Learning Reflection Week 1

● Graded

Student

Piyush Acharya

Total Points

2 / 2 pts

Question 1

Learning Reflection

2 / 2 pts

✔ **– 0 pts** Correct

– 1 pt Missing 2-4 sentence summary

– 1 pt Missing >= 5 concepts/terms/topics with definitions

– 1 pt Missing >= 1 uncertainty/question

– 0 pts Please tag all pages associated with your submission.

Question assigned to the following page: 1

Uncertainties

- Instead of introducing so much complexity with regularization, why can't humans just manually check the graph to see if it overfits too much?
- I still don't understand why the validation data cannot just be used as the final error reading for the model, because we're already calculating the MSE for every model to determine the best one. Since the validation data is unseen data, why can't we just use that as the final measure of the models performance without refitting it to the test data?
- I understand the analogy of gradient descent where a ball rolls down the gradient until it gets to the minimum point. But how does it work algorithmically?
- In gradient descent, why can't we just set the value of alpha to a large number and have a very high learning rate? Isn't a higher learning rate better?
- **How does regularization avoid 2 things being the same category again (bathroom + bathtub discussed in lecture)**

Summary

This week, we learned about assessing the performance of regression models, what the bias-variance tradeoff is, as well as ridge and LASSO regularization. Specifically, assessing performance is about finding the true error of a model, which we find by splitting the dataset into training data, validation data, and testing data. The bias variance tradeoff says that if a model is too complex (too many features, higher degree polynomial, etc), it will have low bias and high variance when seeing new data (overfit) whereas if it is too simple, the model has high bias and low variance (underfit). One way of preventing overfit models is regularization on the weights during training, which consists of ridge (L2; squared), LASSO (L1; absolute value), and Elastic Net (L1 and L2).

Regression Model: Establishes the correlation between continuous variables mathematically

Linear Regression: y hat = w_0 + w_1*x where x is a feature (more features can be added)

Polynomial regression: higher degree x fits the training data better.

How to assess performance of models: minimize the **error/cost**

Minimize **Mean Squared Error**: distance between the predicted line and the datapoints

However, this can lead to the model being overfit on the training data and not be able to accurately predict unseen data in the real-world. This is a problem because the model was originally created to predict new **outcomes** based on the determined relationship between the variables.

True error: the model's MSE on unseen data that it has not been trained on

We can choose the right model with the right complexity for our task by comparing the MSE values for multiple models (basic and advanced) and choosing the lowest one

Validation error: The MSE of the model on the validation set

Test error: How well the model performs on the test set

Not always

What characterizes overfitting?

(Low / **High**) Train Error, (Low / **High**) Test Error

(Low / **High**) Bias, (Low / **High**) Variance

In which scenario is it more likely for a model to overfit?

(Few / **Many**) Features

(Few / **Many**) Parameters

(**Small** / Large) Polynomial Degree

(**Small** / Large) Dataset

Cross validation: split the training set into k folds and train the model on k-1 folds, use the other set for validation, and repeat k times. Finally, train the model one more time on the test set (so it ran **k+1**) times in total.

80% **training set**, 10% **validation set**, 10% **testing set**

This exists because we need some unseen data to evaluate the final performance of the best model we have chosen

Source: Tanmay Shah / CSE 416

Bias-Variance Tradeoff: Simple models have high bias and low variance while complex models have low bias and high variance

Bias: Difference between the predicted value and the actual value in the train set.

A model that is too simple will have a high bias because it cannot fit to the data fully. This means that it does not capture the trend in the data correctly due to its failure to fit the **signal**

This is called **underfitting** the data

This causes it to not be able to **generalize** the trend on future data

Variance: How much a the model's prediction changes by if it is fit to a different portion of the dataset that it has not seen before

It occurs when a model memorizes the training set so well that it is finely tuned to the **noise** in the training data.

High variance is caused by a model that is too complex. This may be due to there being too many **features**, or a polynomial regression model with a large degree

This is called **overfitting** the training data

Regularization: Since large coefficients and too many features cause overfitting, we can use a **LASSO** or **Ridge regularization** model to reduce the values of the coefficients and identify features that aren't as meaningful

$$\hat{w}_{ElasticNet} = \underset{w}{\operatorname{argmin}}\, MSE(w) + \lambda_1 ||w||_1 + \lambda_2 ||w||_2^2$$

Lambda is a hyperparameter

Ridge regression: lambda * the sum of the squared weights are added to the MSE, which is then passed through the **argmin** function, which uses gradient descent to set the weights such that the output is as small as possible

Use Ridge when you have many features and believe that everything contributes to the outcome because it doesn't bring the weights down to zero, it only reduces their value

Elastic net: L1+ L2

LASSO regression: Ridge regression but with lambda * the sum of the absolute values of the weights

Basically we are trying to make the negative weights positive somehow: abs value and squaring are one way

Use LASSO to perform **feature selection**

Favors **sparse** solutions (0s)

Normalization function: make everything in the same scale. Scale everything in the dataset (eg changing sqft to sqmi would increase the weight for that term to make it even with everything else)

$$\bar{h}_j(x_i) = \frac{h_j(x_i) - \mu_j(x_1, \ldots, x_N)}{\sigma_j(x_1, \ldots, x_N)}$$

Decrease the variance by training the model on more data