ELEC8900-57 SPECIAL TOPICS: MACHINE LEARNING

FALL - 2023

LECTURE CONTENT

2

Evaluation Metrics

Evaluation Metrics

Confusion Matrix

- A confusion matrix is a table that summarizes the performance of a classification model.
- It tabulates the counts of true positives, false positives, true negatives, and false negatives.
- It is useful for understanding the types of errors made by the model and assessing its performance across different classes.

Evaluation Metrics

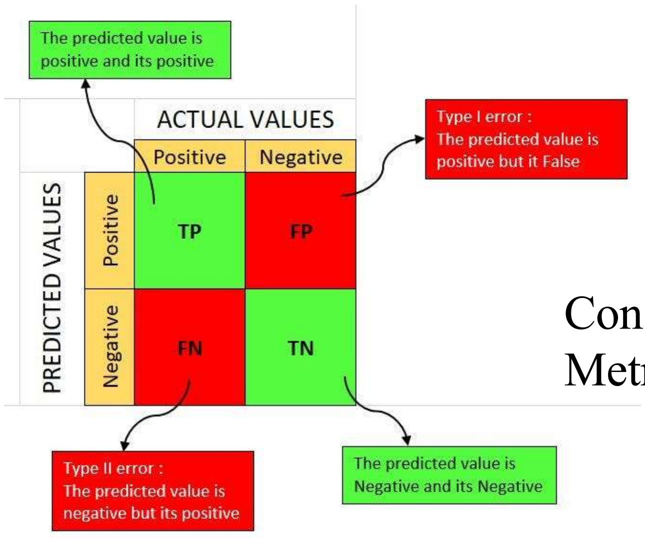
Confusion Metrix

The confusion matrix is typically organized with the following side headings:

- True Positives (TP): Instances that are actually positive and are correctly predicted as positive.
- False Positives (FP): Instances that are actually negative but are incorrectly predicted as positive.
- True Negatives (TN): Instances that are actually negative and are correctly predicted as negative.
- False Negatives (FN): Instances that are actually positive but are incorrectly predicted as negative.

5

Evaluation Metrics


The predicted value is positive and its positive
ACTUAL VALUES
Positive
Negative
Type I error : The predicted value is positive but it False
PREDICTED VALUES
Positive
Negative
TP
FP
FN
TN
Type II error : The predicted value is negative but its positive
The predicted value is Negative and its Negative

Confusion Metrix

[Source: https://medium.com/analytics-vidhya/what-is-a-confusion-matrix-d1c0f8feda5]

6

Evaluation Metrics

Accuracy

- Accuracy is the most basic evaluation metric for classification models.
- It represents the proportion of correct predictions out of the total predictions made.
- It is calculated as the ratio of the number of correctly classified instances to the total number of instances.
- However, accuracy alone may not provide a complete picture, especially when the classes are imbalanced.

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+TN+FN}$$

7

Evaluation Metrics

Precision

- Precision measures the proportion of correctly predicted positive instances out of all predicted positive instances.
- It focuses on the accuracy of positive predictions.
- Precision is calculated as the ratio of true positives (correctly predicted positives) to the sum of true positives and false positives (incorrectly predicted positives).

$$\text{Precision} = \frac{TP}{TP+FP}$$

8

Evaluation Metrics

Recall or Sensitivity or True Positive Rate (TPR)

- Recall measures the proportion of correctly predicted positive instances out of all actual positive instances.
- It focuses on the model's ability to find all positive instances.
- Recall is calculated as the ratio of true positives to the sum of true positives and false negatives (missed positives).

$$\text{Recall (Sensitivity)} = \frac{TP}{TP + FN}$$

9

Evaluation Metrics

F1 Score

- The F1 score is a metric that combines precision and recall into a single value.
- It is calculated as the harmonic mean of precision and recall.
- The F1 score is useful when the class distribution is imbalanced, meaning that one class has significantly more samples than the other.
- It provides a balanced evaluation of the model's performance, taking both precision and recall into account.

$$\text{F1 Score} = \frac{2 * (Precision * Recall)}{(Precision + Recall)}$$

10

Evaluation Metrics

Specificity – True Negative Rate (TNR)

- Specificity is a metric that measures the proportion of correctly predicted negative instances out of all actual negative instances.
- It focuses on the model's ability to identify negative instances.
- Specificity is calculated as the ratio of true negatives (correctly predicted negatives) to the sum of true negatives and false positives (incorrectly predicted negatives).

$$\text{Specificity (True Negative Rate)} = \frac{TN}{TN+FP}$$

11

Evaluation Metrics

Area Under Curve of the Receiver Operating Characteristic (AUC-ROC)

- The ROC curve is a graphical representation of the model's performance at different classification thresholds.
- AUC-ROC represents the area under the ROC curve, which provides an aggregate measure of the model's discrimination ability.
- A higher AUC-ROC indicates better classification performance.

12

Evaluation Metrics

Area Under Curve of the Receiver Operating Characteristic (AUC-ROC)

To find (AUC-ROC)

- Sort the predicted probabilities or scores in descending order.
- Calculate the True Positive Rate (TPR) and False Positive Rate (FPR) for each threshold. FPR = FP/(FP + TN)
- TPR is the ratio of true positives to the total number of actual positives, and FPR is the
- ratio of false positives to the total number of actual negatives.
- Plot the points on a graph with TPR on the y-axis and FPR on the x-axis.
- Calculate the area under the curve using a suitable numerical integration method, such as
- the trapezoidal rule or Simpson's rule

13

EXAMPLE – EVALUATION METRICS

14

EXAMPLE - AUC-ROC

15

Python Libraries for Machine Learning

16

Libraries used in Machine Learning Python

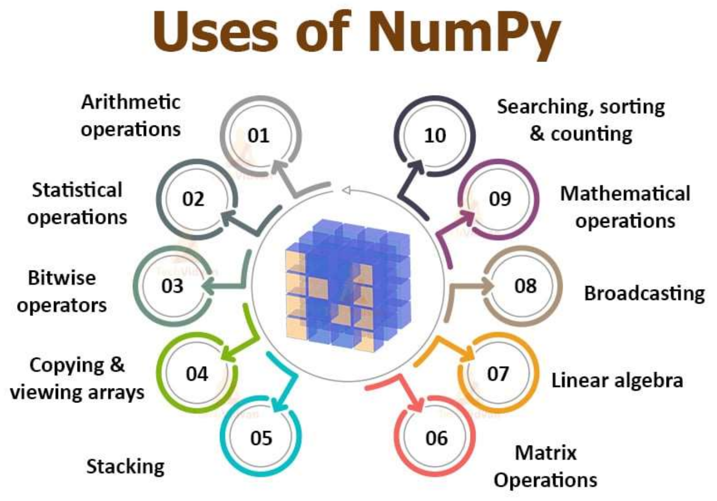
- NumPy (Numerical Python)
- Pandas
- Scikit-learn
- Matplotlib
- TensorFlow
- Keras
- PyTorch
- XGBoost
- … etc.

17

NumPy

- NumPy is a fundamental Python library for numerical computing.
- It offers a robust N-dimensional array object for efficient array manipulation.
- NumPy serves as the foundation for various libraries within the Python scientific ecosystem.
- It is extensively utilized for tasks such as data manipulation and preprocessing in the field of machine learning and data analysis.
- To import; import numpy as np

18



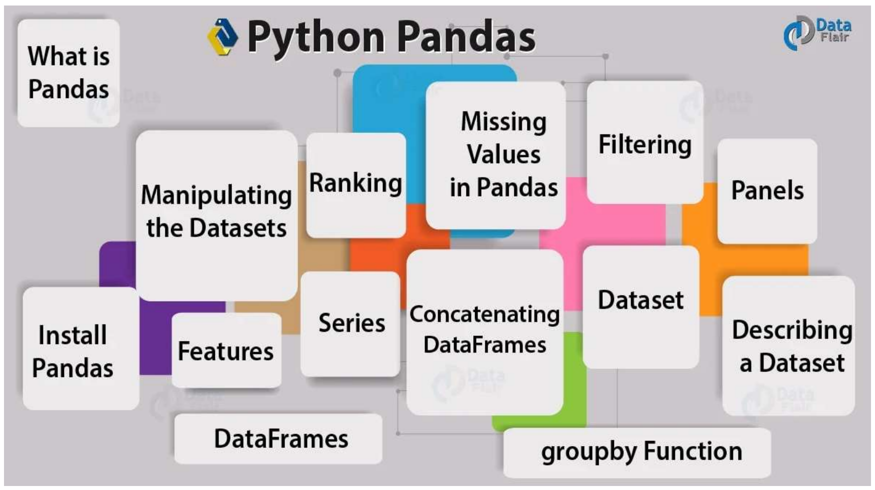
Source: [https://tinyurl.com/yuj6tyhx]

19

Pandas

- Pandas is a widely used Python library for data manipulation and analysis.
- It offers data structures like DataFrames, simplifying the handling of structured data.
- Pandas provides functions for various data-related tasks, including cleaning, transformation, and exploration.
- It plays a crucial role in data preprocessing for machine learning.
- To import Pandas use: import pandas as pd.

20



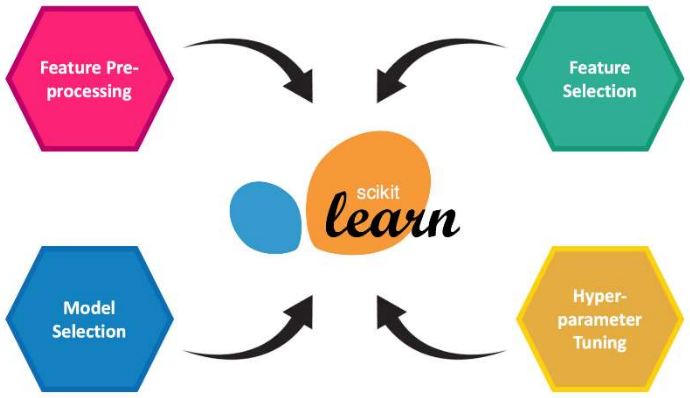
https://data-flair.training/blogs/pandas-tutorial/

21

Scikit-learn

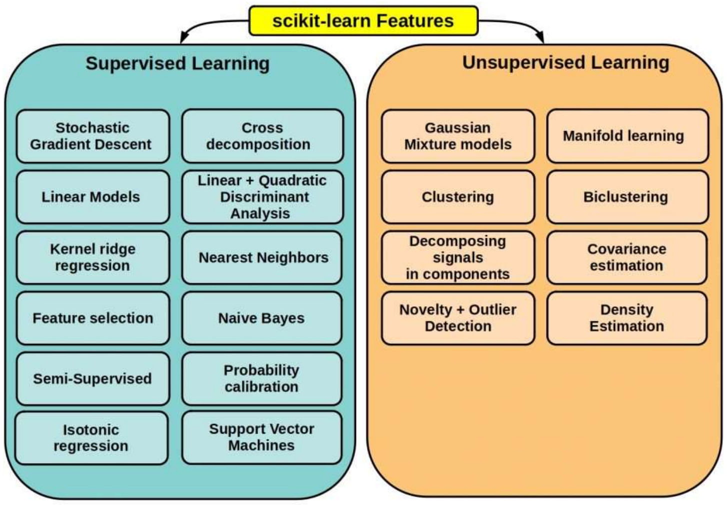
- Scikit-learn is a comprehensive Python library for machine learning.
- It offers a diverse set of algorithms and utilities for various tasks like classification, regression, clustering, and dimensionality reduction.
- Scikit-learn also provides tools for model evaluation and selection.
- It is renowned for its simplicity and beginner-friendly nature.
- To import Scikit-learn, use: import sklearn.

22



[Source: https://tinyurl.com/muvxeudh]

23



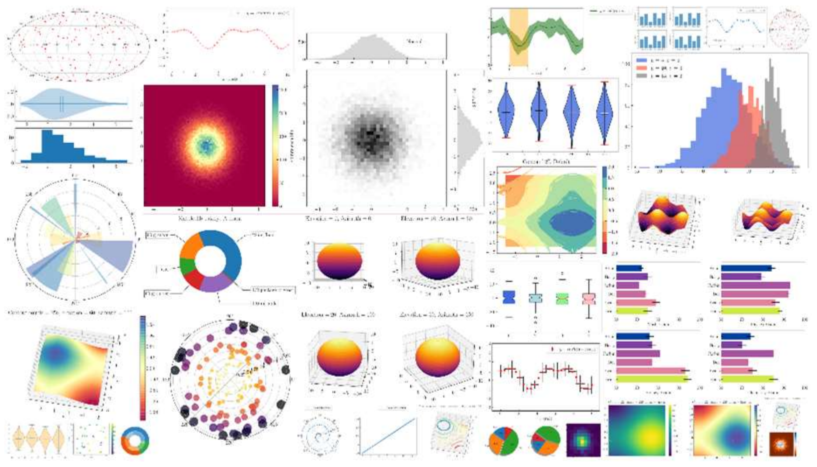
[Source: https://tinyurl.com/tpdzcchr]

24

Matplotlib

- Matplotlib is a popular Python library for creating data visualizations.
- It offers a wide range of functions for generating high-quality plots and charts, such as line plots, scatter plots, bar plots, histograms, and more.
- To import Matplotlib use: import matplotlib.pyplot as plt.

25



Source: https://miro.medium.com/v2/resize:fit:9450/1*OAFElg9w1XHyZk0xBud14A.png

26

TensorFlow

- TensorFlow is a potent open-source library developed by Google for numerical computation and machine learning.
- It provides a versatile framework for creating and deploying machine learning models, especially in the context of deep learning.
- TensorFlow includes a high-level API called Keras, designed to streamline the construction of neural networks.
- To import TensorFlow use: import tensorflow as tf.

27

TensorFlow

[Source: https://www.tensorflow.org/guide]

28

Keras

- Keras is a popular deep learning library for Python that provides a wide range of features for building and training neural networks. Here are some of the most important features of Keras:
- Originally designed as a user-friendly interface for building deep learning models on TensorFlow, Keras has become integrated into TensorFlow's core library.
- Keras offers a straightforward and intuitive interface for creating and training neural networks.
- To import Keras: import keras.

29

Source: https://keras.io/getting_started/

30

PyTorch

- PyTorch is a prominent open-source machine learning library that emphasizes dynamic computation graphs.
- It provides a flexible and efficient framework for training deep learning models.
- PyTorch offers robust support for GPU acceleration, making it suitable for high-performance computing.
- It is widely adopted in the research community and is known for its dynamic nature, enabling flexible model development.
- To import: import torch.

31

PyTorch

[Source: https://pytorch.org/docs/stable/index.html]

32

XGBoost

- XGBoost is an optimized gradient boosting library recognized for its effectiveness with tabular data.
- It is extensively employed for tasks involving classification and regression.
- XGBoost implements the gradient boosting algorithm, combining numerous weak models to form a highly accurate ensemble model.
- To import XGBoost: import xgboost as xgb.

33

XGBoost

[Source: https://xgboost.readthedocs.io/en/stable/python/python_intro.html]

34

Python Libraries

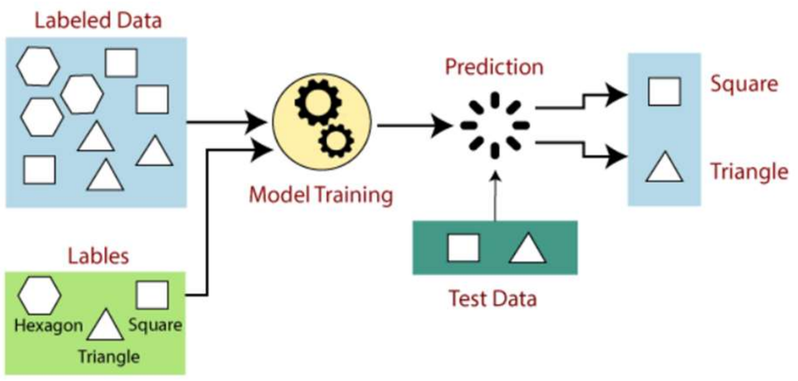
Supervised Learning

Supervised Learning

- Supervised learning is widely used in various domains, including image recognition, natural language processing, recommendation systems, medical diagnosis, and more.
- Supervised learning forms the foundation of many machine learning applications and serves as a powerful tool for solving a wide range of predictive and decision-making problems.
- In supervised learning the algorithm learns from labeled training data to make predictions or decisions without being explicitly programmed for specific tasks.
- In supervised learning, the algorithm is provided with a dataset that includes input-output pairs, and the goal is to learn a mapping from inputs to outputs so that it can generalize and make accurate predictions on new, unseen data

37

Supervised Learning



38

Supervised Learning

- Data: Data includes Input Features (X) and Output Labels or Target (y).
- Output labels can be continuous (for regression tasks) or discrete (for classification tasks).
- Training Data: The dataset is divided into two subsets: the training data and the testing data.
- The training data is used to train the machine learning model. It consists of a large portion of the dataset, and it's the data on which the model learns patterns and relationships between input features and output labels.

39

Supervised Learning

- Model: The model is the algorithm or mathematical function that represents the mapping between input features and output labels.
- The model has parameters that need to be learned from the training data.
- The choice of the model depends on the specific problem you are trying to solve.
- Examples of supervised learning models include linear regression, decision trees, support vector machines, neural networks, and more.

40

Supervised Learning

- Training: The process of training the model involves finding the best set of parameters or coefficients that minimize the difference between the predicted output and the actual output (ground truth) for the training examples. This is typically done using optimization techniques, such as gradient descent.
- Loss Function: The loss function (also known as the cost function) measures how well the model's predictions match the true output values in the training data. The goal during training is to minimize this loss function. Common loss functions include mean squared error (for regression) and cross-entropy (for classification).

41

Supervised Learning

- Learning Algorithm: The learning algorithm is responsible for adjusting the model's parameters iteratively based on the gradients of the loss function.
- Gradient descent is a common optimization technique used in supervised learning to update the model parameters in the direction that reduces the loss.
- Testing/Evaluation: Once the model is trained, it is evaluated on a separate portion of the dataset called the testing data.
- The model's performance is assessed using evaluation metrics appropriate for the task, such as accuracy, precision, recall, F1 score (for classification), or mean squared error, R-squared (for regression

42

Supervised Learning

- Generalization: The ultimate goal of supervised learning is to create a model that can generalize well to make accurate predictions on new, unseen data. Generalization is the ability of the model to apply the learned patterns to data it has never encountered before.
- Hyperparameter Tuning: In addition to learning parameters, supervised learning models often have hyperparameters that control the learning process. Hyperparameter tuning involves finding the best combination of hyperparameters to optimize model performance.
- Deployment: Once the model is trained and evaluated, it can be deployed in real-world applications to make predictions on new data.

43

Classification Algorithms

Classification Algorithms in ML:

- Logistic Regression: Despite its name, logistic regression is a classification algorithm used to model the probability of a binary outcome.
- Decision Trees: Decision trees are tree-like models that split data into branches based on features to make classification decisions.
- Random Forest: A random forest is an ensemble learning method that combines multiple decision trees to improve accuracy and reduce overfitting.
- Support Vector Machines (SVM): SVMs find a hyperplane that best separates data into different classes while maximizing the margin.
- K-Nearest Neighbors (K-NN): K-NN classifies data points based on the majority class among their k nearest neighbors.

44

Classification Algorithms

- Naive Bayes: Naive Bayes classifiers are based on Bayes' theorem and are particularly useful for text classification and spam detection.
- Gradient Boosting Algorithms:
  - Gradient Boosting Machines (GBM)
  - XGBoost
  - LightGBM
  - CatBoost: These algorithms iteratively build an ensemble of weak learners to improve predictive accuracy.
- Neural Networks: Deep learning neural networks, including convolutional neural networks (CNNs) for image classification and recurrent neural networks (RNNs) for sequence data, are used for complex classification tasks.
- Support Vector Machines (SVM): SVMs are also used for multi-class classification by extending the binary classification concept.

45

Regression Algorithms

- Linear Regression: Linear regression models the relationship between the input features and a continuous output variable.

- Ridge Regression and Lasso Regression: These are variants of linear regression that add regularization to prevent overfitting.

- Decision Trees for Regression: Decision trees can also be used for regression tasks, where they predict continuous values.

- Random Forest Regression: Similar to classification, random forests can be used for regression tasks to predict continuous values.

- Support Vector Regression (SVR): SVR extends SVM for regression by finding a hyperplane that best fits the data.

46

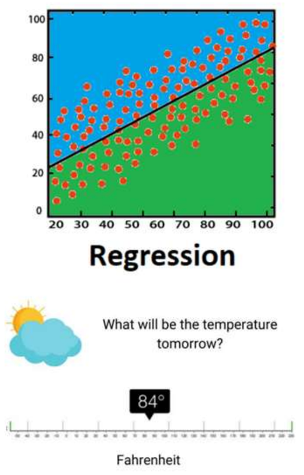
Regression Algorithms

- K-Nearest Neighbors Regression: K-NN can be adapted for regression by averaging the values of the k nearest neighbors.
- Gradient Boosting Algorithms for Regression:
  - Gradient Boosting Machines (GBM)
  - XGBoost
  - LightGBM
  - CatBoost: These algorithms can also be used for regression tasks to predict continuous values.
- Neural Networks for Regression: Deep learning neural networks can predict continuous outputs in regression tasks.
- Polynomial Regression: This is a form of linear regression where higher-degree polynomial terms are added to the model.

47

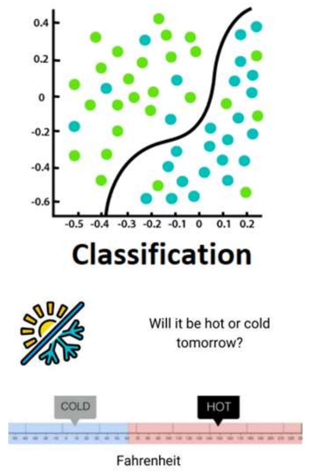
Regression

Regression algorithms are used to predict continuous numerical values based on input variables.



Classification

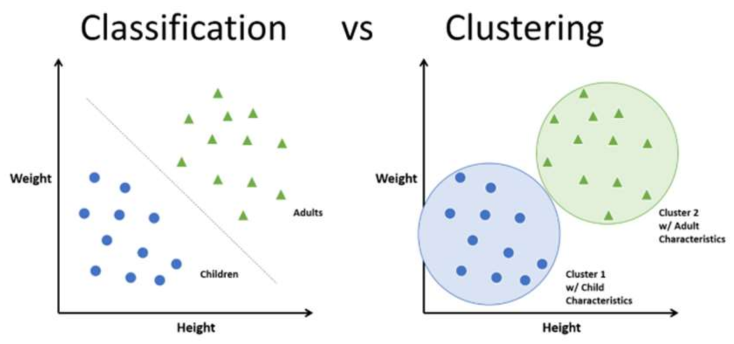
Classification algorithms are used to assign input examples to predefined categories or classes.



48

Clustering

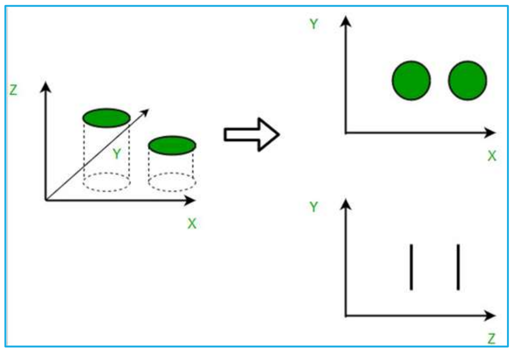
- Clustering: Clustering algorithms group similar data points together based on their characteristics.


Classification vs Clustering
Weight
Height
Adults
Children
Weight
Height
Cluster 2 w/ Adult Characteristics
Cluster 1 w/ Child Characteristics

49

Dimensionality Reduction

- Transforming data from a high-dimensional space into a low-dimensional space while preserving meaningful properties of the original data.
- Commonly used in fields that deal with large numbers of observations/variables, such as signal processing, speech recognition, …
- Dimensionality reduction can be achieved through linear or nonlinear approaches, as well as feature selection or feature extraction.

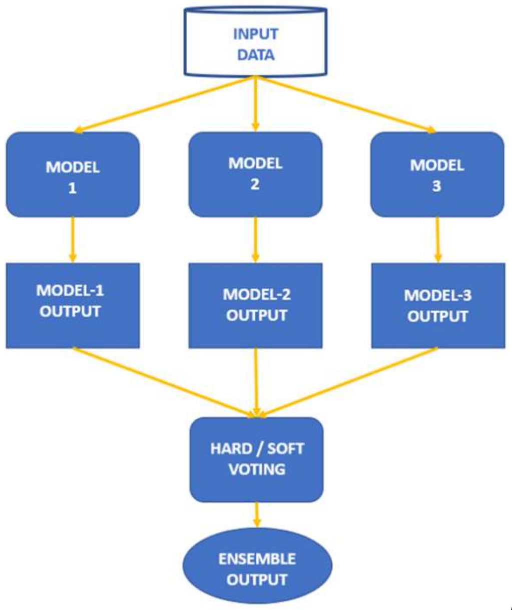


[Source: https://www.geeksforgeeks.org/dimensionality-reduction/]

50

Ensemble Learning

- Ensemble Learning: Ensemble learning combines multiple ML models to make predictions or decisions. It improves accuracy, reduces overfitting, and includes techniques like bagging, boosting, and stacking.



51

Solving Machine Learning Problems

52

Theoretical Approach to Solving ML Problems

- Define the problem
- Gather and explore the data
- Split the dataset
- Feature engineering
- Select an appropriate algorithm
- Train the model
- Test the model
- Evaluate the model
- Iterate and improve
- Deploy the model

53

Programming Approach to Solving ML Problems

- Import required libraries
- Import required datasets
- Check for any null values
- Assign depended and independent variables
- Split the data into training and testing datasets
- Feature scaling
- Fit the ML model
- Compute and visualize the confusion matrix
- Compute the accuracy, precision, recall and F1 score (Evaluation metrics).

54

Linear Regression Example

Linear Regression Example

- Sample problem of predicting home price.

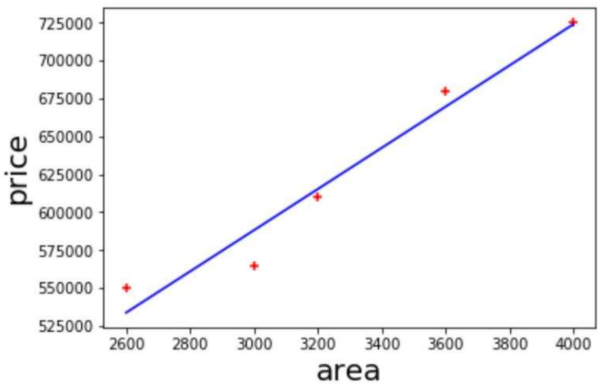
Area	Price
2600	550000
3000	565000
3200	610000
3600	680000
4000	725000

- Problem Statement: Given above data build a machine learning model that can predict home prices based on square feet area

[Source of example: https://github.com/codebasics/py/blob/master/ML/1_linear_reg/1_linear_regression.ipynb]

Linear Regression Example

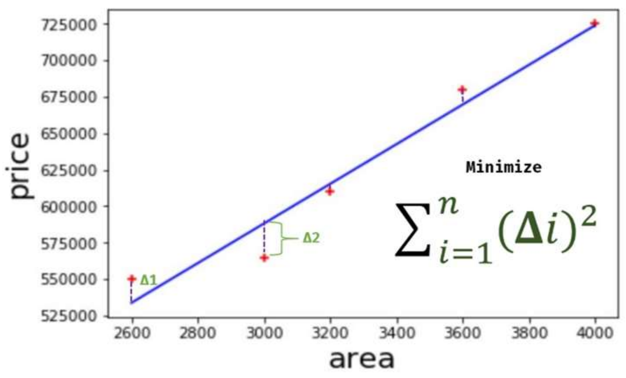
- Plot for data from table.
- The straight line best fits values on chart.



57

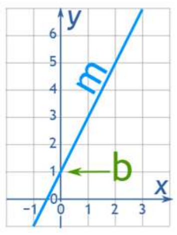
Linear Regression Example

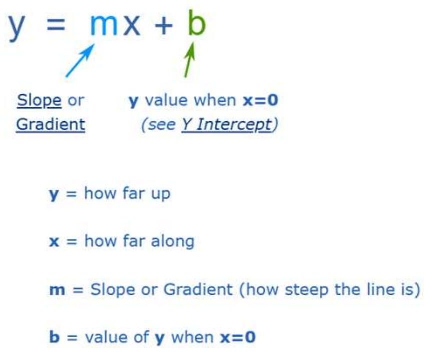
- You can draw multiple lines like this but we choose the one where total sum of error is minimum



58

Linear Regression Example





y = mx + b

Slope or Gradient

y value when **x=0** *(see Y Intercept)*

y = how far up

x = how far along

m = Slope or Gradient (how steep the line is)

b = value of **y** when **x=0**

[Source: https://www.mathsisfun.com/equation_of_line.html]

Linear Regression Example Programming Solution

https://github.com/codebasics/py/blob/master/ML/1_linear_reg/1_linear_regression.ipynb

https://www.youtube.com/watch?v=8jazNUpO3lQ

QUESTIONS

61