Train, Validation sand Test Datasets

Training Dataset

The actual dataset that we use to train the model. The model sees and learns from this data.

Validation Dataset

The validation dataset is used to evaluate a given model, but it is used frequently unlike the test dataset. We fine-tune the model hyperparameters based on the performance on validation dataset. Hence, the model sees this dataset, but never learns from it. The validation dataset affects the model indirectly, and the model becomes more biased as the performance on validation dataset is included in choosing the hyperparameters.

Test Dataset

The test dataset provides the best metric used to evaluate the model. It is only used once a model is completely trained (using the train and validation sets). The test dataset is generally what is used to evaluate competing models. Many a times the validation dataset is used as the test dataset, but it is not good practice, because the model is already biased towards the validation dataset. The test dataset is generally well curated. It contains carefully sampled data that spans the various classes that the model would face, when used in the real world.

Dataset split ratio

The split ratio between training, test and validation dataset should be chosen keeping in mind 2 things, the total number of samples in your data and the actual model you are training. Some models need substantial data to train upon, so in this case you would optimize for the larger training sets. Models with very few hyperparameters will be easy to validate and tune, so you can probably reduce the size of your validation set, but if your model has many hyperparameters, you would want to have a large validation set as well(although you should also consider cross validation). Also, if you happen to have a model with no hyperparameters or ones that cannot be easily tuned, you probably don't need a validation set too! All in all, like many other things in machine learning, the train-test-validation split ratio is also quite specific to your case and it gets easier to make judgement as you train and build more and more models.

Regularization

Regularization helps in avoiding overfitting and also increasing model interpretability. This is a form of regression, that constrains/ regularizes or shrinks the coefficient estimates towards zero. In other words, this technique discourages learning a more complex or flexible model, so as to avoid the risk of overfitting. A simple relation for linear regression looks like Y ≈ β0 + β1X1 + β2X2 + …+ βpXp. The fitting procedure involves minimizing a loss function, known as residual sum of squares or RSS. If there is noise in the training data,

$$\mathrm{RSS} = \sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 .$$

then the estimated coefficients won't generalize well to the future data. This is where regularization comes in and shrinks these learned estimates towards zero.

Ridge Regression

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}\beta_j^2 = \mathrm{RSS} + \lambda\sum_{j=1}^{p}\beta_j^2$$

The RSS is modified by adding the shrinkage quantity. Now, the coefficients are estimated by minimizing this function. Here, λ is the tuning parameter that decides how much we want to penalize the flexibility of our model. Notice that we shrink the estimated association of each variable with the response, except the

$$\tilde{x}_{ij} = \frac{x_{ij}}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_{ij} - \overline{x}_j)^2}},$$

intercept β0. The coefficients that are produced by the standard least squares method are scale equivariant. However, this is not the case with ridge regression, and therefore, we need to standardize the predictors by the formula on left.

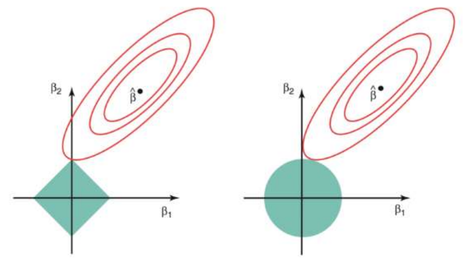
Lasso Regression

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}|\beta_j| \;= \text{RSS} + \lambda\sum_{j=1}^{p}|\beta_j|.$$

This function is minimized. It’s clear that this variation differs from ridge regression only in penalizing the high coefficients. It uses |βj| (modulus) instead of squares of β, as its penalty.

Difference between ridge and lasso regression

The ridge regression can be thought of as solving an equation, where summation of squares of coefficients is less than or equal to s. And the Lasso can be thought of as an equation where summation of modulus of coefficients is less than or equal to s. Here, s is a constant that exists for each value of shrinkage factor λ. Consider there are 2 parameters in a given problem. Then the ridge regression is expressed by $\beta 1^2 + \beta 2^2 \leq s$. Similarly, for lasso regression, the equation becomes, $|\beta 1|+|\beta 2|\leq s$. The image on the left describes these



equations. The image shows the constraint functions (green areas), for lasso(left) and ridge regression(right), along with contours for RSS (red ellipse). Points on the ellipse share the value of RSS. For a very large value of s, the green regions will contain the center of the ellipse, making coefficient estimates of both regression techniques, equal to the least squares estimates. But this is not the case in the above image. In this case, the lasso and ridge regression coefficient estimates are given by the first point at which an ellipse contacts the constraint region. Since ridge regression has a circular constraint with no sharp points, this intersection will not generally occur on an axis, and so the ridge regression coefficient estimates will be exclusively non-zero. However, the lasso constraint has corners at each of the axes, and so the ellipse will often intersect the constraint region at an axis. When this occurs, one of the coefficients will equal zero. In higher dimensions (where parameters are much more than 2), many of the coefficient estimates may equal zero simultaneously. This sheds light on the obvious disadvantage of ridge regression, which is model interpretability. It will shrink the coefficients for least important predictors, very close to zero. But it will never make them exactly zero. In other words, the final model will include all predictors. However, in the case of the lasso, the linear penalty has the effect of forcing some of the coefficient estimates to be exactly equal to zero when the tuning parameter λ is sufficiently large. Therefore, the lasso method also performs variable selection and is said to yield sparse models.

Conclusion

A standard least squares model tends to have some variance in it, i.e. this model won’t generalize well for a data set different than its training data. Regularization, significantly reduces the variance of the model, without substantial increase in its bias. As the value of λ rises, it reduces the value of coefficients and thus reducing the variance. Till a point, this increase in λ is beneficial as it is only reducing the variance (hence avoiding overfitting), without losing any important properties in the data. But after certain value, the model starts losing important properties, giving rise to bias in the model and thus underfitting. Therefore, the value of λ should be carefully selected.