# Complete Machine Learning _/_/___

① Linear Regression

Goal: To find the best fitting line (hyperplane) that minimizes the error b/w ($\hat{y}$ and $y$)

$y = wx + b$ / $y_i = \beta_0 + \beta_1 x_i + \epsilon_i$

Assumptions:
1. Linearity
2. Independence of errors
3. Homoscedasticity: Constant variance of errors
4. Normality: Errors distributed normally
5. No multicollinearity.

* The best fit line will be the one that optimizes the values of m (slope) and b (intercept) so the predicted y values are as close as possible to the actual data points.

To find a best fit line: Least Squares Method

Minimise the sum of squared diff b/w actual & predicted values from the line → residuals. (mean should be 0)

{SSE / RSS} : $\sum_{i=1}^{n} (y_i - \hat{y}_i)^2$ . $\sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 x_i))^2$

(Residual Sum of Squares)

Ensure the diff b/w predicted & actual values is small as possible

② Multivariate Regression

$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots \beta_p + x_p + \epsilon$

Goal: Estimate B's that minimises RSS.

__/__/____

→ Multicollinearity: Occurs when predictors are highly correlated.

- Leads to unstable β estimates.
- Detect using Variation Inflation Factor (VIF)

$$VIF_i = \frac{1}{1 - R^2_i}$$

High VIF (> 10) → strong multicoll...

* Use feature selection / regularization (Ridge / Lasso) to fix this.

(singular)

→ $X^TX$ is non invertible, causes:

- Multicollinearity
- $p > n$ (features more than samples)

↝ If $X^TX$ singular. Linear Regression cannot be solved by normal equations. (use when $p < 1000$)

FIX ? ⇒ GRADIENT DESCENT

We can't min RSS directly using calculus

Cost function (J) = $\frac{1}{n} \sum_{i=1}^{n} (\hat{y}_i - y_i)^2$ [MSE]

(RSS)

[RSS ∝ J]

GD: Optimization technique → used for Linear Regression, by minimizing the prediction error. Repeatedly adjust model parameters to reduce the diff b/w $\hat{y}_i$ and $y_i$.

1. Initialize Parameters: random values for m and b.
2. Calculate Cost function: $\frac{1}{n} \sum_{i=1}^{n} (y_i - (mx_i + b))^2$
3. Compute Gradient

m: $\partial J / \partial m = -2/n \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 x_i$

b: $\partial J / \partial b = -2/n \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$

__/__/____

4. Update parameters :

$m = m - \alpha \, \partial J / \partial m$ (we move opposite to the gradient direction.)

$b = b - \alpha \, \partial J / \partial b$

5. Repeat until convergance.

* Always converges for convex functions like Linear Regression
* Gradient Descent updates $\theta$ values in the direction that reduces the total squared residuals - minimize RSS

→ $R^2$ : Coefficient of Determination

↳ Indicates how much variation the developed model can explain or capture. Range : $[0, 1]$

$$\left( R^2 : 1 - \frac{RSS}{TSS} \right)$$

→ Adjusted $R^2$ : Accounts the number of predictors in the model and penalizes the model for including irrelevant predictors that don't contribute significantly to explain the variance in dependent variables. { Helps prevent overfitting }

$$\text{Adjusted } R^2 : 1 - \left( \frac{(1-R^2) \cdot (n-1)}{n-k-1} \right)$$

→ SUBSET SELECTION

Used when there are many predictors, and we may not want to use all.

1. Forward Selection ($\emptyset$ → select)
2. Backward Selection (full → select / dismiss)

# REGULARIZATION / FEATURE SELECTION

__/__/____

③ Shrinkage Methods

Techniques that constrain / regularize the coeffs of a regression model to reduce variance and prevent overfitting

→ Shrink large coeffs ($\beta$) to 0.

Coeffs can get very large when:

- Multicollinearity
- Model Overfitting

Shrinkage helps by adding a penalty term to the cost function

1. Ridge Regression (L2 Regularization)

$$J(\beta) = RSS + \lambda \sum_{j=1}^{n} \beta_j^2$$

{ coeffs are smaller in magnitude }

$$J(\beta) = \sum_{i=1}^{m} (y_i - \hat{y_i})^2 + \lambda \sum_{j=1}^{n} \beta_j^2$$

- As $\lambda \uparrow$s, coeffs shrink towards 0, not zero exactly.
- Reduces variance but introduces small bias.
- Always scale before (mean = 0, std dev = 1) fitting.

2. Lasso Regression (L1 Regularization)

$$J(\beta) = RSS + \lambda \sum_{j=1}^{n} |\beta_j|$$

(feature)

- Performs both shrinkage & variable selection
- Some $\beta$s exactly becomes 0, eliminate them

(high $\lambda$ → high bias, low var)

3. Elastic Net ($L_1 + L_2$)

- Useful when predictors are highly correlated.
- Ridge → stability + Lasso → Feature Selection.

DIMENSIONALITY REDUCTION / FEATURE EXTRACTION

__/__/____

④ Principal Components Regression (PCR) (PCA + Regression)

Dimensionality Reduction technique → applies PCA to the feature matrix X before Regression.

↳ Top k principal components (PCs) of X - reducing variance retaining most information.

S1: PCA - transforms correlated predictors into uncorrelated components... $[Cov(PC_1, PC_2) = 0]$

$PC_1$ (max Variance) > $PC_2$ (2nd max Variance) > $PC_3$ ...

* - Principal components are linear combinations of many features
- Almost all variables contribute to PCs (even if indirectly)
- Removes multicollinearity completely - PCs are orthogonal

* Use PCR when features are highly correlated, but we want to use all info → Dimensionality Reduction → new uncorrelated feat

PCA
1. Center and scale X (input - features contribute equally)
2. Compute Cov Matrix $X^T X$ (Capture feature relationships)
3. Find Eigenvalues & Eigenvectors (Variance explained)
4. Sort by largest Eigenvalues (Pick PCs with max variance)
5. Project X onto M eigenvectors
6. Choose top M PCs
7. Regress y on these PCs

⇒ Need to use 'y' while forming components ? Use:

Partial least Square (PLS)

__/__/____

## ⑤ Bias - Variance Tradeoff

In supervised learning, we want to reduce the expected prediction error on unseen data.

Total Error can be decomposed to:

$$E\left[(Y - \hat{f}(X))^2\right] = \text{Bias}^2 + \text{Variance} + \sigma^2$$

Bias: error from wrong assumption

Variance: sensitivity of data

$\sigma$ : irreducible error ~ noise inherent in data.

1. Bias: how far model the avg model prediction is from the true function $f(x)$.

↑ High Bias : Model too simple, underfitting

$$\text{Bias}\left(\hat{f}(x)\right) = E\left[\hat{f}(x)\right] - f(x)$$

2. Variance: how much the models prediction changes with different training data.

↑ High Variance : model too sensitive to noise, overfitting

$$\text{Variance}\left(\hat{f}(x)\right) = E\left[\hat{f}(x)\right] - E\left[\hat{f}(x)\right]$$

→ Signs

- Training acc >> Validation acc : Overfitting
- Training & Validation acc low : underfitting
- Train error ↓ Val Error ↑ : overfitting

__/__/____

⑥ Cross Validation

A technique used to check how well a ml model performs on unseen data while preventing overfitting.

- Splitting dataset into diff parts
- Training on some parts and testing on remaining.
- Averaging the results from multiple validation step to get the final performance.

IMP

1. K-fold CV
   - Split data into k roughly equal parts (folds)
   - For each fold,
     train on k-1 folds, test on the remaining fold.
   - Compute mean test error.

2. Leave one out CV (LOOCV)
   - Special case of k-fold where k=n
   - Train on n-1 samples, test on 1, repeat for all

3. Repeated / Stratified CV

Example of

K fold :

| Train | Test |

| Train | Test | Train |

| Test | Train |

→ Discriminative Model predicts the boundary between classes while generative models learn the underlying data distribution to create new data.