Announcements

• Quiz 2 **due tomorrow (Thursday), September 22 at 8pm**

• Quiz 3 will be posted tonight

• Homework 2 posted (**Due Monday, October 3 at 8pm**)
Recap: Maximum Likelihood Estimation

• Compare to loss function minimization:
  • Before: $y_i \approx f_{\beta}(x_i)$
  • Now: $y_i \sim p_{\beta}(\cdot | x_i; \beta)$

• Intuition the difference:
  • $f_{\beta}(x_i)$ just provides a point that $y_i$ should be close to
  • $p_{\beta}(\cdot | x_i; \beta)$ provides a score for each possible $y_i$

• Maximum likelihood estimation combines the loss function and model family design decisions
Recap: Maximum Likelihood Estimation

- Model family is the most likely label:
  \[ f_\beta(x) = \arg \max_y p_\beta(y \mid x) \]

- Loss function is the negative log likelihood (NLL):
  \[ \ell(\beta; Z) = -\log L(\beta; Z) = -\sum_{i=1}^{n} \log p_\beta(y_i \mid x_i) \]
Recap: MLE for Linear Regression

• **Design decision:** We choose the likelihood to be

\[
p_{\beta}(y \mid x) = N(y; \beta^T x, 1) = \frac{1}{\sqrt{2\pi}} e^{-\frac{(\beta^T x - y)^2}{2}}
\]
Recap: MLE for Linear Regression

• Model family:

\[ f_\beta(x) = \arg \max_y p_\beta(y | x) = \beta^T x \]

• Negative log likelihood:

\[ \ell(\beta; Z) = -\sum_{i=1}^{n} \log p_\beta(y_i | x_i) = \frac{n \log(2\pi)}{2} + \sum_{i=1}^{n} (\beta^T x_i - y_i)^2 \]
Recap: MLE for Logistic Regression

• **Design decision:** We choose the likelihood to be

\[
p_\beta(Y = 1 \mid x) = \frac{1}{1 + e^{-\beta^T x}} = \sigma(\beta^T x)
\]

\[
p_\beta(Y = 0 \mid x) = 1 - \sigma(\beta^T x)
\]
Recap: MLE for Logistic Regression

• Model family:

\[ f_\beta(x) = \arg \max_y p_\beta(y \mid x) = 1(\beta^T x \geq 0) \]

• Negative log likelihood:

\[ \ell(\beta; Z) = -\sum_{i=1}^{n} \log p_\beta(y_i \mid x_i) \]
\[ = -\sum_{i=1}^{n} y_i \log(\sigma(\beta^T x_i)) + (1 - y_i) \log(1 - \sigma(\beta^T x_i)) \]
Maximum Likelihood View of ML

• **Two** design decisions
  • **Likelihood**: Probability $p_\beta(y \mid x)$ of data $(x, y)$ given parameters $\beta$
  • **Optimizer**: How do we optimize the NLL? (E.g., gradient descent)

• **Corresponding Loss Minimization View**:
  • **Model family**: Most likely label $f_\beta(x) = \arg\max_y p_\beta(y \mid x)$
  • **Loss function**: Negative log likelihood (NLL) $\ell(\beta; Z) = -\sum_{i=1}^{n} \log p_\beta(y_i \mid x_i)$
Classification Metrics

• While we minimize the NLL, we often evaluate using accuracy

• However, even accuracy isn’t necessarily the “right” metric
  • If 99% of labels are negative (i.e., \( y_i = 0 \)), accuracy of \( f_\beta(x) = 0 \) is 99%!
  • For instance, very few patients test positive for most diseases
  • “Imbalanced data”

• What are alternative metrics for these settings?
Classification Metrics

• Classify test examples as follows:
  • **True positive (TP):** Actually positive, predictive positive
  • **False negative (FN):** Actually positive, predicted negative
  • **True negative (TN):** Actually negative, predicted negative
  • **False positive (FP):** Actually negative, predicted positive

• Many metrics expressed in terms of these; for example:

\[
\text{accuracy} = \frac{TP + TN}{n} \quad \text{error} = 1 - \text{accuracy} = \frac{FP + FN}{n}
\]
## Confusion Matrix

<table>
<thead>
<tr>
<th>Actual Class</th>
<th>Predicted Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>No</td>
<td>FP</td>
</tr>
</tbody>
</table>
## Confusion Matrix

<table>
<thead>
<tr>
<th>Actual Class</th>
<th>Predicted Class</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>3 TP</td>
<td></td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>4 FN</td>
<td></td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>6 FP</td>
<td>37 TN</td>
</tr>
<tr>
<td></td>
<td>No</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Accuracy = 0.8
Classification Metrics

• For imbalanced metrics, we roughly want to disentangle:
  • Accuracy on “positive examples”
  • Accuracy on “negative examples”

• Different definitions are possible (and lead to different meanings)!
Sensitivity & Specificity

• **Sensitivity**: What fraction of actual positives are predicted positive?
  • **Good sensitivity**: If you have the disease, the test correctly detects it
  • Also called **true positive rate**

• **Specificity**: What fraction of actual negatives are predicted negative?
  • **Good specificity**: If you do not have the disease, the test says so
  • Also called **true negative rate**

• Commonly used in medicine
Sensitivity & Specificity

<table>
<thead>
<tr>
<th>Actual Class</th>
<th>Predicted Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>No</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>No</td>
</tr>
</tbody>
</table>

sensitivity = \frac{TP}{TP + FN}

specificity = \frac{TN}{TN + FP}
### Sensitivity & Specificity

<table>
<thead>
<tr>
<th>Actual Class</th>
<th>Predicted Class</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>3 TP</td>
<td>4 FN</td>
</tr>
<tr>
<td>No</td>
<td>6 FP</td>
<td>37 TN</td>
<td></td>
</tr>
</tbody>
</table>

**Sensitivity**

\[
sensitivity = \frac{TP}{TP + FN}
\]

**Specificity**

\[
specificity = \frac{TN}{TN + FP}
\]
Sensitivity & Specificity

<table>
<thead>
<tr>
<th>Actual Class</th>
<th>Predicted Class</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>3 TP</td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>4 FN</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>6 FP</td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>37 TN</td>
</tr>
</tbody>
</table>

sensitivity = 3/7

specificity = 37/43
Precision & Recall

• **Recall**: What fraction of actual positives are **predicted positive**?
  • **Good recall**: If you have the disease, the test correctly detects it
  • Also called the **true positive rate** (and sensitivity)

• **Precision**: What fraction of predicted positives are actual positives?
  • **Good precision**: If the test says you have the disease, then you have it
  • Also called **positive predictive value**

• Used in information retrieval, NLP
Precision & Recall

Recall = \frac{TP}{TP + FN}

Precision = \frac{TP}{TP + FP}
### Precision & Recall

#### Confusion Matrix

<table>
<thead>
<tr>
<th>Actual Class</th>
<th>Predicted Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

**Precision**

\[
precision = \frac{TP}{TP + FP}
\]

**Recall**

\[
recall = \frac{TP}{TP + FN}
\]
Precision & Recall

<table>
<thead>
<tr>
<th>Actual Class</th>
<th>Predicted Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

Recall = 3/7

Precision = 3/9
Classification Metrics

• **How to obtain a single metric?**
  - Combination, e.g., $F_1$ score = \[
  \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}
  \]
    is the harmonic mean
  - More on this later

• **How to choose the “right” metric?**
  - No generally correct answer
  - Depends on the goals for the specific problem/domain
Optimizing a Classification Metric

• We are training a model to minimize NLL, but we have a different “true” metric that we actually want to optimize

• Two strategies (can be used together):
  • **Strategy 1:** Optimize prediction threshold
  • **Strategy 2:** Upweight positive (or negative) examples
Optimizing Prediction Threshold

• Consider hyperparameter $\tau$ for the threshold:

$$f_\beta(x) = 1(\beta^T x \geq 0)$$
Optimizing Prediction Threshold

- Consider hyperparameter $\tau$ for the threshold:

$$f_\beta(x) = 1(\beta^T x \geq \tau)$$
Optimizing Prediction Threshold

\[ 1(\beta^T x \geq 0) \]

\[ 1(\beta^T x \geq -1) \]

\[ 1(\beta^T x \geq -2) \]

- higher sensitivity
- lower specificity

\[ 1(\beta^T x \geq 1) \]

\[ 1(\beta^T x \geq 2) \]

negative

positive

\[ x_1 \]

\[ x_2 \]
Visualization: ROC Curve

Each point on this curve corresponds to a choice of $\tau$

Aside: Area under ROC curve is another metric people consider when evaluating $\hat{\beta}(Z)$
Optimizing Prediction Threshold

• Consider hyperparameter $\tau$ for the threshold:

$$f_\beta(x) = 1(\beta^T x \geq \tau)$$

• Unlike most hyperparameters, we choose this one after we have already fit the model on the training data
  • Then, choose the value of $\tau$ that optimizes the desired metric
  • Fit using validation data (training data is OK if needed)
Optimizing Prediction Threshold

• **Step 1:** Compute the optimal parameters $\hat{\beta}(Z_{\text{train}})$
  • Using gradient descent on NLL loss over the training dataset
  • Resulting model: $f_{\hat{\beta}(Z_{\text{train}})}(x) = 1(\hat{\beta}(Z_{\text{train}})^T x \geq 0)$

• **Step 2:** Modify threshold $\tau$ in model to optimize desired metric
  • Search over a fixed set of $\tau$ on the validation dataset
  • Resulting model: $f_{\hat{\beta}(Z_{\text{train}}),\hat{\tau}(Z_{\text{val}})}(x) = 1(\hat{\beta}(Z_{\text{train}})^T x \geq \hat{\tau}(Z_{\text{val}}))$

• **Step 3:** Evaluate desired metric on test set
Choice of Metric Revisited

- **Common strategy:** Optimize one metric at fixed value of another

Choose $\tau$ corresponding to model at this point

Specificity = 0.9
Optimizing a Classification Metric

• We are training a model to minimize NLL, but we have a different “true” metric that we actually want to optimize

• Two strategies (can be used together):
  • **Strategy 1:** Optimize prediction threshold
  • **Strategy 2:** Upweight positive (or negative) examples
Class Re-Weighting

• **Weighted NLL:** Include a class-dependent weight \( w_y \):

\[
\ell(\beta; Z) = - \sum_{i=1}^{n} w_{y_i} \cdot \log p_{\beta}(y_i | x_i)
\]

• **Intuition:** Tradeoff between accuracy on negative/positive examples
  • To improve sensitivity (true positive rate), upweight positive examples
  • To improve specificity (true negative rate), upweight negative examples

• Can use this strategy to learn \( \beta \), and the first strategy to choose \( \tau \)
Classification Metrics

- NLL isn’t usually the “true” metric
  - Instead, frequently used due to good computational properties

- Many choices with different meanings

- Typical strategy:
  - Learn $\beta$ by minimizing the NLL loss
  - Choose class weights $w_y$ and threshold $\tau$ to optimize desired metric