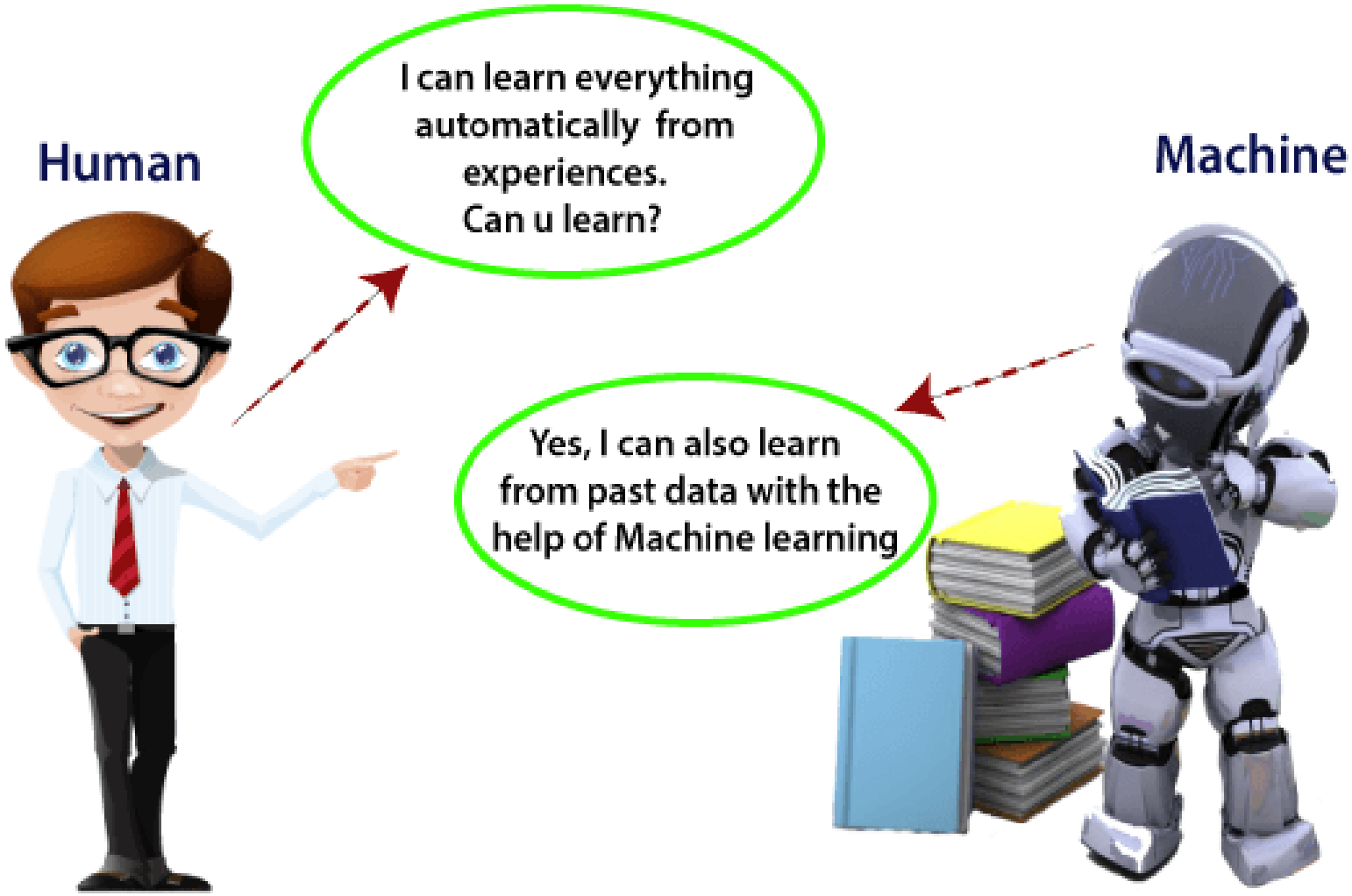
# Introduction to Data Science with Python

Week 09: Supervised Learning

# Learning Objectives

- Types of Machine Learning
- Classification and regression problems
- Classification Algorithms
- Logistic regression
- Metrics for Model Evaluation: Accuracies, confusion matrix, sensitivity (recall), specificity, precision, receiver operating characteristic (ROC) curve, and area under ROC curve (AUC).
- Other terminologies:
  - Overfitting and underfitting
  - Bias and variance
  - Regularization

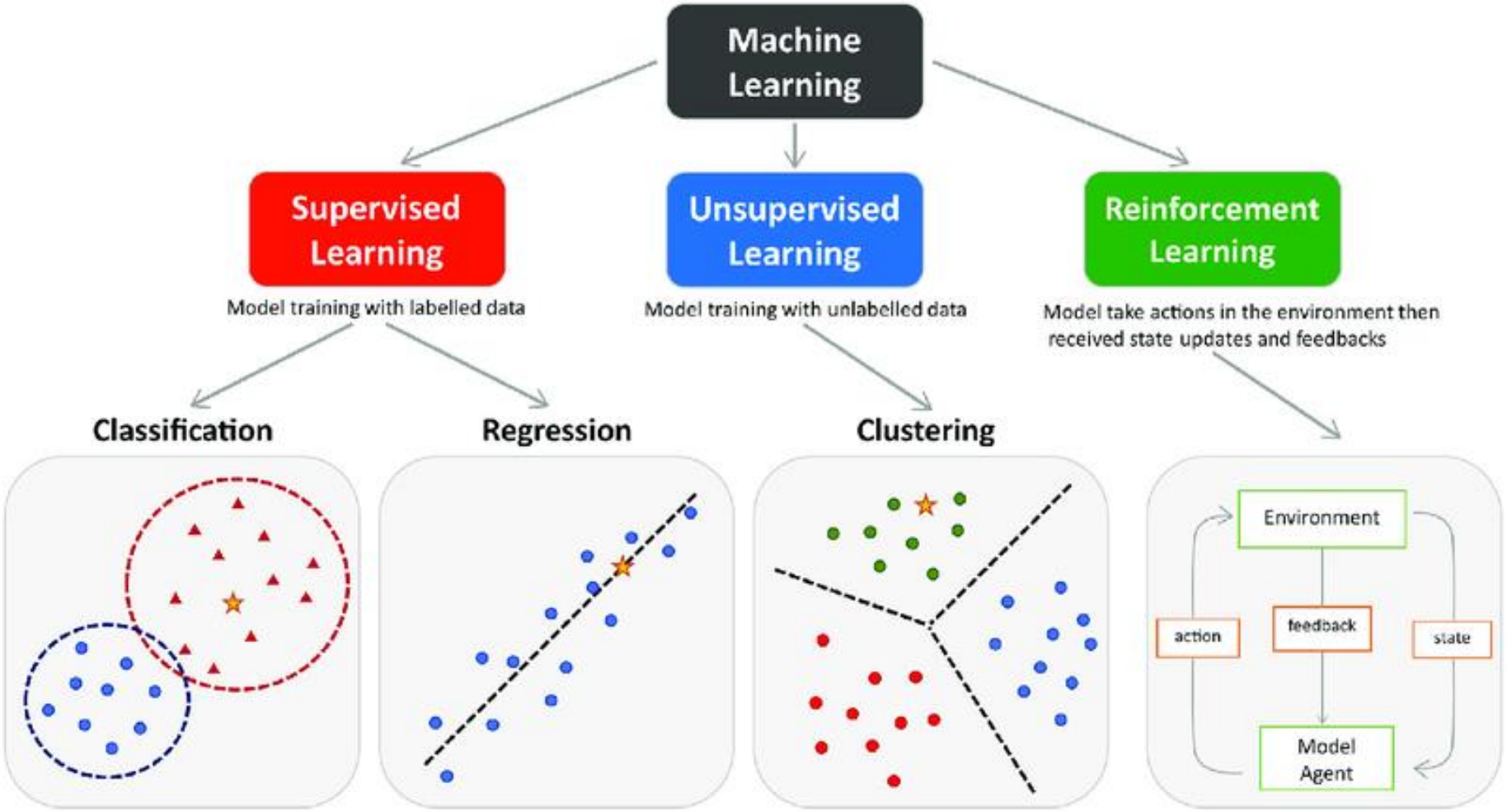
# How machine learn?



# Machine Learning

- Machine learning is **a branch of artificial intelligence (AI)** that involves the development of algorithms and statistical models that enable computer systems to improve their performance on a specific task over time, without being explicitly programmed.
- In essence, it's a way for machines to learn patterns from data and make predictions or decisions without being explicitly programmed for each scenario.

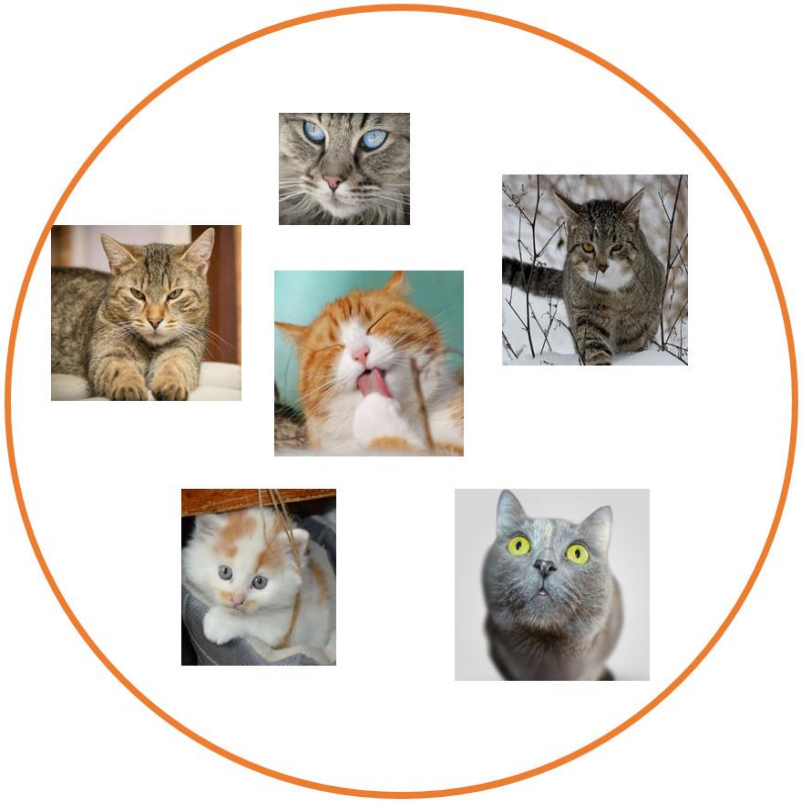
# Types of Machine Learning



# Types of Machine Learning

- Supervised learning
  - Algorithms which learn from a training set of labeled examples (exemplars) to generalize to the set of all possible inputs.
  - Examples of techniques in supervised learning: logistic regression, support vector machines, decision trees, random forest, etc.
- Unsupervised learning
  - Algorithms that learn from a training set of unlabeled examples. Used to explore data according to some statistical, geometric or similarity criterion.
  - Examples of unsupervised learning include k-means clustering and kernel density estimation.
- Reinforcement learning
  - Algorithms that learn via reinforcement from criticism that provides information on the quality of a solution, but not on how to improve it.
  - Improved solutions are achieved by iteratively exploring the solution space.

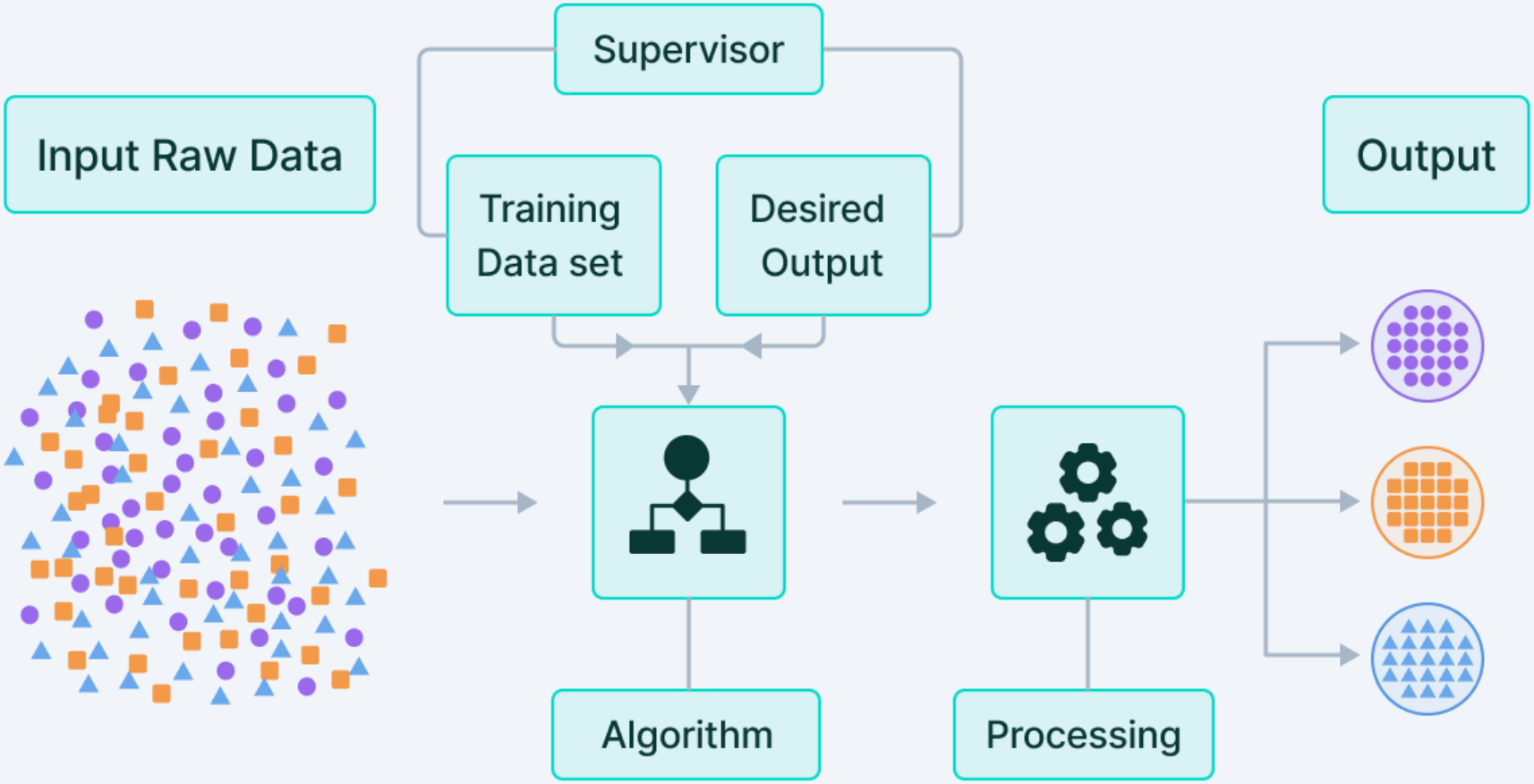
# Supervised Learning



Cat

Dog

# Supervised Learning

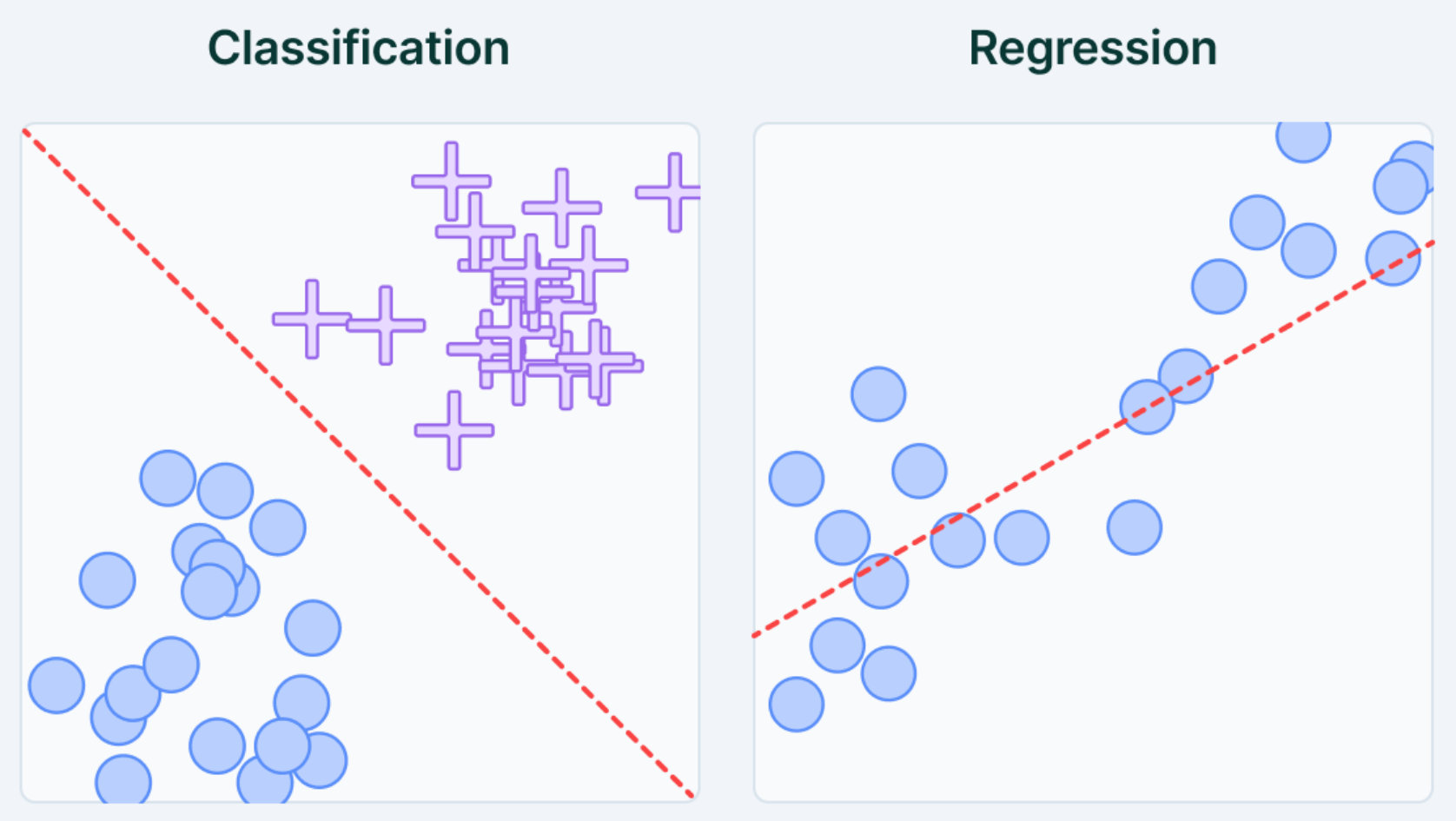



V7 Labs

# Types of Supervised Learning

- There are two main types of supervised learning: where the algorithm is trained on labeled data to either classify input into categories or predict numerical outcomes.
- **Classification:** Predicts the category or class to which a new data point belongs.
  - **Example:** Spam detection (classifying emails as spam or not spam), image recognition (identifying objects in images), sentiment analysis (classifying movie reviews as positive or negative).
- **Regression:** Predicts a numerical value or quantity based on input features.
  - **Example:** Predicting house prices, stock prices, temperature, or any other continuous variable.

# Classification vs Regression




- https://www.v7labs.com/blog/supervised-vs-unsupervised-learning.

# Example: Regression and Classification

- If our question is answered by YES/NO, we are facing a classification problem.
- Classifiers are also the tools to use if our question admits only a discrete set of answers, i.e., we want to select from a finite number of choices.
  - Given the results of a clinical test, e.g., does this patient suffer from diabetes?
  - Given a magnetic resonance image, is it a tumor shown in the image?
  - Given the past activity associated with a credit card, is the current operation fraudulent?
- If our question is a prediction of a real-valued quantity, we are faced with a *regression* problem.
  - Given the description of an apartment, what is the expected market value of the flat? What will the value be if the apartment has an elevator?
  - Given the past records of user activity on Apps, how long will a certain client be connected to our App?
  - Given my skills and marks in computer science and maths, what mark will I achieve in a data science course?

# Classification Algorithm

- There are several techniques used for solving classification problems such as
  - Logistic regression,
  - Classification trees,
  - K - Nearest Neighbours
  - Discriminant analysis,
  - Support vector machines,
  - Random Forest
  - Neural networks,

# Sigmoid Function

- The function maps any real value into another value between 0 and 1.

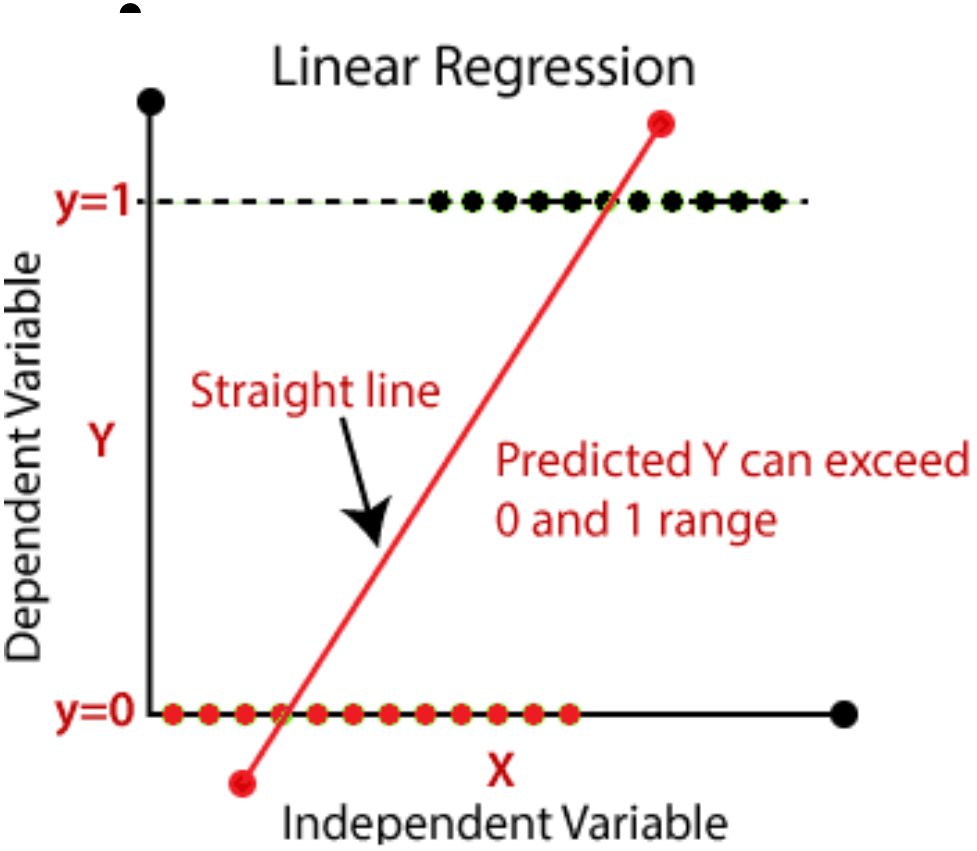
$$S(z) = \frac{1}{1 + e^{-z}}$$

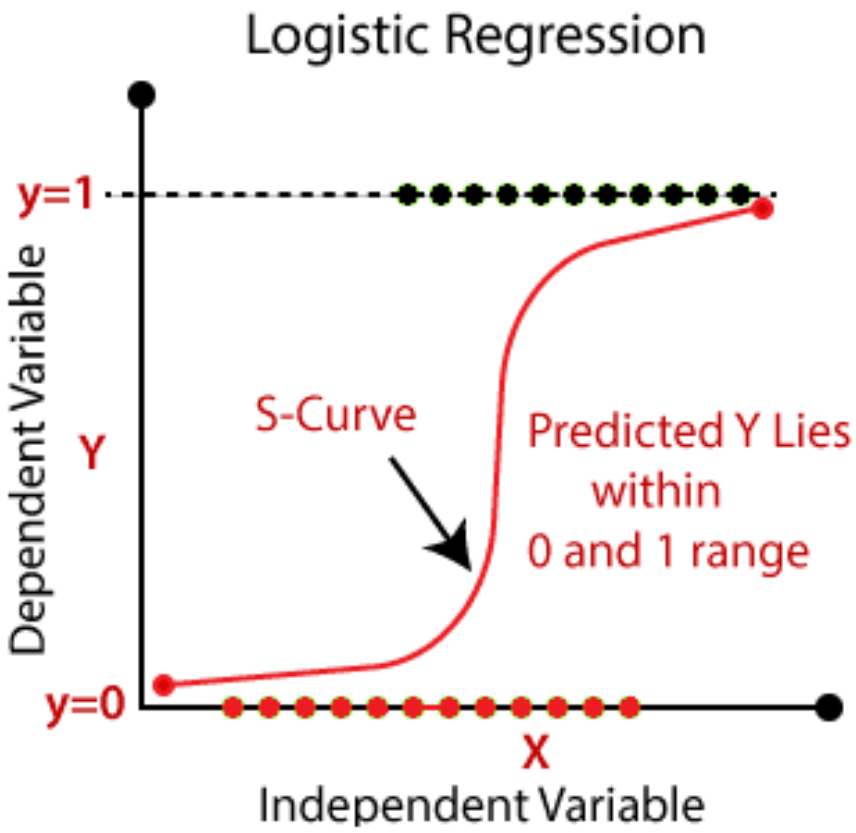
$S(z)$ = output between 0 and 1 (probability estimate)

$z$ = input to the function (e.g. mx + b)

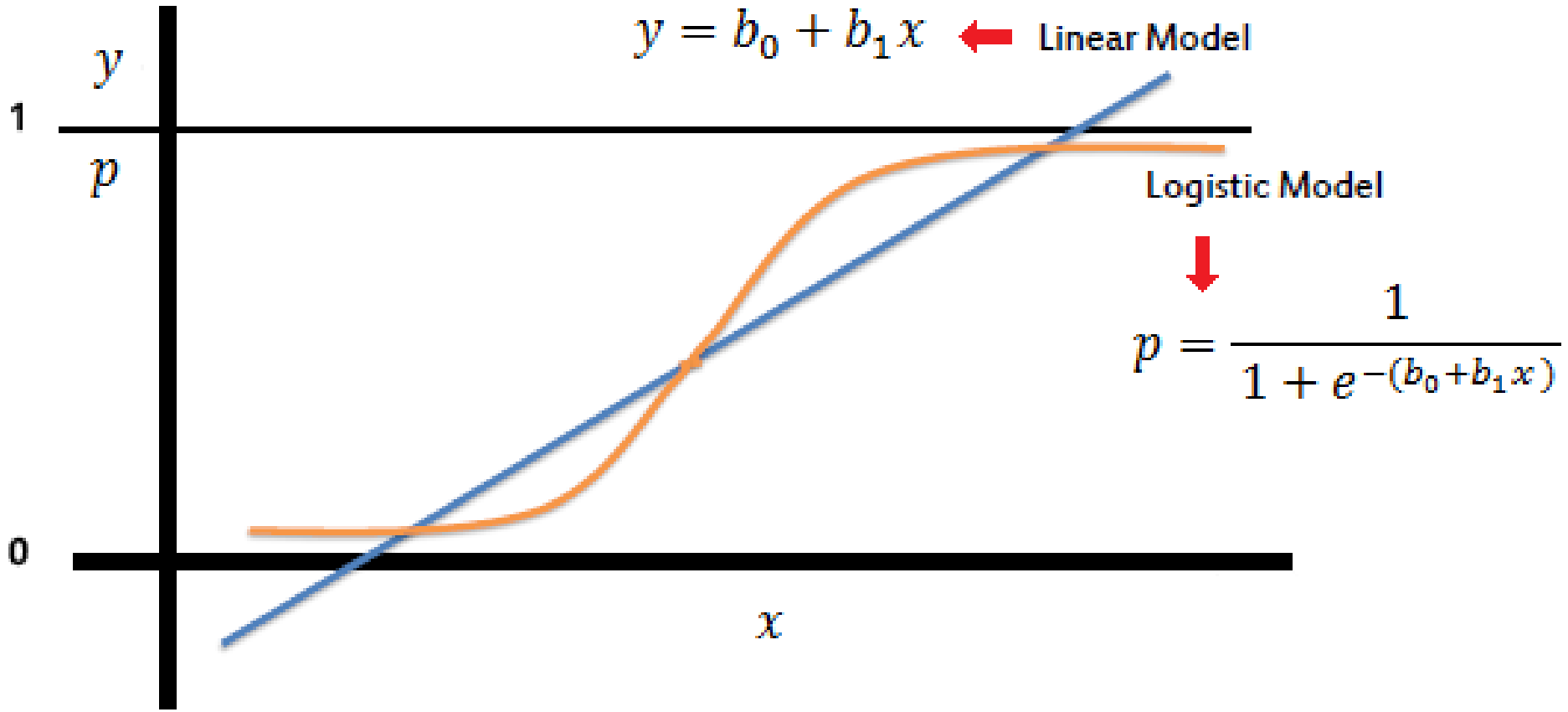
e = base of natural log

# Linear Regression vs Logistic Regression



# Linear Regression vs Logistic Regression



# Logistic Regression

- Logistic regression is used when the dependent variable is binary (i.e., it has only two possible outcomes).
- The logistic regression model is based on the logistic function (also called the sigmoid function).

$$S(z) = \frac{1}{1+e^{-z}}$$

- The logistic regression equation can be written as:
- $P(Y = 1) = \frac{1}{1+e^{-(\beta_0+\beta_1 X_1+\cdots+\beta_k X_k)}}$
- Here, $P(Y=1)$ is the probability of the positive class, $e$ is the base of the natural logarithm, $\beta_0,\beta_1,\ldots,\beta_k$ are the coefficients, and $X_1,\ldots,X_k$ are the features.
- The output of the logistic regression model is the log-odds of the probability of the positive class.

# Logistic Regression

Putting z value to sigmoid function

Linear Regression Equation

$$z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \ldots + \beta_k x_k$$

Sigmoid Function

$$p = \frac{1}{1 + e^{-z}}$$

$$p = \frac{e^z}{e^z + 1}$$

$$p = \frac{e^{\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \ldots + \beta_k x_k}}{e^{\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \ldots + \beta_k x_k} + 1}$$

Looks like very hard to solve it, so let's try to transform it into some easy to solve equation with the help of Odds ratio.

Odds Ratio $S = \frac{p}{1-p}$

Replace p and solve

$$S = e^{\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \ldots + \beta_k x_k}$$

Take log each side and solve

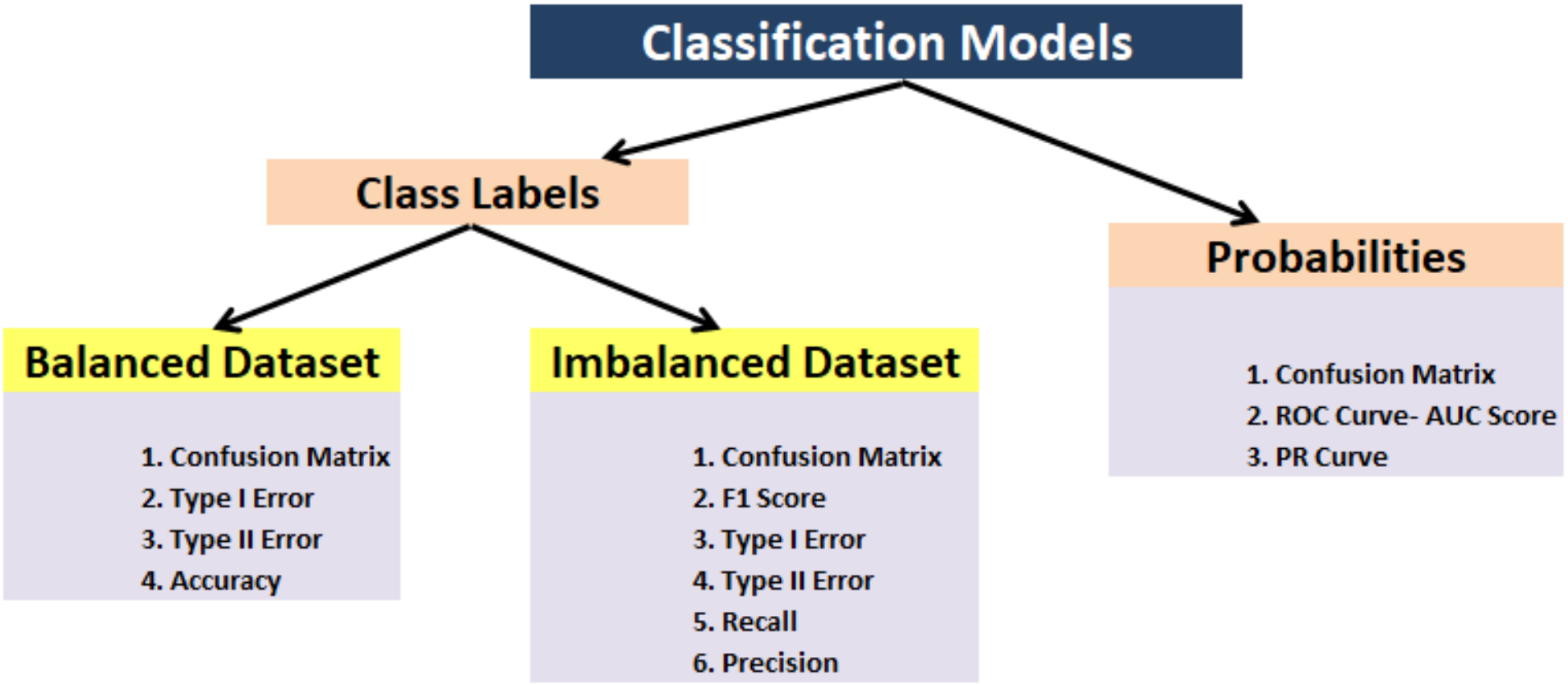
$$\mathbf{Ln(S) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \ldots + \beta_k x_k}$$

Transformed into Linear Regression

Notes:

- The log of Odds is called Logit and transformed model is linear in $\beta_s$
- So solving the logistic regression problem essentially reduces to finding the $\beta_s$ that minimizes the error.
- Now suppose with one predictor we got the Linear Regression eq. ln(s) = -20.40782+.42592*x. And now we want to classify for given x = 50 then:
- Ln(s) = -20.40782+.42592*50 = 0.89 => s = $e^{0.89}$ = 2.435
- s = $\frac{p}{1-p}$ => p = $\frac{s}{s+1}$ ⇒ $p$ = 2.435/(1+2.435) = .709
- So using a probability of 0.50 as a cut-off between predicting the two classes 1 or 0, this member would be classified as class 1 with a probability of 70%

# Model Performance Metrics



# Accuracy

- Accuracy is defined as the percentage of correct predictions out of all the observations.

$$Accuracy = \frac{Correct\ prediction}{Total\ cases} * 100\%$$

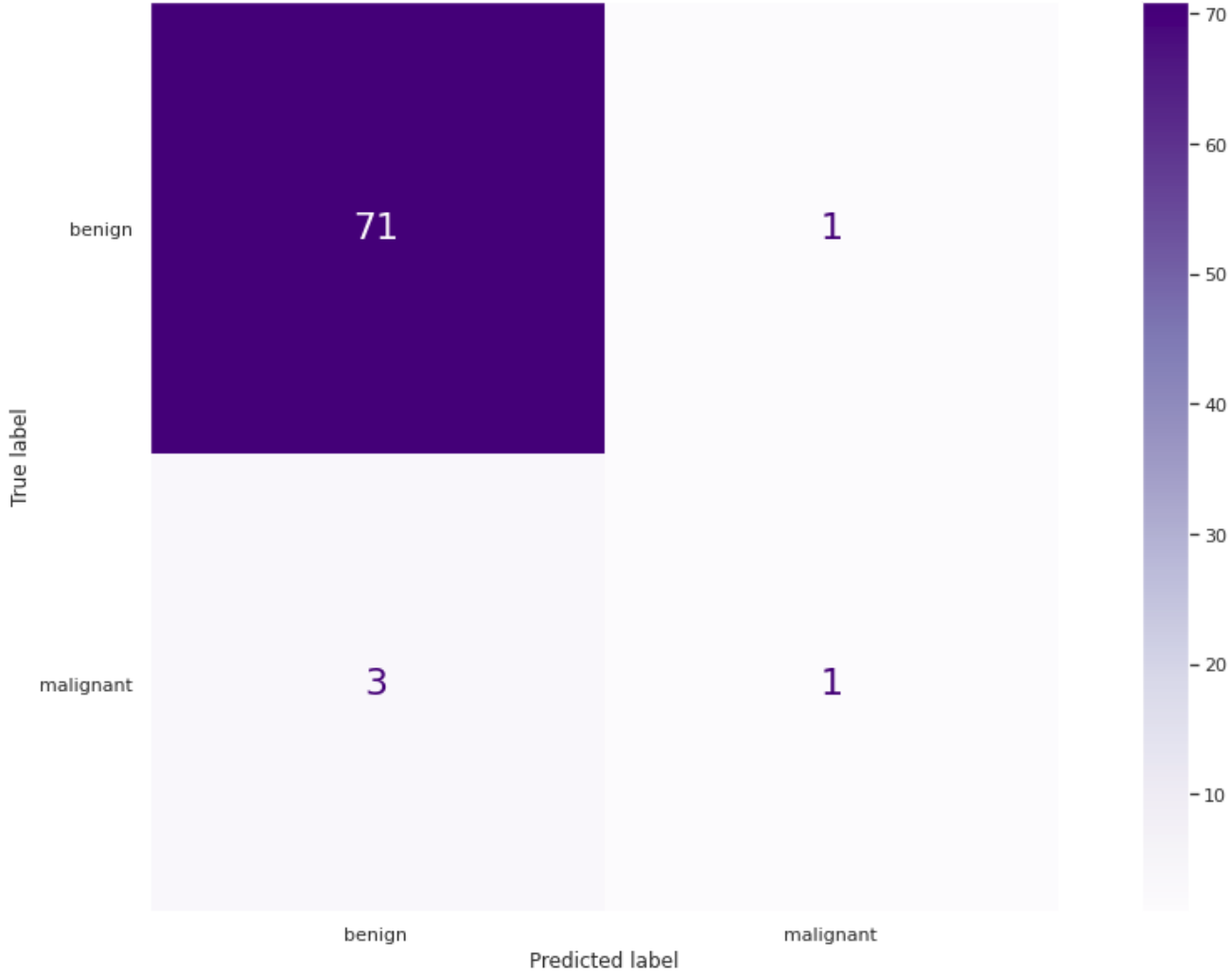
$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)} * 100\%$$

- Where TP = True Positives, TN = True Negatives, FP = False Positives, and FN = False Negatives.

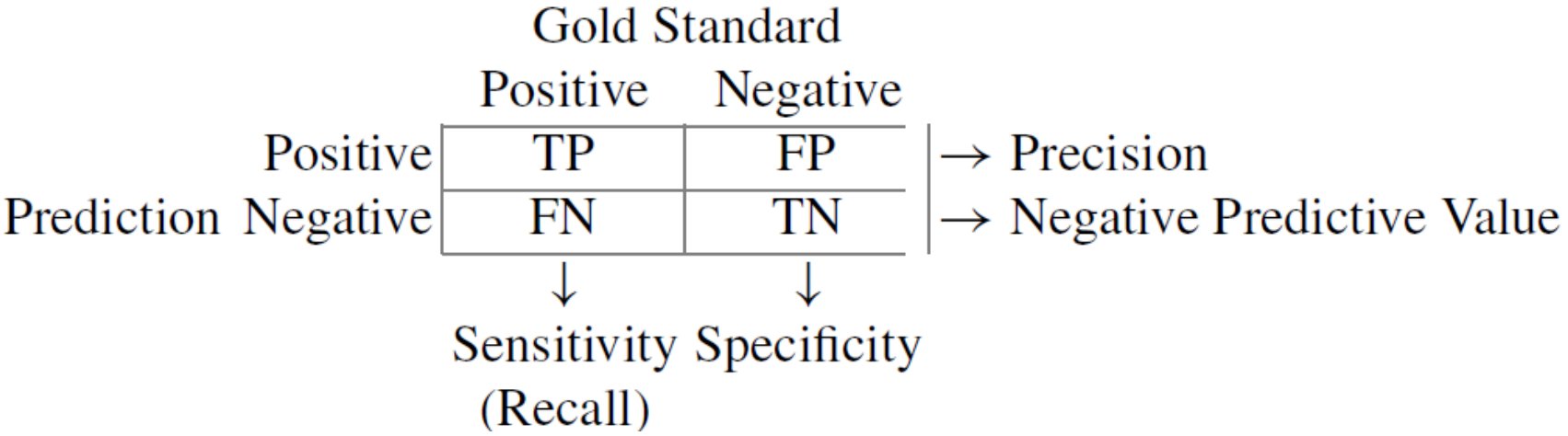
# The Accuracy Paradox

- overall accuracy in machine learning classification models can be misleading when the class distribution is imbalanced, and it is critical to predict the minority class correctly.
- In this case, the class with a higher occurrence may be correctly predicted, leading to a high accuracy score, while the minority class is being misclassified.
- This gives the wrong impression that the model is performing well when it is not.
- For example, in cancer prediction, we cannot miss malignant cases.
  - Neither should we diagnose benign ones as malignant.
  - Doing so would put healthy people through serious treatment and decrease trust in the whole diagnostic process.
  - But most times, the dataset contains a lot of data points in the benign class and few in the malignant class.

# Wisconsin Breast Cancer dataset

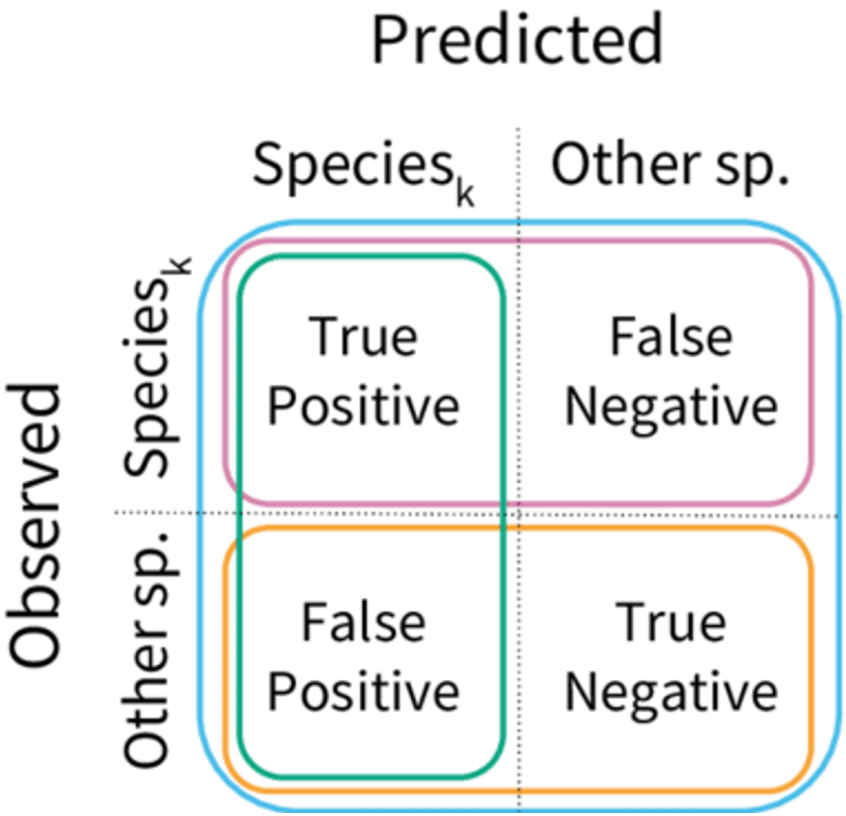


# Confusion Matrix




- *TP* : True Positives
  - positive classes that are correctly predicted as positive.
- *FP* : False Positives
  - negative classes that are falsely predicted as positive.
- *TN* : True Negatives
  - negative classes that are correctly predicted as negative.
- *FN* : False Negatives
  - positive classes that are falsely predicted as negative.

# Confusion Matrix for Multiclass




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

$$\text{Specificity} = \frac{TN}{TN + FP}$$

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

$$\text{Recall} = \frac{TP}{TP + FN}$$

# Performance Evaluation Metrices

- *Accuracy:*

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

- Column-wise we find these two partial performance metrics:
  - *Sensitivity or Recall:*

$$\text{sensitivity} = \frac{\text{TP}}{\text{Real Positives}} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

  - *Specificity:*

$$\text{specificity} = \frac{\text{TN}}{\text{Real Negatives}} = \frac{\text{TN}}{\text{TN} + \text{FP}}$$

- Row-wise we find these two partial performance metrics:
  - *Precision or Positive Predictive Value:*

$$\text{precision} = \frac{\text{TP}}{\text{Predicted Positives}} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

  - *Negative predictive value:*

$$\text{NPV} = \frac{\text{TN}}{\text{Predicted Negative}} = \frac{\text{TN}}{\text{TN} + \text{FN}}$$

# Precision

- It explains how many of the correctly predicted cases actually turned out to be positive.
- Precision is useful in the cases where False Positive is a higher concern than False Negatives.
- The importance of *Precision is in music or video recommendation systems, e-commerce websites, etc. where wrong results could lead to customer churn and this could be harmful to the business.*
- **Precision for a label is defined as the number of true positives divided by the number of predicted positives.**

# Recall (Sensitivity)

- It explains how many of the actual positive cases we were able to predict correctly with our model.
- Recall is a useful metric in cases where False Negative is of higher concern than False Positive.
- It *is important in medical cases where it doesn't matter whether we raise a false alarm but the actual positive cases should not go undetected!*
- **Recall for a label is defined as the number of true positives divided by the total number of actual positives.**

# F1 Score

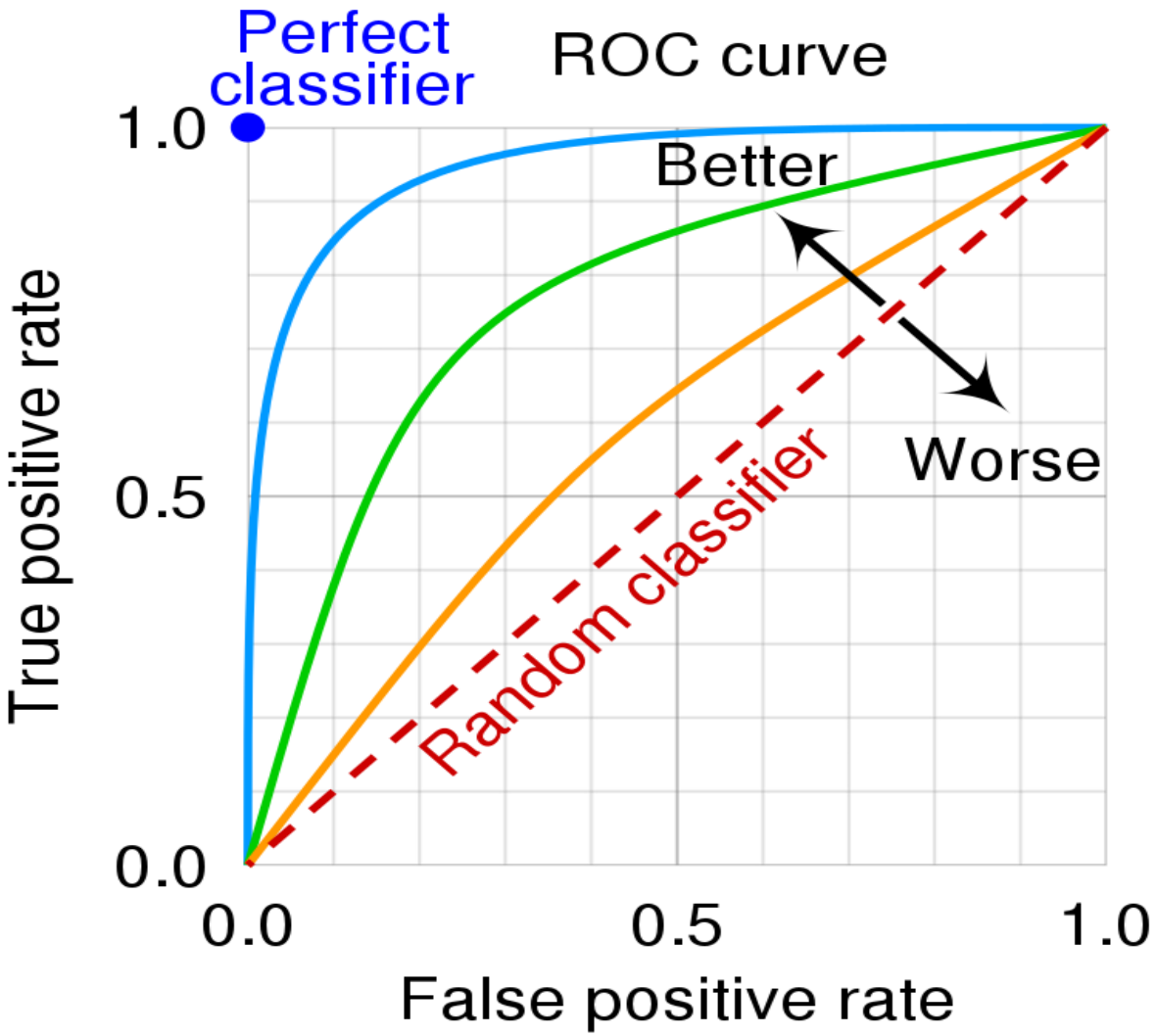
- It gives a combined idea about Precision and Recall metrics. It is maximum when Precision is equal to Recall.
- **F1 Score is the harmonic mean of precision and recall.**

# ROC curve

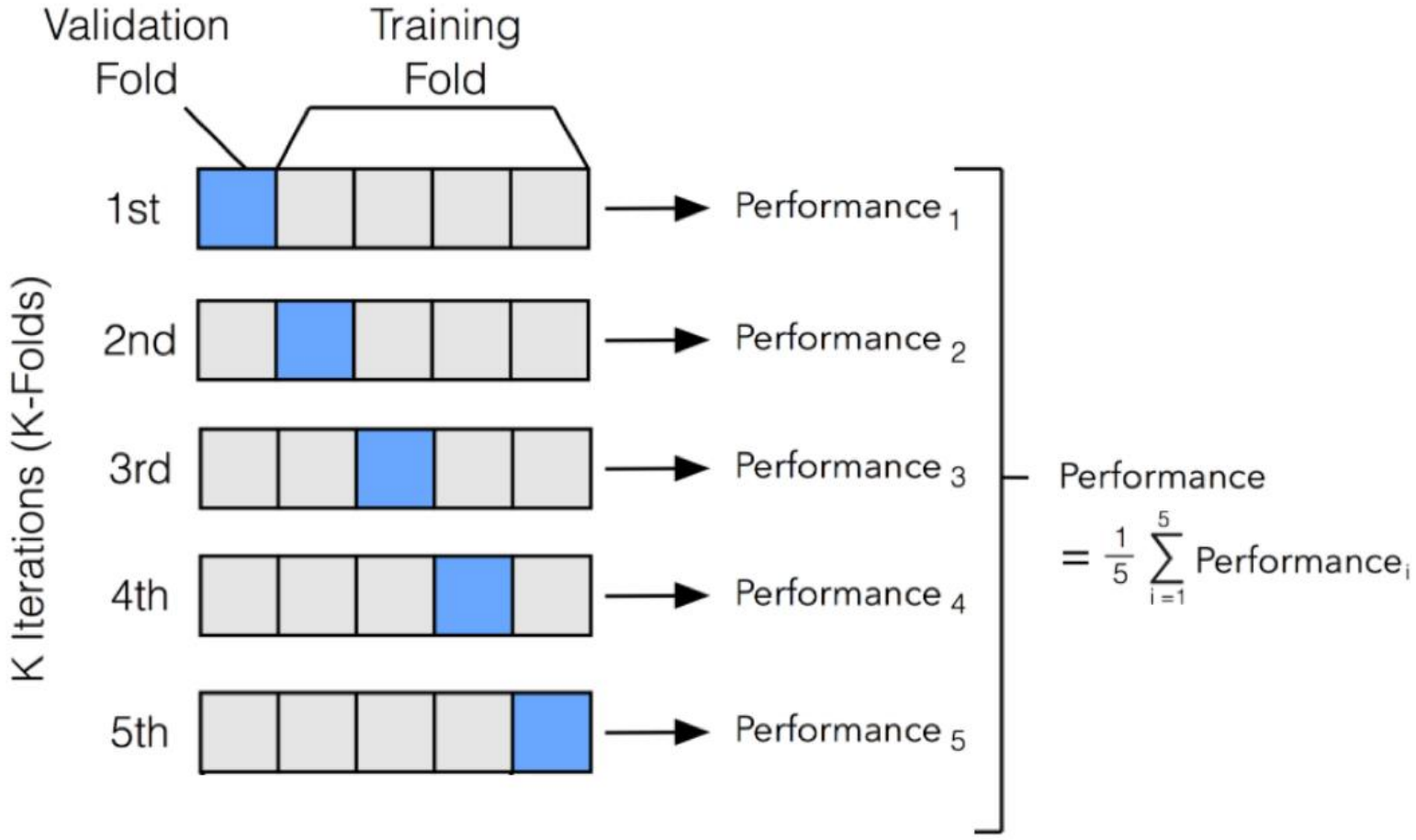
- Another useful tool for visualizing the trade-off between true positives and false positives in order to choose the operating point of the classifier is the receiver-operating characteristic (ROC) curve.
- This curve plots the true positive rate/sensitivity/recall (TP/(TP+FN)) with respect to the false positive rate (FP/(FP+TN)).

# ROC curve

- .

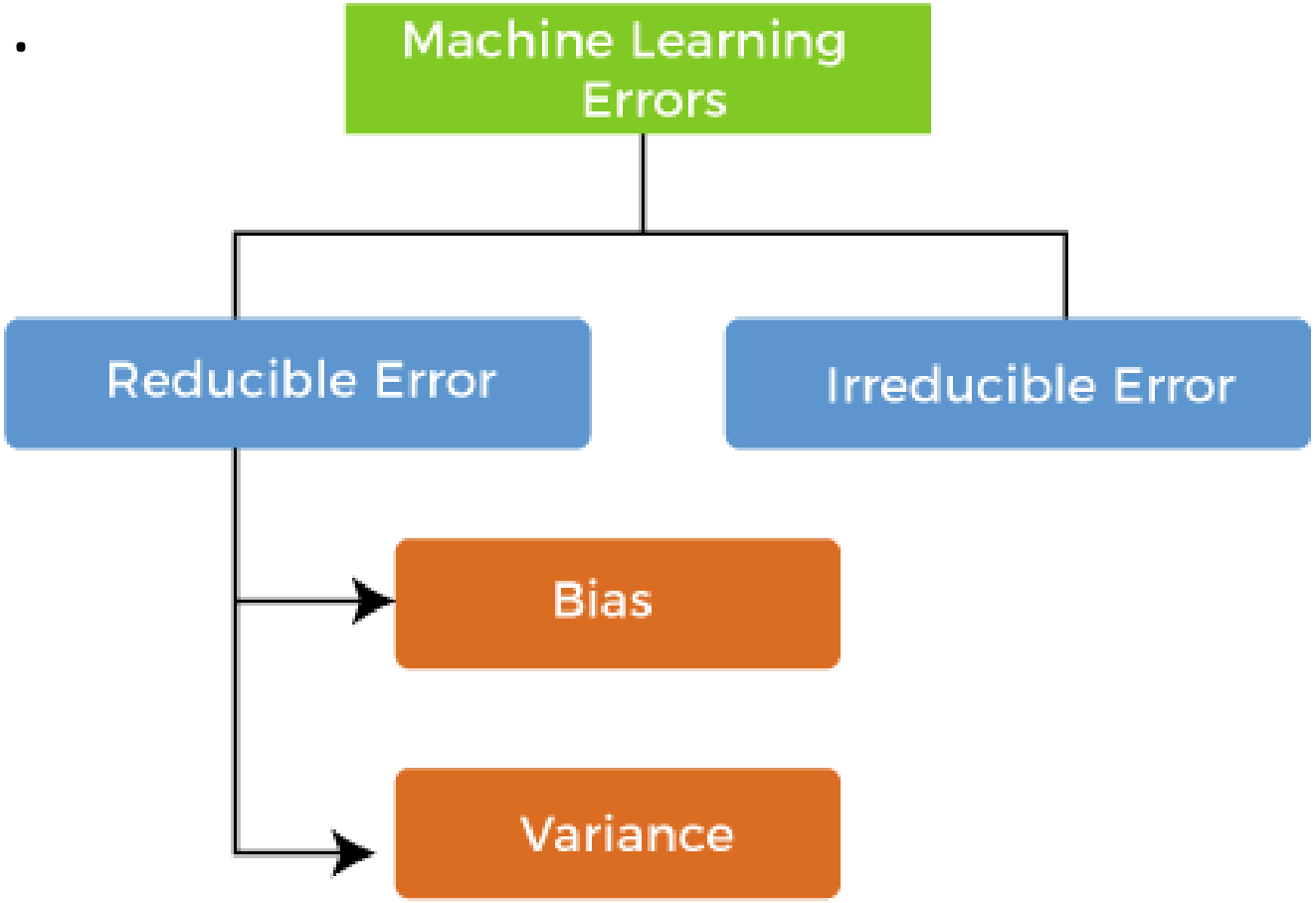

Perfect classifier
ROC curve
1.0
Better
0.5
Worse
Random classifier
True positive rate
0.0
0.0
0.5
1.0
False positive rate

# K – Fold Cross Validation



# Sources of Errors in machine learning

- .



# Bias

- ***While making predictions, a difference occurs between prediction values made by the model and actual values/expected values, and this difference is known as bias errors or Errors due to bias***.
  - It can be defined as an inability of machine learning algorithms such as Linear Regression to capture the true relationship between the data points.
- **Low Bias:** A low bias model will make fewer assumptions about the form of the target function.
- **High Bias:** A model with a high bias makes more assumptions, and the model becomes unable to capture the important features of our dataset. **A high bias model also cannot perform well on new data.**
- Generally, a linear algorithm has a high bias, as it makes them learn fast. The simpler the algorithm, the higher the bias it has likely to be introduced. Whereas a nonlinear algorithm often has low bias.
- **Ways to reduce High Bias:**
  - High bias mainly occurs due to a much simple model. Below are some ways to reduce the high bias:
  - Increase the input features as the model is underfitted.
  - Decrease the regularization term.
  - Use more complex models, such as including some polynomial features.

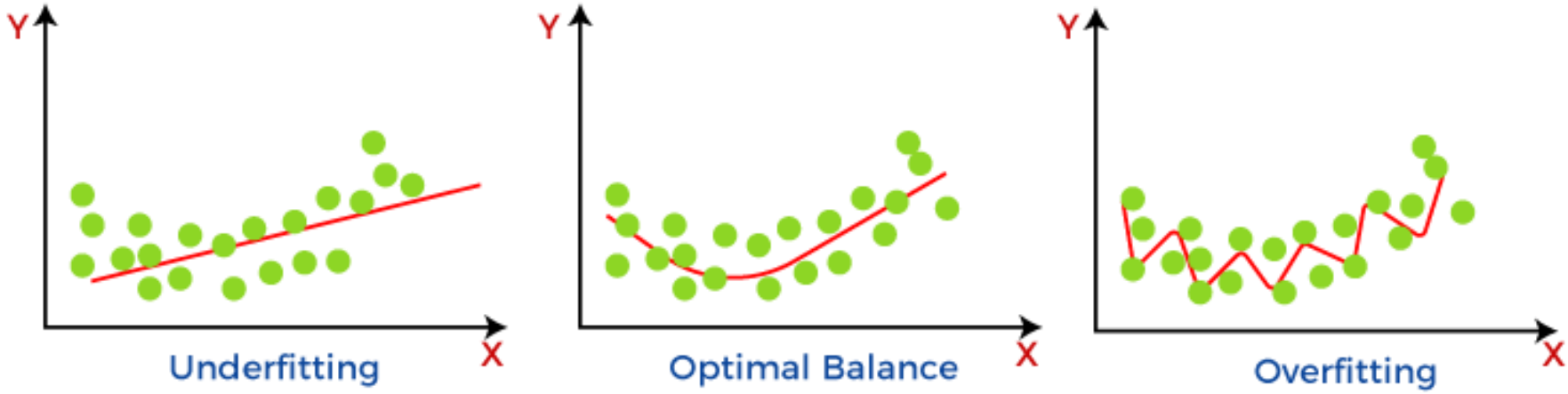
# Variance

- A model that shows high variance learns a lot and perform well with the training dataset, and does not generalize well with the unseen dataset. As a result, such a model gives good results with the training dataset but shows high error rates on the test dataset.
- Since, with high variance, the model learns too much from the dataset, it leads to overfitting of the model. A model with high variance has the below problems:
  - A high variance model leads to overfitting.
  - Increase model complexities.
- Usually, nonlinear algorithms have a lot of flexibility to fit the model, have high variance.
- Some algorithms with low variance are, **Linear Regression, Logistic Regression, and Linear discriminant analysis**.
- Algorithms with high variance are **decision tree, Support Vector Machine, and K-nearest neighbours.**
- Ways to Reduce High Variance:
  - Reduce the input features or number of parameters as a model is overfitted.
  - Do not use a much complex model.
  - Increase the training data.
  - Increase the Regularization term.

# Overfitting

- In the learning curve that plot the learning behavior for a fixed number of examples with respect to the complexity of the model, as the complexity increases the training error is reduced; but above a certain level of complexity, the test error also increases. This effect is called *overfitting.*

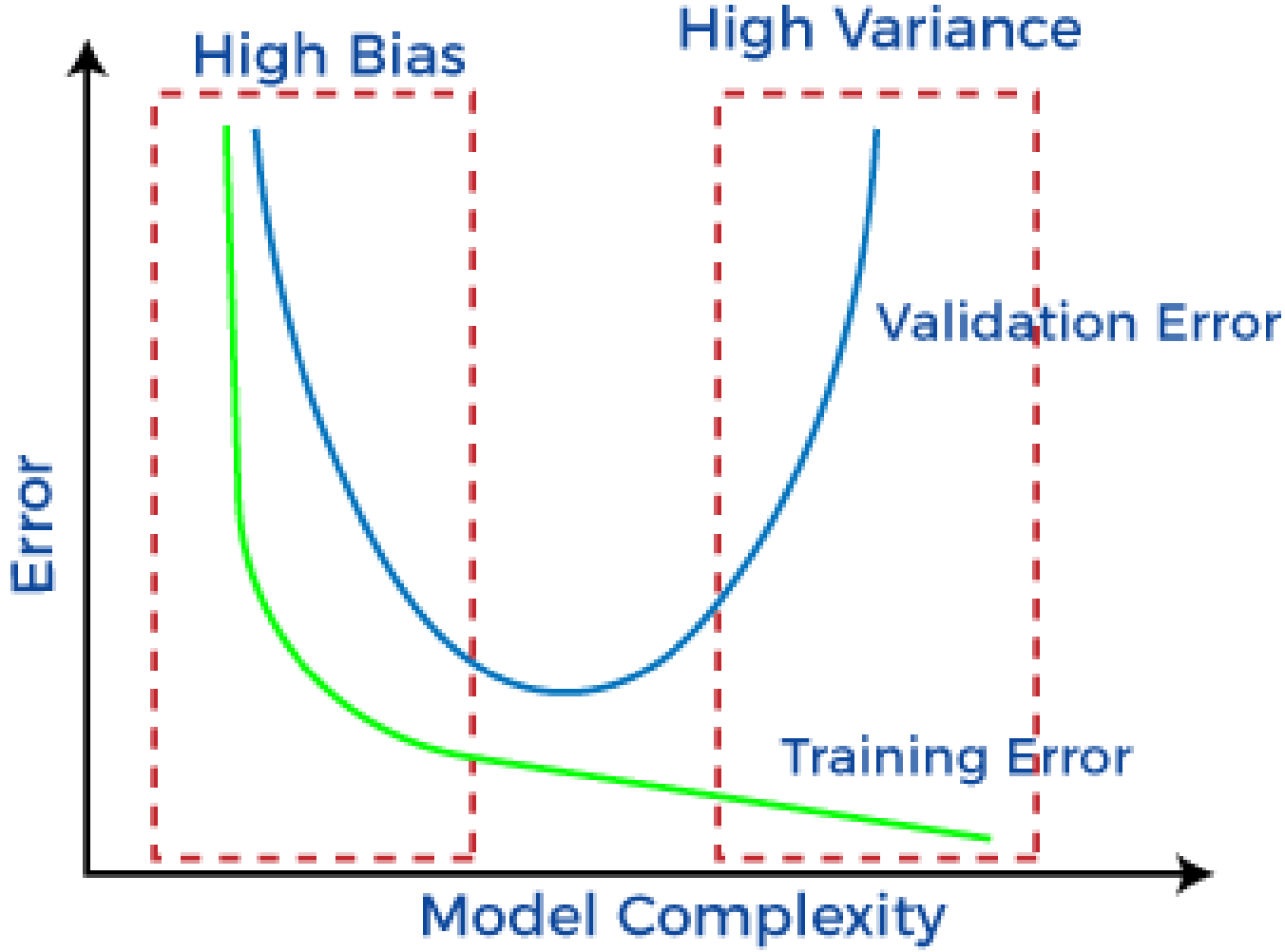
# Underfitting vs Overfitting



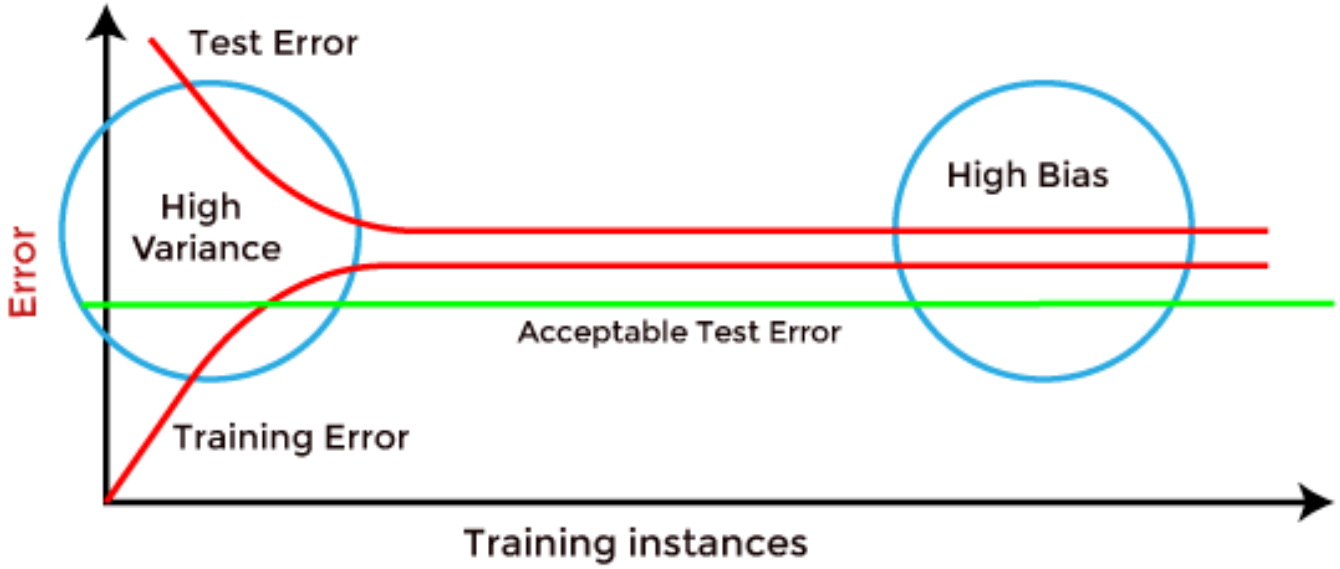

- https://www.javatpoint.com/bias-and-variance-in-machine-learning.

# Bias and Variance

- .



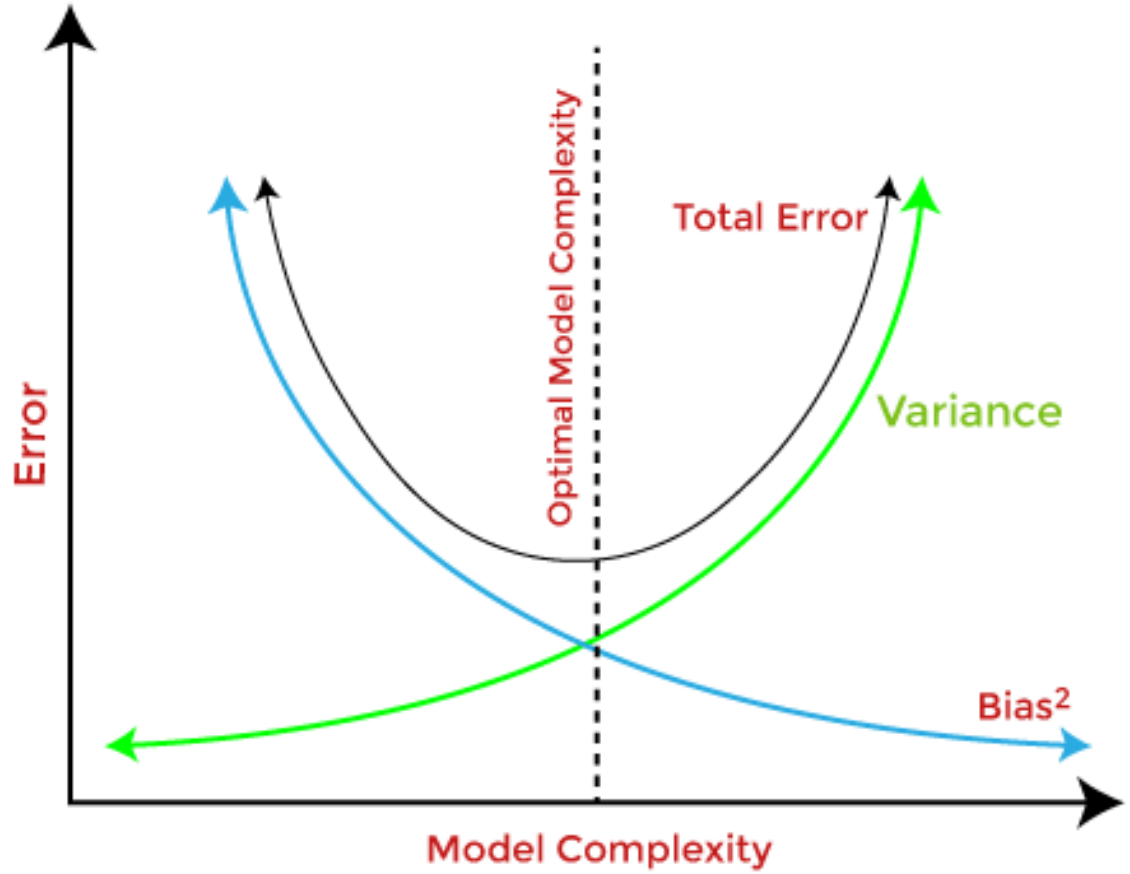
# How to identify High variance or High Bias?




- High variance :Low training error and high test error.
- High Bias : High training error and the test error is almost similar to training error.

# Bias-Variance Tradeoff

- .



# Cures for Overfitting

We may enact several cures for overfitting:

- Hyperparameter tuning [e.g., Grid Search]
- Regularization [e.g., LASSO, Ridge]
- Ensemble techniques [e.g., Random Forest]

# Hyperparameter Tuning

- Models are usually  parameterized by some hyperparameters.
- Selecting the complexity is usually governed by some such parameters.
- Thus, we are faced with a model selection problem.
- A good heuristic for selecting the model is to choose the value of the hyperparameters that yields the smallest estimated test error.

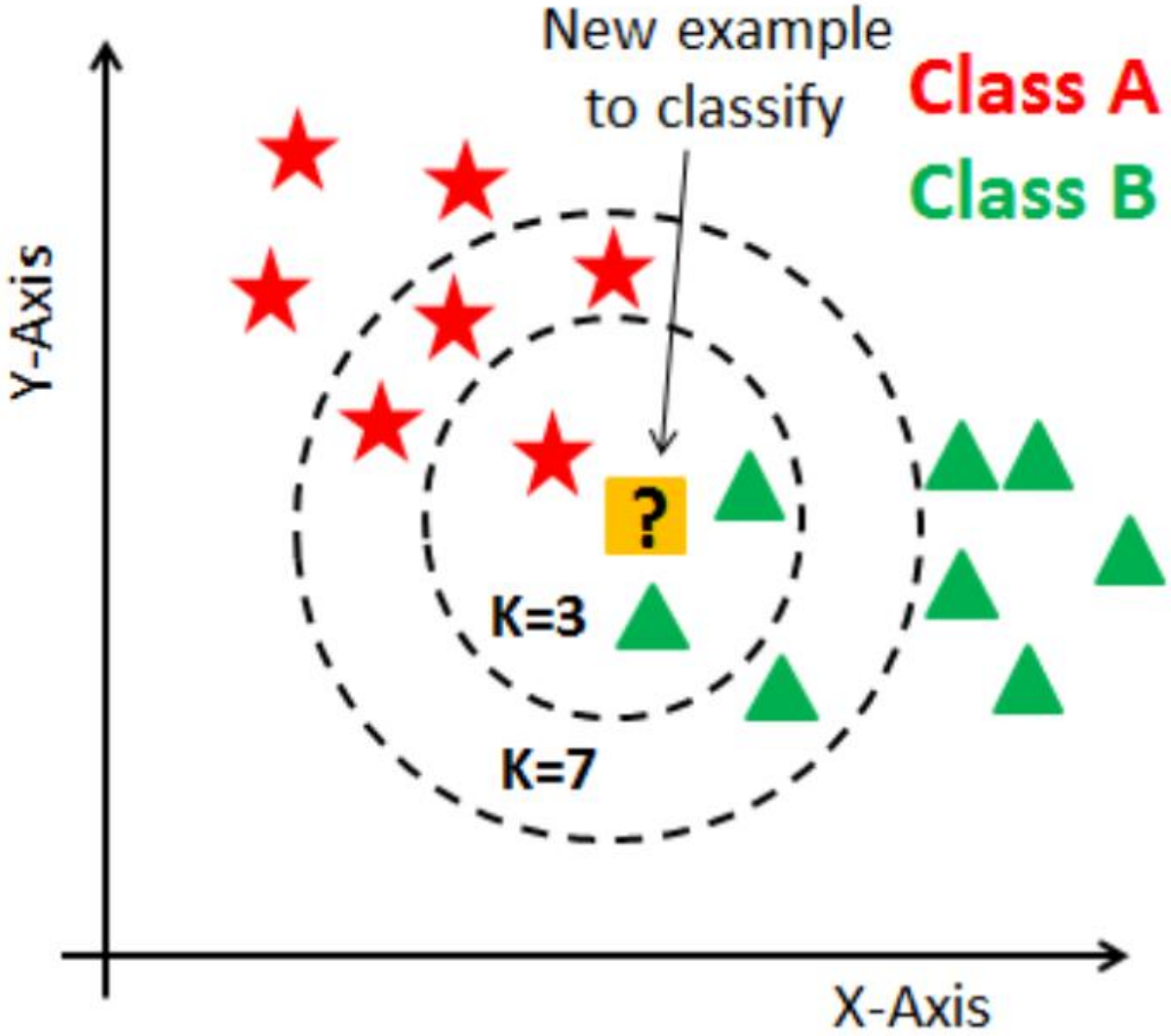
# Regularization

- We may also change the formulation of the objective function to penalize complex models. This is called *regularization*.
- **L2 weight regularization:**
  - Adding an L2 penalization term to the weights of a weight-controlled model implies looking for solutions with small weight values. Intuitively, adding an L2 penalization term can be seen as a surrogate for the notion of smoothness. In this sense, a low complexity model means a very smooth model.
- **L1 weight regularization:**
  - Adding an L1 regularization term forces sparsity in the weights of the model. In this sense, a low complexity model means a model with few components or few active terms.
- These terms are added to the objective function. They trade off with the error function in the objective and are governed by a hyperparameter. Thus, we still have to select this parameter by means of model selection.

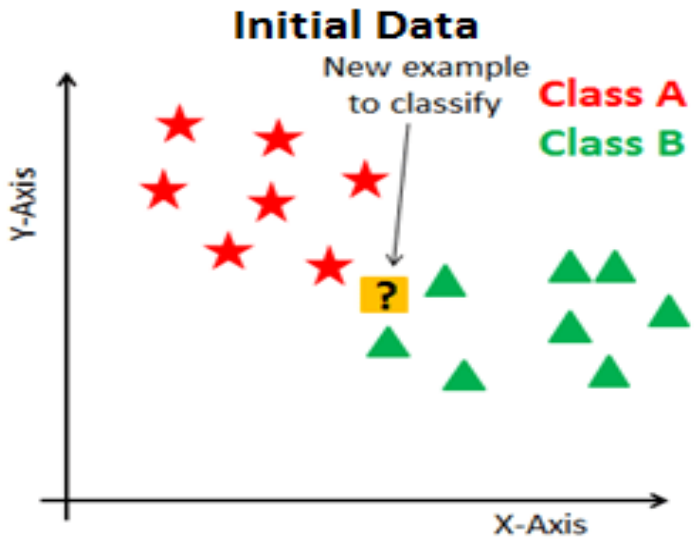
# Ensemble Techniques

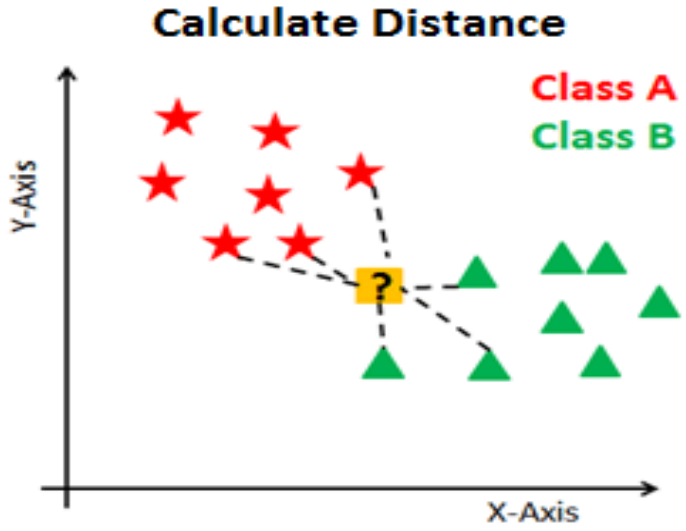
- Another cure for overfitting is to use ensemble techniques.
- The best known are *bagging* and *boosting*.

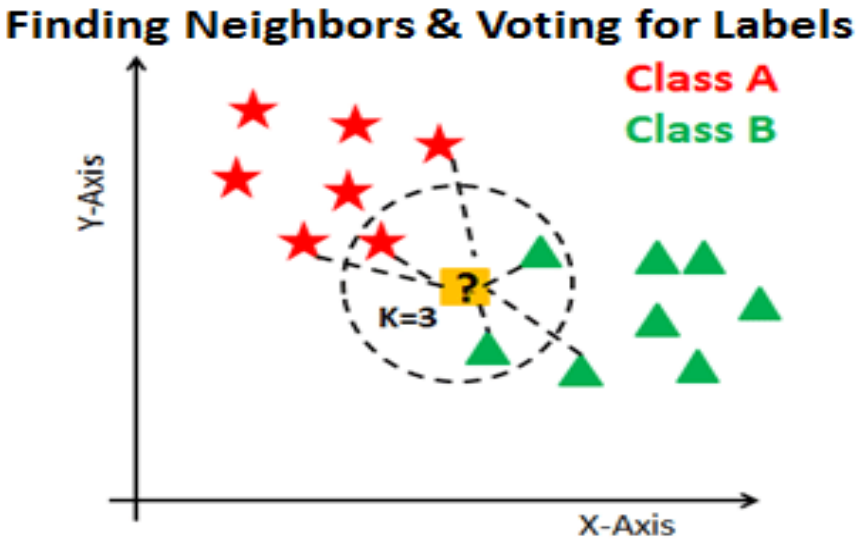
# KNN Classifier



# KNN Classifier



# Support Vector Machines

- SVM is a learning technique initially designed to fit a linear boundary between the samples of a binary problem, ensuring the maximum robustness in terms of tolerance to isotropic uncertainty. (Fig. 5.9).
- Note that the boundary displayed has the largest distance to the closest point of both classes.
- Any other separating boundary will have a point of a class closer to it than this one.
- The figure also shows the closest points of the classes to the boundary. These points are called ***support vectors***.
  - In fact, the boundary only depends on those points.
  - If we remove any other point from the dataset, the boundary remains intact.
  - However, in general, if any of these special points is removed the boundary will change.

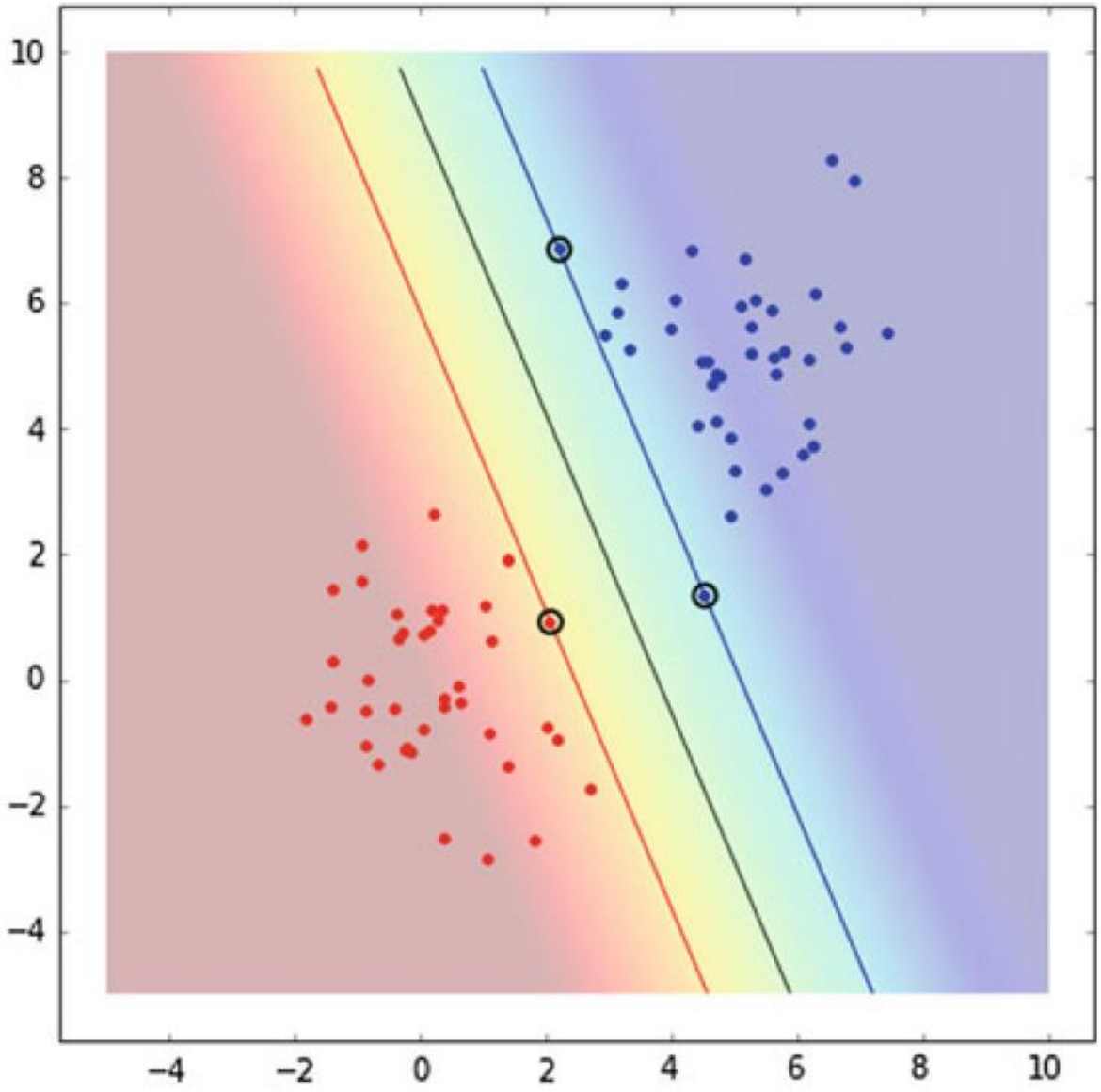



**Fig. 5.9** Support vector machine decision boundary and the support vectors

# Kernel SVM

The decision boundary of most problems cannot be well approximated by a linear model. In SVM, the extension to the nonlinear case is handled by means of kernel theory. In a pragmatic way, a kernel can be referred to as any function that captures the similarity between any two samples in the training set. The kernel has to be a positive semi-definite function as follows:

- *Linear kernel*:

$$k(x_i, x_j) = x_i^T x_j$$

- *Polynomial kernel*:

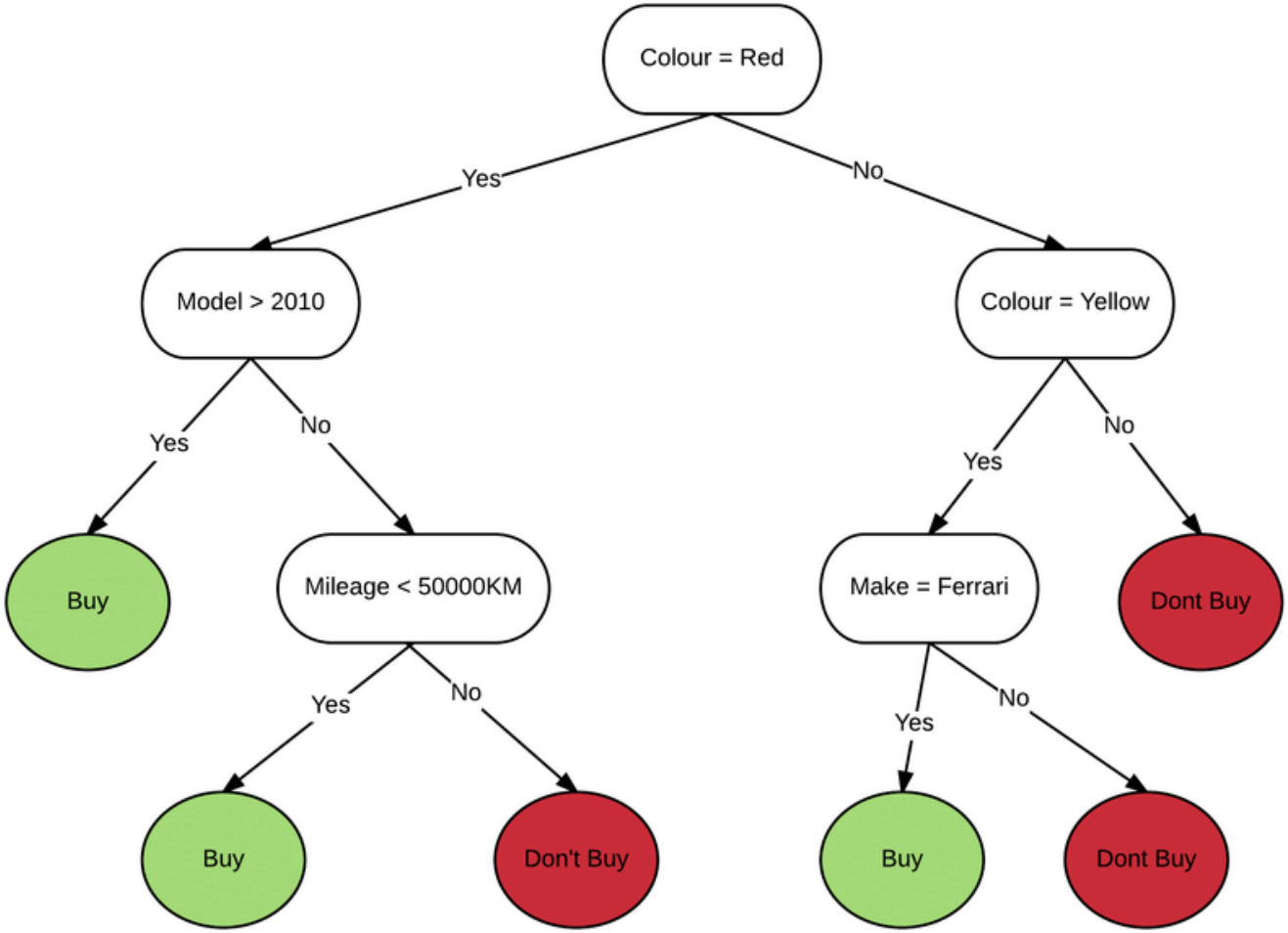
$$k(x_i, x_j) = (1 + x_i^T x_j)^p$$

- *Radial Basis Function kernel*:

$$k(x_i, x_j) = e^{-\frac{\|x_i - x_j\|^2}{2\sigma^2}}$$

# Decision Trees

- A decision tree is one of the most simple and intuitive techniques in machine learning, based on the divide and conquer paradigm.
- The basic idea behind decision trees is to partition the space into patches and to fit a model to a patch.
- There are two questions to answer in order to implement this solution:
  - How do we partition the space?
  - What model shall we use for each patch?
- Tackling the first question, most techniques use a threshold in a single feature. For example, in our problem “Does the applicant have a home mortgage?”. This is the key that allows the results of this method to be interpreted.
- In decision trees, the second question is straightforward, each patch is given the value of a label, e.g., the majority label, and all data falling in that part of the space will be predicted as such.


Colour = Red
Yes
No
Model > 2010
Colour = Yellow
Yes
No
Yes
No
Buy
Mileage < 50000KM
Make = Ferrari
Dont Buy
Yes
No
Yes
No
Buy
Don't Buy
Buy
Dont Buy

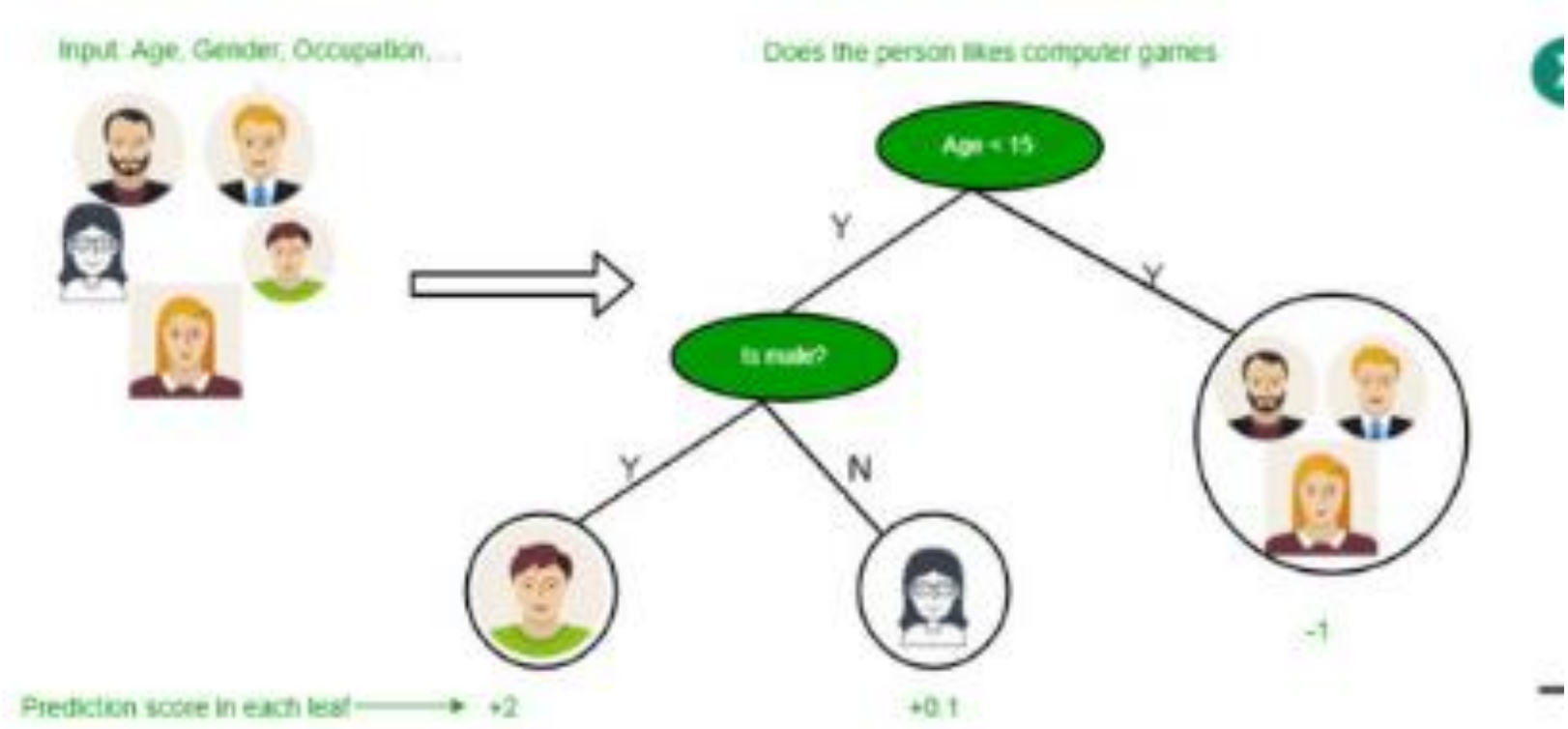
# Random Forest

- Random Forest (RF) is an ensemble technique. The base classifiers in RF are decision trees.
- The RF technique creates different trees over the same training dataset.
- The word “random” in RF refers to the fact that only a subset of features is available to each of the trees in its building process.
- The two most important parameters in RF are the **number of trees** in the ensemble and the **number of features** each tree is allowed to check.
- the aggregation of classifiers using a voting technique
- RF rely on combining different decision tree classifier using a majority voting as an aggregation technique.

# Ensemble Techniques

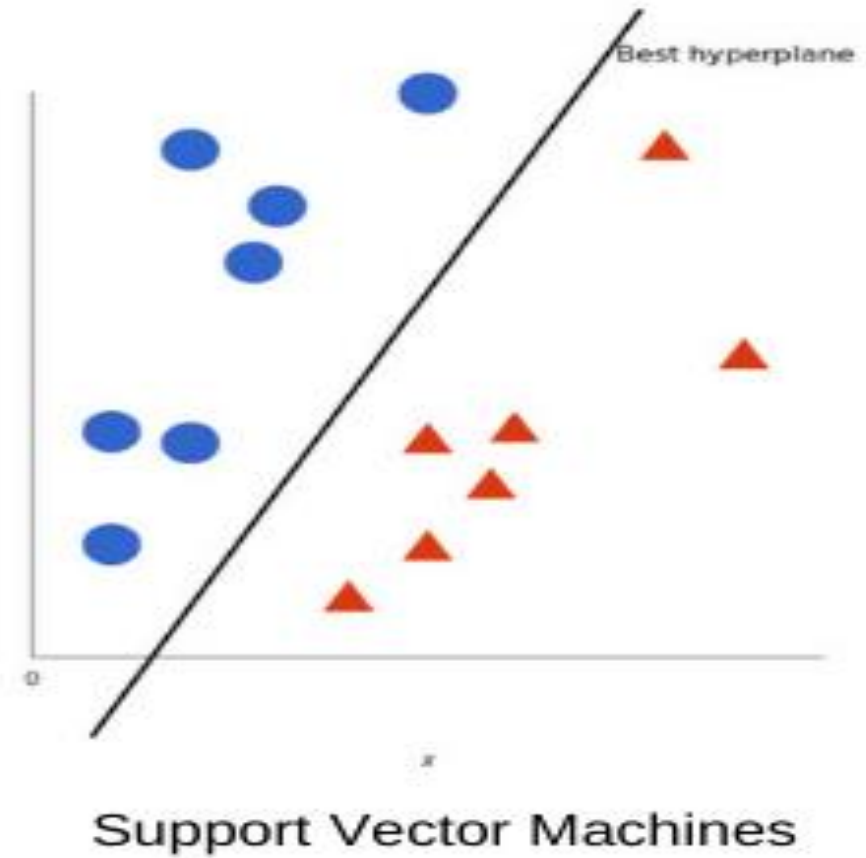
- Ensemble techniques rely on combining different classifiers using some aggregation technique, such as majority voting. Ensemble techniques usually have good properties for combating overfitting.
- In this case, the aggregation of classifiers using a voting technique reduces the variance of the final classifier. This increases the robustness of the classifier and usually achieves a very good classification performance.
- A critical issue in the ensemble of classifiers is that for the combination to be successful, the errors made by the members of the ensemble should be as uncorrelated as possible. This is sometimes referred to the diversity of the classifiers.

# DT




Decision Tree

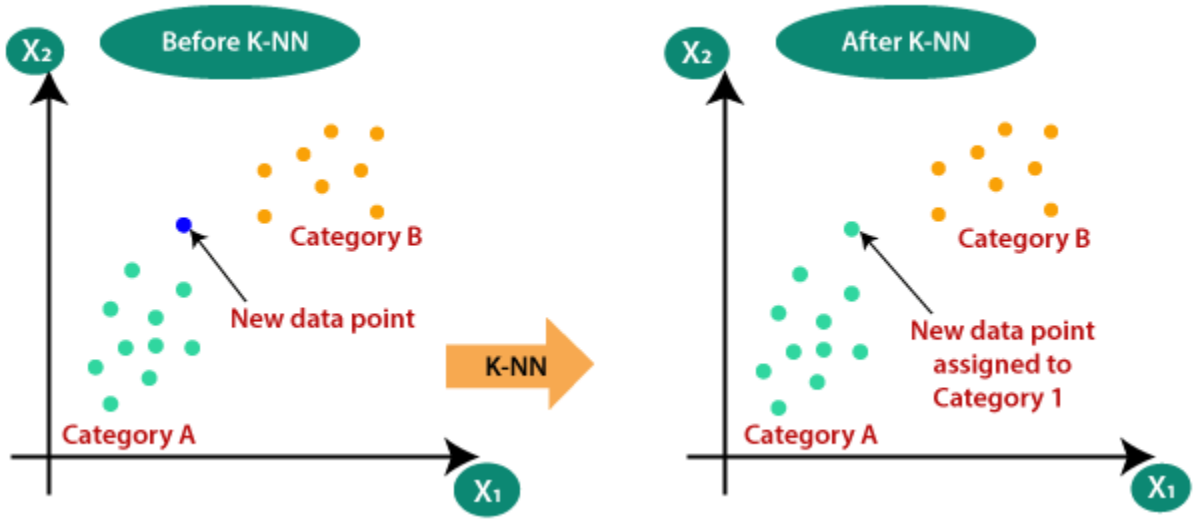
# SVM

- 

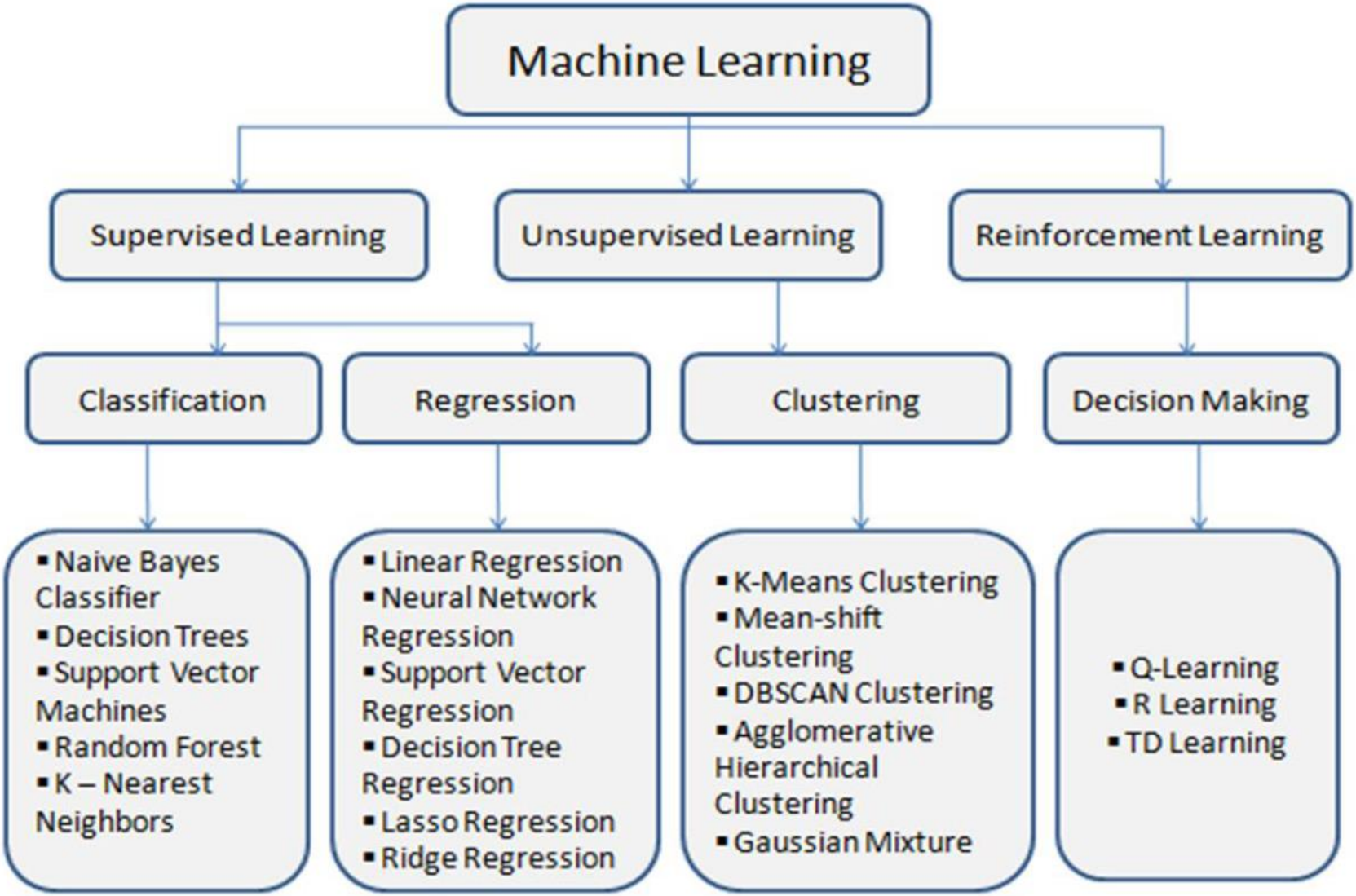

Support Vector Machines

# KNN

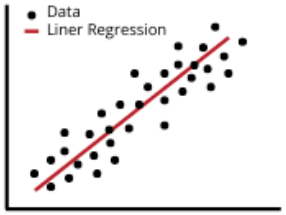
- .


Before K-NN
X2
Category B
New data point
Category A
X1
K-NN
After K-NN
X2
Category B
New data point
assigned to
Category 1
Category A
X1

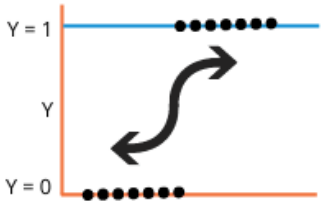
# Machine Learning Algorithms



## Linear Regression

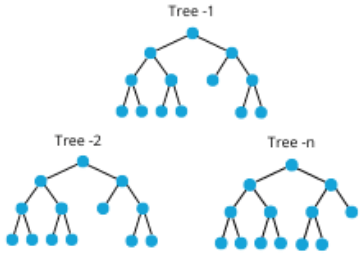



## Logistic Regression

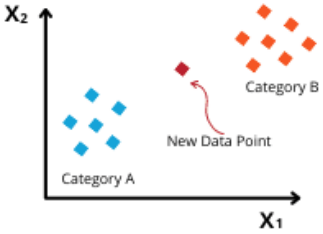



## Decision Trees

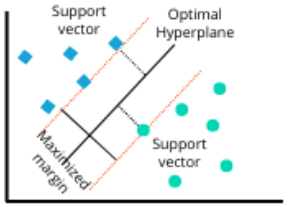

## Random Forest



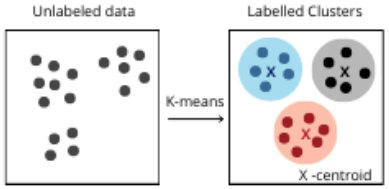

## K-Nearest Neighbor



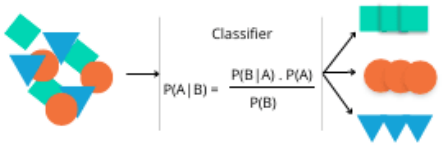

## Support Vector Machine




## K-Means Clustering




## Naïve Bayes



# Terminologies

- Target and features
- Supervised Learning and Unsupervised Learning
  - Classification problem and regression problem
- Test set and training set
- Learning curve
- Training error or in-sample error
- Test error or out of sample error
- Bias-variance tradeoff
- Overfitting and under-fitting problem
- K-fold cross validation
- Regularization
- KPI- key performance indicator
- Hyperparameter
- Confusion matrix

# Classification Algorithm Workflow

- Preprocessing
  - Handling Missing value
    - drop the variable/ case,
    - replace with mean/median/mode etc
  - Handling Outliers
  - Feature Scaling
    - MinMax scaler, Standardize scaler
  - Feature Engineering
  - Drop the redundant variable
  - Encoding categorical variable
    - one hot encoding, label encoding]
- EDA
  - Graphical presentation
  - Tabular presentation
- Fit the Model
  - Split the data into test set and training set
  - Train the model by passing the training set to the model
- Prediction
  - using the fitted model predict target for test set
- Performance Evaluation and Model Selection
  - Accuracy
  - Confusion matrix
  - Precision/Recall /Specificity/sensitivity
  - ROC-AUC
- Improve Model performance
  - Hyperparameter tuning
  - K-fold cross validation
  - Regularization [L1 and L2]

# THANK YOU