Day 5: Mini-Project

Summer STEM: Machine Learning

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

June 26, 2020

Recap: machine learning pipeline

0. Determine your task:
   regression / classification
1. Collect the data
   - training / validation
     randomly 80% 20%
     test set will be given separately
   - feature / label
2. Pick your model

Ex • linear models

Scalar-valued features

$\hat{y} = w_0 + w_1 x$ or $\hat{y} = w_0 + w_1 x + w_2 x^2 + \cdots + w_M x^M$

$\hat{y} = w^T \phi(x)$

• vector-valued features

$x_i = \begin{bmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{iD} \end{bmatrix}$ $\phi(x_i) = \begin{bmatrix} 1 \\ x_{i1} \\ x_{i2} \\ \vdots \\ x_{iD} \end{bmatrix}$

$w = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \end{bmatrix}$

$$\begin{bmatrix} \vdots \\ w_D \end{bmatrix}$$

$$\hat{y} = w^T \phi(x)$$

3. Loss function : MSE / MAE

$$\text{MSE} : J(w) = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2$$

$$\text{MAE} : J(w) = \frac{1}{N} \sum_{i=1}^{N} |y_i - \hat{y}_i|$$

4 Training

i.e. find optimal model parameters w^* such that $J(w^*)$ is the minimum among all possible w.

Linear models and MSE

$$w^* = (X^T X)^{-1} X y$$

X: design matrix, each row of X is not x_i but $\phi(x_i)$

Scalar-valued feature

fit polynomials

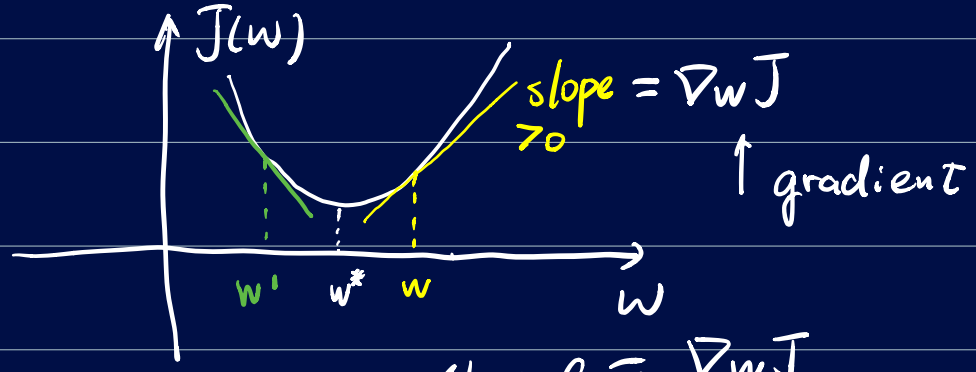
$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}$$

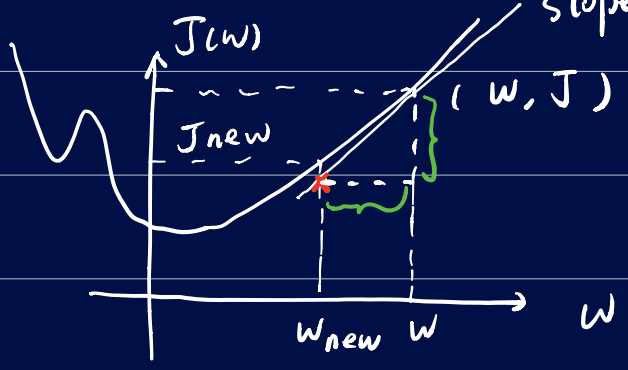
original

$$\begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^M \\ 1 & x_2 & x_2^2 & & x_2^M \\ & & & & \\ 1 & x_N & x_N^2 & \cdots & x_N^M \end{bmatrix}$$

design matrix

- What if we don't have a formula for $\underset{\sim}{w}^*$? Gradient descent update rule: $w_{new} = w - \alpha \nabla_w J(w)$





slope $g = \nabla_w J$

(w_{new}, J')

$$g = \frac{J - J'}{w - w_{new}}$$

$$J - J' = g(w - w_{new})$$

$$J' - J = g(w_{new} - w)$$

$$J_{new} \approx J' = J + g(w_{new} - w)$$

the closer w_{new} and w are, the better J' approximates J_{new}

$$J_{new} \approx J + (\nabla_w J)(w_{new} - w)$$

We want $J_{new} < J$ $\quad (< 0)$

Plug in the update rule

$$J_{new} \approx J + \nabla_w J \left(w - \alpha \nabla_w J - w \right)$$

$$\approx J + \nabla_w J \cdot (-\alpha \nabla_w J)$$

$$\approx J - \alpha (\nabla_w J)^2$$

$$J_{new} - J \approx \underline{-\alpha (\nabla_w J)^2} < 0$$

$\alpha > 0$: learning rate

α is a scalar

is a hyper-parameter

Tuning hyperparameters on the Validation set

$(\alpha, \lambda, \text{intercept}, M)$

α : Learning rate

λ : regularization coeff

M : order/degree of the polynomials

$(\ldots)$

(Lasso)
L1-norm regularization enforces some weights to be 0
→ could be used to determine the order M of the polynomial to be fit.
L2-norm regularization (Ridge) enforces the weights to be small (but not necessarily 0)

5. Evaluate the trained model on the test set. At this stage, you Should NOT change your model anymore!
You don't have to use the same loss as the training loss here.

Outline

Classification Vs. Regression

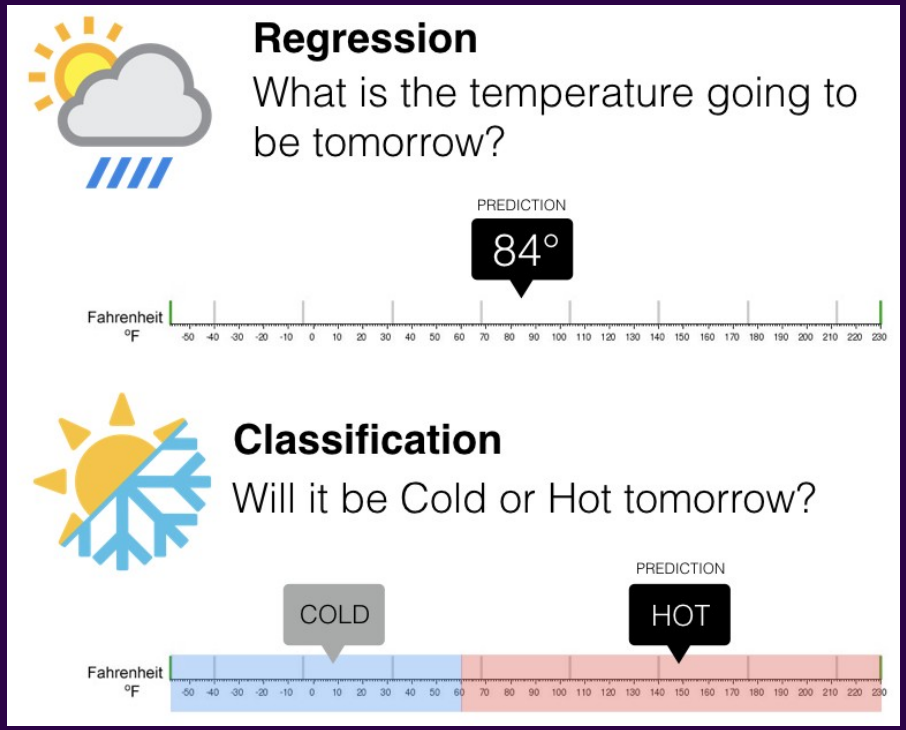
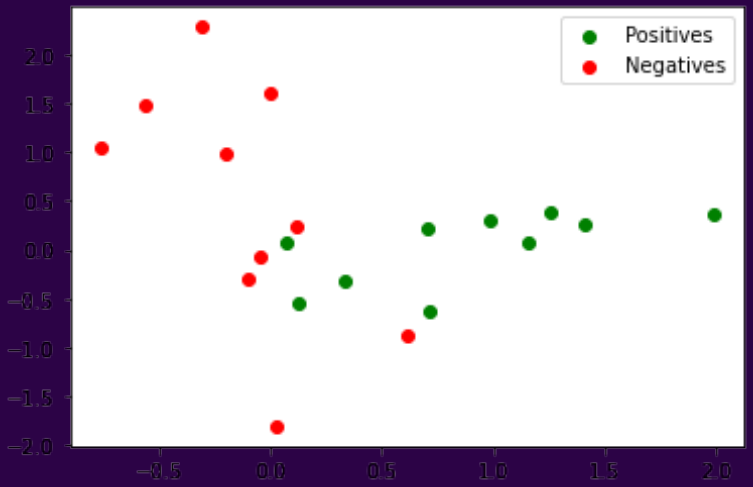


Figure: https://www.pinterest.com/pin/672232681855858622/?lp=true

Classification

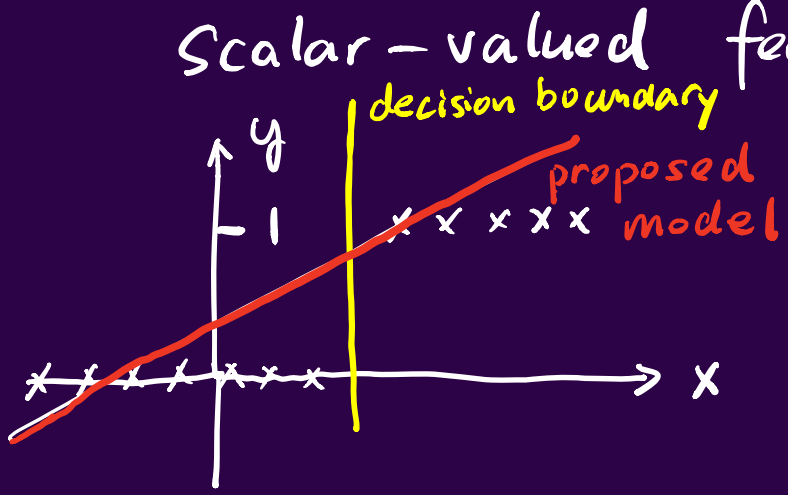
Given the dataset (x_i, y_i) for $i = 1, 2, \ldots, N$, find a function $f(x)$ (model) so that it can predict the label $\hat{y}$ for some input x, even if it is not in the dataset, i.e. $\hat{y} = f(x)$.

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



Classification via regression

- Proposal: train a model to fit the data with linear regression!

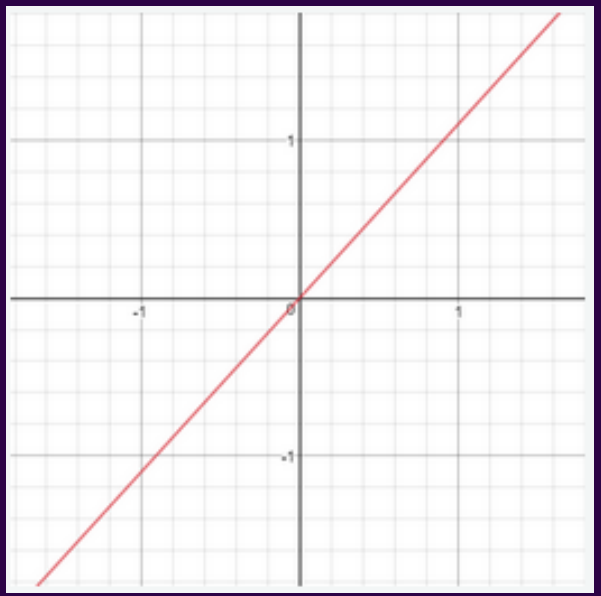


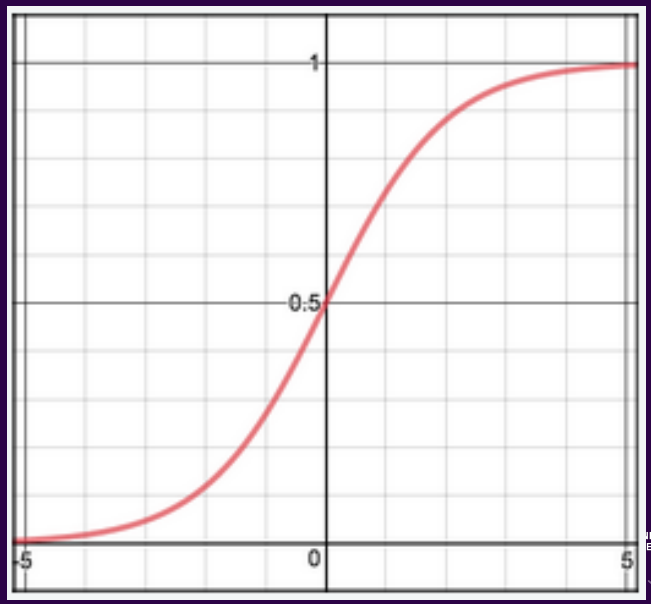
Classification via regression

- Proposal: train a model to fit the data with linear regression!
- What could be the problem?

Sigmoid Function

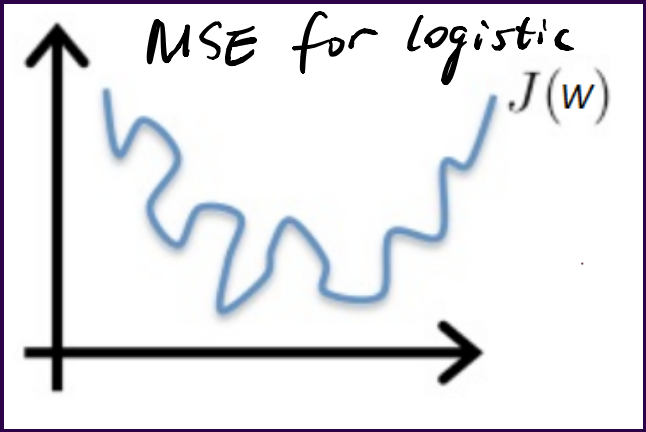
- Recall from linear regression $z = w_0 + w_1 x$ $w^T \phi(x)$
- By applying the sigmoid function to z, we enforce $0 \leq \hat{y} \leq 1$
  - $\hat{y} = sigmoid(z) = \frac{1}{1+e^{-z}}$

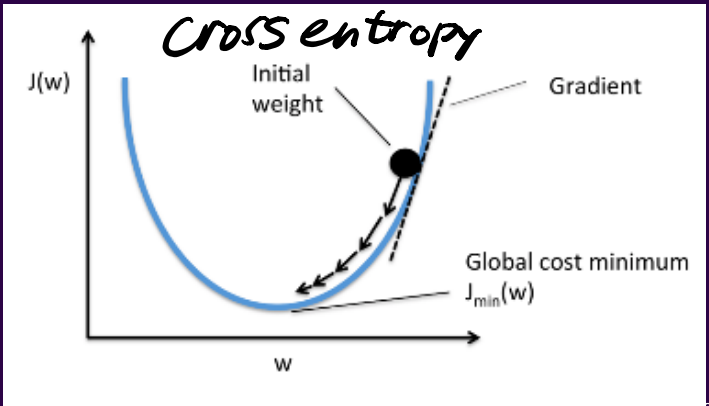




Classification Loss Function

- Cannot use the same cost function that we used for linear regression
  - MSE of a logistic function has many local minima
- Use $\frac{1}{N}\sum_{i=1}^{N}\left[-ylog(\hat{y})-(1-y)log(1-\hat{y})\right]$
  - This loss function is called binary cross entropy loss
  - This loss function has only one minimum

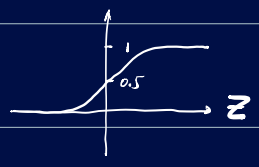




Logistic Regression

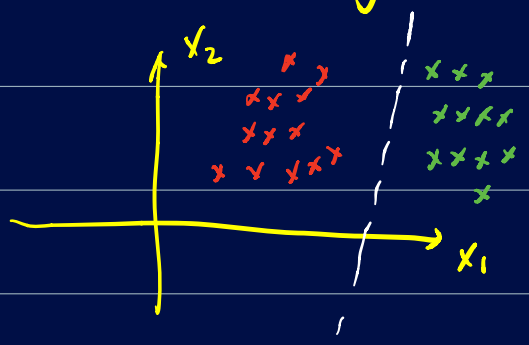
$$\text{predicted label} = \begin{cases} 0 & \text{if sigmoid}(w^T\phi(x)) < t \\ 1 & \text{if sigmoid}(w^T\phi(x)) \geq t \end{cases}$$

sigmoid(z): $\frac{1}{1+e^{-z}}$



decision threshold t: typically $t = 0.5$

decision boundary: $\text{sigmoid}(w^T\phi(x)) = t$



negative $y = 0$

positive $y = 1$

Cross-entropy loss:

$$J(w) = \frac{1}{N}\sum_{i=1}^{N}\left[-y\log(\hat{y}) - (1-y)\log(1-\hat{y})\right]$$

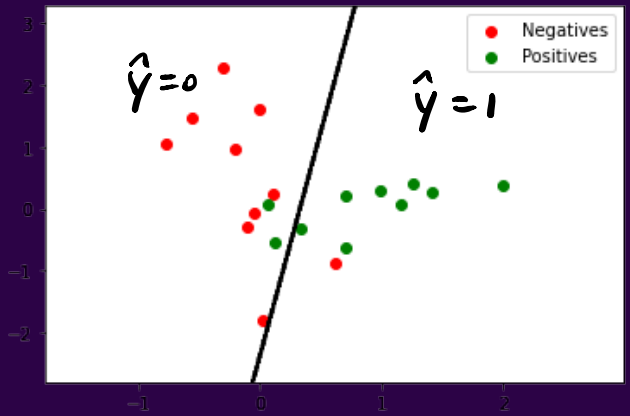
* here

$\hat{y} = \text{sigmoid}(w^T\phi(x))$

$$y=0 \quad \frac{1}{N}\sum_{i=1}^{N_{y=0}}\left(-\log(1-\hat{y})\right)$$

$$y=1 \quad \frac{1}{N}\sum_{i=1}^{N_{y=1}}\left(-\log(\hat{y})\right)$$

Decision Boundary

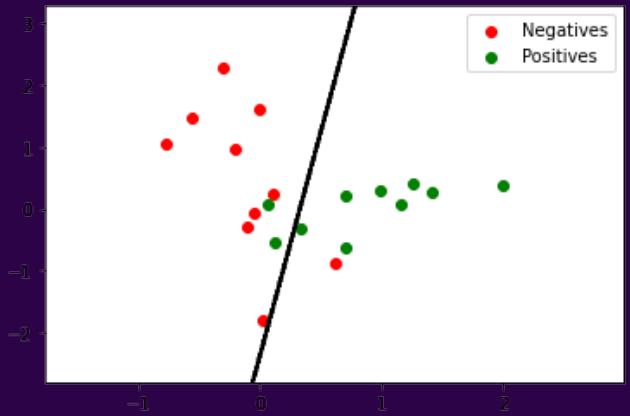


- Evaluation metric : (on test set)

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

$Y = \hat{Y}$

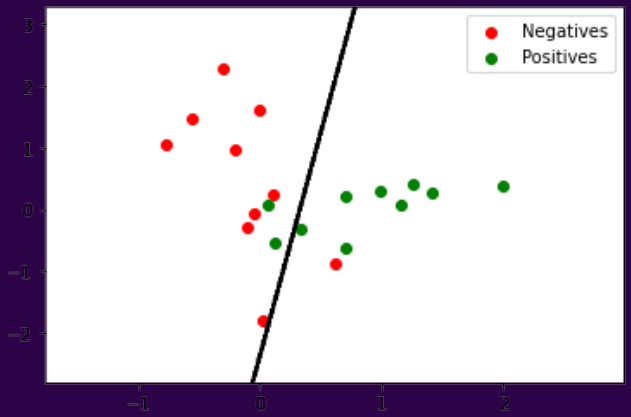
Decision Boundary



- Evaluation metric :

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

- What is the accuracy in this example ?



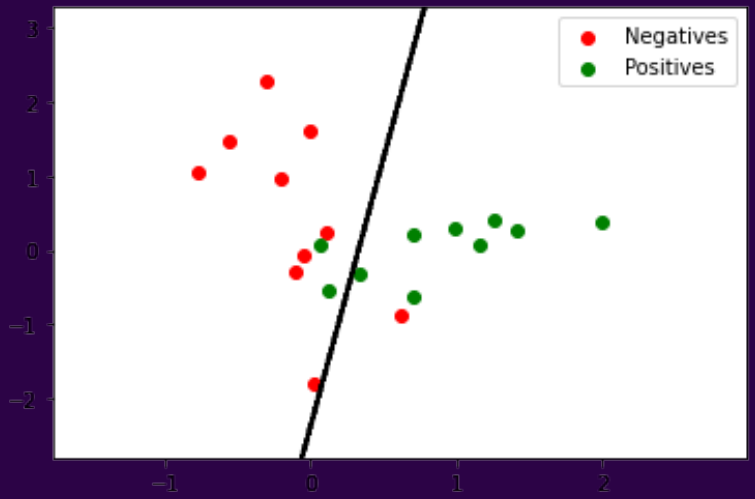
- Evaluation metric :

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

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 ?

Performance metrics for a classifier

- Accuracy of a classifier:
  - (TP + TF)/(TP+FP+TN+FN) (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:
  - (TP + TF)/(TP+FP+TN+FN) (percentage of correct classification)
- Why accuracy alone is not a good measure for assessing the model
  - There might be an overwhelming proportion of one class over another (unbalanced classes)
  - 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

Other metrics

Some other metrics

- Sensitivity/Recall/TPR = TP/(TP+FN) (How many positives are detected among all positive?)
- Precision = TP/(TP+FP) (How many detected positives are actually positive?)
- Specificity/TNR = TN/(TN+FP) (How many negatives are detected among all negatives?)

Exercise: think of tasks for which sensitivity, precision, or specificity is a better metric.

Outline

Mini Project

- Task: Predict fish weight!
- You should split the given dataset into training and validation set.
- Test set will be released on Sunday night.
- Next Monday morning: present your project and the model performance on the test set.
- Each team should present for 8-10 minutes.

Presentation Template

- Slide 1: Title and introduction
- Slide 2: Your model and loss function
- Slide 3 & 4: What is your choice of feature transformation, regularizer (Ridge/Lasso?) hyper-parameters, etc.
- Slide 5: Model performance on training and test set?
- Slide 6: Challenges and how you resolve them.
- Slide 7: Conclusion

Thank You!

- Next Week: Deep Learning
- Have a fun weekend!
- Revise Revise Revise!