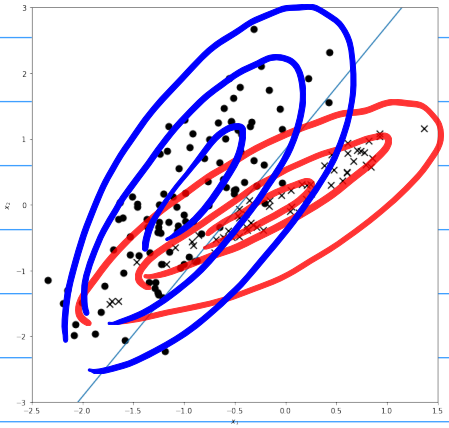
Lecture 3: Evaluating SL Models

In the last lecture we saw a simple example of a linear regression model for SL.

$$\hat{f}(\underline{x}) = \hat{\beta}_0 + \sum_{j=0}^{d-1} \hat{\beta}_j x_j$$

$$\hat{\beta} = \underset{\beta \in \mathbb{R}^d}{\arg\min} \| f(X) - \underline{Y} \|^2$$



You may be wondering why we specifically chose f to be linear in $\underline{x}$, hence leading to the linear decision boundary.

We can easily consider extensions

$$f(\underline{x}) = \beta_0 + \sum_{j=0}^{d-1} \beta_j x_j + \sum_{j=0}^{d-1} \sum_{k=j}^{d-1} \beta_{(d+j)d+(k-j)} x_j x_k$$

or more broadly

$$f(\underline{x}) = \sum_{j=0}^{J-1} \beta_j \psi_j(\underline{x}), \quad \psi_j : \mathcal{X} \longrightarrow \mathbb{R}$$

called feature maps

Such models can still be trained via least squares under some assumptions.

Define

$$A = \begin{bmatrix} \psi_0(\underline{x}_0) & \psi_1(\underline{x}_0) & \cdots & \psi_{J-1}(\underline{x}_0) \\ \psi_0(\underline{x}_1) & \cdots & & \psi_{J-1}(\underline{x}_1) \\ & & \vdots & \\ \psi_0(\underline{x}_{N-1}) & \cdots & & \psi_{J-1}(\underline{x}_{N-1}) \end{bmatrix} \in \mathbb{R}^{N \times J}$$

Then we can find the regression soln (fit) by solving

$$\hat{f} \equiv \hat{\underline{\beta}} = \underset{\underline{\beta} \in \mathbb{R}^J}{\operatorname{argmin}} \frac{1}{2\sigma^2} \| A\underline{\beta} - \underline{Y} \|^2$$

the soln is still given by normal eqn's

$$\hat{\underline{\beta}} = (A^T A)^{-1} A^T \underline{y}.$$

Then so long as $A^T A$ is invertible we can find a unique $\hat{\underline{\beta}}$. But this is not always the case, certainly not in large scale & high dimensional problems.

If $A^T A$ is not invertible ie, columns of A are not linearly independent, we can fix the problem by regularization / penalization / shrinkage
(math) (optimization) (statistics)

that is, we consider

$$\hat{\underline{\beta}} = \underset{\underline{\beta} \in \mathbb{R}^J}{\operatorname{argmin}} \frac{1}{2\sigma^2} \| A\underline{\beta} - \underline{Y} \|^2 + \frac{\lambda}{2} \|\underline{\beta}\|_p^p,$$

$\lambda \geq 0$ is called the regularization/penalty parameter & $p \geq 1$ denotes the choice of the norm $\underline{\beta}$. we will see $p=1$ later in the course & only consider $p=2$ for now. This particular case is called Ridge regression.

$$\hat{\underline{\beta}} = \underset{\underline{\beta}}{\operatorname{argmin}} \frac{1}{2\sigma^2} \| A\underline{\beta} - \underline{Y} \|^2 + \frac{\lambda}{2} \|\underline{\beta}\|^2$$

once again we can solve for $\tilde{\underline{\beta}}$ exactly,

$$\frac{\partial}{\partial \underline{\beta}} \left(\frac{1}{2\sigma^2} (A\underline{\beta} - \underline{Y})^T (A\underline{\beta} - \underline{Y}) + \frac{\lambda}{2} \underline{\beta}^T \underline{\beta} \right)$$

$$= \frac{1}{\sigma^2} A^T (A\underline{\beta} - \underline{Y}) + \lambda \underline{\beta}$$

$$= \left(\frac{1}{\sigma^2} A^T A + \lambda I \right) \underline{\beta} - \frac{1}{\sigma^2} A^T \underline{Y} = 0$$

$$\Rightarrow \underline{\beta} = \underbrace{\left(A^T A + \sigma^2 \lambda I \right)^{-1}}_{\text{always invertible if } \lambda > 0} A^T \underline{Y}$$

SVD of $A = U\Sigma V^T$ then $A^TA = V\Sigma^2V^T$

& also $VV^T = I$ so

$$A^TA + \sigma^2\lambda I = V(\Sigma^2 + \sigma^2\lambda I)V^T$$

all singular vals $\geq \sigma^2\lambda > 0$

ex Back to our example.

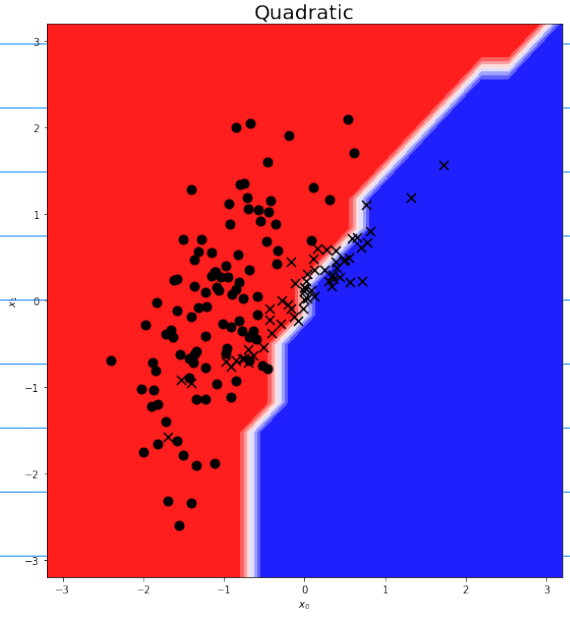
We consider two variations of Ridge regression on our data set from the previous lecture:

Case 1: quadratic feature maps

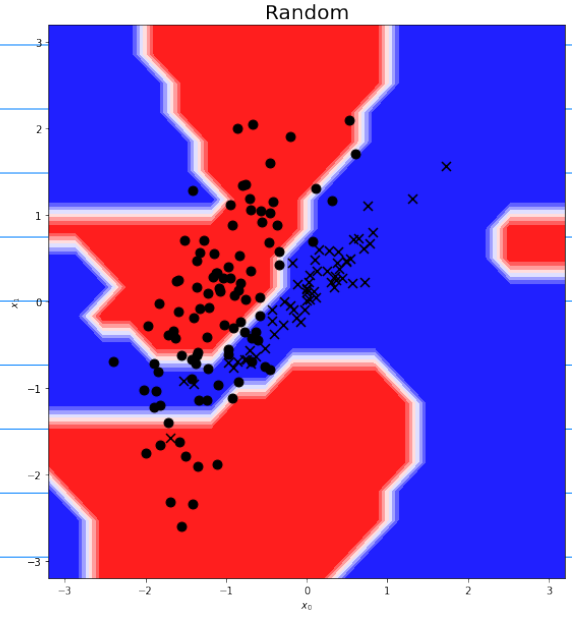
$$f(x) = \beta_0 + \beta_1 x_0 + \beta_2 x_1 + \beta_3 x_0^2 + \beta_4 x_1^2 + \beta_5 x_0 x_1$$

Case 2: Random feature maps

$$f(x) = \sum_{j=1}^{100} \beta_j \psi_j(x), \quad \psi_j = \exp\left(\frac{1}{2(w_j^{(0)})^2}(x_0 - w_j^{(1)})^2 + \frac{1}{2(w_j^{(2)})^2}(x_1 - w_j^{(3)})^2\right)$$



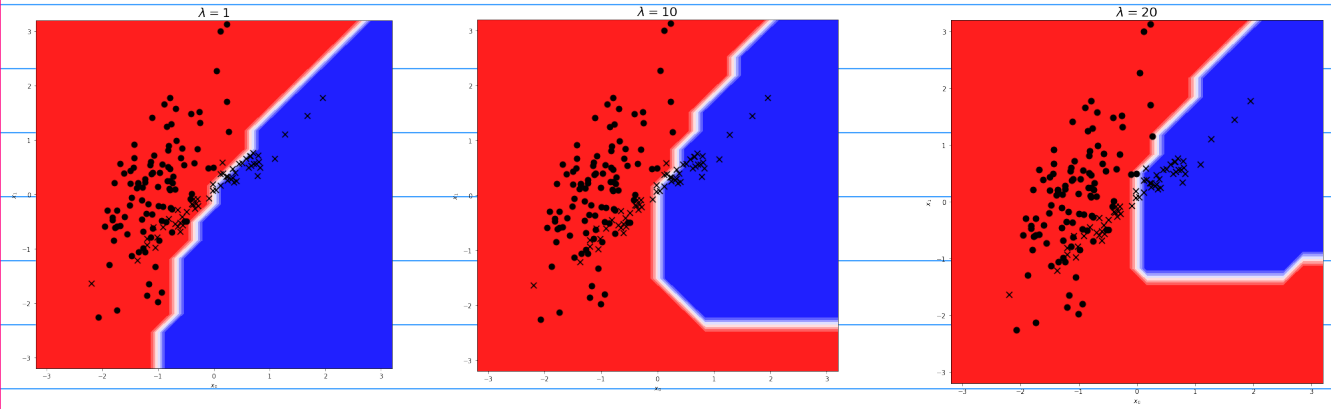
$(\lambda = .1)$



$(\lambda = 0.1)$

the quadratic model is more stable & can tolerate smaller values of λ. This is due to the fact that matrix A here has full column rank.

Regardless, choosing a large λ still ruins the decision boundary.



So, the choice of λ is important

- if $\lambda \to +\infty$ then $\hat{\beta} \to 0$ so model is not good.
- if $\lambda \to 0$ then A^TA may become singular & values of $\hat{\beta}$ are unstable.

* This clearly indicates that λ needs to be tuned. But inorder to do so we first need to assess the performance of a soln $\hat{\beta}$ obtained by a certain choice of λ or features.

3.1 Training & Testing Errors

It is clear now that regardless of the SL model chosen we need to make an assessment on the performance of our model.

The first step is to check the error of our model on the training data set, ie, the $\{X, Y\}$ that we used to estimate the $\hat{\beta}$.

Let $\hat{f}(\underline{x}) = \sum_{j=1}^{J} \hat{\beta}_j \psi_j(\underline{x})$ the trained regression model obtained previously. Then we can consider the training mean squared error (MSE)

$$MSE_{train}(\hat{f}, \underline{Y}) = \frac{1}{N} \sum_{n=0}^{N-1} |\hat{f}(\underline{x}_n) - y_n|^2$$

for $\underline{x}_n \in X$, $y_n \in Y$.

- This is equivalent to $\frac{1}{N} \|\hat{f}(X) - \underline{Y}\|^2$ in our compact notation.
- We call this the Training MSE or simply training error because it is computed over the dataset $\{X, Y\}$ that was also used to train $\hat{f}$.

In many ML applications, in particular benchmark data sets, we have access to an additional data set specifically for testing.

$\{X, Y\}$ - training set, used to find $\hat{f}(\equiv \hat{\beta})$

$\{X', Y'\}$ - test set, used only for evaluating $\hat{f}$.

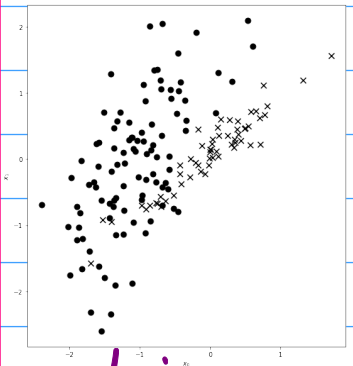
Given the training & test sets we then consider a measure of error such as MSE on both training & test sets

$$MSE_{train} = \frac{1}{N} \sum_{n=0}^{N-1} |\hat{f}(\underline{x}_n) - y_n|^2, \quad \underline{x}_j \in X, y_j \in Y$$

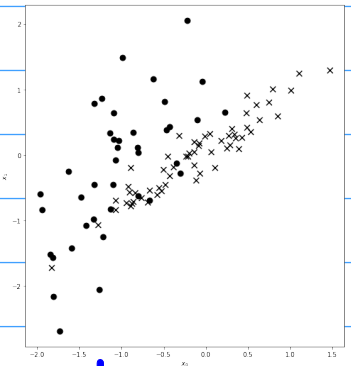
$$MSE_{test} = \frac{1}{N'} \sum_{n=0}^{N'-1} |\hat{f}(\underline{x}_n) - y_n|^2, \quad \underline{x}_n \in X', y_n \in Y'$$

If a dataset does not come with a test set we can simply split our training set into two parts, use one part for training & another part for testing. This is highly recommended!

ex back to regression demo



train



test

```
MSE_quad_training: 0.6998016769091164
MSE_rndf_training: 0.7614587072342567
MSE_quad_test: 1.1242952833690438
MSE_rndf_test: 0.8033131949828711
```

Observe that the train error of the quadratic model is lower but in testing, the random feature model did better.

WARNING: This does not mean random feature model is always better. In fact sometimes it isn't. This example reflects the delicacy of SL.

Your train & test sets are often not the same & this leads to errors.

You will also notice that both train & test errors are highly dependent on the choice of λ!

$\lambda = 1$, $\lambda = 0.1$

```
MSE_quad_training: 0.6998016769091164
MSE_rndf_training: 0.7614587072342567
MSE_quad_test: 1.1242952833690438
MSE_rndf_test: 0.8033131949828711
```

$\lambda = 2$, $\lambda = 0.2$

```
MSE_quad_training: 0.6715629548994052
MSE_rndf_training: 0.8225190877071444
MSE_quad_test: 0.8275766363722283
MSE_rndf_test: 0.9289597665418734
```