CSE/STAT 416

Classification

Hunter Schafer
Paul G. Allen School of Computer Science & Engineering
University of Washington

April 12, 2021
1. Housing Prices - Regression
   - Regression Model
   - Assessing Performance
   - Ridge Regression
   - LASSO

2. Sentiment Analysis – Classification
   - Classification Overview
   - Logistic Regression
Spam Filtering

Input: x
Text of email
Sender
Subject
...

Output: y
Spam
Not Spam (ham)
Object Detection

Input: $x$ Pixels

Output: $y$ Class (+ Probability)

Top Predictions
- Labrador retriever
- Golden retriever
- Redbone
- Bloodhound
- Rhodesian ridgeback
Training Data → Feature extraction → ML model → \( \hat{y} \) → Quality metric → ML algorithm → \( \hat{f} \) → Feature extraction → \( h(x) \) → \( x \) → Training Data
In our example, we want to classify a restaurant review as positive or negative.

Sentence from review → Classifier Model → Predicted class

Input: x

Output: y
Implementation 1: Simple Threshold Classifier

Idea: Use a list of good words and bad words, classifier by most frequent type of word

- Positive Words: great, awesome, good, amazing, ...
- Negative Words: bad, terrible, disgusting, sucks, ...

Simple Threshold Classifier

Input $x$: Sentence from review

- Count the number of positive and negative words, in $x$
- If $\text{num\_positive} > \text{num\_negative}$:
  - $\hat{y} = +1$
- Else:
  - $\hat{y} = -1$

Example: "Sushi was great, the food was awesome, but the service was terrible"
How do we get list of positive/negative words?

Words have different degrees of sentiment.
- Great > Good
- How can we weigh them differently?

Single words are not enough sometimes...
- “Good” → Positive
- “Not Good” → Negative
Implementation 2: Linear Classifier

**Idea:** Use labelled training data to learn a weight for each word. Use weights to score a sentence.

<table>
<thead>
<tr>
<th>Word</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>good</td>
<td>1.0</td>
</tr>
<tr>
<td>great</td>
<td>1.5</td>
</tr>
<tr>
<td>awesome</td>
<td>2.7</td>
</tr>
<tr>
<td>bad</td>
<td>-1.0</td>
</tr>
<tr>
<td>terrible</td>
<td>-2.1</td>
</tr>
<tr>
<td>awful</td>
<td>-3.3</td>
</tr>
<tr>
<td>restaurant, the, we, where, ...</td>
<td>0.0</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Score a Sentence

<table>
<thead>
<tr>
<th>Word</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>good</td>
<td>1.0</td>
</tr>
<tr>
<td>great</td>
<td>1.5</td>
</tr>
<tr>
<td>awesome</td>
<td>2.7</td>
</tr>
<tr>
<td>bad</td>
<td>-1.0</td>
</tr>
<tr>
<td>terrible</td>
<td>-2.1</td>
</tr>
<tr>
<td>awful</td>
<td>-3.3</td>
</tr>
<tr>
<td>restaurant, the,</td>
<td>0.0</td>
</tr>
<tr>
<td>we, where, ...</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Input $x_i$:
“Sushi was great, the food was awesome, but the service was terrible”

Linear classifier, because output is linear weighted sum of inputs.
Will learn how to learn weights soon!
Implementation 2: Linear Classifier

Idea: Use labelled training data to learn a weight for each word. Use weights to score a sentence.

- See last slide for example weights and scoring.

Linear Classifier

Input $x$: Sentence from review

- Compute $Score(x)$
- If $Score(x) > 0$:
  - $\hat{y} = +1$
- Else:
  - $\hat{y} = -1$
Model: $\hat{y}_i = \text{sign}(\text{Score}(x_i))$

\begin{align*}
\text{Score}(x_i) &= w_0 h_0(x_i) + w_1 h_1(x_i) + \ldots + w_D h_D(x_i) \\
&= \sum_{j=0}^{D} w_j h_j(x_i) \\
&= w^T h(x)
\end{align*}

We will also use the notation

\begin{align*}
\hat{s}_i &= \text{Score}(x_i) = w^T h(x_i) \\
\hat{y}_i &= \text{sign}(\hat{s}_i)
\end{align*}
Consider if only two words had non-zero coefficients

\[ \hat{s} = 1 \cdot \text{#awesome} - 1.5 \cdot \text{#awful} \]

<table>
<thead>
<tr>
<th>Word</th>
<th>Coefficient</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>( w_0 )</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>awesome</td>
<td>( w_1 )</td>
<td>1.0</td>
</tr>
<tr>
<td>awful</td>
<td>( w_2 )</td>
<td>-1.5</td>
</tr>
</tbody>
</table>
Decision Boundary

Score(x) = 1 \cdot #\text{awesome} - 1.5 \cdot #\text{awful}

Generally, with classification we don’t us a plot like the 3d view since it’s hard to visualize, instead use 2d plot with decision boundary
\[ Score(x) = 1 \cdot \#\text{awesome} - 1.5 \cdot \#\text{awful} \]
What happens to the decision boundary if we add an intercept?

\[ \text{Score}(x) = 1.0 + 1 \cdot \#\text{awesome} - 1.5 \cdot \#\text{awful} \]
What happens to the decision boundary if we add an intercept?

\[ \text{Score}(x) = 1.0 + 1 \cdot \#\text{awesome} - 1.5 \cdot \#\text{awful} \]
What if we want to use a more complex decision boundary?

- Need more complex model/features!
- Covered next lecture!
Evaluating Classifiers
Training Data \rightarrow \text{Feature extraction} \rightarrow h(x) \rightarrow \text{ML model} \rightarrow \hat{s} \rightarrow \text{ML algorithm} \rightarrow \hat{w} \rightarrow \text{Quality metric} \rightarrow y \leftarrow \text{Training Data}

\hat{y} = \text{sign}(\hat{s})
Classification Error

Ratio of examples where there was a mistaken prediction

What’s a mistake?

- If the true label was positive ($y = +1$), but we predicted negative ($\hat{y} = -1$)
- If the true label was negative ($y = -1$), but we predicted positive ($\hat{y} = +1$)

Classification Error

Classification Accuracy
What’s a good accuracy?

For binary classification:
- Should at least beat random guessing...
- Accuracy should be at least 0.5

For multi-class classification ($k$ classes):
- Should still beat random guessing
- Accuracy should be at least $\frac{1}{k}$
  - 3-class: 0.33
  - 4-class: 0.25
  - ...

Besides that, higher accuracy means better, right?
Detecting Spam

Imagine I made a “Dummy Classifier” for detecting spam

▪ The classifier ignores the input, and always predicts spam.
▪ This actually results in 90% accuracy! Why?
  - Most emails are spam...

This is called the **majority class classifier**.

A classifier as simple as the majority class classifier can have a high accuracy if there is a **class imbalance**.

▪ A class imbalance is when one class appears much more frequently than another in the dataset

This might suggest that accuracy isn’t enough to tell us if a model is a good model.
Always digging in and ask critical questions of your accuracy.

- Is there a class imbalance?
- How does it compare to a baseline approach?
  - Random guessing
  - Majority class
  - ...
- Most important: What does my application need?
  - What's good enough for user experience?
  - What is the impact of a mistake we make?
Brain Break
Confusion Matrix

For binary classification, there are only two types of mistakes:

- $\hat{y} = +1, \ y = -1$
- $\hat{y} = -1, \ y = +1$

Generally we make a **confusion matrix** to understand mistakes.

![Confusion Matrix Diagram](image)
Confusion Matrix Example

<table>
<thead>
<tr>
<th>True Label</th>
<th>Predicted Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>True Positive (TP)</td>
</tr>
<tr>
<td>-</td>
<td>False Positive (FP)</td>
</tr>
</tbody>
</table>
Which is Worse?

What’s worse, a false negative or a false positive?
- It entirely depends on your application!

**Detecting Spam**
- False Negative: Annoying
- False Positive: Email lost

**Medical Diagnosis**
- False Negative: Disease not treated
- False Positive: Wasteful treatment

In almost every case, how treat errors depends on your context.
We mentioned on the first day how ML is being used in many contexts that impact crucial aspects of our lives.

Models making errors is a given, what we do about that is a choice:
- Are the errors consequential enough that we shouldn’t use a model in the first place?
- Do different demographic groups experience errors at different rates?
  - If so, we would hopefully want to avoid that model!

Will talk more about how to define whether or not a model is fair / discriminatory in a later lecture! Will use these notions of error as a starting point!
Binary Classification Measures

Notation
- $C_{TP} = \#TP$, $C_{FP} = \#FP$, $C_{TN} = \#TN$, $C_{FN} = \#FN$
- $N = C_{TP} + C_{FP} + C_{TN} + C_{FN}$
- $N_P = C_{TP} + C_{FN}$, $N_N = C_{FP} + C_{TN}$

Error Rate
$$\frac{C_{FP} + C_{FN}}{N}$$

Accuracy Rate
$$\frac{C_{TP} + C_{TN}}{N}$$

False Positive rate (FPR)
$$\frac{C_{FP}}{N_N}$$

False Negative Rate (FNR)
$$\frac{C_{FN}}{N_P}$$

True Positive Rate or Recall
$$\frac{C_{TP}}{N_P}$$

Precision
$$\frac{C_{TP}}{C_{TP} + C_{FP}}$$

F1-Score
$$\frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$
Consider predicting *(Healthy, Cold, Flu)*

<table>
<thead>
<tr>
<th>True Label</th>
<th>Predicted Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>Healthy</td>
<td>60</td>
</tr>
<tr>
<td>Cold</td>
<td>4</td>
</tr>
<tr>
<td>Flu</td>
<td>0</td>
</tr>
</tbody>
</table>
Suppose we trained a classifier and computed its confusion matrix on the training dataset. **Is there a class imbalance in the dataset and if so, which class has the highest representation?**

<table>
<thead>
<tr>
<th>True Label</th>
<th>Predicted Label</th>
<th>Pupper</th>
<th>Doggo</th>
<th>Boofer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pupper</td>
<td>Pupper</td>
<td>2</td>
<td>27</td>
<td>4</td>
</tr>
<tr>
<td>Doggo</td>
<td>Pupper</td>
<td>4</td>
<td>25</td>
<td>4</td>
</tr>
<tr>
<td>Boofer</td>
<td>Pupper</td>
<td>1</td>
<td>30</td>
<td>2</td>
</tr>
</tbody>
</table>
Suppose we trained a classifier and computed its confusion matrix on the training dataset. Is there a class imbalance in the dataset and if so, which class has the highest representation?

<table>
<thead>
<tr>
<th>True Label</th>
<th>Predicted Label</th>
<th>Pupper</th>
<th>Doggo</th>
<th>Boofer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pupper</td>
<td>Pupper</td>
<td>2</td>
<td>27</td>
<td>4</td>
</tr>
<tr>
<td>Doggo</td>
<td>Doggo</td>
<td>4</td>
<td>25</td>
<td>4</td>
</tr>
<tr>
<td>Boofer</td>
<td>Boofer</td>
<td>1</td>
<td>30</td>
<td>2</td>
</tr>
</tbody>
</table>
Learning Theory
How much data?

The more the merrier

- But data quality is also an extremely important factor

Theoretical techniques can bound how much data is needed

- Typically too loose for practical applications
- But does provide some theoretical guarantee

In practice

- More complex models need more data
How does the true error of a model relate to the amount of training data we give it?

- Hint: We’ve seen this picture before
Learning Curve

What if we use a more complex model?

True error vs. Amount of training data
Threshold Model
What if I never want to make a false positive prediction?

What if I never want to make a false negative prediction?

One way to control for our application is to change the scoring threshold. (Could also change intercept!)

- If $\text{Score}(x) > \alpha$:
  - Predict $\hat{y} = +1$
- Else:
  - Predict $\hat{y} = -1$
What happens to our TPR and FPR as we increase the threshold?
Next Time

We will talk about learning classifiers that model the probability of seeing a particular class at a given input.

\[ P(y|x) \]

Normally assume some structure on the probability (e.g. linear)

\[ P(y|x, w) \approx w^T x \]

Use machine learning algorithm to learn approximate \( \hat{w} \) such that

\[ \hat{P}(y|x) = P(y|x, \hat{w}) \]

And \( P(y|x) \) and \( \hat{P}(y|x) \) are close.
Recap

**Theme:** Describe high level idea and metrics for classification

**Ideas:**
- Applications of classification
- Linear classifier
- Decision boundaries
- Classification error / Classification accuracy
- Class imbalance
- Confusion matrix
- Learning theory
- ROC Curve