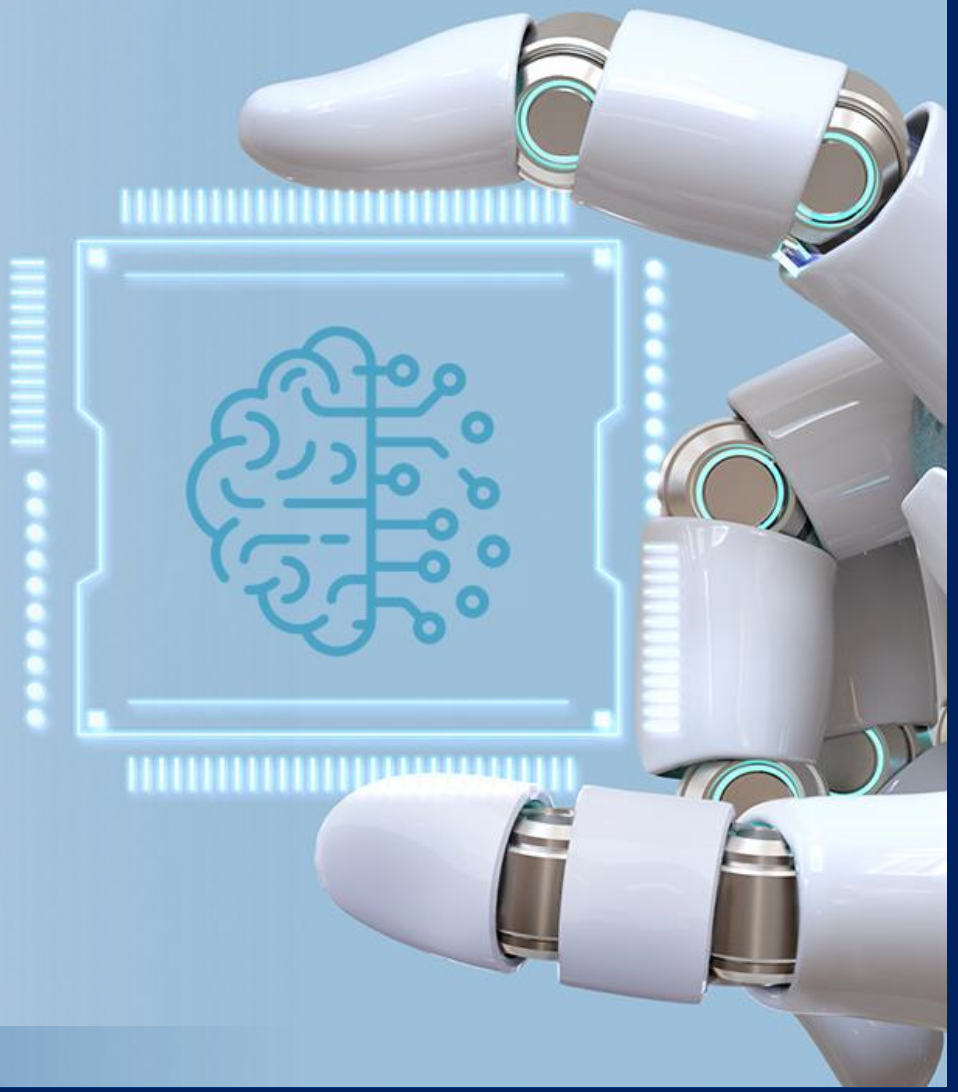
edunet foundation

Key Concepts in AI & ML Algorithms



Disclaimer

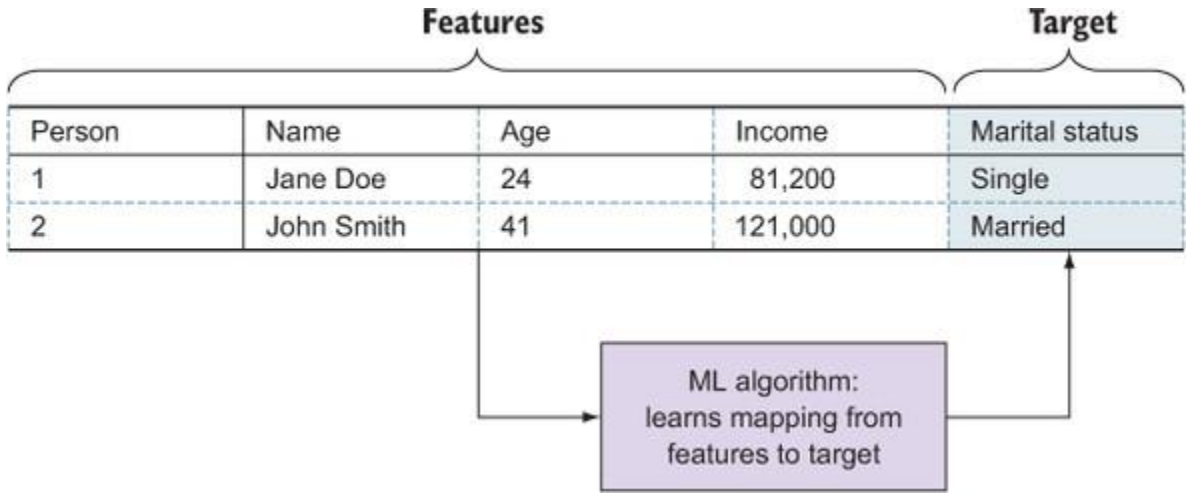
The content is curated from online/offline resources and used for educational purpose only

Agenda

- Feature engineering and representation learning
- Loss functions and optimization algorithms
- Training and testing data, overfitting, and generalization in machine learning
- Introduction to Machine Learning Algorithms
- Linear Regression

What is a Feature?

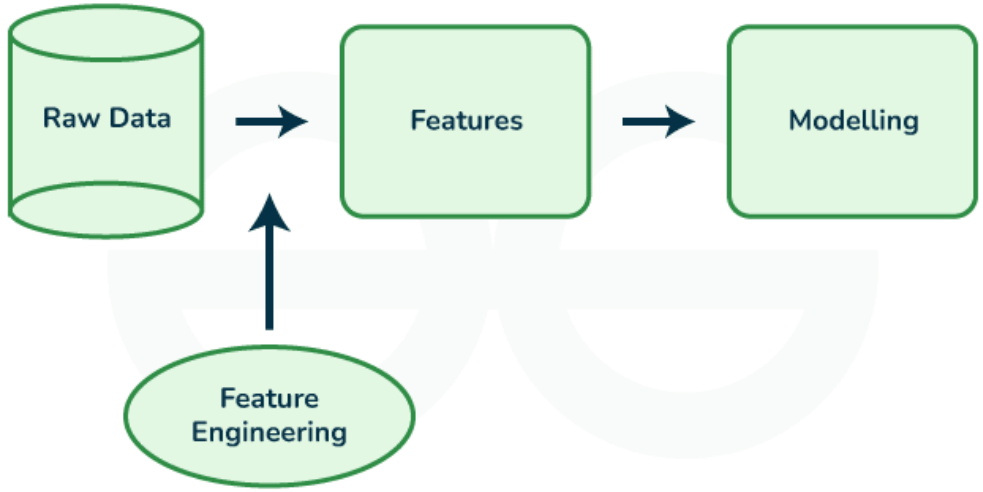
- In the context of machine learning, a feature (also known as a variable or attribute) is an individual measurable property or characteristic of a data point that is used as input for a machine learning algorithm.
- Features can be numerical, categorical, or text-based, and they represent different aspects of the data that are relevant to the problem at hand.
- For example, in a dataset of housing prices, features could include the number of bedrooms, the square footage, the location, and the age of the property.
- The choice and quality of features are critical in machine learning, as they can greatly impact the accuracy and performance of the model.



Source: https://livebook.manning.com/concept/machine-learning/input-feature

Feature Engineering

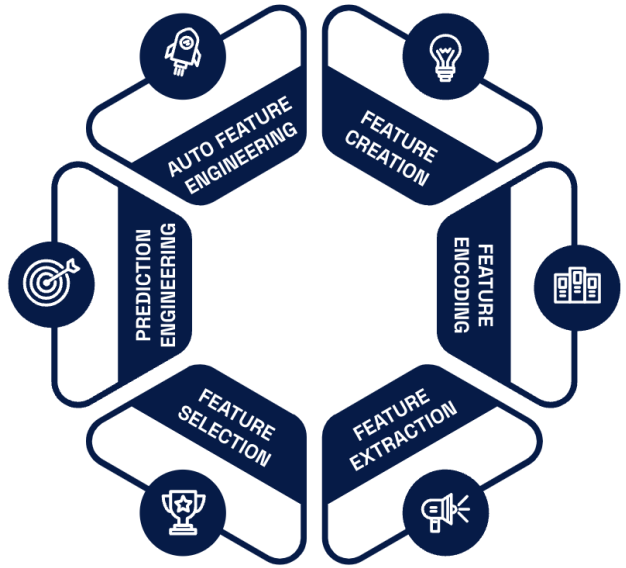
- Feature Engineering is the process of creating new features or transforming existing features to improve the performance of a machine-learning model.
- It involves selecting relevant information from raw data and transforming it into a format that can be easily understood by a model.
- The success of machine learning models heavily depends on the quality of the features used to train them.
- Feature engineering involves a set of techniques that enable us to create new features by combining or transforming the existing ones.



Source: https://medium.com/@CertifiedAuthur/lets-discuss-feature-engineering-d2bf610e51db

Common Feature Engineering Techniques

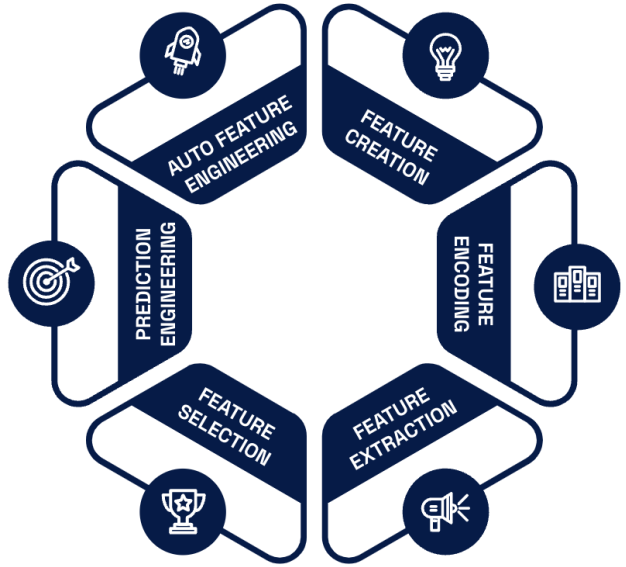
- **Imputation:** Dealing with missing values by filling them in with some value. Common methods include mean, median, or mode imputation, or more complex methods like predictive imputation.
- **Normalization/Scaling:** Ensuring that all features have similar scales. Techniques like Min-Max scaling or standardization (z-score normalization) are often used.
- **One-Hot Encoding:** Converting categorical variables into binary vectors. Each category becomes a new feature with a value of 1 or 0.
- **Label Encoding:** Converting categorical variables into numerical format, assigning each category a unique integer.
- **Feature Scaling:** Rescaling features to ensure they have similar scales. This is important for algorithms that are sensitive to feature scales, like gradient descent-based methods.
- **Binning/Discretization:** Grouping numerical variables into bins or categories. This can help to handle outliers and noise.



Source: https://medium.com/@jordan_volz/solving-customer-churn-on-the-modern-data-stack-607127cbc33a

Common Feature Engineering Techniques

- **Feature Interactions:** Creating new features based on interactions between existing ones, e.g., multiplying or dividing two features.
- **Feature Selection:** Choosing the most relevant features to include in the model.
- **Dimensionality Reduction:** Techniques like Principal Component Analysis (PCA) or t-distributed Stochastic Neighbor Embedding (t-SNE to reduce the number of features while preserving important informatio
- **Text Features:** For natural language processing, this involves techniques like tokenization, stemming/lemmatization, and TF-IDF (Ter Frequency-Inverse Document Frequency).
- **Feature Engineering from Domain Knowledge:** Utilizing your understanding of the domain to create new features that might be relevant but not explicitly present in the data.



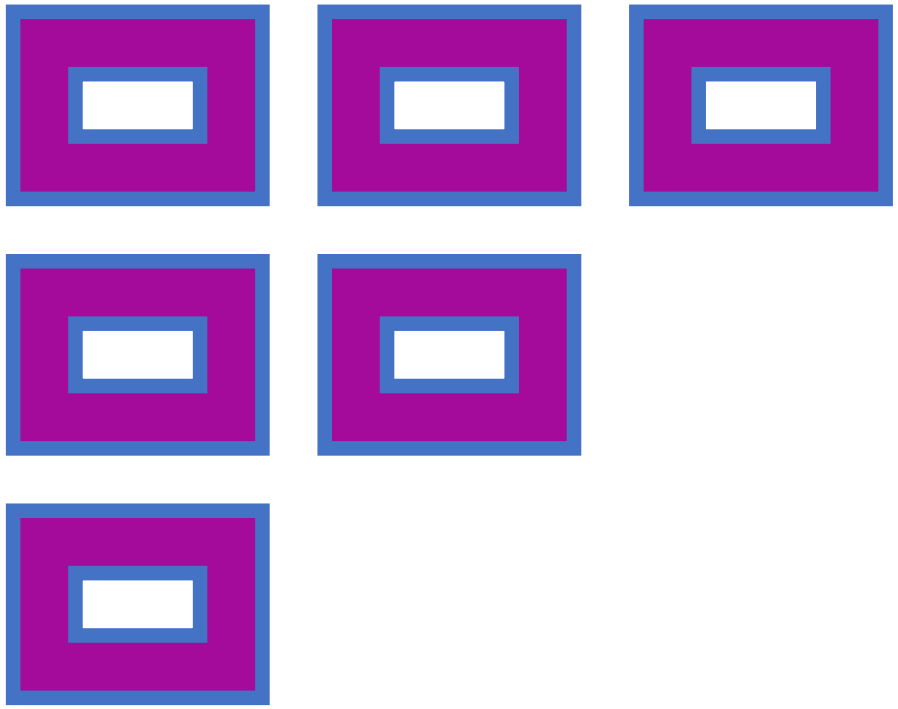
Source: https://medium.com/@jordan_volz/solving-customer-churn-on-the-modern-data-stack-607127cbc33a

Feature Creation

- Feature Creation is the process of generating new features based on domain knowledge or by observing patterns in the data. It is a form of feature engineering that can significantly improve the performance of a machine-learning model.

Types of Feature Creation:

- **Domain-Specific:** Creating new features based on domain knowledge, such as creating features based on business rules or industry standards.
- **Data-Driven:** Creating new features by observing patterns in the data, such as calculating aggregations or creating interaction features.
- **Synthetic:** Generating new features by combining existing features or synthesizing new data points.



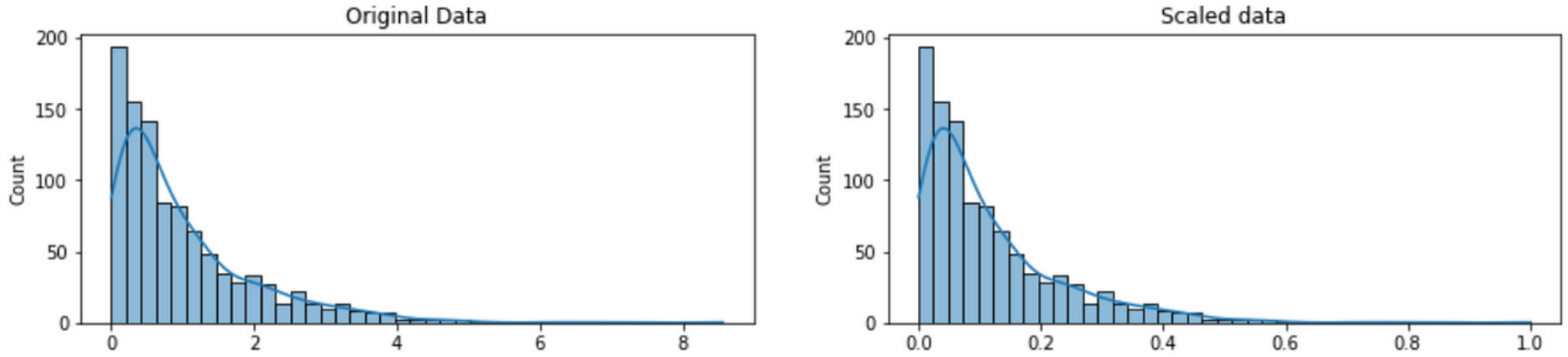
Why Feature Creation?

- **Improves Model Performance:** By providing additional and more relevant information to the model, feature creation can increase the accuracy and precision of the model.
- **Increases Model Robustness:** By adding additional features, the model can become more robust to outliers and other anomalies.
- **Improves Model Interpretability:** By creating new features, it can be easier to understand the model's predictions.
- **Increases Model Flexibility:** By adding new features, the model can be made more flexible to handle different types of data.

Normalization/Scaling

Types of Feature Scaling:

- Min-Max Scaling: Rescaling the features to a specific range, such as between 0 and 1, by subtracting the minimum value and dividing by the range.
- Standard Scaling: Rescaling the features to have a mean of 0 and a standard deviation of 1 by subtracting the mean and dividing by the standard deviation.
- Robust Scaling: Rescaling the features to be robust to outliers by dividing them by the interquartile range.



Source: https://medium.com/geekculture/scaling-vs-normalization-are-they-the-same-348035afe5ca

Lab Exercise

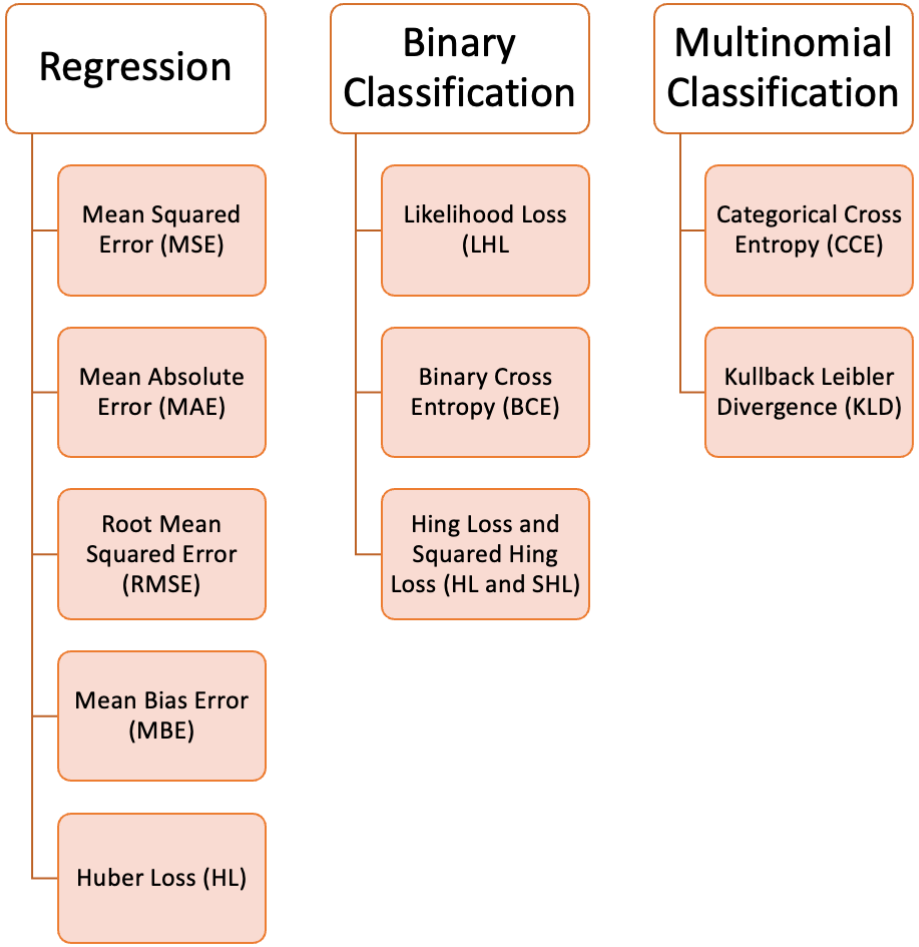
Hands on

Refer: Lab 1

Use a sample dataset to demonstrate different techniques of feature engineering.

Loss Functions

- Loss functions, also known as cost functions or objective functions, are essential components in machine learning algorithms. They measure how well a model performs by quantifying the difference between predicted values and actual values. The goal is to minimize this difference during model training.
- The loss functions depends on the type of problem in the hand. The loss functions for regression problems and classification problems are different.



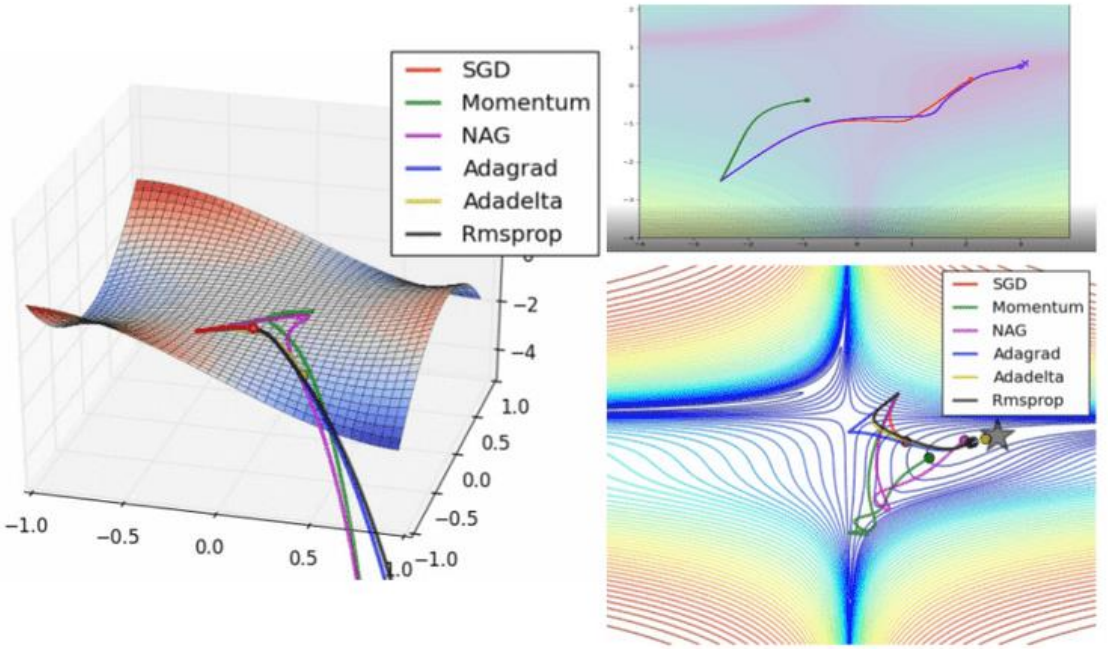
Source: https://uk.linkedin.com/posts/mustafa-gomleksiz_activity-6969996018918551552-MtJI

Optimization Algorithms

- Optimization algorithms play a crucial role in training machine learning models by iteratively updating model parameters to minimize a chosen loss function.

Common algorithms:

- Gradient Descent
- Momentum
- Adaptive Learning Rate Methods
- Adaptive Learning Rate with Second Order Methods
- Regularization



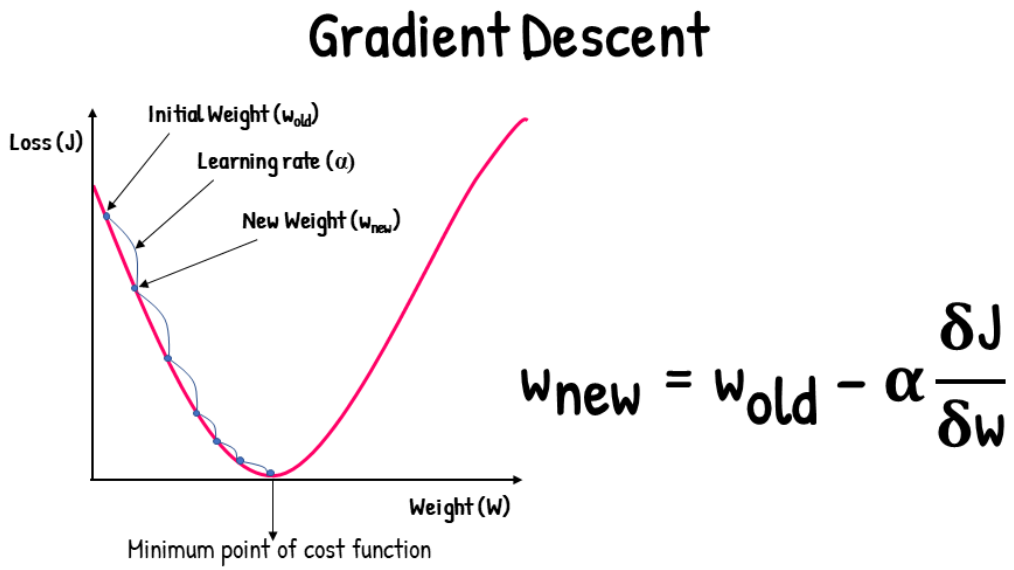
Source: https://medium.com/@fraidoonomarzai99/common-optimization-algorithms-989092830ec3

Gradient Descent

Gradient descent is a fundamental optimization algorithm used to minimize the loss function of a machine learning model. It iteratively updates the parameters of the model in the opposite direction of the gradient of the loss function, aiming to reach the minimum of the loss function.

Types of Gradient descent:

- Batch Gradient Descent
- Stochastic Gradient Descent (SGD)
- Mini-batch Gradient Descent



Source: https://dev.to/ctrlaltvictoria/ai-log-3-what-is-gradient-descent-4a0k

Batch Gradient Descent

Computes the gradient of the loss function with respect to the parameters for the entire training dataset. It updates parameters in the opposite direction of the gradient.

Advantages:

- **Convergence to the Global Minimum:** Batch gradient descent computes the gradient over the entire dataset, ensuring a more accurate estimation of the gradient direction. This can lead to convergence to the global minimum for convex optimization problems.
- **Stable Convergence:** Batch gradient descent typically has a smoother convergence trajectory compared to stochastic gradient descent because it considers the entire dataset for each update.

Disadvantages:

- **Computational Cost:** Computing gradients over the entire dataset can be computationally expensive, especially for large datasets. This makes batch gradient descent slower compared to its stochastic counterpart.
- **Memory Requirements:** Since batch gradient descent requires storing the entire dataset in memory for computing gradients, it may not be feasible for very large datasets that cannot fit into memory.

Stochastic Gradient Descent (SGD)

Computes the gradient and updates parameters for each training example individually. It can be less computationally expensive but more noisy compared to batch gradient descent.

Advantages:

- **Faster Updates:** Since stochastic gradient descent updates parameters after each training example, it converges faster in terms of iterations. This is especially advantageous for large datasets as updates are made more frequently.
- **Less Memory Usage:** SGD requires only a single training example to compute gradients, leading to significantly lower memory requirements compared to batch gradient descent.

Disadvantages:

- **High Variance in Updates:** The updates in SGD can be noisy due to the frequent fluctuation in the gradient estimates. This can lead to high variance in the convergence trajectory, making it less stable compared to batch gradient descent.
- **Less Accurate Gradient Estimates:** Due to the use of a single training example for computing gradients, SGD may provide less accurate gradient estimates compared to batch gradient descent. This can lead to slower convergence, especially near the minimum.

Mini-batch Gradient Descent

Computes the gradient and updates parameters using a small batch of training examples. It combines advantages of both batch and stochastic gradient descent.

Advantages:

- **Balanced Approach:** Mini-batch gradient descent combines the advantages of both batch and stochastic gradient descent. It offers a balance between the stability of batch gradient descent and the efficiency of stochastic gradient descent.
- **Efficient Use of Memory:** Mini-batch gradient descent processes a small batch of training examples at a time, making it more memory-efficient than batch gradient descent for large datasets.

Disadvantages:

- **Hyperparameter Sensitivity:** Mini-batch gradient descent introduces an additional hyperparameter—the batch size. Choosing an inappropriate batch size can lead to suboptimal convergence or increased computational overhead.
- **Parameter Tuning:** Like SGD, mini-batch gradient descent requires tuning of the learning rate and other hyperparameters. This can be more complex due to the additional hyperparameters introduced by the batch size.

Machine Learning Algorithms

Machine learning algorithms are a fundamental part of artificial intelligence, designed to enable computers to learn from data and improve their performance over time without being explicitly programmed. These algorithms are used in a wide range of applications, from spam detection in emails to self-driving cars.

Some common machine learning algorithms:

Supervised Learning Algorithms:

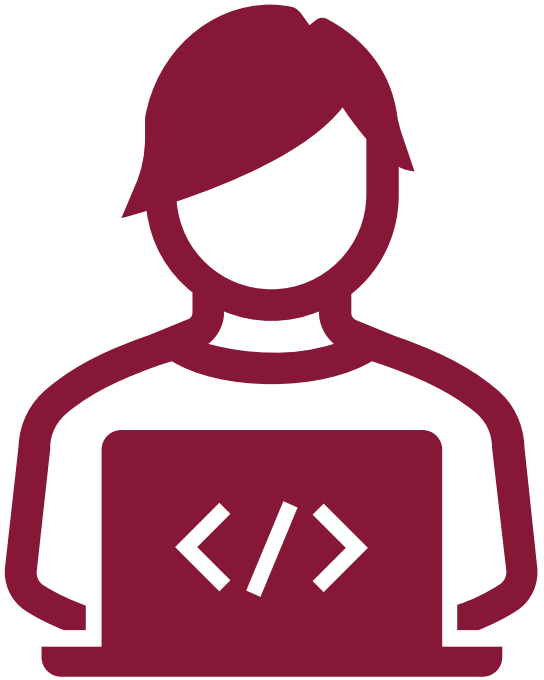
- Linear Regression
- Logistic Regression
- Support Vector Machines (SVM)
- Decision Trees

Unsupervised Learning Algorithms:

- K-means Clustering
- Hierarchical Clustering
- Principal Component Analysis (PCA)

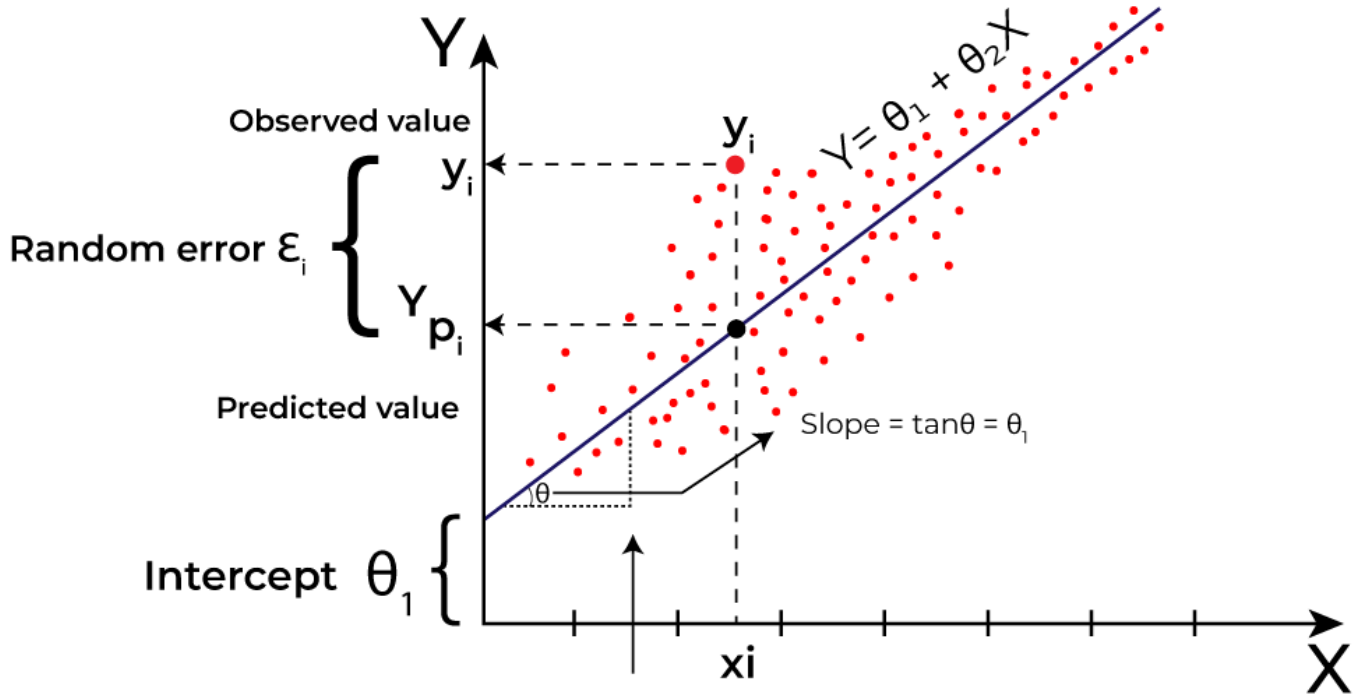
Reinforcement Learning Algorithms:

- Q-Learning
- Deep Q-Networks (DQN)



Regression

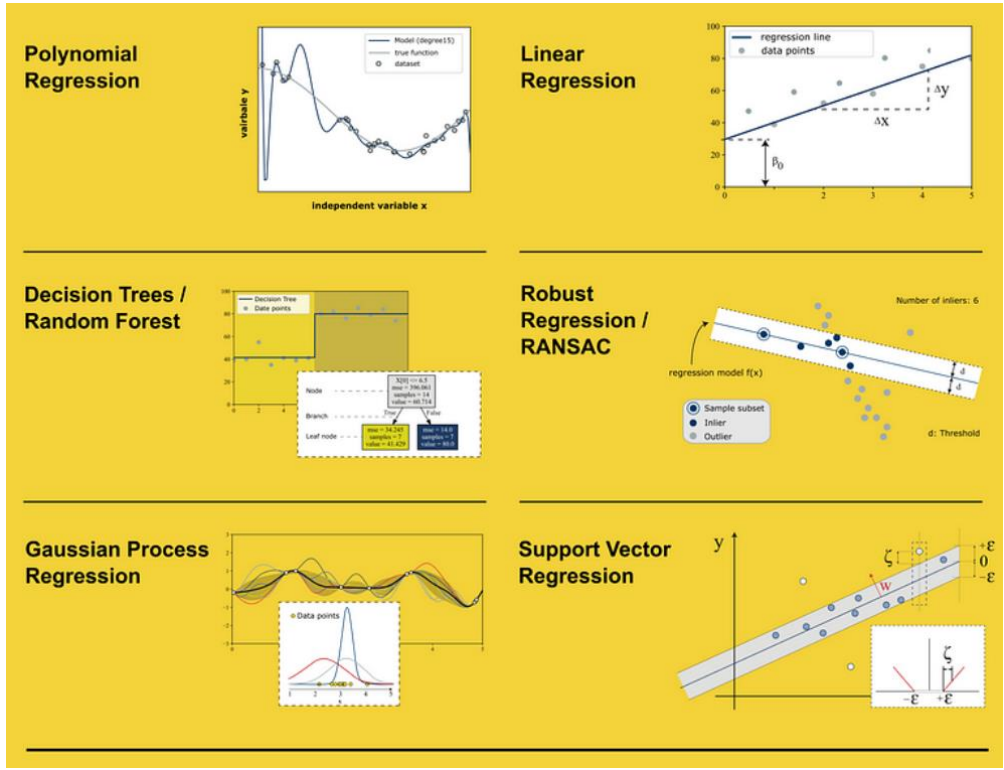
Regression is a type of supervised learning used to predict continuous values based on input data. It is widely used in various fields such as finance, economics, engineering, and science. Here's an overview of regression.



Source: https://www.geeksforgeeks.org/ml-linear-regression/

Regression Algorithms

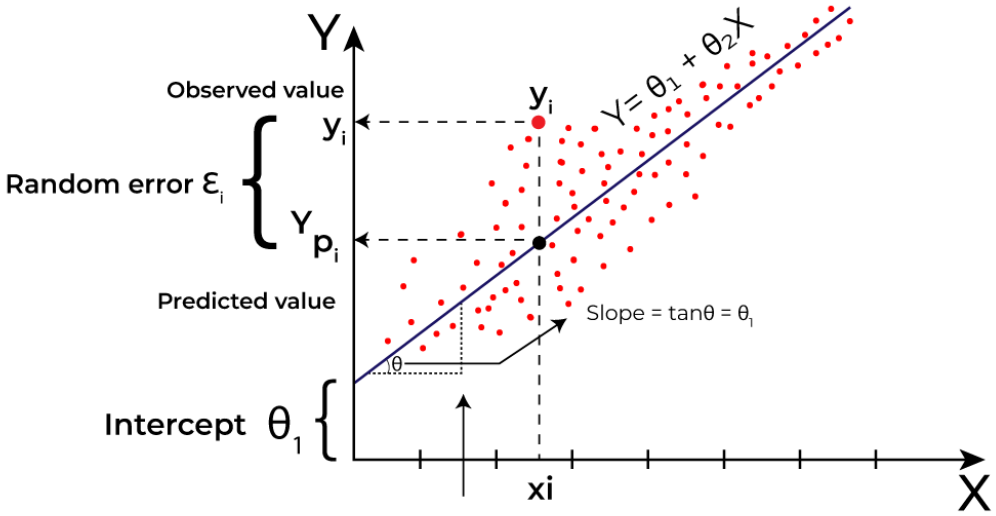
- **Linear Regression:** Assumes a linear relationship between the input variables and the output variable.
- **Polynomial Regression:** Extends linear regression by fitting a polynomial function to the data.
- **Ridge Regression (L2 regularization):** Adds a penalty term to linear regression to prevent overfitting.
- **Lasso Regression (L1 regularization):** Similar to ridge regression but uses the absolute value of coefficients, leading to feature selection.
- **Support Vector Regression (SVR):** An extension of support vector machines (SVM) for regression tasks.
- **Decision Tree Regression:** Uses decision trees to partition the feature space and make predictions.
- **Random Forest Regression:** Ensemble of decision trees where predictions are averaged over multiple trees.



Source: https://towardsdatascience.com/7-of-the-most-commonly-used-regression-algorithms-and-how-to-choose-the-right-one-fc3c8890f9e3

Linear Regression

- Linear regression is a type of supervised machine learning algorithm that computes the linear relationship between the dependent variable and one or more independent features by fitting a linear equation to observed data.
- When there is only one independent feature, it is known as Simple Linear Regression, and when there are more than one feature, it is known as Multiple Linear Regression.
- It assumes a linear relationship between the predictors and the target variable.



Source: https://www.geeksforgeeks.org/ml-linear-regression/

Why Linear Regression is Important?

Linear regression is easy to understand, which is one of its best qualities. Its equation gives clear numbers that show how much each independent thing affects the outcome. This helps us understand how everything works together. It's simple, easy to use, and it's the basis for more complicated methods.

Types of Linear Regression

There are two main types of linear regression:

Simple Linear Regression

This is the simplest form of linear regression, and it involves only one independent variable and one dependent variable. The equation for simple linear regression is:

$$Y = \beta_0 + \beta_1 X$$

Multiple Linear Regression

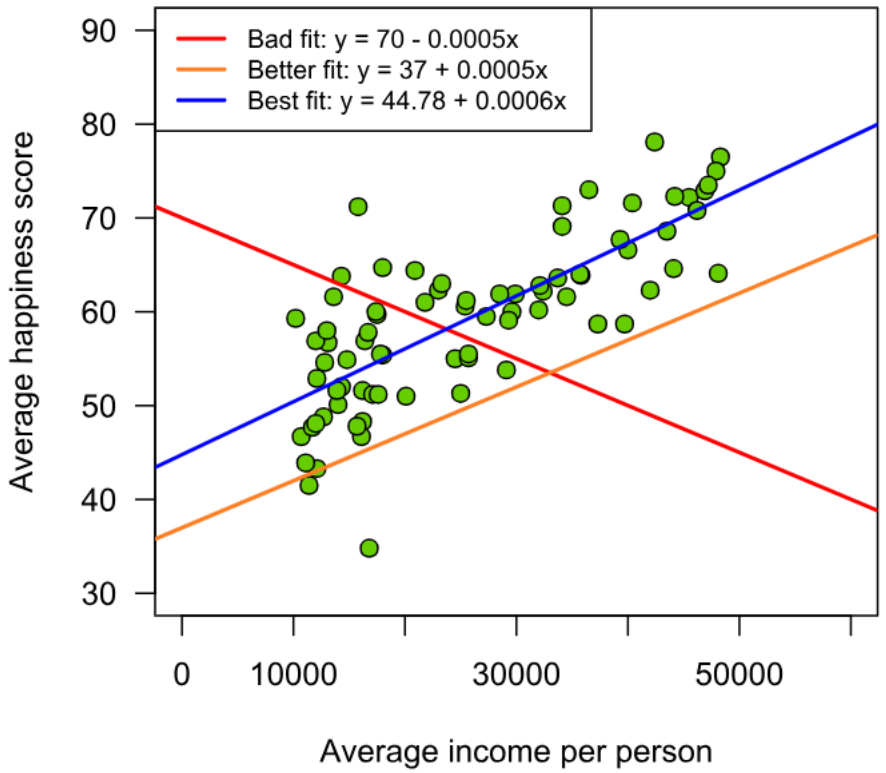
This involves more than one independent variable and one dependent variable. The equation for multiple linear regression is:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_p X_p$$

- Here betas are the coefficients.
- The goal of the algorithm is to find the best Fit Line equation that can predict the values based on the independent variables.

What is the best Fit Line?

- Our primary objective while using linear regression is to locate the best-fit line, which implies that the error between the predicted and actual values should be kept to a minimum. There will be the least error in the best-fit line.
- The best Fit Line equation provides a straight line that represents the relationship between the dependent and independent variables. The slope of the line indicates how much the dependent variable changes for a unit change in the independent variable(s).



Source: https://aiml.com/what-is-linear-regression/

How The Model Learns The Coefficients To Get The Best Fit?

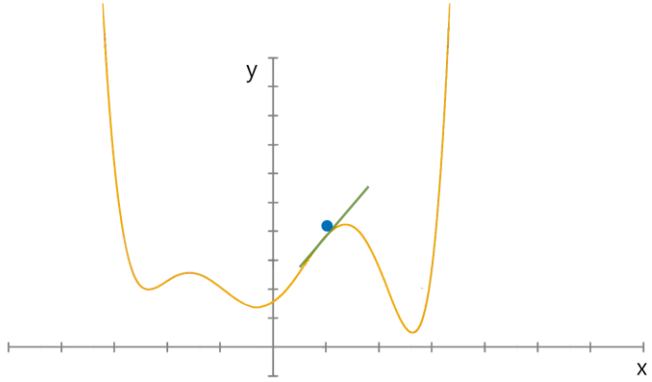
- To achieve the best-fit regression line, the model aims to predict the target value such that the error difference between the predicted value and the true value is minimum. So, it is very important to update the coefficients, to reach the best value that minimizes the error between the predicted value and the true value.
- In Linear Regression, the Mean Squared Error (MSE) cost function is employed, which calculates the average of the squared errors between the predicted values and the actual values.
- Utilizing the MSE function, the iterative process of gradient descent is applied to update the values of coefficients.
- This ensures that the MSE value converges to the global minima, signifying the most accurate fit of the linear regression line to the dataset.

Gradient Descent Algorithm

- Gradient descent is a method used in machine learning to minimize the cost (or error) function of a model by adjusting its parameters iteratively. It's commonly used in training machine learning models, particularly in cases where the cost function is not directly solvable.

Here's how gradient descent works:

- Initialization: Start with initial values for the parameters of the model.
- Compute Gradient: Calculate the gradient (or derivative) of the cost function with respect to each parameter. The gradient indicates the direction of the steepest increase of the function.
- Update Parameters: Adjust the parameters in the direction that minimizes the cost function. This adjustment is made using the gradient and a step size called the learning rate.
- Repeat: Continue steps 2 and 3 until the algorithm converges to a minimum of the cost function or reaches a predefined number of iterations.



Source: https://adatis.co.uk/introduction-to-artificial-neural-networks-part-two-gradient-descent-backpropagation-supervised-unsupervised-learning/

Evaluation Metrics for Regression

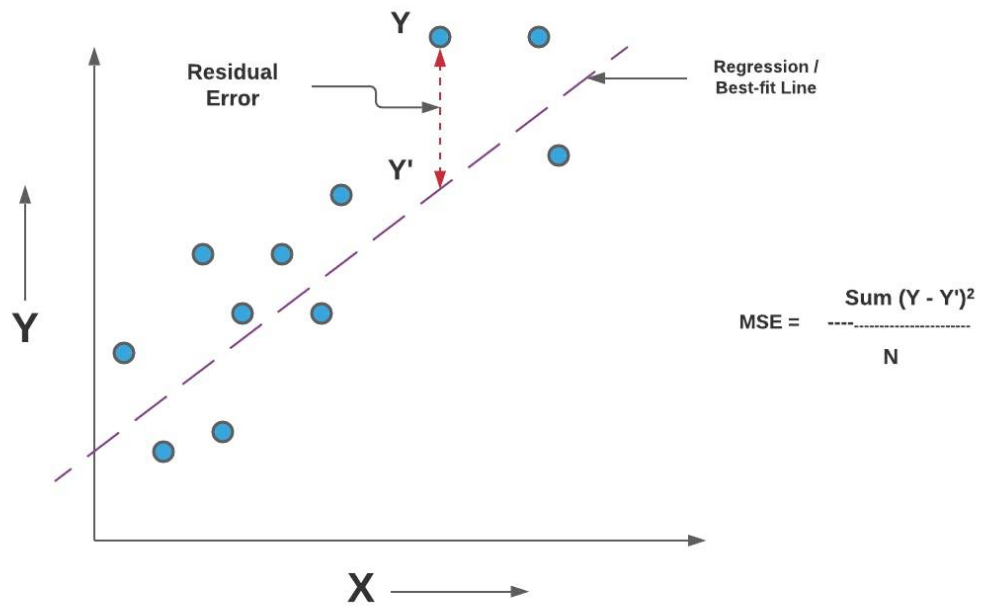
- Evaluation metrics for regression models are used to measure the accuracy and performance of the model in predicting continuous values.
- **Mean Squared Error (MSE):** Measures the average squared difference between predicted and actual values.
- **R-squared (R^2) Score:** Indicates the proportion of the variance in the dependent variable that is predictable from the independent variables.
- **Mean Absolute Error (MAE):** Measures the average absolute difference between predicted and actual values.

Mean Squared Error (MSE)

- Mean Squared Error (MSE) is a widely used metric in statistics and machine learning to measure the average squared difference between the actual (observed) values and the values predicted by a model. It's a way of quantifying how close the predictions are to the actual values.
- The formula for calculating MSE is:

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

- Here, n is the number of data points, y_i is the actual (observed) value of the ith data point and y_i^ is the predicted value of the ith data point.
- A lower MSE indicates better agreement between the predicted and actual values, while a higher MSE means the predictions are further away from the actual values.



Source: https://vitalflux.com/mean-square-error-r-squared-which-one-to-use/

R-squared (R^2) Score

- The coefficient of determination, commonly known as (R-squared), is a statistical measure that represents the proportion of the variance in the dependent variable that is predictable from the independent variables in a regression model. It's used to assess how well the independent variables explain the variability of the dependent variable.
- The formula for calculating (R-squared) is: $$R^2 = 1 - \frac{SS_{RES}}{SS_{TOT}} = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \overline{y})^2}$$

- Here, n is the number of data points, y_i is the actual (observed) value of the ith data point and y_i^ is the predicted value of the ith data point.
- (R-squared) score ranges from 0 to 1, where: (R-squared)=0 indicates that the model does not explain any of the variability of the response data around its mean.(R-squared) =1 indicates that the model perfectly explains the variability of the response data around its mean.
- (R-squared) is a useful metric for evaluating the goodness of fit of a regression model, but it has limitations. For instance, it can increase when more variables are added to the model, even if those variables are not relevant. Adjusted (R-squared) is often used to address this issue, as it penalizes for adding unnecessary variables.

Mean Absolute Error (MAE)

- Mean Absolute Error (MAE) is another commonly used metric to evaluate the performance of a regression model. It measures the average absolute difference between the actual (observed) values and the predicted values.
- The formula for calculating MAE is: $$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

- Here, n is the number of data points, y_i is the actual (observed) value of the ith data point and y_i^ is the predicted value of the ith data point.
- MAE doesn't penalize large errors as heavily as MSE. This means it may not fully capture the impact of large errors on model performance. Nevertheless, it provides a straightforward measure of the average magnitude of errors in predictions.
- When comparing models, a lower MAE indicates better performance, as it signifies smaller average prediction errors. MAE is particularly useful when the distribution of errors is not Gaussian (normal), or when you want to give more importance to smaller errors and be less influenced by outliers.

Applications

- Predicting house prices based on features like size, location, and number of bedrooms.
- Forecasting sales revenue based on advertising expenditure and other factors.
- Estimating the impact of temperature, humidity, and other weather variables on crop yields.
- Predicting stock prices based on historical data and market indicators.

Lab Exercise - Simplest Linear Regression Model Demonstration.

Hands On

Refer: Lab 2

Train a very basic linear regression model using a sample data.

Lab Exercise - Train a linear regression model for car price prediction without using feature engineering techniques

Hands On

Refer: Lab 3

Use car price data from Kaggle to train a Linear Regression Model.

Lab Manual - Train a linear regression model for car price prediction also using feature engineering techniques

Hands On

Refer: Lab 4

Use car price data from Kaggle to train a Linear Regression Model.

Conclusion

Congratulations! You have completed this course and now have a thorough understanding of several crucial aspects of machine learning. You have learned about feature engineering and representation learning, the roles of loss functions and optimization algorithms, the importance of training and testing data, and the concepts of overfitting and generalization. Additionally, you have been introduced to various machine learning algorithms, with a particular focus on linear regression.

Quiz

1. **What is feature engineering?**

A. The process of collecting data

B. The process of using domain knowledge to create features from raw data

C. The process of validating a model

D. The process of deploying a model

Answer: B
The process of using domain knowledge to create features from raw data

Quiz

2. What is the purpose of a loss function in machine learning?

A) To evaluate the performance of a model

B) To collect data

C) To deploy a model

D) To preprocess data

Answer: A
To evaluate the performance of a model

Quiz

3. Which of the following is an example of a supervised learning algorithm?

A) K-means Clustering

B) Linear Regression

C) Principal Component Analysis (PCA)

D) DBSCAN

Answer: B
Linear Regression

Quiz

4. What is the main difference between supervised and unsupervised learning?

A. Supervised learning requires labeled data, whereas unsupervised learning does not

B. Unsupervised learning requires labeled data, whereas supervised learning does not

C. Supervised learning is used for clustering, whereas unsupervised learning is used for regression

D. There is no difference between supervised and unsupervised learning

Answer: A
Supervised learning requires labeled data, whereas unsupervised learning does not

Quiz

5. What is the goal of linear regression?

A. To classify data into categories

B. To predict a continuous target variable based on one or more predictor variables

C. To cluster data into groups

D. To reduce the dimensionality of data

Answer: B
To predict a continuous target variable based on one or more predictor variables

References

- https://www.datacamp.com/blog/what-is-feature-learning
- https://www.geeksforgeeks.org/what-is-feature-engineering/
- https://medium.com/data-science-group-iitr/loss-functions-and-optimization-algorithms-demystified-bb92daff331c
- https://medium.com/geekculture/loss-functions-and-optimizers-in-ml-models-b125871ff0dc
- https://machinelearningmastery.com/overfitting-and-underfitting-with-machine-learning-algorithms/
- https://www.geeksforgeeks.org/introduction-machine-learning/
- https://www.geeksforgeeks.org/ml-linear-regression/

edunet
foundation

Thank You!