Introduction to Machine Learning

Homework 04

Atharva Patil- B01014288

LFD Exercise 4.3
Deterministic noise depends on 1-l, as some models approximate f better than others.

(a) Assume 1-l is fixed and we increase the complexity of f. Will deterministic noise in general go up or down? Is there a higher or lower tendency to overfit?

(b) Assume f is fixed and we decrease the complexity of 1-l. Will deterministic noise in general go up or down? Is there a higher or lower tendency to overfit?

(a) When considering a given hypothesis set H:
- If H's best approximation is simpler than the original target function, then increasing the complexity of the actual function f generally leads to an increase in deterministic noise, making it more challenging for H to fit the target function, thereby raising the risk of overfitting.
- Conversely, if H's best approximation is more complex than the target function, an initial increase in f's complexity may initially reduce deterministic noise and the likelihood of overfitting. However, if the increase continues past H's closest function approximation, the deterministic noise will rise again, enhancing the propensity for overfitting.

(b) With a constant f :
- Should H's best approximation be less intricate than the target function, a reduction in H's complexity leads to a higher deterministic noise, which can increase overfitting potential.
- If H's best approximation is more intricate than the target function, reducing H's complexity initially lowers deterministic noise and overfitting likelihood. Nevertheless, if we keep on reducing H's complexity beyond the point where it matches f, we begin to boost deterministic noise again, which, in turn, increases the risk of overfitting.

LFD Exercise 4.6

We have seen both the hard-order constraint and the soft-order constraint. Which do you expect to be more useful for binary classification using the perceptron model?

The VC dimension of the perceptron reduces when the hard-order restriction is applied, and it is less likely to categorize the same number of points with more parameters. Even in cases when 'w' is tiny, the signs of $w^T x$ remain unchanged if the soft-order constraint is applied. As a result, we can still categorize the points.

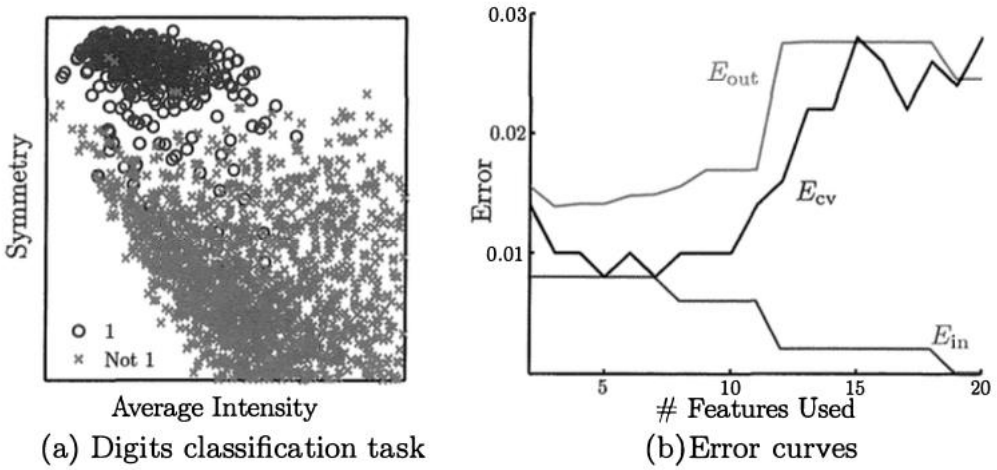
Exercise 4.8

Is E_m an unbiased estimate for the out of sample error $E_{\text{out}}(g_m^-)$?

From the validation set, H_m, $\overline{g_m}$. is individually learned for each model. The expectation varies only on x_n, when we take it in relation to the validation data set. The $\overline{g_m}$ remains unchanged. Therefore, I believe that E_m, is a fair estimate of the out-of-sample error $E_{out}(\overline{g_m})$.

LFD Exercise 4.11

In this experiment, the black curve E_{cv} is sometimes below and sometimes above the red curve E_{out}. If we repeated this experiment many times, and plotted the average black and red curves, would you expect the black curve to lie above or below the red curve?



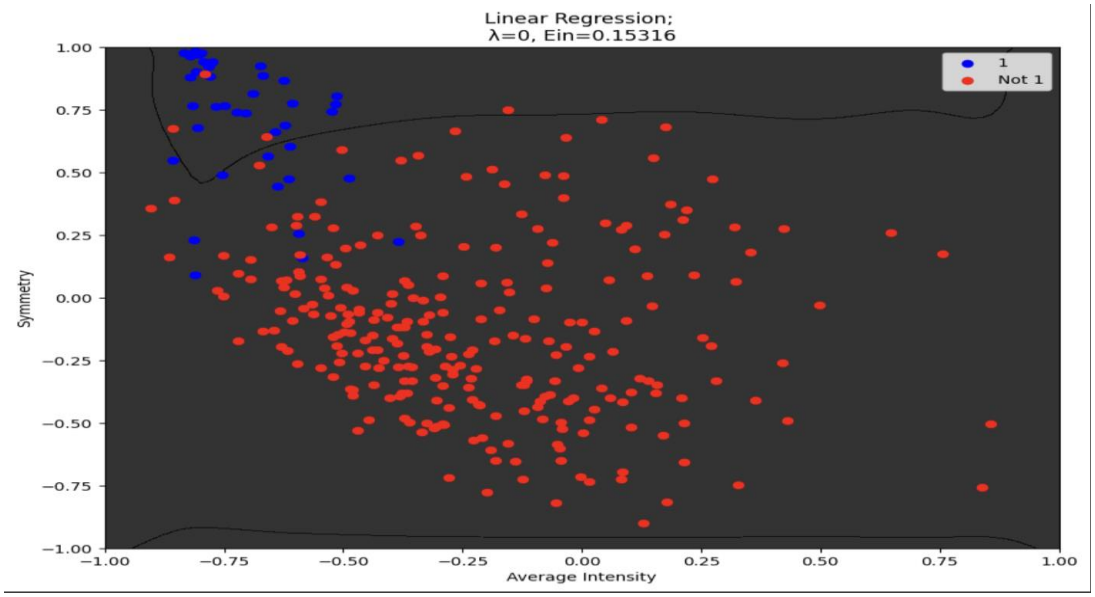
When repeating the experiment multiple times and averaging the black and red curves, we would generally expect the average black curve E_{cv} to be below the average red curve E_{out}, indicating that even after multiple iterations, cross-validation tends to provide a lower error estimate compared to the actual out-of-sample error.

Q.5: MNIST

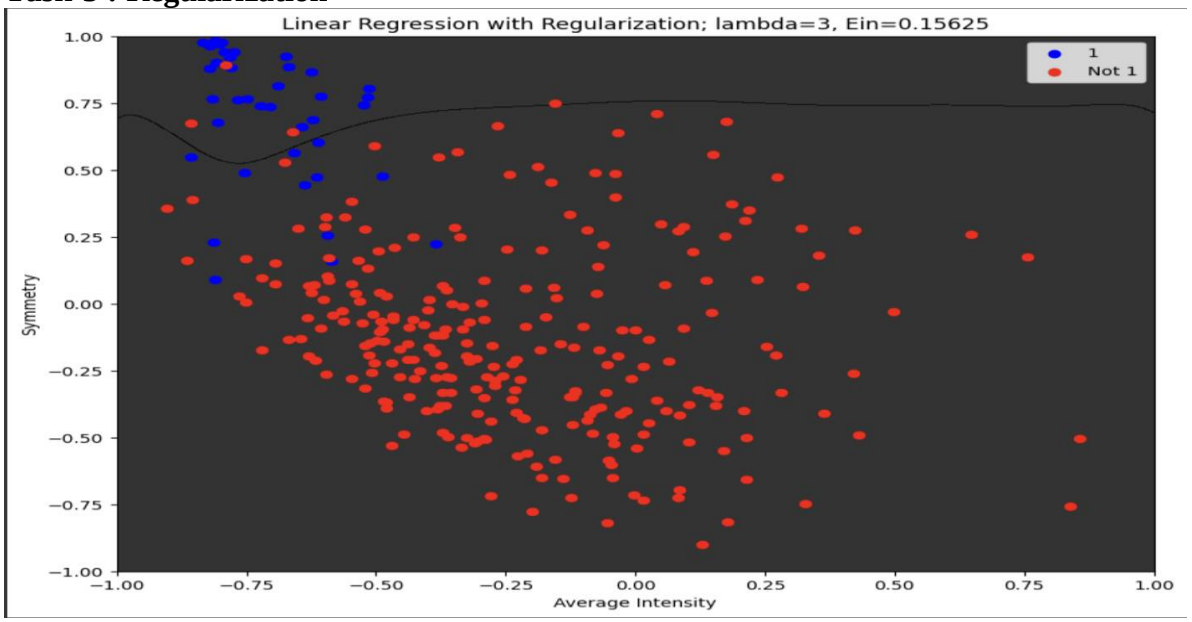
Task-1 : 10th Order Polynomial Transform

```
Z shape (300, 31)
Ztest shape (8998, 31)
```

Task-2 : Overfitting

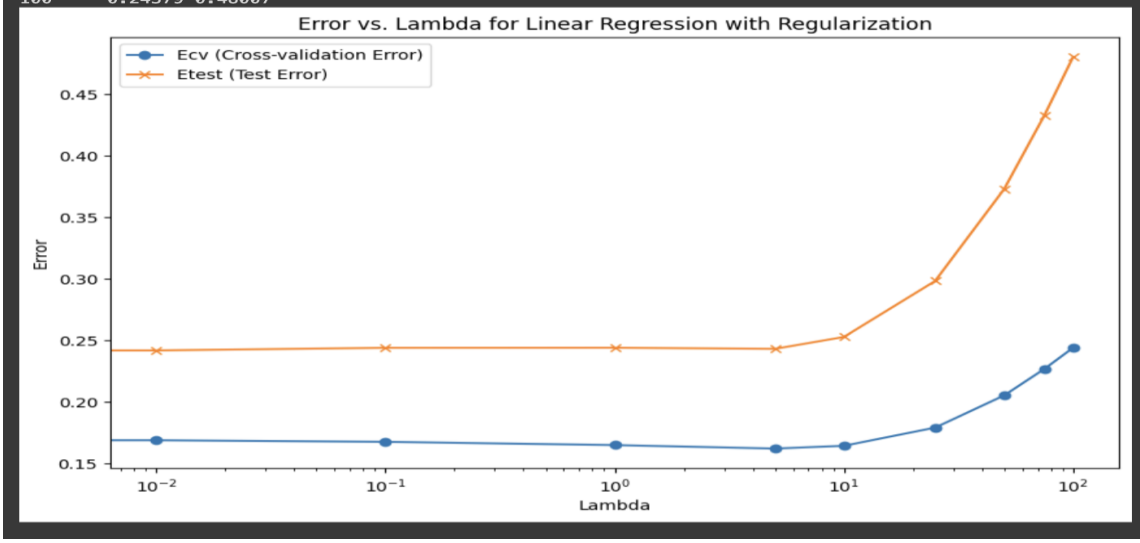


Task-3 : Regularization

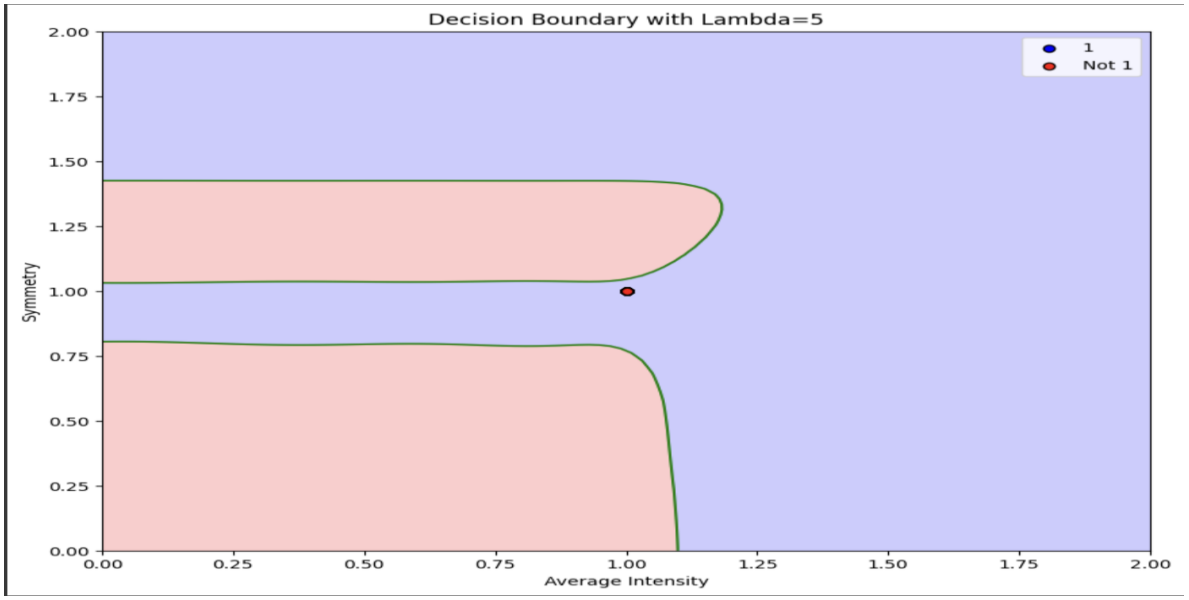


Task-4 : Cross Validation

```
Lambda  Ecv             Etest
0       0.17550 0.25037
0.01    0.16863 0.24159
0.1     0.16735 0.24372
1       0.16470 0.24379
5       0.16189 0.24287
10      0.16411 0.25254
25      0.17887 0.29799
50      0.20502 0.37264
75      0.22659 0.43271
100     0.24379 0.48007
```



Task-5 : Pick λ



Task-6 : Estimate Classification Error

```
Classification error on test set (Eout(g)): 0.08333333333333333
99% confidence interval for classification error: [0, 0.17524201801951844]
```

Task-7 : Is E_{cv} Biased?

While not totally unbiased, $E_{cv}(\lambda^*)$ is typically a good estimator for out-of-sample error. Choosing λ^* to reduce E_{cv} may cause the error estimate to be a little bit optimistic because:

Models for cross-validation are trained using datasets that are marginally smaller than the final model.
If λ values are tested frequently or the dataset is limited, choosing λ^* based on E_{cv} may cause an overfit to the validation set.

In spite of these reservations, E_{cv} is nevertheless a valuable approximation in situations involving huge datasets and rather thorough hyperparameter adjustment.

Task-8 : Data Snooping

E_{out} is usually estimated using $E_{test}(\text{wlin}(\lambda^*))$, yet data snooping—the practice of choosing a model or hyperparameter based only on how well it performs on the test set—may cause it to be less than entirely objective. This is especially true if the test set has had any bearing on the selection of λ^*, e.g., if the test set is utilized indirectly in model selection or for validation.

This explains why $E_{test}(\text{wlin}(\lambda^*))$ could be skewed:
If the test data was used, even if only partially, in determining λ^*, it indicates that the model is tuned to perform well on this particular set and that the estimate of E_{out} may be excessively optimistic.

Because the model's complexity was chosen to minimize the test set error, the error estimate may be less than the genuine out-of-sample error.

In order to obtain an unbiased estimator of E_{out} and prevent data snooping, one should make sure:
The training or validation procedures never make use of the test set.
Model selection and hyperparameter adjustment should be done using a different validation set or cross-validation.
The performance of the finished model should only be evaluated once using the test set.

To correct for data snooping:
If possible, collect a new test set that was not seen during the entire model selection and training process.
Use nested cross-validation, where one level of cross-validation is used for hyperparameter tuning and another level for performance estimation.

Google Colab Link:
https://colab.research.google.com/drive/1299XR6cl_UnKKXenR_jx0QWDteEs1qpa#scrollTo=ppJFs8NWY3Fk

Collaborator: Shreyas Kadam