CptS 315 - HW 3

Zach Fechko (011711215)

15 November, 2022

Contents

Question 1

Answer the following with a yes or no along with proper justification.

Is the decision boundary of voted perceptron linear? Why or why not?
Is the decision boundary of averaged perceptron linear? Why or why not?

Answer

A

The decision boundary of the voted perceptron is not linear. If we have two weight vectors $(0, 1)$ and $(-1, 0)$ and we tried to discern positive and negative examples. Both of these vectors would last the same number of iterations in training, and only instances which would be positive would be in the top left quadrant.

B

The decision boundary of the averaged perceptron is linear. If we use the same example from part A, we would get a linear boundary because the average of the two vectors is $\frac{1(0,1)+1(-1,0)}{2} = (-\frac{1}{2}, \frac{1}{2})$ which is a linear boundary.

Question 2

Consider the following setting. You are provided with n training examples, $(x_1, y_1, h_1), (x_2, y_2, h_2), \ldots, (x_n, y_n, h_n)$ where x_i is the input example, y_i is the class label ($+1$ or -1), and $h_i > 0$ is the importance weight of the example. The teacher gave you some additional information by specifying the importance of each training example. How will you modify the perceptron algorithm to be able to leverage this extra information?

Answer

We can modify the perceptron algorithm to leverage this extra information by multiplying the weight vector by the importance weight of the example. This will allow us to weight the importance of each example and allow us to learn from the examples that are more important.

Question 3

Consider the following setting. You are provided with n training examples, $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ where x_i is the input example, y_i is the class label ($+1$ or -1). However the training data is highly imbalanced (say 90 of the examples are negative and 10 are positive) and we care about the accuracy of positive examples. How will you modify the perceptron algorithm to solve this learning problem? Explain your approach.

Answer

What we could do to modify the perceptron algorithm to solve this learning problem is to weight the positive examples more than the negative examples. We could do this by multiplying the weight vector by the number of positive examples divided by the number of negative examples. This would allow us to weight the positive examples more than the negative examples and allow us to learn from the positive examples more than the negative examples.

Question 4

You were just hired by MetaMind. MetaMind is expanding rapidly, and you decide to use your machine learning skills to assist them in their attempts to hire the best. To do so, you have the following available to you for each candidate i in the pool of candidates $\mathcal{I}$.

Their GPA

Whether they took a Data Mining course and got an A

Whether they took an algorithms course and got an A

Whether they have a job offer from Google

Whether they have a job offer from Facebook

The number of typos in their resume

You decide to repsent each candidate $i \in \mathcal{I}$ by a corresponding 6 dimensional feature vector $f(x^{(i)})$. You believe that if you just knew the right weight vector $w \in \mathbb{R}^6$ you could reliably predict the quality of a candidate i by computing $w \cdot f(x^{(i)})$. To determine w your boss lets you sample pairs of candidates from the pool. For a pair of candidates (k, l) you can have them face off in a "Data Mining Fight". The result is score$(k \succ l)$, which tells you that candidate k is at least score$(k \succ l)$ better than candidate l. Note that the score will be negative when l is a better candidate than k. Assume you collected scores for a set of pairs of candidates $\mathcal{P}$.

Describe how you would use a perceptron based algorithm to learn the weight vector w. Make sure to describe the basic intuition; how the weight updates will be done; and psuedocode for the entire algorithm.

Answer

Intuition

The intuition for the problem is:

1. The model produces a list of candidates in descending order of quality.
2. If a candidate is better than another based on the score (k, l), they will be placed ahead of the other candidate in the final list.
3. For each pair of candidates in pair $\mathcal{P}$, each candidate will be compared to each other to make the final list
4. The candidates who compare better than others will be placed higher on the list
5. If $k \succ l$, then $A > 0$ and the candidate makes the final list
6. If $k \leq l$, then $A \leq 0$ and the candidate does not make the final list
7. The bias B can be used to select the first B candidates in the final list

Psuedocode

```
1. set b = number of desired candidates based on demand
2. for each set of candidates in the pool based on the score (k, l)
    - Activation = weight * inputs + bias
    - if activation > 0
        - output = 1
    - else
        - output = 0
    - otherwise if lk <= 1, then Activation <= 0 and the candidate will not appear in final list
3. Get final list of candidates in descending order based on selection capability and merit
4. select first b candidates from the final list
```

Question 5

Suppose we have n_+ positive training examples and n_- negative training examples. Let C_+ be the center of the positive examples and C_- be the center of the negative examples. i.e. $C_+ = \frac{1}{n_+}\sum_{i:y_i=+1} x_i$ and $C_- = \frac{1}{n_-}\sum_{i:y_i=-1} x_i$. Consider a simple classifier called CLOSE that classifies a test example x by assigning it to the class whose center is closest

Show that the decision boundary of the CLOSE classifier is a linear hyperplane of the form $\text{sign}(w \cdot x + b)$. Compute the values of w and b in terms of C_+ and C_-.

Recall that the weight vector can be written as a linear combination of all the training examples $w = \sum_{i=1}^{n_+ + n_-} \alpha_i \cdot y_i \cdot x_i$. Compute the dual weights (α's). How many of the training examples are support vectors?

Answer

Part 1

The decision boundary of the CLOSE classifier is a linear hyperplane of the form $\text{sign}(w \cdot x + b)$. The values of w and b can be computed in terms of C_+ and C_- as follows: $w = C_+ - C_-$ and $b = -\frac{1}{2}(C_+ + C_-)$

Part 2

The dual weights (α's) can be computed as follows: $\alpha_i = y_i \cdot (w \cdot x_i + b)$ There are n_+ support vectors, which means that the majority of the training examples are used to support the decision boundary.

Question 6

Read the following paper and write a brief summary of the main points in at most two pages
A few useful things to know about machine learning

Answer

The article *A few useful things to know about machine learning* by Pedro Domingos is a brief summary of the main points of machine learning. The article is broken down into 9 main sections, each of which is a main point of machine learning.
The 9 sections are:

1. Learning = Representation + Evaluation + Optimization
    - This section addresses the fact that it's hard to choose the best algorithm for a given problem.
    - The key for not getting lost in the sea of algorithms is to know that it consists of combinations of 3 components, how the data is represented, how the outcome is evalueated, and how to find the best algorithm/parameters.
2. It's generalization that counts
    - The main goal is to be able to generalize beyond the examples in the training set and handle new unseen data
    - It's unlikely that we will encounter the exact same samples when we start testing
    - It's generally good practice to split your training data into a number of subsets and use cross validation to test the model
    - Using the same set of data for training and testing brings the illusion of success, meaning it will look accurate, but it's essentially the same as just memorizing the answers to a test instead of learning the concepts.
3. Overfitting has many faces
    - Overfitting is when you have great results with the training data but poor results with the testing data.
    - Domingos says that overfitting can come in "many forms that are not immediately obvious"
    - He also talks about dividing generalization error into bias (accuracy) and variance (precision) and selecting a model based on how the model is behaving
        - Linear models have high bias
        - Decision trees have high variance but low bias
4. Intuition fails in high dimensions
    - Domingos refers to a concept called the curse of dimensionality, which means that machine learning algorithms start to suffer as the demensionality of the data increases
    - This is because the ratio between the number of training examples and the number of features decreases extremely low
    - It's important to choose the most relevant features to avoid feeding noise to your learner.
5. Theoretical guarantees are not what they seem
    - Just because a model has a theoretical guarantee, it doesn't mean that it will work in practice
6. Feature Engineering is the key
    - Domingos calls feature engineering the most important factor of machine learning
    - If the features correlate well with the class, then learning is going to be really easy
    - He says that it's the most interesting parts of machine learning and attributes it to "black art"
7. More data beats a cleverer algorithm
    - When your model isn't that accurate, you reach a crossroads situation, where you can either "design a better learning algorithm, or gather more data", Domingos recommends the latter.
    - It's often quicker to just get more data than build a new algorithm
    - Algorithms learn from data and the more data you have, the more accurate the model will be because it has more examples to learn

8. Learn many models, not just one
    - In this section Domingos talks about the importance of ensemble learning, saying "one would test a lot of models and select the best model"
    - He also talks about a better approach which is to combine the strengths of multiple models into one super model.
9. Simplicity does not imply accuracy
    - "Given two classifiers with the same training error, the simpler of the two will likely have the lowest test error"
    - This section essentially tells the reader to follow Occam's razor, opting for the simpler model most of the time.