Day 4: Classification

Summer STEM: Machine Learning

Department of Electrical and Computer Engineering
NYU Tandon School of Engineering
Brooklyn, New York

July 15, 2021

Outline

Review

- Machine learning pipeline:
  - Process Data
  - Train on training data
  - Test on testing data
- Is it possible have a high accuracy for the training data and a low accuracy for the testing data? What should we do?

Review

- Imagine you are preparing for the SATs and you come across a book full of practice questions you did not understand how to solve any of the problems. However, you memorized all of the answers.
- What do you think will happen if you try to solve practice questions in a different book.
- Why are you studying actual problem solving techniques instead of just memorizing solutions from practice questions?
- Assuming you have an eidetic memory will memorizing solutions from practice questions be a good strategy?

Review

- $J(w) = \frac{1}{N}\|Y - Xw\|^2 + \lambda\|w\|^2$
- $w = [10000, 20000, 30000, 10000]$ does this look good?

Outline

Motivation

- Cannot rely on closed form solutions
  - Computation efficiency: operations like inverting a matrix is not efficient
  - For more complex problems such as neural networks, a closed-form solution is not always available
- Need an optimization technique to find an optimal solution
  - Machine learning practitioners use **gradient**-based methods

Gradient Descent Algorithm

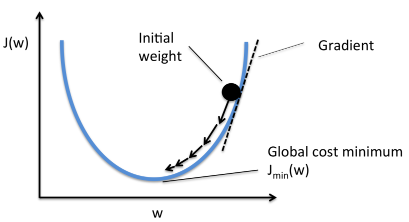
- Update Rule

  $Repeat\{$

  $$\mathbf{w}_{new} = \mathbf{w} - \alpha \nabla J(\mathbf{w})$$

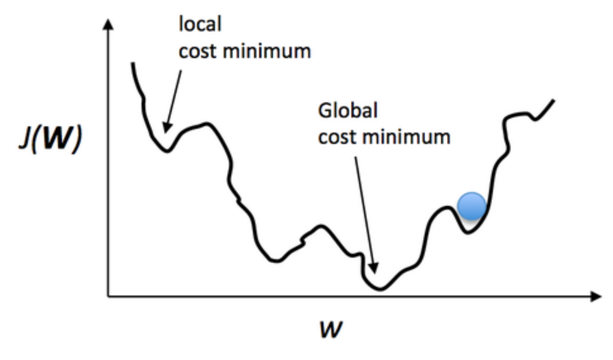
  $\}$

  α is the learning rate

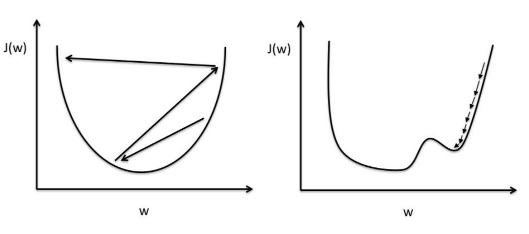


General Loss Function Contours

- Most loss function contours are not perfectly parabolic
- Our goal is to find a solution that is very close to global minimum by the right choice of hyper-parameters

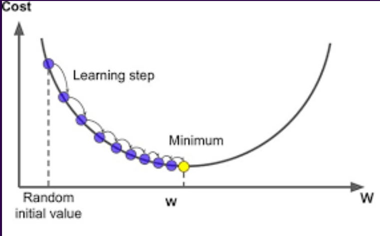


Understanding Learning Rate



Large learning rate: Overshooting.

Small learning rate: Many iterations until convergence and trapping in local minima.



Correct learning rate

Outline

Classification Vs. Regression

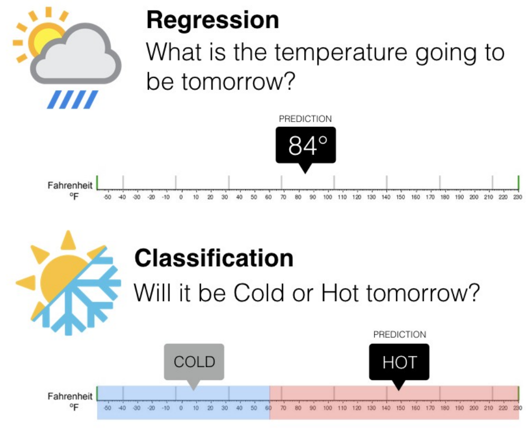
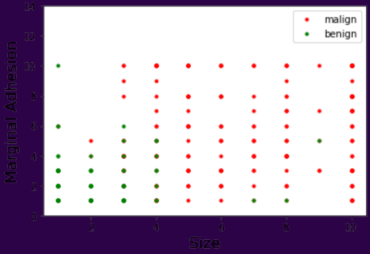
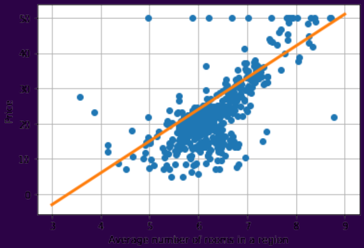


Figure: https://www.pinterest.com/pin/672232681855858622/?lp=true

Classification Vs. Regression



(a) Breast cancer dataset

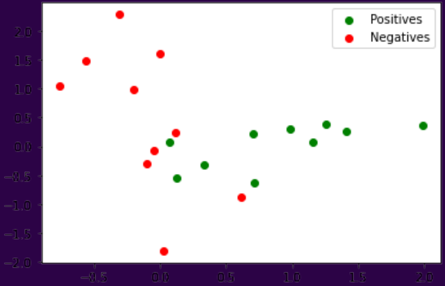


(b) Boston Housing dataset

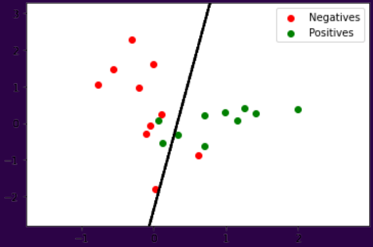
Classification

Given the dataset (x_i, y_i) for $i = 1, 2, \ldots, N$, find a function $f(x)$ (model) so that it can predict the label $\hat{y}$ for some input x, even if it is not in the dataset, i.e. $\hat{y} = f(x)$.

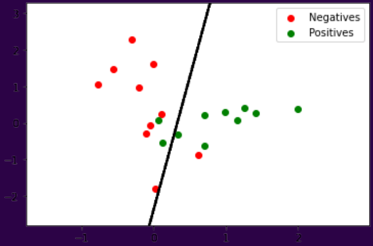
- Positive : $y = 1$
- Negative : $y = 0$



Decision Boundary



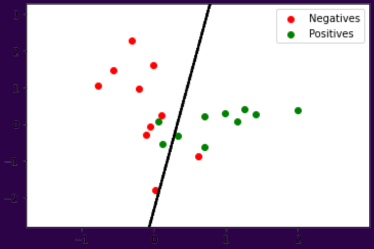
Decision Boundary



- Evaluation metric :

$$\text{Accuracy} = \frac{\text{Number of correct prediction}}{\text{Total number of prediction}}$$

- What is the accuracy in this example ?



- Evaluation metric :

$$\text{Accuracy} = \frac{\text{Number of correct prediction}}{\text{Total number of prediction}} = \frac{17}{20} = 0.85 = 85\%$$

Need for a new model

- What would happen if we used the linear regression model :

$$\hat{y} = w_0 + w_1 x$$

Need for a new model

- What would happen if we used the linear regression model :

$$\hat{y} = w_0 + w_1 x$$

- y is 0 or 1
- $\hat{y}$ will take any value between $-\infty$ and ∞

Need for a new model

- What would happen if we used the linear regression model :

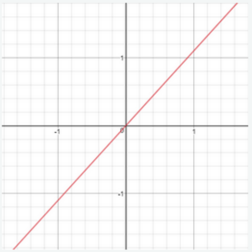
$$\hat{y} = w_0 + w_1 x$$

- y is 0 or 1
- $\hat{y}$ will take any value between $-\infty$ and ∞
- It will be hard to find w_0 and w_1 that make the prediction $\hat{y}$ match the label y.

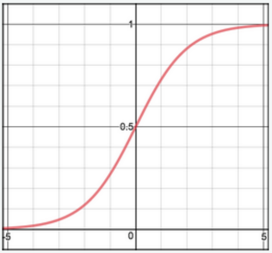
Sigmoid Function

- By applying the sigmoid function, we enforce $0 \leq \hat{y} \leq 1$

$$\hat{y} = \text{sigmoid}(w_0 + w_1 x) = \frac{1}{1 + e^{-(w_0 + w_1 x)}}$$



(a) Linear model



(b) Sigmoid model

A new loss function

- Binary cross entropy loss :

$$\text{Loss} = \frac{1}{N}\sum_{i=1}^{N}\Big[-y_i\log(\hat{y}_i) - (1-y_i)\log(1-\hat{y}_i)\Big]$$

pause

- What happens if $y_i = 0$:
$\Big[-y_i\log(\hat{y}_i) - (1-y_i)\log(1-\hat{y}_i)\Big] = ?$

A new loss function

- Binary cross entropy loss :

$$\text{Loss} = \frac{1}{N}\sum_{i=1}^{N}\Big[-y_i\log(\hat{y}_i) - (1-y_i)\log(1-\hat{y}_i)\Big]$$

- If $y_i = 0$:
$$\Big[-y_i\log(\hat{y}_i) - (1-y_i)\log(1-\hat{y}_i)\Big] = -\log(1-\hat{y}_i)$$

A new loss function

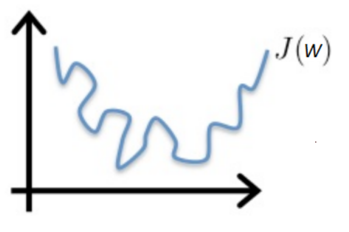
- Binary cross entropy loss :

$$\text{Loss} = \frac{1}{N}\sum_{i=1}^{N}\Big[-y_i\log(\hat{y}_i) - (1-y_i)\log(1-\hat{y}_i)\Big]$$

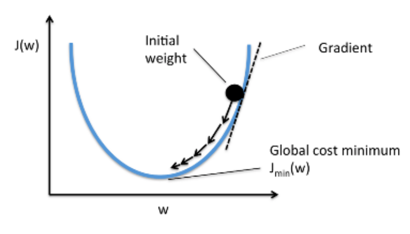
- If $y_i = 0$:
  $\Big[-y_i\log(\hat{y}_i) - (1-y_i)\log(1-\hat{y}_i)\Big] = -\log(1-\hat{y}_i)$
- If $y_i = 1$:
  $\Big[-y_i\log(\hat{y}_i) - (1-y_i)\log(1-\hat{y}_i)\Big] = -\log(\hat{y}_i)$

MSE vs Binary cross entropy loss

- MSE of a logistic function has many local minima.
- The Binary cross entropy loss has only one minimum.



(a) MSE



(b) Binary cross entropy loss

Classifier

$$\hat{y} = \text{sigmoid}(w_0 + w_1 x) = \frac{1}{1 + e^{-(w_0 + w_1 x)}}$$

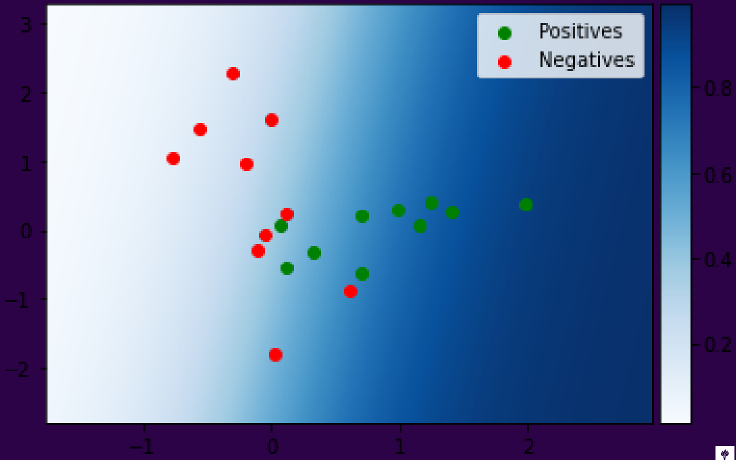
- How to deal with uncertainty ?
  - Thanks to the sigmoid, $\hat{y} = f(x)$ is between 0 and 1.

Classifier

$$\hat{y} = \text{sigmoid}(w_0 + w_1 x) = \frac{1}{1 + e^{-(w_0 + w_1 x)}}$$

- How to deal with uncertainty ?
  - Thanks to the sigmoid, $\hat{y} = f(x)$ is between 0 and 1.
- If $\hat{y}$ is close to 0, the data is probably negative
- If $\hat{y}$ is close to 1, the data is probably positive
- If $\hat{y}$ is around 0.5, we are not sure.

Classifier



Decision Boundary

- Once, we have a classifier outputting a score $0 < \hat{y} < 1$, we need to create a decision rule.

Decision Boundary

- Once, we have a classifier outputting a score $0 < \hat{y} < 1$, we need to create a decision rule.
- Let $0 < t < 1$ be a Threshold :
  - If $\hat{y} > t$, $\hat{y}$ is classified as positive.
  - If $\hat{y} < t$, $\hat{y}$ is classified as negative.

Decision Boundary

- Once, we have a classifier outputting a score $0 < \hat{y} < 1$, we need to create a decision rule.
- Let $0 < t < 1$ be a Threshold :
  - If $\hat{y} > t$, $\hat{y}$ is classified as positive.
  - If $\hat{y} < t$, $\hat{y}$ is classified as negative.
- How to choose t ?

Impact of the threshold

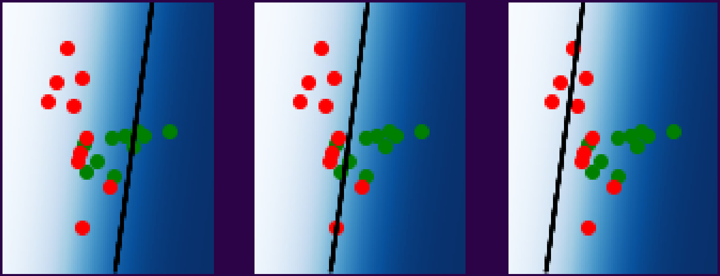


Figure: $t = 0.2, 0.5, 0.8$

Performance metrics for a classifier

- Accuracy of a classifier: percentage of correct classification
- Why accuracy alone is not a good measure for assessing the model ?

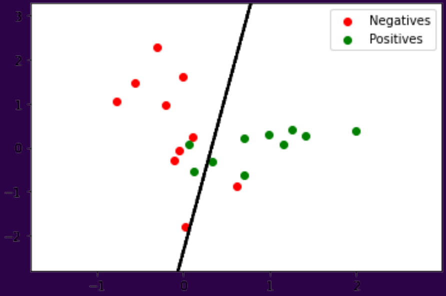
Performance metrics for a classifier

- Accuracy of a classifier: percentage of correct classification
- Why accuracy alone is not a good measure for assessing the model ?
  - Example: A rare disease occurs 1 in ten thousand people
  - A test that classifies everyone as free of the disease can achieve 99.999% accuracy when tested with people drawn randomly from the entire population

Types of Errors in Classification

- Correct predictions:
  - True Positive (TP) : Predict $\hat{y} = 1$ when $y = 1$
  - True Negative (TN) : Predict $\hat{y} = 0$ when $y = 0$
- Two types of errors:
  - False Positive/ False Alarm (FP): $\hat{y} = 1$ when $y = 0$
  - False Negative/ Missed Detection (FN): $\hat{y} = 0$ when $y = 1$

Example



- How many True Positive (TP) are there ?
- How many True Negative (TN) are there ?
- How many False Positive (FP) are there ?
- How many False Negative (FN) are there ?

Other metrics

- Sensitivity/Recall/TPR (How many positives are detected among all positive?)

$$\frac{\text{TP}}{\text{TP} + \text{FN}}$$

- Precision (How many detected positives are actually positive?)

$$\frac{\text{TP}}{\text{TP} + \text{FP}}$$

Outline

Lab: Diagnosing Breast Cancer

- We're going to use the breast cancer dataset to predict whether the patients' scans show a malignant tumour or a benign tumour.
- Let's try to find the best linear classifier using logistic regression.

Outline

Multiclass Classificaiton

- Previous model: $f(\mathbf{x}) = \sigma(\phi(\mathbf{x})w)$
- Representing Multiple Classes:
  - One-hot / 1-of-K vectors, ex : 4 Class
  - Class 1 : $\mathbf{y} = [1, 0, 0, 0]$
  - Class 2 : $\mathbf{y} = [0, 1, 0, 0]$
  - Class 3 : $\mathbf{y} = [0, 0, 1, 0]$
  - Class 4 : $\mathbf{y} = [0, 0, 0, 1]$

Multiclass Classificaiton

- Previous model: $f(\mathbf{x}) = \sigma(\phi(\mathbf{x})w)$
- Representing Multiple Classes:
  - One-hot / 1-of-K vectors, ex : 4 Class
  - Class 1 : $\mathbf{y} = [1, 0, 0, 0]$
  - Class 2 : $\mathbf{y} = [0, 1, 0, 0]$
  - Class 3 : $\mathbf{y} = [0, 0, 1, 0]$
  - Class 4 : $\mathbf{y} = [0, 0, 0, 1]$
- Multiple outputs: $f(\mathbf{x}) = \text{softmax}(\phi(\mathbf{x})W)$
- Shape of $\phi(\mathbf{x})W$: $(N, K) = (N, D) \times (D, K)$
- $\text{softmax}(\mathbf{z})_k = \dfrac{e^{z_k}}{\sum_j e^{z_j}}$

Multiclass Classificaiton

- Multiple outputs: $f(\mathbf{x}) = \text{softmax}(\mathbf{z})$ with $\mathbf{z} = \phi(\mathbf{x})W$
- $\text{softmax}(\mathbf{z})_k = \dfrac{e^{\mathbf{z}_k}}{\sum_j e^{\mathbf{z}_j}}$
- Softmax example: If $\mathbf{z} = \begin{bmatrix} -1 \\ 2 \\ 1 \\ -4 \end{bmatrix}$ then,

$$\text{softmax}(z) = \begin{bmatrix} \frac{e^{-1}}{e^{-1}+e^{2}+e^{1}+e^{-4}} \\ \frac{e^{2}}{e^{-1}+e^{2}+e^{1}+e^{-4}} \\ \frac{e^{1}}{e^{-1}+e^{2}+e^{1}+e^{-4}} \\ \frac{e^{-4}}{e^{-1}+e^{2}+e^{1}+e^{-4}} \end{bmatrix} \approx \begin{bmatrix} 0.035 \\ 0.704 \\ 0.259 \\ 0.002 \end{bmatrix}$$

NYU TANDON SCHOOL OF ENGINEERING

Cross-entropy

- Multiple outputs: $\hat{\mathbf{y}}_{\mathbf{i}} = \text{softmax}(\phi(\mathbf{x}_i)W)$
- Cross-Entropy: $J(W) = -\sum_{i=1}^{N}\sum_{k=1}^{K} \mathbf{y}_{ik} log(\hat{\mathbf{y}}_{ik})$
- Example : $\text{K} = 4$

$$\text{If, } \mathbf{y}_i = [0, 0, 1, 0] \text{ then, } \sum_{k=1}^{K} \mathbf{y}_{ik} log(\hat{\mathbf{y}}_{ik}) = log(\hat{\mathbf{y}}_{i3})$$

Outline

Lab: Iris Dataset

- Open demo_iris.ipynb