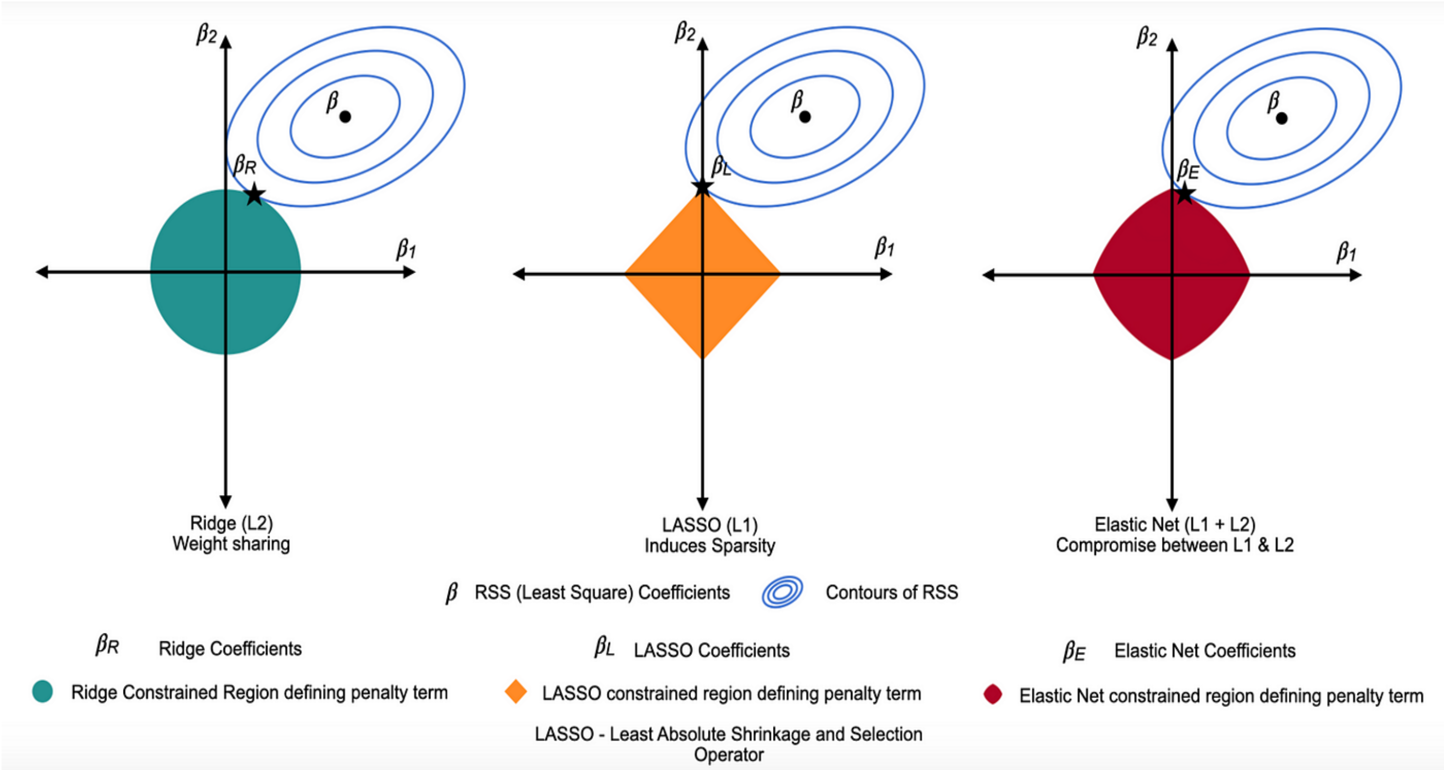

Table Of Contents

multi collinearity :-

→ we can define multi-collinearity as the situation where the independent var have strong correlation among themselves.

→ the coeff. in a linear Reg model represent the extent of change in 'y' when a certain x (x_1, x_2, x_3 ..) is changed keeping others constant. But if x_1 & x_2 are dependent then this assumption itself is wrong that we are changing one vars keeping others constant as the dependent var will also be changed.

It means model becomes a bit flawed.

$y = m x_1 + n x_2 + c$

while,

$x_2 = a x_1 + d$

Independent var (x_2, x_1) are related.

More than one linear eq,

Hence called multi collinearity

→ we have redundancy in our model as two variables (or more than two) are trying to convey the same information.

→ As the extent of collinearity increases, there is a chance that we might produce an "overfitted model".

→ An overfitted model works well with the test data but its accuracy fluctuates when exposed to other datasets.

* can result in a dummy variable trap.

→ By the heatmap we can check collinearity. Generally a correlation greater than 0.9 (or) less than -0.9 are to be avoided.

(#) Variance Inflection Factor (VIF):

Regression of one X var against other X variables.

$$\boxed{VIF = \frac{1}{1-R^2}}$$

VIF < 5 ✓ perfect

→ If VIF > 5, extreme correlation is avoided.

Remedies for multi-collinearity

1. Do-nothing: If corr not extreme / the var not used then ignore
2. Remove one-var: like in dummy var trap (one hot encoding)
3. combine correlated var: combine variables.
4. Principle componet Analysis (PCA)

Regularization :

→ when we use Regr. models to train some data, there is a good chance that the model will overfit the given training dataset.

→ Regularization helps sort this overfitting problem by restricting the degrees of freedom of given eq.

i.e. simply reducing the no. of degrees of a polynomial by reducing their corresponding weights.

Reason for Regularization

→ In a linear eq, we don't want huge coeff, as a small change in coeff can make a large diff. so regular constraints weights of such features to avoid overfitting.

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots \beta_p \cdot x_p$$

$$\boxed{RSS = \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j \cdot x_{ij} \right)^2} \quad (y_i - \hat{y})$$

* To regularize a model, shrinkage penalty is added to cost function,

LASSO (Least Absolute Shrinkage & Selection Operator) (L1 form)

→ LASSO regression penalizes the model based on the sum of magnitude of the coefficients.

The regularization term is given by,

$$\boxed{\text{regularization} = \lambda * \sum |\beta_j|}$$

λ — shrinkage factor

→ Loss after regularization is:-

new loss func.

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

Here, Loss is not calculated just by previous values $(y - \hat{y})$, now it has been split into several parts which decreases loss,,

Ridge Regression (L2 form)

→ Ridge Regression penalizes the model based on the sum of squares of mag of coeff.

$$\boxed{\text{regularization} = \lambda * \sum |\beta_j^2|}$$

λ - can be calculated by cross validation

→ Loss after regularization,

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

Note:- consider β_1 & β_2 be coeff of LR & $\lambda = 1$.

① For LASSO, $\boxed{\beta_1 + \beta_2 <= S}$ → where 'S' is the max value the eq. can achieve. At

② for Ridge, $\boxed{\beta_1^2 + \beta_2^2 <= S}$

Elastic net :-

→ middle ground b/w Ridge & Lasso.

→ The regularization term is a simple mix of both Ridge & Lasso, and you can control mix ratio α.

$$\boxed{L_{enet}(\hat{\beta}) = \frac{\sum_{i=1}^{n}(y_i - x_i\hat{\beta})^2}{2n} + \lambda\left[\left(\frac{1-\alpha}{2}\right)\cdot\sum_{j=1}^{m}\hat{\beta}_j^2 + \alpha\sum_{j=1}^{m}|\hat{\beta}_j|\right]}$$

→ where 'α' is the mixing parameter

Ridge ($\alpha=0$)

Lasso ($\alpha=1$) //

(*) Difference between Ridge & LASSO :-

→ Ridge Regression shrinks the coeff for those predictors which contribute very less in the model but have huge weights, very close to '0'.

$y = 0.04x$ — less
$\beta^2 = 0.0016$ — very less

Thus final model will still contain all those predictors, though with less weights.

→ Whereas in LASSO, the L1 penalty does reduce some coeff exactly to zero, when we use a sufficiently large tuning parameter λ.

So, in addition to regularizing, LASSO also performs feature selection.

Q— Why use Regularization?

Ans
→ It helps to reduce the variance of the model, without a substancial increase in the bias.

→ If there's variance in the model that means the model won't fit well for dataset other than training data (which is called overfitting).

→ The tuning parameter ('λ') controls this bias & variance trade off. selected using cross-validation

→ When 'λ' is increased to certain limit, it reduces the variance without losing any imp prop of data.

→ But after certain limit model will start losing imp prop which will increase bias in the data.

Q) What should be used plain linear, Ridge, Lasso or Elastic net?

Ans
→ It is always preferrable to have atleast a little bit of Regularization, so generally avoid plain linear Regr.

→ Ridge is a good default,

→ But if you suspect that only a few features are actually useful, you should prefer Lasso/EN, since they tend to reduce the useless features weights down to zero.

→ In general Elastic net is preffered over Lasso since Lasso may behave erratically when the no. of features is greater than no. of training instances or when several features are strongly correlated.

whenever, mean is not 0,
it's better to do standard scalar

Note:-

① StandardScaler() — $\frac{x-\bar{x}}{SD}$ $\bar{x} = 0$, $SD = 1$

② Two ways to find multicollinearity — correlation, VIF

③ 4 ways to remove multicollinearity

④ train set, high accuracy ; test set, low accuracy.
i.e. overfitting, so regularize it.
→ Even after checking Lasso and Ridge,
if it gives the same accuracy values,
then you can it is not over-fitted.

i.e. Just by having huge difference in train accuracy & test accuracy then it doesn't just mean overfitted. Test before judge. In this case model is not problem, more data is needed.

⑤ The linear in Linear Regression does not talk about the degree of polynomial eq in terms of dependent var(x).
Instead, it talks about the degree of coeff.

$\boxed{y = a + bx + cx^2 + \cdots nx^n}$

it's not talking about power of x,
but the power of a, b, c etc. And their powers
Hence it is linear Regression

Polynomial Regression

→ It is a mechanism to predict a dependent var based on the polynomial relationship with the independent variable.

In the eq, $y = a + bx + cx^2 + \cdots nx^n + \cdots$

max. power of x is called degree of polynomial eq.

Deg ① → $y = a + bx$

Deg ② → $y = a + bx + cx^2$ —

when to use?

✱ If data is scatter in a polynomial curve shape (not linear) then keep trying different degrees until the line fits to data.

codes

```
from sklearn.preprocessing import PolynomialFeatures
Poly_reg = PolynomialFeatures(degree=2)
X_Poly = Poly_reg.fit_transform(x)
```

→ keep changing degree until it fits to data.