Multiclass Classification

CS 6956: Deep Learning for NLP

So far: Binary Classification

- We have seen linear models for binary classification

- We can write down a loss for binary classification
  - Common losses: Hinge loss and log loss

This lecture

- Multiclass classification
- Modeling multiple classes
- Loss functions for multiclass classification
  – Once we have a loss, we can minimize it to train

Where are we?

- Multiclass classification

- Modeling multiple classes

- Loss functions for multiclass classification
  - Once we have a loss, we can minimize it to train

What is multiclass classification?

- An input can belong to one of K classes

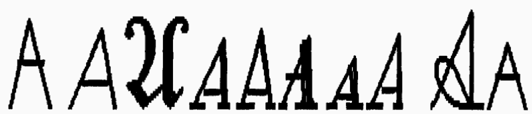
- Training data: Input associated with class label (a number from 1 to K)
- Prediction: Given a new input, predict the class label

Each input belongs to exactly one class. Not more, not less.

- Otherwise, the problem is not multiclass classification
- If an input can be assigned multiple labels (think tags for emails rather than folders), it is called *multi-label classification*

Example applications: Images

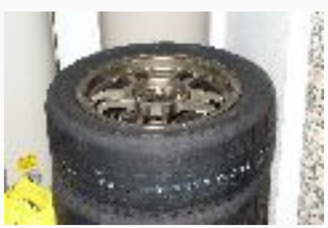
– *Input*: hand-written character; *Output*: which character?



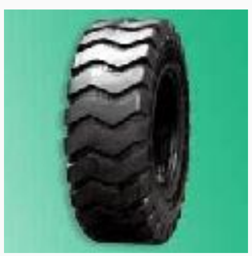
all map to the letter A

– *Input*: a photograph of an object; *Output*: which of a set of categories of objects is it?

- Eg: the Caltech 256 dataset

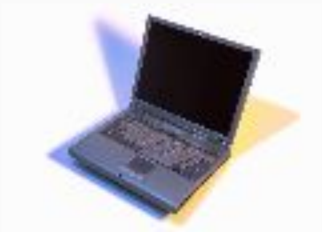


Car tire



Car tire

Duck



laptop

Example applications: Language

- *Input*: a news article
- *Output*: Which section of the newspaper should be be in

- *Input*: an email
- *Output*: which folder should an email be placed into

- *Input*: an audio command given to a car
- *Output*: which of a set of actions should be executed

Where are we?

- Multiclass classification
- Modeling multiple classes
- Loss functions for multiclass classification
  - Once we have a loss, we can minimize it to train

Multiclass prediction

- Suppose we have K classes: Given an input $\boldsymbol{x}$, we need to predict one of these classes.
  - Let us number the labels as 1, 2, …, K

Multiclass prediction

- Suppose we have K classes: Given an input $\boldsymbol{x}$, we need to predict one of these classes.
  - Let us number the labels as 1, 2, …, K

- Modeling K classes:
  - For a label i, we can define a scoring function $score(\boldsymbol{x}, i)$
  - The score is a real number. Higher score means that the label is preferred

Multiclass prediction

- Suppose we have K classes: Given an input $\boldsymbol{x}$, we need to predict one of these classes.
  - Let us number the labels as 1, 2, …, K

- Modeling K classes:
  - For a label i, we can define a scoring function $score(\boldsymbol{x}, i)$
  - The score is a real number. Higher score means that the label is preferred

> We haven't committed to the actual functional form of the *score* function.
>
> For now, we will assume that there is some function that is parameterized. Our eventual goal would be to learn the parameters.

Multiclass prediction

- Suppose we have K classes: Given an input $\boldsymbol{x}$, we need to predict one of these classes.
  - Let us number the labels as 1, 2, …, K

- Modeling K classes:
  - For a label i, we can define a scoring function $score(\boldsymbol{x}, i)$
  - The score is a real number. Higher score means that the label is preferred

- Prediction: find the label with the highest score

$$\operatorname*{argmax}_i score(\boldsymbol{x}, i)$$

Scores to probabilities

Suppose you wanted a model that predicts the probability that the label is i for an example $\boldsymbol{x}$.

The most common probabilistic model involves the softmax operator and is defined as:

$$P(i \mid \boldsymbol{x}) = \frac{\exp(score(i, \boldsymbol{x}))}{\sum_{j=1}^{K} \exp(score(j, \boldsymbol{x}))}$$

The softmax function

A general method to normalize scores into probabilities to produce a categorical probability distribution.

- Converts a vector of scores into a vector of probabilities

If we have a collection of K scores $z_1, z_2, \cdots, z_K$ that could be any real numbers, then their softmax gives K probabilities, each of which is defined as:

$$\frac{e^{z_1}}{e^{z_1} + e^{z_2} + \cdots + e^{z_K}}, \frac{e^{z_2}}{e^{z_1} + e^{z_2} + \cdots + e^{z_K}}, \cdots, \frac{e^{z_K}}{e^{z_1} + e^{z_2} + \cdots + e^{z_K}}$$

The numerator is the un-normalized probability for each outcome.

The denominator adds up the un-normalized probabilities for all ***competing*** outcomes.

What we didn't see:
How are the scores constructed?

They could be linear functions of the input features

$$score(\boldsymbol{x}, i) = \mathbf{w}_i^T \boldsymbol{x}$$

- This gives us multiclass SVM (if we use hinge loss) or multinomial logistic regression (if we use cross-entropy loss)

They could be a neural network

- Most commonly used with the softmax function

Important lesson: If you want multiple decisions to compete with each other, then place a softmax on top of them.

Is this the only way to predict multiple classes?

Is this the only way to predict multiple classes?

- Not really

- Historically, there have been several approaches

Is this the only way to predict multiple classes?

- Not really

- Historically, there have been several approaches
  - Reducing multiclass classification to several binary classification problems

Is this the only way to predict multiple classes?

- Not really

- Historically, there have been several approaches
  - Reducing multiclass classification to several binary classification problems
  - **One-vs-all**: K binary classifiers. For the i^{th} label, the binary classification problem is "label i vs. not label i".

Is this the only way to predict multiple classes?

- Not really

- Historically, there have been several approaches
  - Reducing multiclass classification to several binary classification problems
  - **One-vs-all**: K binary classifiers. For the i^{th} label, the binary classification problem is "label i vs. not label i".
  - **All-vs-all**: O(K^2) classifiers. One classifier for each pair of labels.

Is this the only way to predict multiple classes?

- Not really

- Historically, there have been several approaches
  - Reducing multiclass classification to several binary classification problems
  - **One-vs-all**: K binary classifiers. For the i^{th} label, the binary classification problem is "label i vs. not label i".
  - **All-vs-all**: $O(K^2)$ classifiers. One classifier for each pair of labels.
  - **Error correcting output codes**: Encode each label as a binary string and train one classifier for each position of the string

Is this the only way to predict multiple classes?

- Not really

- Historically, there have been several approaches
  - Reducing multiclass classification to several binary classification problems
  - **One-vs-all**: K binary classifiers. For the i^{th} label, the binary classification problem is "label i vs. not label i".
  - **All-vs-all**: $O(K^2)$ classifiers. One classifier for each pair of labels.
  - **Error correcting output codes**: Encode each label as a binary string and train one classifier for each position of the string

- **Exercise**: How would you construct the output in each case?

Exercises

1. What is the connection between the softmax function and the sigmoid function used in logistic regression?
   - To explore this, consider what happens when we have two classes and use softmax

2. Come up with at least two different prediction schemes for the all-vs-all setting

Where are we?

- Multiclass classification
- Modeling multiple classes
- Loss functions for multiclass classification
  - Once we have a loss, we can minimize it to train

The big picture

- We want to solve a multiclass classification problem with K classes

- We have defined the functional form of a scoring function
  - That is, a function that assigns a score to each label
  - We will call this $score(\boldsymbol{x}, i)$ for input $\boldsymbol{x}$ and label i
  - We could convert this to a probability via softmax too

- Our goal: Learn this scoring function
  - Actually the parameters that define it

- Or equivalently: Our goal is to define a loss function using that scoring function

The ingredients for defining a loss function

- We have a function that can assign scores (or probabilities) to a label
  - $score(\boldsymbol{x}, i)$ or $P(i \mid x)$ defined via softmax
  - The score is parameterized by some weights which are not shown

- We have an example $\boldsymbol{x}$ that has the ground truth label y
  - y is an integer between 1 and K

- Our goal: Penalize scoring functions that do not assign the highest score (or probability) to the label y

Two kinds of losses

- Multiclass hinge loss
  - Or max-margin loss
  - The multiclass version of the SVM
- Multiclass log loss
  - Or cross-entropy loss
  - The multinomial (i.e. multiclass) version of logistic regression

Multiclass hinge loss

The intuition:

- We want the true label to get a score that is *at least one more than* the score for any other label
- That is, there is a margin of one between the score for the true label and the score for any other label.

$$L(x, y) = \max_i(score(x, i) - score(x, y) + \Delta(y, i))$$

Multiclass hinge loss

The intuition:

- We want the true label to get a score that is *at least one more than* the score for any other label
- That is, there is a margin of one between the score for the true label and the score for any other label.

$$L(x, y) = \max_i(\underbrace{score(x, i)}_{\text{The score for label i}} - score(x, y) + \Delta(y, i))$$

Multiclass hinge loss

The intuition:

- We want the true label to get a score that is *at least one more than* the score for any other label
- That is, there is a margin of one between the score for the true label and the score for any other label.

$$L(x, y) = \max_i(\underbrace{score(x, i)}_{\text{The score for label i}} - \underbrace{score(x, y)}_{\text{The score for label y}} + \Delta(y, i))$$

Multiclass hinge loss

The intuition:

- We want the true label to get a score that is *at least one more than* the score for any other label
- That is, there is a margin of one between the score for the true label and the score for any other label.

$$L(x,y) = \max_i\big(\underbrace{score(x,i)}_{\text{The score for label i}} - \underbrace{score(x,y)}_{\text{The score for label y}} + \Delta(y,i)\big)$$

The "loss" term defined as:

$$\Delta(y,i) = \begin{cases} 0 & y = i \\ 1 & y \neq i \end{cases}$$

Multiclass hinge loss

The intuition:

- We want the true label to get a score that is *at least one more than* the score for any other label
- That is, there is a margin of one between the score for the true label and the score for any other label.

$$L(x, y) = \max_i \left(\underbrace{score(x, i)}_{\text{The score for label i}} - \underbrace{score(x, y)}_{\text{The score for label y}} + \Delta(y, i) \right)$$

The loss is defined by the label whose score, when augmented by the Δ is more than the score of the true label by the greatest amount.

The "loss" term defined as:

$$\Delta(y, i) = \begin{cases} 0 & y = i \\ 1 & y \neq i \end{cases}$$

The cross-entropy loss

The intuition:

- We want the true label to get the highest probability
- The loss is the negative log of the probability of the true label

$$L(\boldsymbol{x}, y) = -\log P(y \mid \boldsymbol{x})$$

The cross-entropy loss

The intuition:

- We want the true label to get the highest probability
- The loss is the negative log of the probability of the true label

$$L(\boldsymbol{x}, y) = -\log P(y \mid \boldsymbol{x})$$

Or sometimes, this is written using the indicator function

$$L(\boldsymbol{x}, y) = -\sum_{i} I[y = i] \log P(i \mid \boldsymbol{x})$$

$I[y = i]$ is zero for all values of i except when it is equal to the true label y, when it takes the value 1.

Exercises

- Show that the multiclass hinge loss is the same as the binary hinge loss when we have two labels.

- Show that the cross-entropy loss is the same as the logistic loss when we have two labels.

Multiclass classification: Wrapup

- Label belongs to a set that has more than two elements

- We saw how we can convert a label scoring function into:
  1. A probability for a label
  2. A prediction rule

- We saw two loss functions for multiclass classification
  - Hinge loss
  - Cross-entropy loss

Questions?