CM146 HW2

Stephanie Chen

October 30 2019

1. a)AND
We can use w^T = [1 1], b = -1, such that w^Tx + b = ([1 1] x -1) as a linear model for AND function. Let $y^{test} = sgn(w^T x)$, then for the following input data:

$$(w[-1 \quad -1]) - 1 = (-2) - 1 = -3, y^{test} = y = -1$$

$$(w[-1 \quad 1]) - 1 = (0) - 1 = -1, y^{test} = y = -1$$

$$(w[1 \quad -1]) - 1 = (0) - 1 = -1, y^{test} = y = -1$$

$$(w[1 \quad 1]) - 1 = (2) - 1 = 1, y^{test} = y = 1$$

Another linear model that can be used to model the AND function is
w^T = [1 $\frac{1}{2}$], b = -1, such that w^Tx + b = ([1 $\frac{1}{2}$] x - 1)
Using this w and b, we get the following results when we test our model with the same input data from above:
(w [-1 -1]) - 1 = ($\frac{-3}{2}$) - 1 = -$\frac{5}{2}$, $y^{test} = sgn(\frac{-5}{2}) = y = -1$
(w [-1 1]) - 1 = ($\frac{-1}{2}$) - 1 = $\frac{-3}{2}$, $y^{test} = sgn(\frac{-3}{2}) = y = -1$
(w [1 -1]) - 1 = ($\frac{1}{2}$) - 1 = $\frac{-1}{2}$, $y^{test} = sgn(\frac{-1}{2}) = y = -1$
(w [1 1]) - 1 = ($\frac{3}{2}$) - 1 = $\frac{1}{2}$, $y^{test} = sgn(\frac{1}{2}) = y = 1$

b)XOR
XOR is not linearly separable since we cannot draw a line to separate data points [-1,-1] and [1,1] (the points with label y = -1 by the XOR function) from points [-1, 1] and [1, 1] (the points with label y = 1), so no linear model exists.

2. a) The partial derivative is

$$\sum_{n=1}^{N} \left(\sigma\left(\boldsymbol{w}^T \boldsymbol{x}_n\right) - y_n\right) x_{n,j}$$

3. a)

$$J(w_0, w_1) = \sum_{n=1}^{N} \alpha_n \left(w_0 + w_1 x_{n,1} - y_n\right)^2$$

$$\frac{\partial J}{\partial w_0} = \sum_{n=1}^{N} 2\alpha_n (w_0 + w_1 x_{n,1} - y_n)$$

$$\frac{\partial J}{\partial w_1} = \sum_{n=1}^{N} 2\alpha_n (w_0 + w_1 x_{n,1} - y_n) x_{n,1}$$

b) Let $\frac{\partial J}{\partial w_0} = 0$, then

$$\sum_{n=1}^{N} \alpha_n w_0 + \sum_{n=1}^{N} \alpha_n w_1 x_{n,1} - \sum_{n=1}^{N} \alpha_n y_n = 0$$

$$(\sum_{n=1}^{N} \alpha_n) w_0 + (\sum_{n=1}^{N} \alpha_n x_{n,1}) w_1 = (\sum_{n=1}^{N} \alpha_n) y_n \tag{1}$$

Let $\frac{\partial J}{\partial w_1} = 0$, then

$$\sum_{n=1}^{N} \alpha_n w_0 x_{n,1} + \sum_{n=1}^{N} \alpha_n w_1 {x_{n,1}}^2 - \sum_{n=1}^{N} \alpha_n y_n x_{n,1} = 0$$

$$(\sum_{n=1}^{N} \alpha_n x_{n,1}) w_0 + (\sum_{n=1}^{N} \alpha_n {x_{n,1}}^2) w_1 = (\sum_{n=1}^{N} \alpha_n x_{n,1}) y_n \tag{2}$$

From (1), we can get

$$w_0 = \frac{\sum_{n=1}^{N} \alpha_n y_n - \sum_{n=1}^{N} (\alpha_n x_{n,1}) w_1}{\sum_{n=1}^{N} \alpha_n}$$

From (2), we can get

$$w_1 = \frac{\sum_{n=1}^{N} \alpha_n x_{n,1} y_n - \sum_{n=1}^{N} \alpha_n x_{n,1} (w_0)}{\sum_{n=1}^{N} \alpha_n {x_{n,1}}^2}$$

We can then plug in w_0 we get from (1) to the equation for w_1 we get from (2) to solve for w_0 and w_1 that minimizes J.

4. a) If the data is linearly separable, then by the convergence theorem, there exists a unit vector $\vec{u}$ and some b such that $y_i(\vec{u}^T\vec{x_i} + b) \geq \gamma$, where γ is the margin of the data set.
We can divide both sides of this equation by γ to get: yi($\frac{\vec{u}^T}{\gamma}\vec{x_i} + \frac{b}{\gamma}$) ≥ 1
This is an optimal solution to the linear program (2) with $\delta = 0$

b)An optimal solution with $\delta = 0$ means $y_i(\vec{w}^T\vec{x_i} + b) \geq 1$ for all training instances from $D = \{(\vec{x_i}, y_i)\}_{i=1}^m$. For such a solution, y_i and $\vec{w}^T\vec{x_i} + b$ have the same sign for all m instances.
We can define a solution as follow:

$y_i = 1$ if $(\vec{w}^T\vec{x_i} + b) \geq 0$
y_i = -1 if $(\vec{w}^T\vec{x_i} + b) < 0$

c) What can we say about the linear separability of the data set if there exists a hyper-plane that satisfies condition (2) with $\delta > 0$?

If $\delta > 0$ and there exists a hyper-plane that satisfies (2), then we know the data is linearly separable if $\delta > 0$ and $\delta < 1$. If $\delta = 1$, then the data is not separable. If $\delta > 1$, we also cannot tell if the data is separable or not.

d)An alternative LP formulation to (2) may be

$$\begin{aligned} \min \quad & \delta \\ \text{subject to} \quad & y_i(\vec{w}^T\vec{x_i} + b) \geq -\delta, \qquad \forall(\vec{x_i}, y_i) \in D \\ & \delta \geq 0 \end{aligned}$$

Find the optimal solution to this formulation (independent of D) to illustrate the issue with such a formulation.

An optimal solution to this formulation is to set $\vec{w} = 0$, b = 0, and $\delta = 0$. This would satisfy the condition of the LP formulation, but then $\vec{w}^T\vec{x_i} + b$ will not form a hyper-plane and thus we cannot derive any information from the data set.

e) Since there are only two points in D, we know D is separable. By (2), we need to satisfy the formulation $y_i(\vec{w}^T\vec{x_i} + b) \geq \gamma$. Let $\vec{w}^T = [w_1\ w_2$

$w_3]$, then given this data set D, we need $\vec{w}^T$ and b such that

$$w_1 + w_2 + w_3 + b \geq 1$$

and

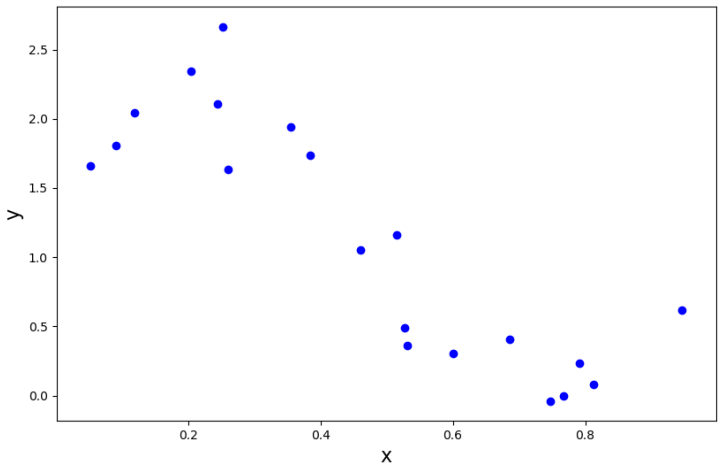
$$-w_1 - w_2 - w_3 + b \geq 1$$

. This means we need

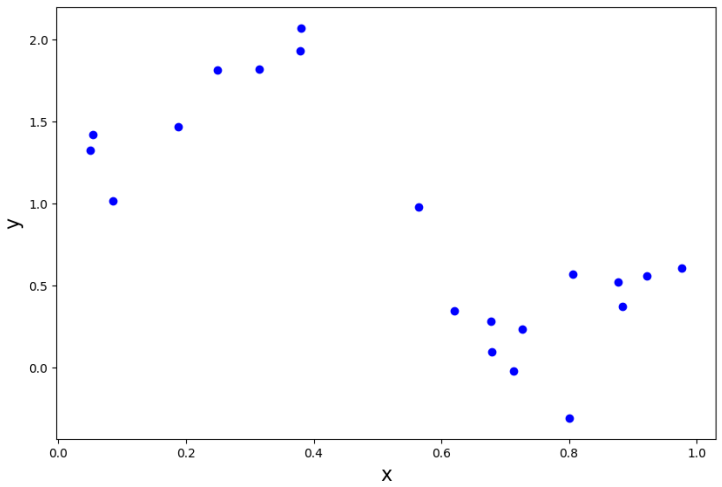
$$w_1 + w_2 + w_3 \geq |1 - b|$$

A possible optimal solution is $\delta = 0$, b=2, and
$\vec{w}^T = [w_1 \ w_2 \ w_3] = [\frac{1}{3} \ \frac{1}{3} \ \frac{1}{3}]$

5. a) Visualize the training and test data using the `plot_data(...)` function. What do you observe? For example, can you make an educated guess on the effectiveness of linear regression in predicting the data?



For the plot for training data, a linear regression does a moderate job in representing the data points as there is an overall downward trend.



For the test data, a polynomial regression would be better as the data does not show a nice linear trend.

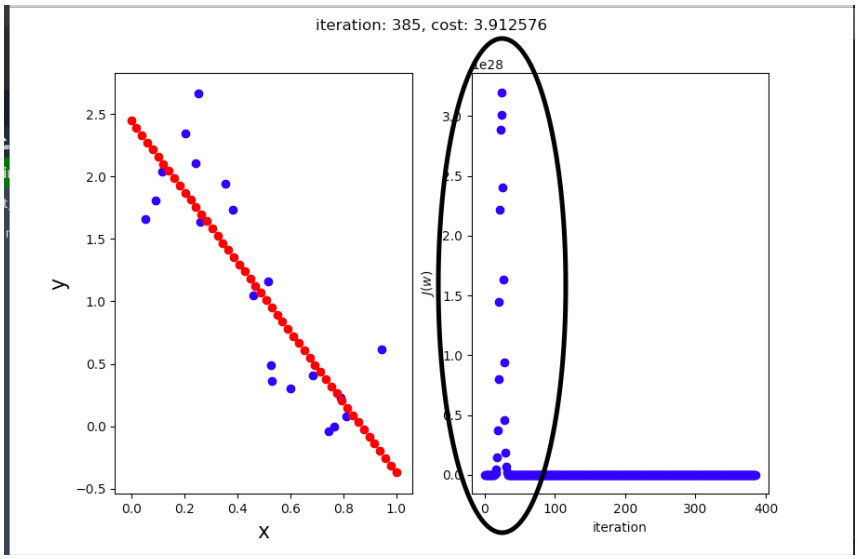
b) Modified `PolynomialRegression.generate_polynomial_features(...)` as required.
c) Modified `PolynomialRegression.predict(...)` to predict y. d)

learning rate (eta)	coefficients	number of iterations	final cost	time
0.0407	[-9.40470931e+18 -4.65229095e+18]	10000	e^(30+) order	more than an hour
10^-2 = 0.01	[2.44640703 -2.81635346]	764	3.912576	132.98944 seconds
10^-3 = 0.001	[2.4464068 -2.816353]	7020	3.912576	12247.992311 seconds
10^-4	[2.27044798 -2.46064834]	10000	4.086397	14595.427741 seconds

The final coefficients generally converge to [2.44634,-2.81623]. When eta is too large (0.0407) or too small 10^{-4}, fit-GD does not converge and would go through 10000 iterations. Only when eta is 0.01 do the final coefficients converge in a reasonable amount of time in 764 iterations.

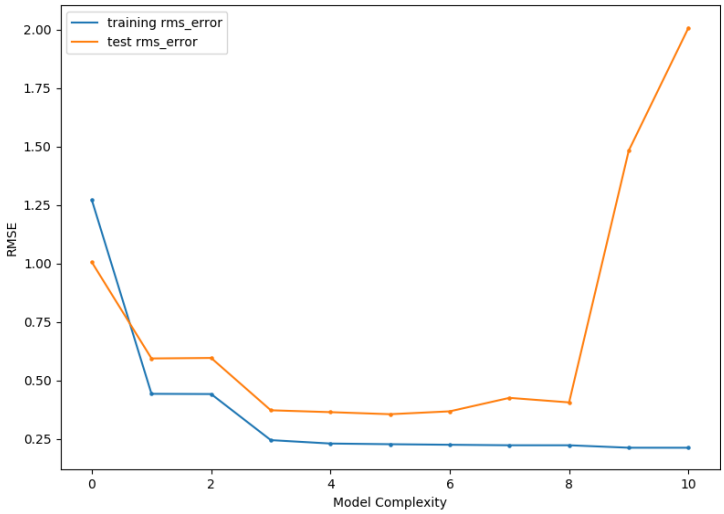
e) The closed form solution takes only 0.004046 seconds to compute the final coefficient, which is a lot faster compared to fit-GD. This makes sense since fit-GD find the final coefficients by looping 10000 times or until value of our objective function J(w) stops changing across iterations. Since computing the closed form solution requires using matrix inversion, which can run quite slow, it comes down to choosing the correct step-size (eta) to help us arrive at the final coefficient efficiently. The final coefficient obtained from the closed form solution is [2.4464,-2.8163] with cost 3.912576, which matches the answer we obtained from using gradient descent with most values of eta.

f) With the variable learning rate, the algorithm takes 1678 iterations and 451.783536 seconds (around 7.5 minutes). The final coefficient is still around [2.44634,-2.81623] with cost 3.91257. When eta is large (at small iterations), the cost for the model grew exponentially big, as shown in the black circle in the image below. As we increase the number of iterations however, the cost quickly dropped from order of e^{28} to 3.912576, which is a huge difference.



g)Updated `PolynomialRegression.generate_polynomial_features(...)` as required.

h) RMSE is preferred because it is an average of the error over N instances. Taking the square root also makes the calculation more robust against large errors since in J(W), the difference between the prediction and actual value is squared.



i)

The best polynomial degree is 5 as both the training and test error obtained are the lowest at this degree. Starting from degree 9 and 10, we can start seeing over-fitting as the training error continues to decrease while the test error blows up. This is a perfect fit to the training data but generalizes poorly to test data.