Linear Regression:

Linear regression is a statistical method used to model the relationship between a dependent variable and one or more independent variables. The relationship is modeled by fitting a linear equation to the observed data. The linear equation has the form:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n + \in$$

where:

- y is the dependent variable.

- $x_1, x_2, \dots, x_n$ are the independent variables.

- β_0 is the intercept.

- $\beta_1, \beta_2, \dots \beta_n$ are the coefficients.

- $\in$ is the error term.

Non-Linear Regression:

Non-linear regression is a form of regression analysis in which data is modeled by a function that is a non-linear combination of the model parameters and depends on one or more independent variables. The model can take various forms, such as exponential, logarithmic, polynomial, and others. A general form of a non-linear regression model is:

$$y = f(x_1, x_2, \dots, x_n;\ \theta) + \in$$

where:

- y is the dependent variable.

- $x_1, x_2, \dots, x_n$ are the independent variables.

- f is a non-linear function of the independent variables and the parameters θ

- θ are the parameters of the model.

- $\in$ is the error term.

Verifying Linear and Non-Linear Regression Models:

1. Goodness of Fit:

- **Linear Regression:** Use the coefficient of determination (R^2) to assess how well the model explains the variability of the dependent variable. An R^2 value close to 1 indicates a good fit.

- **Non-Linear Regression:** Similarly, use R^2 or adjusted R^2 for non-linear models. Other metrics like Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Akaike Information Criterion (AIC) can also be used.

2. Residual Analysis:

- **Linear Regression:** Plot the residuals (the differences between observed and predicted values) to check for randomness. If the residuals exhibit a pattern, it may indicate a non-linear relationship.

- **Non-Linear Regression:** Residual plots should also be checked for non-linear models. Patterns in residuals may suggest that the model does not capture the underlying relationship adequately.

3. Model Assumptions:

- **Linear Regression:** Check for linearity, independence, homoscedasticity (constant variance of residuals), and normality of residuals.

- **Non-Linear Regression:** Verify that the chosen non-linear function appropriately fits the data. Assumptions may vary depending on the specific non-linear model used.

4. Cross-Validation:

- Use techniques like k-fold cross-validation to assess the model's performance on different subsets of the data, ensuring that the model generalizes well to unseen data.

5. Comparison of Models:

- Compare linear and non-linear models using information criteria like AIC or Bayesian Information Criterion (BIC) to determine which model provides a better fit while penalizing for model complexity.

6. Visualization:

- Plot the observed data and the fitted regression line (for linear) or curve (for non-linear) to visually assess the fit of the model.

Multiple Variable Linear Regression

Multiple variable linear regression, also known as multiple linear regression, is a statistical technique used to model the relationship between one dependent variable and two or more independent variables. The equation of a multiple linear regression model is:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n + \in$$

where:

- y is the dependent variable.

- $x_1, x_2, \ldots, x_n$ are the independent variables.

- β_0 is the intercept.

Multiple Variable Linear Regression Example

- $\beta_1, \beta_2, \ldots \beta_n$ are the coefficients.

- $\in$ is the error term.

Polynomial Regression:

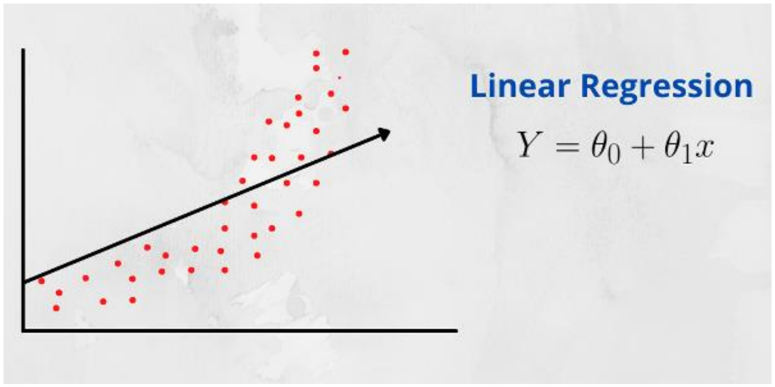
Polynomial regression is a type of regression analysis in which the relationship between the independent variable xxx and the dependent variable y is modeled as an nth degree polynomial. Polynomial regression fits a nonlinear relationship between the value of xxx and the corresponding conditional mean of y, denoted $E(y|x)$

The polynomial regression model can be written as:

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_3 x^n + \in$$

Why Polynomial Regression?

A simple linear regression algorithm only works when the relationship between the data is linear. But suppose we have non-linear data, then linear regression will not be able to draw a best-fit line. Simple regression analysis fails in such conditions. Consider the below diagram, which has a non-linear relationship, and you can see the linear regression results on it, which does not perform well, meaning it does not come close to reality. Hence, we introduce it to overcome this problem, which helps identify the curvilinear relationship between independent and dependent variables.



How Does Polynomial Regression Handle Non-Linear Data?

Polynomial regression is a form of Linear regression where only due to the Non-linear relationship between dependent and independent variables, we add some polynomial terms to linear regression to convert it into Polynomial Regression in Machine Learning.

The relationship between the dependent variable and the independent variable is modeled as an nth-degree polynomial function. When the polynomial is of degree 2, it is called a quadratic model; when the degree of a polynomial is 3, it is called a cubic model, and so on.

Suppose we have a dataset where variable X represents the independent data and Y is the dependent data. Before feeding data to a mode in the preprocessing stage, we convert the input variables into polynomial terms using some degree.

Consider an example my input value is 35, and the degree of a polynomial is 2, so I will find 35 power 0, 35 power 1, and 35 power 2 this helps to interpret the non-linear relationship in data.
The equation of polynomials becomes something like this.

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_3 x^n + \in$$

The degree of order which to use is a Hyperparameter, and we need to choose it wisely. But using a high degree of polynomial tries to overfit the data, and for smaller values of degree, the model tries to underfit, so we need to find the optimum value of a degree. Polynomial Regression in Machine Learning models are usually fitted with the method of least squares. The least square method minimizes the variance of the coefficients under the Gauss-Markov Theorem.

Why Is Polynomial Regression Called Polynomial Linear Regression?

If you look at the equation of polynomial regression python carefully, then we can see that we are trying to estimate the relationship between coefficients and y. And the values of x and y are already given to us, only we need to determine coefficients, and the degree of coefficient here is 1 only, and degree one represents simple linear regression Hence, Polynomial Regression in Machine Learning is also known as Polynomial Linear Regression as it has a polynomial equation, and this is only the simple concept behind this. I hope you got the point right.

Underfitting: Underfitting occurs when a model is too simple to capture the underlying structure of the data. It performs poorly on both training and testing data because it fails to learn the relationships in the data. In regression, underfitting might happen if a linear model is used for data that has a more complex, non-linear relationship. *Occurs when a model is too simple to capture the data's patterns, resulting in high bias and poor performance on both training and test data*.

Best Fitting: Best fitting, or a well-fit model, strikes the right balance between bias and variance. It captures the underlying patterns in the training data and generalizes well to unseen data. In regression, a best-fit model accurately represents the true relationship between the independent and dependent variables without being too simple or overly complex*. Occurs when the model complexity is just right, capturing the underlying data patterns without fitting the noise.*

Overfitting: Overfitting occurs when a model is too complex and captures the noise in the training data rather than the underlying pattern. It performs well on training data but poorly on testing data because it doesn't generalize well. In regression, overfitting might happen if a high-degree polynomial is used for data that has a simpler underlying relationship. *Occurs when the model is too complex, capturing noise along with the underlying data patterns. This results in excellent performance on the training data but poor generalization of new, unseen data.*

CONCEPT FOR OVERFITTING AND UNDERFITTING:

***Overfitting**: A model is said to be overfitted when we train it with a lot of data. after getting trained with a large amount of data, it starts **learning from the noise** and inaccurate data entries in our data set. Now the model may not be able to differentiate between noise and data. The main step to prevent overfitting is Increase the dataset and reduce the complexity of model. Using **dropouts** in NN can also be another way to reduce overfitting*

***Underfitting**: An ML model is said to have underfitting when is not able to gather the generalized trend of the data. Underfitting is one of the major reasons for affecting the accuracy of our model. Its occurrence simply means that our model or the algorithm does not fit the data well enough. It usually happens when we have fewer data to build an accurate model. Issue can be solved mostly by adding new features to the data / removing noise from the data. Another easy method is to increase the number of epochs for the data training.*

MODEL COMPLEXITY VS OVERFITTING

Model complexity refers to the capacity of a model to fit a wide variety of functions. It involves the number of parameters or the flexibility of the model. In machine learning, managing model complexity is crucial for balancing bias and variance, which directly relates to underfitting and overfitting.

- **Simple Models (Low Complexity)**: These models have fewer parameters and are less flexible. Examples include linear regression and low-degree polynomial regression. Simple models may suffer from high bias, leading to underfitting.
- **Complex Models (High Complexity)**: These models have many parameters and are highly flexible. Examples include high-degree polynomial regression and deep neural networks. Complex models may suffer from high variance, leading to overfitting.

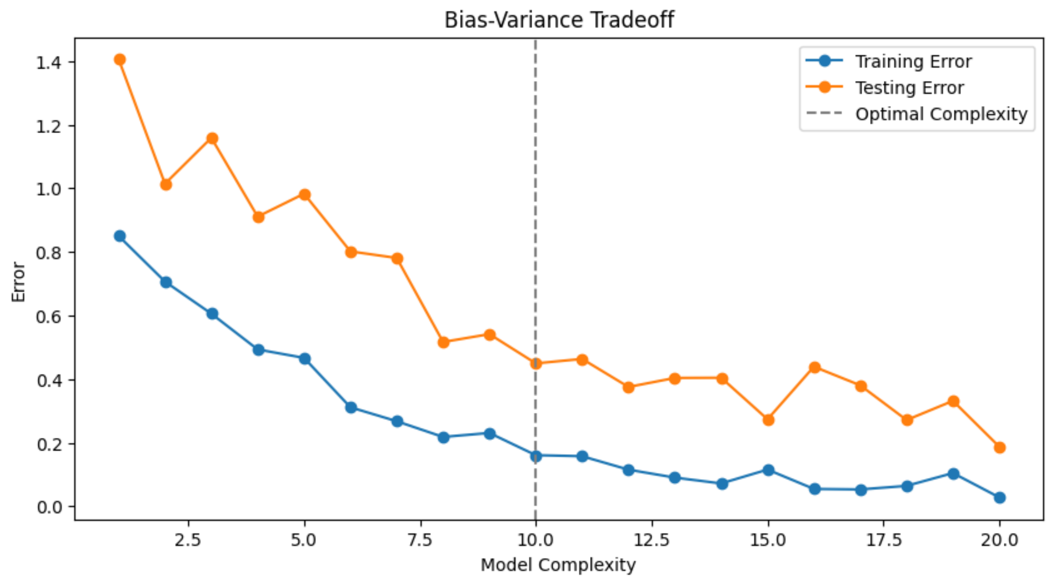
MODEL COMPLEXITY AND THE BIAS-VARIANCE TRADEOFF

The bias-variance tradeoff is a fundamental concept in understanding model complexity:

- **Bias**: Error due to overly simplistic assumptions in the learning algorithm. High bias can cause the model to miss relevant relations (underfitting).
- **Variance**: Error due to too much complexity in the learning algorithm. High variance can cause the model to model the random noise in the training data (overfitting).

Visual Representation of Bias-Variance Tradeoff

A common way to visualize the bias-variance tradeoff is with a graph showing training and testing errors as functions of model complexity:



In the graph, you can see:

- At low complexity, both training and test errors are high (underfitting).
- At medium complexity, training and test errors are minimized (best fitting).
- At high complexity, training error is low, but test error is high (overfitting).

  overfitting and underfitting example

R-squared (Coefficient of Determination)

Measures the proportion of the variance in the dependent variable that is predictable from the independent variables. It provides an indication of goodness of fit and therefore a measure of how well unseen samples are likely to be predicted by the model.

$$R^2 = 1 - \frac{SSR}{SST} \quad ; R^2 \text{ will be between 0 and 1.}$$

Sum of Squared Regression (SSR) or Residual Sum of Squares: This measures the variation in the dependent variable that is explained by the regression model. In other words, it quantifies how well the model fits the data. A higher SSR indicates a better fit.

$$SSR = \sum(y_i - \hat{y}_i)^2$$

Total Sum of Squares (SST): This measures the total variation in the dependent variable. It represents the overall variability of the data.

$$SST = \sum(y_i - \bar{y})^2$$

In essence:

- *SSR tells you how much of the variation in the data is captured by your model.*
- *SST tells you the total variation present in the data.*

Cost Function: A cost function is a specific type of objective function that quantifies the error between a predicted value and an actual value. In machine learning, the goal is typically to minimize the cost function.

Sometimes, the term loss function is used interchangeably with the cost function. However, there can be subtle differences:

- *Loss function often refers to the error for a single data point.*
- *Cost function is typically the average loss over the entire dataset.*

The objective function, or the cost function or loss function, is a mathematical function that a machine learning algorithm seeks to minimize during the training process.
It represents a measure of the error or discrepancy between the predicted values of the model and the actual target values in the training data. The ***primary goal*** *of a machine learning algorithm is to learn a set of parameters that minimizes this objective function, thereby* ***improving the model's ability to make accurate predictions on new, unseen data.***
The objective function guides the optimization algorithm, which iteratively adjusts the model parameters to find the optimal values. or loss function is a mathematical function that a machine learning algorithm seeks to minimize during training loss function, is a mathematical function that a machine learning algorithm seeks to minimize during training loss function, is a mathematical function that a machine learning algorithm seeks to minimize during training.

Differences among these evaluation metrics

- Mean Squared Error (MSE) and Root Mean Square Error (RMSE) penalizes the large prediction errors vi-a-vis Mean Absolute Error (MAE). However, RMSE is widely used than MSE to evaluate the performance of the regression model with other random models as it has the same units as the dependent variable (Y-axis).
- MSE is a differentiable function that makes it easy to perform mathematical operations in comparison to a non-differentiable function like MAE. Therefore, in many models, RMSE is used as a default metric for calculating Loss Function despite being harder to interpret than MAE.
- The lower value of MAE, MSE, and RMSE implies higher accuracy of a regression model. However, a higher value of R square is considered desirable.
- R Squared & Adjusted R Squared are used for explaining how well the independent variables in the linear regression model explains the variability in the dependent variable. R Squared value always increases with the addition of the independent variables which might lead to the addition of the redundant variables in our model. However, the adjusted R-squared solves this problem.
- Adjusted R squared considers the number of predictor variables, and it is used to determine the number of independent variables in our model. The value of Adjusted R squared decreases if the increase in the R square by the additional variable isn't significant enough.
- For comparing the accuracy among different linear regression models, RMSE is a better choice than R Squared.

What Is Regularization in Machine Learning?

Regularization in machine learning serves as a method to forestall a model from overfitting. Overfitting transpires when a model not only discerns the inherent pattern within the training data but also incorporates the noise, potentially leading to subpar performance on fresh, unobserved data. ***If the number of features is large then it will cause overfitting.*** Regularization is a set of methods for reducing overfitting in machine learning models. Typically, regularization trades a marginal decrease in training accuracy for an increase in generalizability.

The primary goal of regularization is to reduce the model's complexity to make it more generalizable to new data, thus improving its performance on unseen datasets.

L1 Regularization (Lasso): This adds a penalty equal to the absolute value of the magnitude of coefficients. This can lead to some coefficients being zero, which means the model ignores the corresponding features. It is useful for feature selection.

L2 Regularization (Ridge): Adds a penalty equal to the square of the magnitude of coefficients. All coefficients are shrunk by the same factor, and none are eliminated, as in L1.

- ***Bias*** *measures the average difference between predicted values and true values. As bias increases, a model predicts less accurately on a training dataset. High bias refers to high error in training.*

- ***Variance*** *measures the difference between predictions across various realizations of a given model. As variance increases, a model predicts less accurately on unseen data. High variance refers to high error during testing and validation.*

Explain L1 Regularization (Lasso) and L2 Regularization (Ridge)?

- Lasso regression (or L1 regularization) is a regularization technique that penalizes high-value, correlated coefficients. It introduces a regularization term (also called, penalty term) into the model's sum of squared errors (SSE) loss function. This penalty term is the absolute value of the sum of coefficients. Controlled in turn by the hyperparameter lambda (λ), it reduces select feature weights to zero. Lasso regression thereby removes MULTICOLLINEAR FEATURES from the model altogether.

$$Cost = \frac{1}{n} \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \lambda \sum_{i=1}^{m} |w_i|$$

Where, m = number of features n = number of examples λ (lambda) is the regularization strength

y_i = actual target value $\hat{y}_i$ = predicted target value

- Ridge regression (or L2 REGULARIZATION) is regularization technique that similarly penalizes high-value coefficients by introducing a penalty term in the SSE loss function. It differs from lasso regression, however. First, the penalty term in ridge regression is the squared sum of coefficients rather than the absolute value of coefficients. Second, ridge regression does not enact feature selection. While lasso regression's penalty term can remove features from the model by shrinking coefficient values to zero, ridge regression only shrinks feature weights towards zero but never to zero.

$$Cost = \frac{1}{n} \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \lambda \sum_{i=1}^{m} w_i^2$$

- Elastic net regularization essentially combines both ridge and lasso regression but inserting both the L1 and L2 penalty terms into the SSE loss function. L2 and L1 derive their penalty term value, respectively, by squaring or taking the absolute value of the sum of the feature weights. Elastic net inserts both of these penalty values into the cost function (SSE) equation. In this way, elastic net addresses multicollinearity while also enabling feature selection.

$$Cost = \frac{1}{n} \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \lambda\{(1-\alpha) \sum_{i=1}^{m} |w_i| + \alpha \sum_{i=1}^{m} |w_i^2|\}$$

How Does Regularization Work?

Regularization adds a penalty term to the standard loss function that a machine learning model minimizes during training. This penalty encourages the model to keep its parameters (like weights in neural networks or coefficients in regression models) small, which can help prevent overfitting. Here's a step-by-step breakdown of how regularization functions:

1. Modifying the Loss Function

Regularized Loss = Original Loss + λ × Penalty

Here, λ (lambda) is the regularization strength, which controls the trade-off between fitting the data well and keeping the model parameters small.

2. Types of Penalties

- **L1 Regularization (Lasso):** The penalty is the sum of the absolute values of the parameters. This can lead to a sparse model where some parameter values are exactly zero, effectively removing those features from the model.
- **L2 Regularization (Ridge):** The penalty is the sum of the squares of the parameters. This evenly distributes the penalty among all parameters, shrinking them towards zero but not exactly zeroing any.
- **Elastic Net:** A mix of L1 and L2 penalties. It is useful when there are correlations among features or when you want to combine the feature selection properties of L1 with the shrinkage properties of L2.

3. Effect on Training

- Minimizing a larger penalty term (due to larger values of λ) emphasizes smaller model parameters, leading to simpler models that might generalize better but could underfit the training data.
- Minimizing a smaller penalty term (lower values of λ) allows the model to fit the training data more closely, possibly at the expense of increased complexity and overfitting

4. Balancing Overfitting and Underfitting

Choosing the right value of λ is crucial:

- Too high a value can make the model too simple and fail to capture important patterns in the data (underfitting).
- Too low a value might not sufficiently penalize large coefficients, leading to a model that captures too much noise from the training data (overfitting).

In practice, the optimal value of λ and the type of regularization (L1, L2, or Elastic Net) are often selected through cross-validation, where multiple models are trained with different values of λ and possibly different types of regularization. The model that performs best on a validation set or through a cross-validation process is then chosen.

WHAT ARE THE ROLS OF REGULARIZATION?

- **Preventing Overfitting:** Regularization's most significant role is to prevent overfitting, a common issue in which a model learns the underlying pattern and noise in the training data. This usually results in high performance on the training set but poor performance on unseen data. Regularization reduces overfitting by penalizing larger weights, encouraging the model to prioritize simpler hypotheses.
- **Improving Model Generalization:** Regularization helps ensure the model performs well on the training and new, unseen data by constraining its complexity. A well-regularized model will likely capture the data's underlying trends rather than the training set's specific details and noise.

- **Handling Multicollinearity**: Regularization is particularly useful in scenarios where features are highly correlated (multicollinearity). L2 regularization (Ridge) can reduce the variance of the coefficient estimates, which are otherwise inflated due to multicollinearity. This stabilization makes the model's PREDICTIONS more reliable.
- **Feature Selection:** L1 regularization (Lasso) encourages sparsity in the model coefficients. By penalizing the absolute value of the coefficients, Lasso can shrink some of them to exactly zero, effectively selecting a smaller subset of the available features. This can be extremely useful in scenarios with high-dimensional data where feature selection is necessary to improve model interpretability and efficiency.
- **Improving Robustness to Noise:** Regularization makes the model less sensitive to the idiosyncrasies of the training data. This includes noise and outliers, as the penalty discourages fitting them too closely. Consequently, the model focuses more on the robust features that are more generally applicable, enhancing its robustness.
- **Trading Bias for Variance:** Regularization introduces bias into the model (assuming that smaller weights are preferable). However, it reduces variance by preventing the model from fitting too closely to the training data. This trade-off is beneficial when the unconstrained model is highly complex and prone to overfitting.
- **Enabling the Use of More Complex Models:** Regularization sometimes allows practitioners to use more complex models than they otherwise could. For example, regularization techniques like dropout can be used in neural networks to train deep networks without overfitting, as they help prevent neuron co-adaptation.
- **Aiding in Convergence:** For models trained using iterative OPTIMIZATION TECHNIQUES (like gradient descent), regularization can help ensure smoother and more reliable convergence. This is especially true for problems that are ill-posed or poorly conditioned without regularization.

What Are Overfitting and Underfitting?

Overfitting

OVERFITTING happens when a model gets too caught up in the nuances and random fluctuations of the training data to the point where its ability to perform well on new, unseen data suffers. Essentially, the model becomes overly intricate, grasping at patterns that don't hold up when applied to different datasets.

Characteristics:

- High accuracy on training data but poor accuracy on validation or test data.
- The model has learned the training data's underlying structure and random fluctuations.
- Often occurs when the model is too complex relative to the amount and noisiness of the input data.

Common Causes:

- Too many parameters in the model (high complexity).
- Too little training data.
- Insufficient use of regularization.
- Training for too many epochs or without early stopping.

Mitigation Strategies:

- Simplify the model by reducing the number of parameters or using a less complex model.
- Increase training data.
- Use regularization techniques like L1, L2, and dropout.
- Employ techniques like cross-validation to ensure the model performs well on unseen data.
- Implement early stopping during training.

Underfitting

Underfitting arises when a model lacks the complexity to capture the underlying patterns within the data. Consequently, it inadequately fits the training data, leading to subpar performance when applied to new data.

Characteristics:

- Poor performance on both the training and testing datasets.
- The model is too simple and does not capture the basic trends in the data.

Common Causes:

- The model is too simple and has very few parameters.
- Features used in the model do not adequately capture the complexities of the data.
- Excessive use of regularization (too strong a penalty for model complexity).

Mitigation Strategies:

- Increase the complexity of the model by using more parameters or choosing a more sophisticated model.
- Feature engineering: Create more features or use different techniques to extract and select relevant features.
- Reduce the regularization force if the model is overly penalized.
- Ensure the model is properly trained and tweak training parameters like the number of epochs or learning rate.

What Are Bias and Variance?

Bias and variance are two fundamental concepts that describe different types of errors in predictive models in machine learning and statistics. Understanding bias and variance is crucial for diagnosing model performance issues and navigating the trade-offs between underfitting and overfitting.

Bias

Bias in machine learning arises when a simplified model fails to capture the complexities of a real-world problem. This oversight can lead to underfitting, where the algorithm overlooks important relationships between input features and target outputs.

Characteristics:

- Bias is the difference between our model's expected (or average) prediction and the correct value we try to predict. Models with high bias pay little attention to the training data and oversimplify the model, often leading to underfitting.
- High bias can lead to a model that is too simple and does not capture the complexity of the data.

Variance

Variance refers to the amount by which the model's predictions would change if we estimated it using a different training data set. Essentially, variance indicates how much the model's predictions are spread out from the average prediction. Excessive variability can lead an algorithm to mimic the random fluctuations in the training data instead of focusing on the desired outcomes, resulting in overfitting.

Characteristics:

- Variance quantifies the extent to which predictions for a specific point fluctuate across various model instances.
- Elevated variance may cause the model to capture the noise within the training data instead of the desired outcomes, thereby causing subpar performance when applied to unseen data.

Bias Variance Tradeoff

The relationship between bias and variance is referred to as the bias-variance trade-off. Minimizing both bias and variance is ideal:

- High Bias, Low Variance: The models are consistent but inaccurate on average, typical of overly simplified models.
- Low Bias, High Variance: Models are accurate on average but inconsistent across different datasets. This is typical of overly complex models.
- Low Bias, Low Variance: Models are accurate and consistent on training and new data, indicating a good balance between model complexity and performance on unseen data.
- High Bias, High Variance: Models are inaccurate and inconsistent, performing poorly in training and on new data.

Balancing the Trade-off:

- Underfitting: Occurs when the model is too simple, characterized by low variance and high bias.
- Overfitting: Occurs when the model is too complex, characterized by high variance and low bias.

Benefits of Regularization

1. **Reduces Overfitting:** Regularization helps prevent models from learning noise and irrelevant details in the training data.
2. **Improves Generalization:** By discouraging complex models, regularization ensures better performance on unseen data.
3. **Enhances Stability:** Regularization stabilizes model training by penalizing large weights.
4. **Enables Feature Selection:** L1 regularization can zero out some coefficients, effectively selecting more relevant features.
5. **Manages Multicollinearity:** Reduces the problem of high correlations among features, particularly useful in linear models.
6. **Encourages Simplicity:** Promotes simpler models that are easier to interpret and less likely to overfit.
7. **Controls Model Complexity:** Provides a mechanism to balance the complexity of the model with its performance on the training and test data.
8. **Facilitates Robustness:** Makes models less sensitive to individual peculiarities in the training set.
9. **Improves Convergence:** Helps optimization algorithms converge more quickly and reliably by smoothing the error landscape.
10. **Adjustable Complexity:** The strength of regularization can be tuned to fit the data's specific needs and desired model complexity.

Different Combinations of Bias-Variance

High Bias, Low Variance: *A model that has high bias and low variance is considered to be underfitting.*

High Variance, Low Bias: *A model that has high variance and low bias is considered to be overfitting.*

High-Bias, High-Variance: *A model with high bias and high variance cannot capture underlying patterns and is too sensitive to training data changes. On average, the model will generate unreliable and inconsistent predictions.*

Low Bias, Low Variance: *A model with low bias and low variance can capture data patterns and handle variations in training data. This is the perfect scenario for a machine learning model where it can generalize well to unseen data and make consistent, accurate predictions. However, in reality, this is not feasible.*

What is the Difference Between Ridge Regression and Lasso Regression?

Parameter	Ridge Regression	Lasso Regression
Regularization Type	L2 regularization adds a penalty equal to the square of the magnitude of coefficients.	L1 regularization adds a penalty equal to the absolute value of the magnitude of coefficients.
Primary Objective	To shrink the coefficients towards zero to reduce model complexity and multicollinearity.	To shrink some coefficients towards zero for both variable reduction and model simplification.
Feature Selection	Does not perform feature selection: all features are included in the model, but their impact is minimized.	Performs feature selection: can completely eliminate some features by setting their coefficients to zero.
Coefficient Shrinkage	Coefficients are shrunk towards zero but not exactly to zero.	Coefficients can be shrunk to exactly zero, effectively eliminating some variables.
Suitability	Suitable in situations where all features are relevant, and there is multicollinearity.	Suitable when the number of predictors is high and there is a need to identify the most significant features.
Bias and Variance	Introduces bias but reduces variance.	Introduces bias but reduces variance, potentially more than Ridge due to feature elimination.
Interpretability	Less interpretable in the presence of many features as none are eliminated.	More interpretable due to feature elimination, focusing on significant predictors only.
Sensitivity to λ	Gradual change in coefficients as the penalty parameter λ changes.	Sharp thresholding effect where coefficients can abruptly become zero as λ changes.
Model Complexity	Generally, results in a more complex model compared to Lasso.	This leads to a simpler model, especially when irrelevant features are abundant.

sparsity refers to a matrix of numbers that includes many zeros or values that will not significantly impact a calculation.

Lasso and Ridge Regression with examples