought resistant?</td>
</tr>
<tr>
<td>(binary output)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Regression</td>
<td>How many wheat kernels are in this picture?</td>
<td>What will the yield of this plant be?</td>
</tr>
<tr>
<td>(continuous output)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Oracles and Sampling

Whiteboard

- Sampling from common probability distributions
  - Bernoulli
  - Categorical
  - Uniform
  - Gaussian
- Pretending to be an Oracle (Regression)
  - Case 1: Deterministic outputs
  - Case 2: Probabilistic outputs
- Probabilistic Interpretation of Linear Regression
  - Adding Gaussian noise to linear function
  - Sampling from the noise model
- Pretending to be an Oracle (Classification)
  - Case 1: Deterministic labels
  - Case 2: Probabilistic outputs (Logistic Regression)
  - Case 3: Probabilistic outputs (Gaussian Naïve Bayes)
In-Class Exercise

1. With your neighbor, write a function which returns samples from a Categorical
   – Assume access to the rand() function
   – Function signature should be: categorical_sample(theta)
     where theta is the array of parameters
   – Make your implementation as efficient as possible!

2. What is the expected runtime of your function?
Generative vs. Discriminative

Whiteboard

– Generative vs. Discriminative Models
  • Chain rule of probability
  • Maximum (Conditional) Likelihood Estimation for Discriminative models
  • Maximum Likelihood Estimation for Generative models
Categorical Distribution

Whiteboard

– Categorical distribution details
  • Independent and Identically Distributed (i.i.d.)
  • Example: Dice Rolls
Takeaways

• One view of what ML is trying to accomplish is **function approximation**

• The principle of **maximum likelihood estimation** provides an alternate view of learning

• **Synthetic data** can help **debug** ML algorithms

• Probability distributions can be used to **model** real data that occurs in the world (don’t worry we’ll make our distributions more interesting soon!)
Learning Objectives

**Oracles, Sampling, Generative vs. Discriminative**

You should be able to…

1. Sample from common probability distributions
2. Write a generative story for a generative or discriminative classification or regression model
3. Pretend to be a data generating oracle
4. Provide a probabilistic interpretation of linear regression
5. Use the chain rule of probability to contrast generative vs. discriminative modeling
6. Define maximum likelihood estimation (MLE) and maximum conditional likelihood estimation (MCLE)
PROBABILISTIC LEARNING
Probabilistic Learning

**Function Approximation**
Previously, we assumed that our output was generated using a deterministic target function:

\[
\begin{align*}
x^{(i)} & \sim p^*(\cdot) \\
y^{(i)} & = c^*(x^{(i)})
\end{align*}
\]

Our goal was to learn a hypothesis \( h(x) \) that best approximates \( c^*(x) \)

**Probabilistic Learning**
Today, we assume that our output is sampled from a conditional probability distribution:

\[
\begin{align*}
x^{(i)} & \sim p^*(\cdot) \\
y^{(i)} & \sim p^*(\cdot|x^{(i)})
\end{align*}
\]

Our goal is to learn a probability distribution \( p(y|x) \) that best approximates \( p^*(y|x) \)
# Robotic Farming

## Deterministic vs. Probabilistic Classification

<table>
<thead>
<tr>
<th>Classification (binary output)</th>
<th>Deterministic</th>
<th>Probabilistic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Is this a picture of a wheat kernel?</td>
<td>Is this plant drought resistant?</td>
<td></td>
</tr>
</tbody>
</table>

## Regression (continuous output)

<table>
<thead>
<tr>
<th>Regression (continuous output)</th>
<th>Deterministic</th>
<th>Probabilistic</th>
</tr>
</thead>
<tbody>
<tr>
<td>How many wheat kernels are in this picture?</td>
<td>What will the yield of this plant be?</td>
<td></td>
</tr>
</tbody>
</table>
Oracles and Sampling

Whiteboard

– Sampling from common probability distributions
  • Bernoulli
  • Categorical
  • Uniform
  • Gaussian
– Pretending to be an Oracle (Regression)
  • Case 1: Deterministic outputs
  • Case 2: Probabilistic outputs
– Probabilistic Interpretation of Linear Regression
  • Adding Gaussian noise to linear function
  • Sampling from the noise model
– Pretending to be an Oracle (Classification)
  • Case 1: Deterministic labels
  • Case 2: Probabilistic outputs (Logistic Regression)
  • Case 3: Probabilistic outputs (Gaussian Naïve Bayes)
Takeaways

• One view of what ML is trying to accomplish is **function approximation**

• The principle of **maximum likelihood estimation** provides an alternate view of learning

• **Synthetic data** can help **debug** ML algorithms

• Probability distributions can be used to **model** real data that occurs in the world (don’t worry we’ll make our distributions more interesting soon!)
Learning Objectives

**Oracles, Sampling, Generative vs. Discriminative**

You should be able to...

1. Sample from common probability distributions
2. Write a generative story for a generative or discriminative classification or regression model
3. Pretend to be a data generating oracle
4. Provide a probabilistic interpretation of linear regression
5. Use the chain rule of probability to contrast generative vs. discriminative modeling
6. Define maximum likelihood estimation (MLE) and maximum conditional likelihood estimation (MCLE)