Homework #1

CSE 446/546: Machine Learning
Prof. Kevin Jamieson
Due: **Wednesday** October 18, 2023 11:59pm
A: 60 points, **B**: 30 points

Short Answer and "True or False" Conceptual questions

A1. The answers to these questions should be answerable without referring to external materials. Briefly justify your answers with a few words.

a. *[2 points]* In your own words, describe what bias and variance are. What is the bias-variance tradeoff?

b. *[2 points]* What **typically** happens to bias and variance when the model complexity increases/decreases?

c. *[2 points]* True or False: Suppose you're given a fixed learning algorithm. If you collect more training data from the same distribution, the variance of your predictor increases.

d. *[2 points]* Suppose that we are given train, validation, and test sets. Which of these sets should be used for hyperparameter tuning? Explain your choice and detail a procedure for hyperparameter tuning.

e. *[1 point]* True or False: The training error of a function on the training set provides an overestimate of the true error of that function.

What to Submit:

- **Parts c, e:** True or False
- **Parts a-e:** Brief (2-3 sentence) explanation justifying your answer.

a. Bias is the different between the model and true feathers of the data. Variance is the flexibility of the model based on the training data. A model with high variance may fit the training data too well but generalize poorly, while a model with high bias may be too simplistic to capture the features within the data. The bias-variance tradeoff is find the right balance between variance and bias to optimize our model that minimize errors.

b. As the model complexity increases, the bias would decrease while variance increase.

c. False. As more data collected, the variance becomes less sensitive to individual data point, it tends to be more stable and the prediction will become more stable. This leads to a decreasing of variance.

d. The validation set should be used in different hyperparameters to decide which one should be used to maximize the performance of a model. And the test set will remain untouched during this process.

e. False. The training error of a function on the training set is the error that a model evaluated on the same training data. It tends to work well on the training data since it learns from it. But it may not generalize well on unseen data like the test set.

Maximum Likelihood Estimation (MLE)

A2. You're the Reign FC manager, and the team is five games into its 2021 season. The numbers of goals scored by the team in each game so far are given below:

$$[2, 4, 6, 0, 1].$$

Let's call these scores $x_1, \ldots, x_5$. Based on your (assumed iid) data, you'd like to build a model to understand how many goals the Reign are likely to score in their next game. You decide to model the number of goals scored per game using a *Poisson distribution*. Recall that the Poisson distribution with parameter λ assigns every non-negative integer $x = 0, 1, 2, \ldots$ a probability given by

$$\text{Poi}(x|\lambda) = e^{-\lambda}\frac{\lambda^x}{x!}.$$

a. *[5 points]* Derive an expression for the maximum-likelihood estimate of the parameter λ governing the Poisson distribution in terms of goal counts for the first n games: $x_1, \ldots, x_n$. (Hint: remember that the log of the likelihood has the same maximizer as the likelihood function itself.)

b. *[2 points]* Give a numerical estimate of λ after the first five games. Given this λ, what is the probability that the Reign score exactly 6 goals in their next game?

c. *[2 points]* Suppose the Reign score 8 goals in their 6th game. Give an updated numerical estimate of λ after six games and compute the probability that the Reign score exactly 6 goals in their 7th game.

What to Submit:

- **Part a:** An expression for the MLE of λ after n games and relevant derivation
- **Parts b-c:** A numerical estimate for λ and the probability that the Reign score 6 next game.

a. Observe that $x_1, \ldots, x_5$ are drawn IID from $\text{Poi}(x|\lambda) = e^{-\lambda}\frac{\lambda^x}{x!}$. The likelihood function $L_n(\theta) = \prod_{i=1}^n (e^{-\lambda}\frac{\lambda^x}{x!})$. The log-likelihood function $l_n(\lambda) = \log(L_n(\theta)) = \sum_{i=1}^n \log(e^{-\lambda}\frac{\lambda^x}{x!})$.
The MLE

$$\begin{aligned}\widehat{\lambda}_{MLE} &= \underset{\lambda}{\text{argmax}}\, l_n(\lambda) \\ &= \underset{\lambda}{\text{argmax}} \sum_{i=1}^n \log(e^{-\lambda}\frac{\lambda^{x_i}}{x_i!})\end{aligned}$$

$$\begin{aligned}\frac{\partial}{\partial\lambda}\sum_{i=1}^n \log(e^{-\lambda}\frac{\lambda^x}{x_i!}) &= \sum_{i=1}^n (\frac{x_i}{\lambda} - 1) \\ &= \frac{\sum_{i=0}^n x_i}{\lambda} - n\end{aligned}$$

Set derivative to zero, we have:

$$\lambda = \frac{\sum_{i=0}^n x_i}{n}$$

b. After first 5 games, $\lambda = (2+4+6+0+1)/5 = 2.6$. Then $\text{Poi}(x_6|\lambda) = e^{-2.6}\frac{2.6^6}{6!} \approx 0.0319$.

c. If $x_6 = 8$, $\lambda = (2+4+6+0+1+8)/6 = 3.5$. $\text{Poi}(x_7|\lambda) = e^{-3.5}\frac{3.5^6}{6!} \approx 0.0771$

b

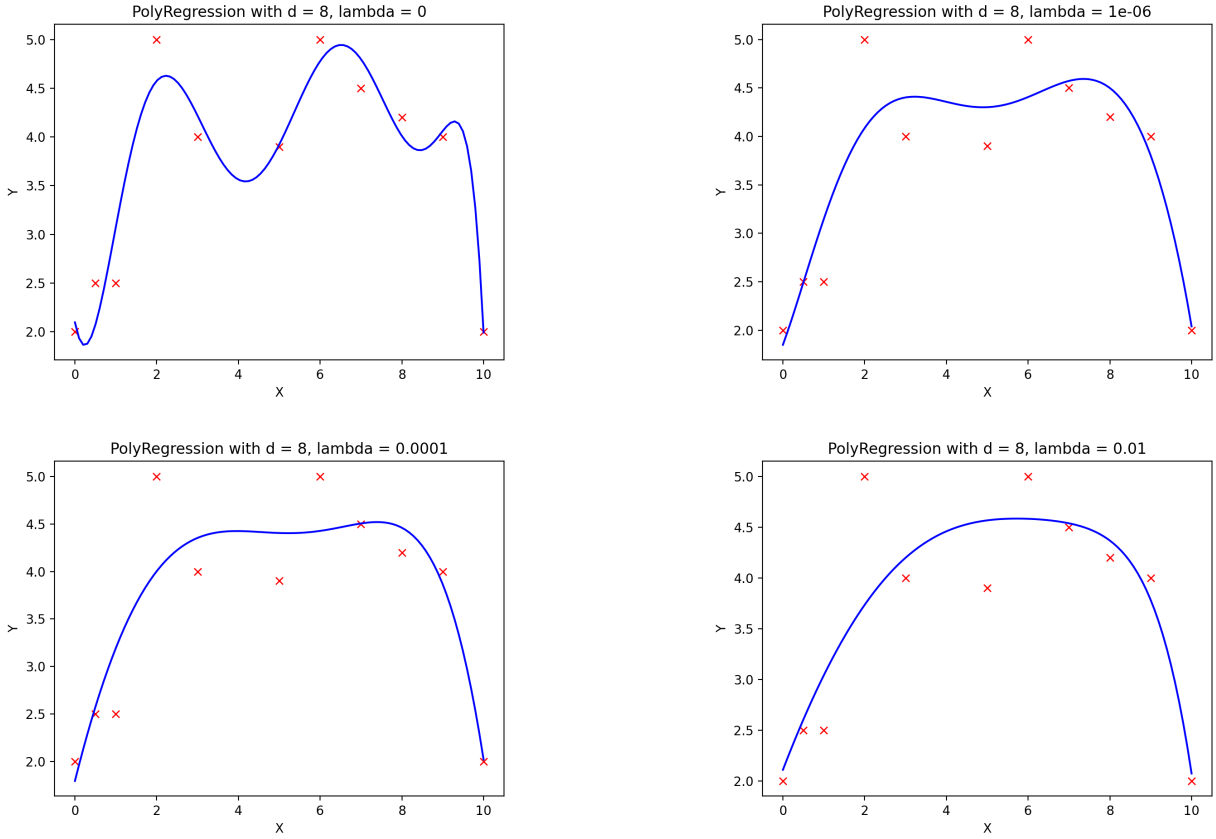

Figure 1: Model with different $\lambda : 0, 10^{-6}, 10^{-4}, 10^{-2}$

.

Polynomial Regression

Relevant Files[1]:

- **polyreg.py**
- `linreg_closedform.py`
- `plot_polyreg_univariate.py`
- `plot_polyreg_learningCurve.py`

In the context of polynomial regression, increasing lambda will lead to a smoother prediction curve. The higher the lambda, the more the model will tend to produce a simple, low-degree polynomial rather than a complex, high-degree one. See figure 1 .

[1] **Bold text** indicates files or functions that you will need to complete; you should not need to modify any of the other files.

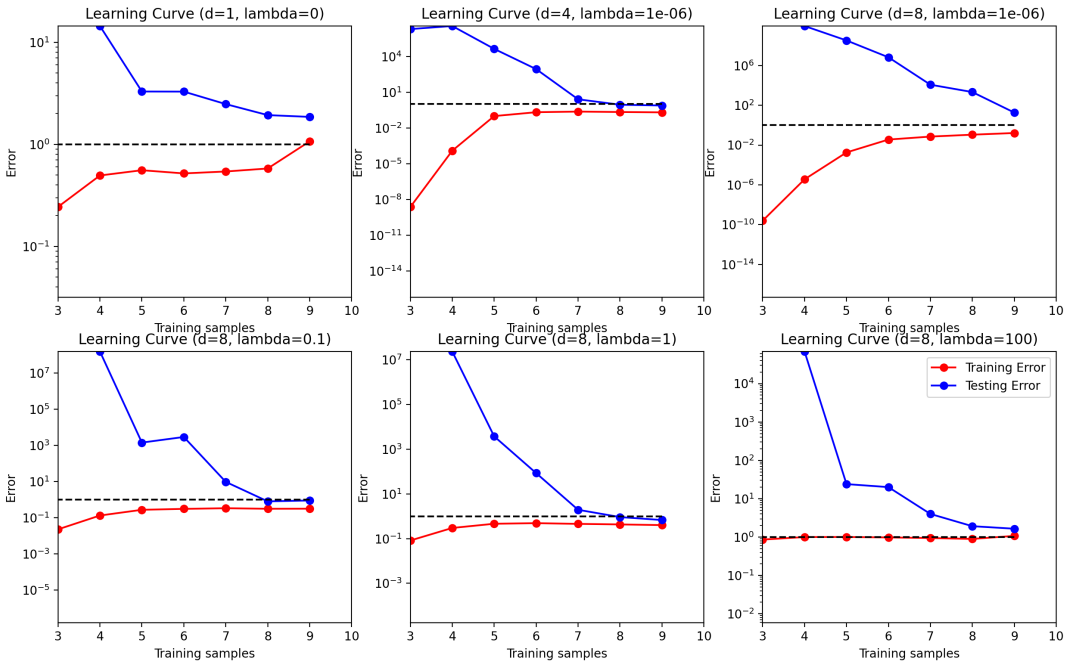

Figure 2: Learning curves for various values of d and λ. The blue lines represent the testing error, while the red lines the training error.

Ridge Regression on MNIST

Relevant Files (you should not need to modify any of the other files for this part):

- **ridge_regression.py**

$$
\begin{aligned}
\widehat{W} &= \text{argmin}_{W\in\mathbb{R}^{d\times k}} \sum_{j=1}^{k} \left[\|Xw_j - Ye_j\|^2 + \lambda\|w_j\|^2\right] \\
&= \text{argmin}_{W\in\mathbb{R}^{d\times k}} \sum_{j=1}^{k} \left[(Xw_j - Ye_j)^T(Xw_j - Ye_j) + \lambda\|w_j\|^2\right] \\
&= \text{argmin}_{W\in\mathbb{R}^{d\times k}} \sum_{j=1}^{k} \left[(Xw_j - Ye_j)^T(Xw_j - Ye_j) + \lambda w_j^T w_j\right] \\
&= \text{argmin}_{W\in\mathbb{R}^{d\times k}} \sum_{j=1}^{k} \left[w_j^T X^T X w_j + e_j^T Y^T Y e_j - e_j^T Y^T X w_j - w_j^T X^T Y e_j + w_j^T \lambda I w_j\right] \\
&= \text{argmin}_{W\in\mathbb{R}^{d\times k}} \sum_{j=1}^{k} \left[w_j^T (X^T X + \lambda I) w_j + e_j^T Y^T Y e_j - e_j^T Y^T X w_j - w_j^T X^T Y e_j\right]
\end{aligned}
$$

$$
\begin{aligned}
&\frac{\partial}{\partial w_j} \sum_{j=1}^{k} \left[w_j^T (X^T X + \lambda I) w_j + e_j^T Y^T Y e_j - e_j^T Y^T X w_j - w_j^T X^T Y e_j\right] \\
=&2(X^T X + \lambda I) w_j + 0 - (e_j^T Y^T X)^T - X^T Y e_j \qquad\qquad j \in \{1, 2, ..., 10\} \\
=&2(X^T X + \lambda I) w_j - 2X^T Y e_j
\end{aligned}
$$

Let the derivative be zero, we have

$$
\begin{aligned}
2(X^T X + \lambda I) w_j - 2X^T Y e_j &= 0 \\
(X^T X + \lambda I) w_j &= X^T Y e_j \\
w_j &= (X^T X + \lambda I)^{-1} X^T Y e_j
\end{aligned}
$$

Now, we have jth column of $\widehat{W}$ equals jth column of $(X^T X + \lambda I)^{-1} X^T Y$. So,

$$
\widehat{W} = (X^T X + \lambda I)^{-1} X^T Y
$$

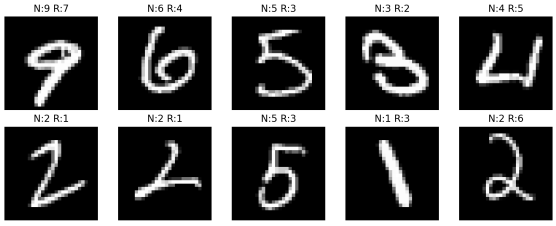

Figure 3: Test images that incorrectly predicted

Ridge Regression Problem
Train Error: 14.805%
Test Error: 14.66%
From figure 3, most wrong prediction due to bad hand writing, make these numbers looks similar to others.

Administrative

A3.

a. *[2 points]* About how many hours did you spend on this homework? There is no right or wrong answer :)

≈ 25HOURS