CS155 Set 1

Timothy Liu

January 11, 2018

1 Problem 1

1.1 Part A

A hypothesis set is the collection of possible candidates functions representing a model.

1.2 Part B

The hypothesis set of a linear model is a weight vector w that has a selected number of dimensions (depending on the number of parameters we are fitting to) and a scalar b.

1.3 Part C

Overfitting is when the model fits too closely to the training set. Overfitting occurs when the in sample error is decreasing but the out of sample error increases.

1.4 Part D

One way of preventing overfitting is to increase the size of the training set. The second way is to use a simpler model with fewer parameters.

1.5 Part E

The training data is the data that you look at to construct you learning model. The model is "trained" using the training data. The test data is a separate set of data that the learning algorithm is not trained on. Instead, the learning model, after being trained on the training data, is then applied on the test data to evaluate the model's performance. If you change the model based on the test set, then you are using the test set as the training set. Doing this will compromise the test set's ability to predict how well your model performs on out of sample data.

1.6 Part F

We assume that the dataset is sampled independently and identically. This means each dataset has the same distribution as other datasets and that they are independent of each other.

1.7 Part G

The input space X is a set of emails and the output space Y are labels that indicate if each email is either spam or not spam.

1.8 Part H

k-fold validation is a technique for estimating the out of sample error of an algorithm. In this technique, the original training data is broken into multiple partitions. One partition is set aside as the validation set, and the algorithm is trained on the other partitions. The trained algorithm is then evaluated on the untouched validation set. The process is repeated, and each partition is used as the validation set.

2 Problem 2

2.1 Problem A

$$\mathbb{E}_s[E_{out}(f_s)]$$

Substitution

$$\mathbb{E}_s[\mathbb{E}_x[(f_s(x) - y(x))^2]]$$

Swap order of expected value

$$\mathbb{E}_x[\mathbb{E}_s[(f_s(x) - y(x))^2]]$$

Add zero

$$\mathbb{E}_x[\mathbb{E}_s[(f_s(x) - F(x) + F(x) - y(x))^2]]$$

Expand

$$\mathbb{E}_x[\mathbb{E}_s[(f_s(x) - F(x))^2 + (F(x) - y(x))^2 + 2(f_s(x) - F(x))(F(x) - y(x))]]$$

Expected value of cross term is zero

$$\mathbb{E}_x[\mathbb{E}_s[(f_s(x) - F(x))^2 + (F(x) - y(x))^2]]$$

Linearity

$$\mathbb{E}_x[\mathbb{E}_s[(f_s(x) - F(x))^2] + \mathbb{E}_s[(F(x) - y(x))^2]]$$

$$\mathbb{E}_x[Bias(x) + Var(x)]]$$

2.2 Problem B

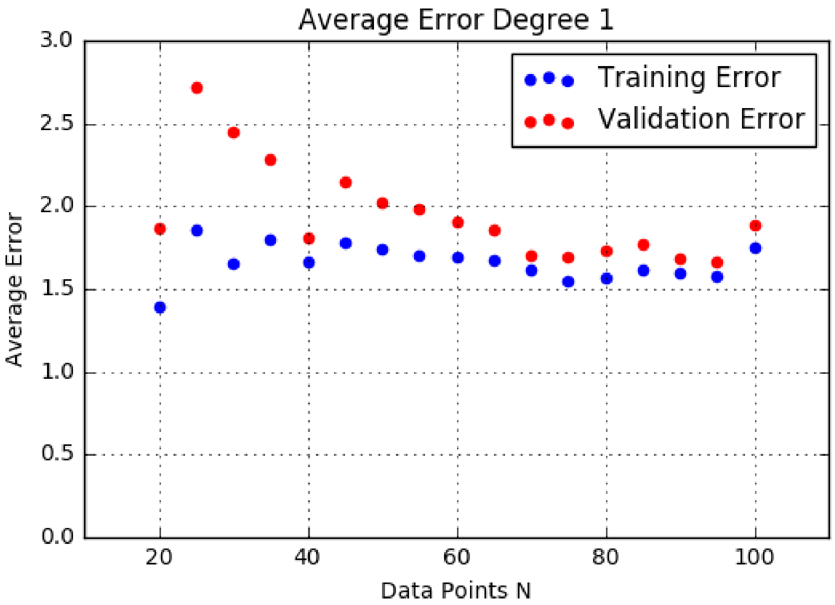


Figure 1: Training and validation error for polynomial model degree 1.

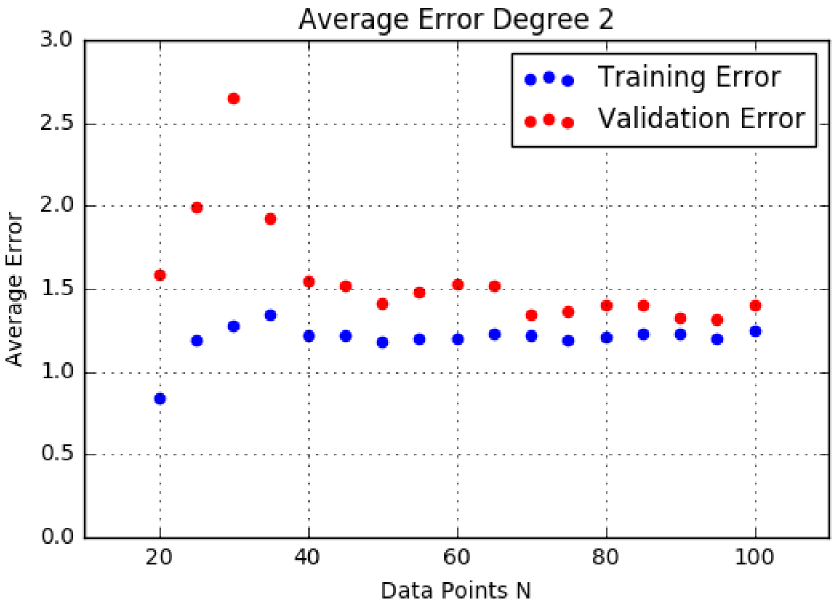


Figure 2: Training and validation error for polynomial model degree 2.

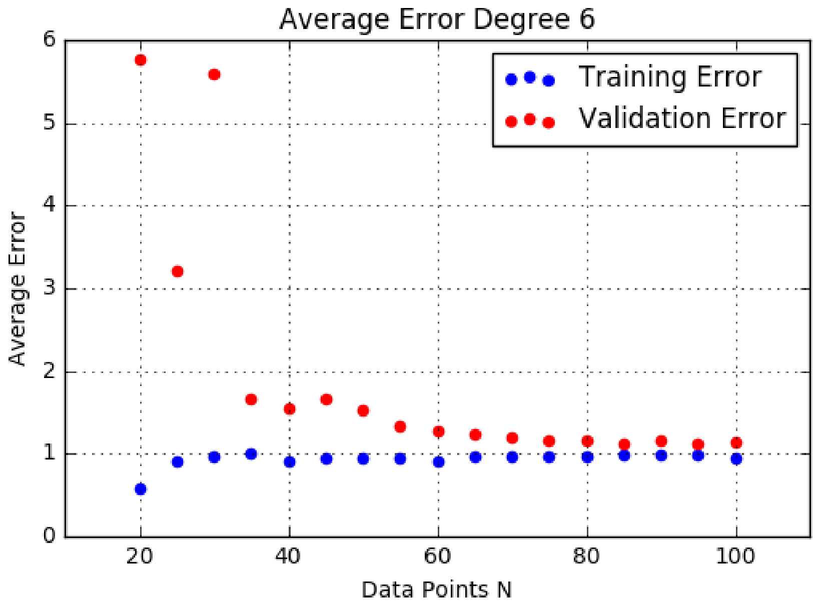


Figure 3: Training and validation error for polynomial model degree 6.

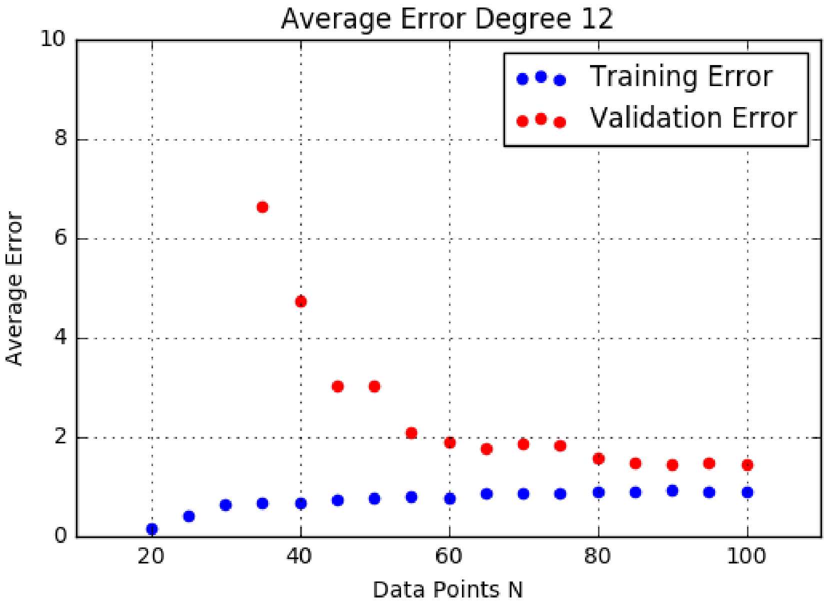


Figure 4: Training and validation error for polynomial model degree 12. Some validation errors are not shown because their value greatly exceeds that of the other points.

2.3 Problem C

The 1-degree model has the highest bias. This model has an average training error of about 1.6 to 1.7 regardless of the size of the training set, which is the highest of the four models. Models with lower complexity also generally have more bias.

2.4 Problem D

The 12-degree model has the largest variance. The validation error for this model is the highest, and for a small training set the validation error is many orders of magnitude greater than that of the other models.

2.5 Problem E

The quadratic model may improve slightly but not greatly if we included more training points. After roughly 70 points the average validation error is fairly stable, suggesting that adding more points won't necessarily lead to improved performance.

2.6 Problem F

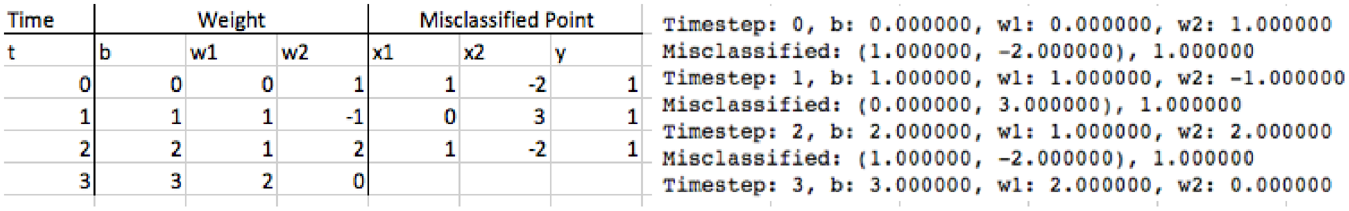
The training error is generally lower because the models are optimized to minimize the error of the points in the training set. Since the points from the validation set are not used to help create the model, the validation error is generally greater.

2.7 Problem G

The 6 degree model trained on the most number of points possible will perform the best on out-of-sample data. This model has the lowest validation error - slightly greater than 1 - and the validation error is an approximate indicator of how well the model will perform on unseen data.

3 Problem 3

3.1 Problem A



Time	Weight			Misclassified Point		
t	b	w1	w2	x1	x2	y
0	0	0	1	1	-2	1
1	1	1	-1	0	3	1
2	2	1	2	1	-2	1
3	3	2	0			

```
Timestep: 0, b: 0.000000, w1: 0.000000, w2: 1.000000
Misclassified: (1.000000, -2.000000), 1.000000
Timestep: 1, b: 1.000000, w1: 1.000000, w2: -1.000000
Misclassified: (0.000000, 3.000000), 1.000000
Timestep: 2, b: 2.000000, w1: 1.000000, w2: 2.000000
Misclassified: (1.000000, -2.000000), 1.000000
Timestep: 3, b: 3.000000, w1: 2.000000, w2: 0.000000
```

Figure 5: Weights of perceptron at each timestep and misclassified point.

3.2 Problem B

In 2D the smallest non-linearly-separable data set has 4 points. If there are 3 points, then if two points are classified the same then a line can be drawn between the third point and the two similarly classified point. If there are 4 points, 3 points can be classified +1 and arranged in a triangle. If the fourth point is labeled -1 and placed at the center of the triangle, the data is not linearly separable.

Similarly, for a 3D dataset the smallest non-linearly-separable data set has 5 points. 4 points can be arranged in a tetrahedron and classified +1, and the fifth point is in the middle of the tetrahedron and classified -1. Any dataset with 4 points can be linearly separated in 3D, since a plane can be drawn between 3 points similarly classified and then shifted slightly towards the fourth, differently classified point.

For an N-dimensional set, the smallest non-linearly separable set has N+2 points.

3.3 Problem C

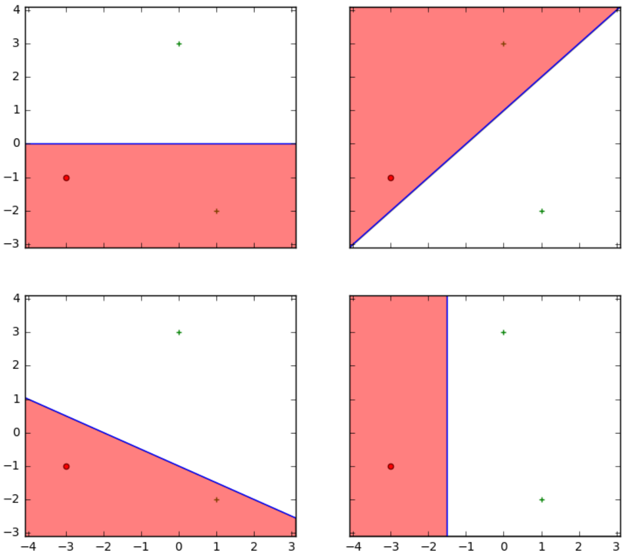


Figure 6: Visualization of data set from part A.

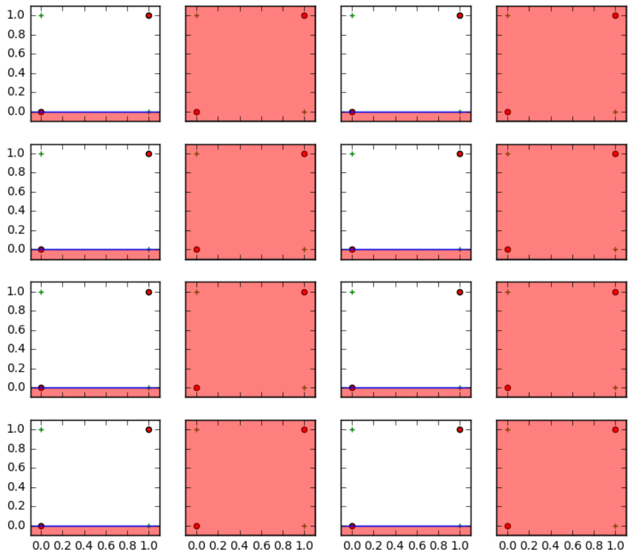


Figure 7: Visualization of non-separable data set.

If the dataset is not linearly separable, then the PLA will never converge. As demonstrated in the graph above, the PLA will oscillate between two solutions, each of which misclassifies a different point. Updating the weights to correctly classify one solution will lead to the other solution being misclassified.

4 Problem 4

4.1 Problem A

We define a zeroth coordinate for the weight vector $\mathbf{w}$ and for each training point $\mathbf{x}$. Every former $\mathbf{x} = (x_1, x_2, x_3.....)$ is instead $(1, x_1, x_2, x_3.....)$ and the weight vector now includes a w_0 term: $(w_0, w_1, w_2.....)$. The w_0 is the bias term.

4.2 Problem B

We want the derivative of the squared loss function for linear regression with respect to the vector $\mathbf{w}$.

$$\partial_w \sum_{i=1}^{N} (y_i - \mathbf{w}^T \mathbf{x}_i)^2$$

$$\sum_{i=1}^{N} \partial_w (y_i - \mathbf{w}^T \mathbf{x}_i)^2$$

$$\sum_{i=1}^{N} 2 * (y_i - \mathbf{w}^T \mathbf{x}_i) \partial_w (y_i - \mathbf{w}^T \mathbf{x}_i)$$

$$\sum_{i=1}^{N} -2 * (y_i - \mathbf{w}^T \mathbf{x}_i) \mathbf{x}_i$$

4.3 Problem C

See src.

4.4 Problem D

As the starting point is changed, the weights take a different path to convergence but they reach the same minimum. The starting point has some influence on how quickly the weights converge, though it seems like they all converge about as quickly. Dataset 2 has certain points that converge more quickly than others. The points further from the minimum initially descend faster.

4.5 Problem E

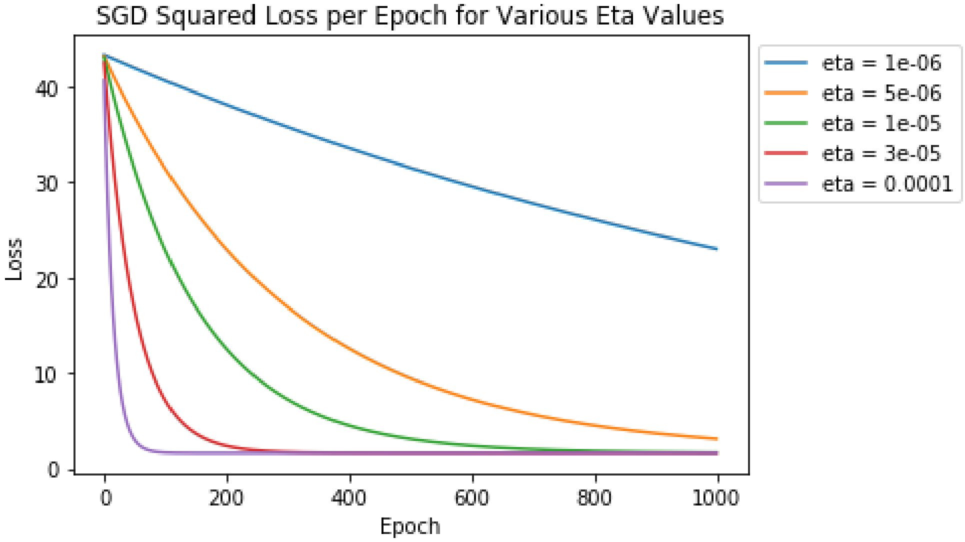


Figure 8: Loss function across epochs for different etas.

The squared loss falls much more rapidly for a larger eta. A larger eta leads to a faster convergence to a solution.

4.6 Problem F

The final weight vector is **w** = (-5.94224311, 3.94373944, -11.72398258, 8.78553623) and the bias term is -0.227205913368.

4.7 Problem G

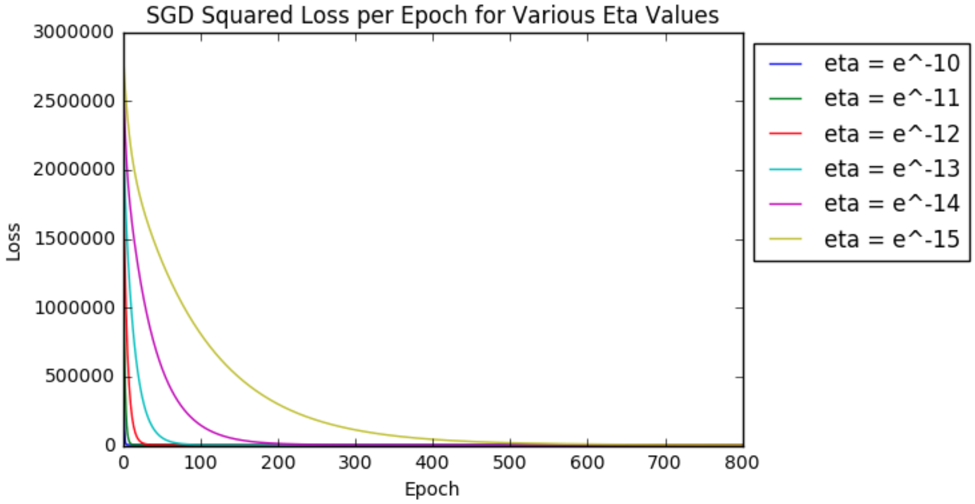


Figure 9: Loss function across epochs for different etas.

The value of the loss function falls much more rapidly for larger etas.

4.8 Problem H

The final weight vector from the analytical method is **w** = (-5.99157048, 4.0150995, -11.93325972, 8.99061096) and the bias term is -0.316442513271. These values are similar to but not exactly the same as the weights from stochastic gradient descent.

4.9 Problem I

If you have a dataset with a large number of dimensions, computing the solution analytically may be more computationally expensive. Taking the inverse of a large, multidimensional matrix and summing the product of $\mathbf{x}$ and $\mathbf{x}^T$

4.10 Problem J

One technique could be to keep track of how much the loss function changes from one epoch to the next. When the loss function only falls by a small amount (below some threshold value) then the algorithm can be stopped.

4.11 Problem K

The weight vector for the SGD produces monotonically better solutions as more iterations happen. In contrast, the perceptron can have intermediate weight vectors that have a higher loss than previous weight vectors.