Linear Classifiers, Binary Classification, Multi-class classification, Evaluating model performance: resampling methods (cross-validation, bootstrap), regularisation, overfitting, bias-variance tradeoff

PRESENTATION BY:
- SUUNA CONRAD
- HILARY KANSIIME
- FREEDOM GEMMAR

DATASETS USED:
- IRIS

Iris Dataset

Species in the data set

Iris Versicolor

Iris Setosa

Iris Virginica

Features shared across the species

- sepal_length.
- sepal_width.
- petal_length.
- petal_width

Classification

- A systematic arrangement in groups or categories according to established criteria.
- Process of identifying to which of a set of categories (sub-populations) a new observation belongs, based on a training set of data containing observations (or instances) whose category membership is known.
- Classification is a task that requires the use of machine learning algorithms that learn how to assign a class label to examples from the problem domain. An easy to understand example is classifying emails as “spam” or “not spam.” It falls under supervised learning.

Classification types

- Binary Classification
- Multiclass Classification
- Multi-label
- Imbalance

Binary classification

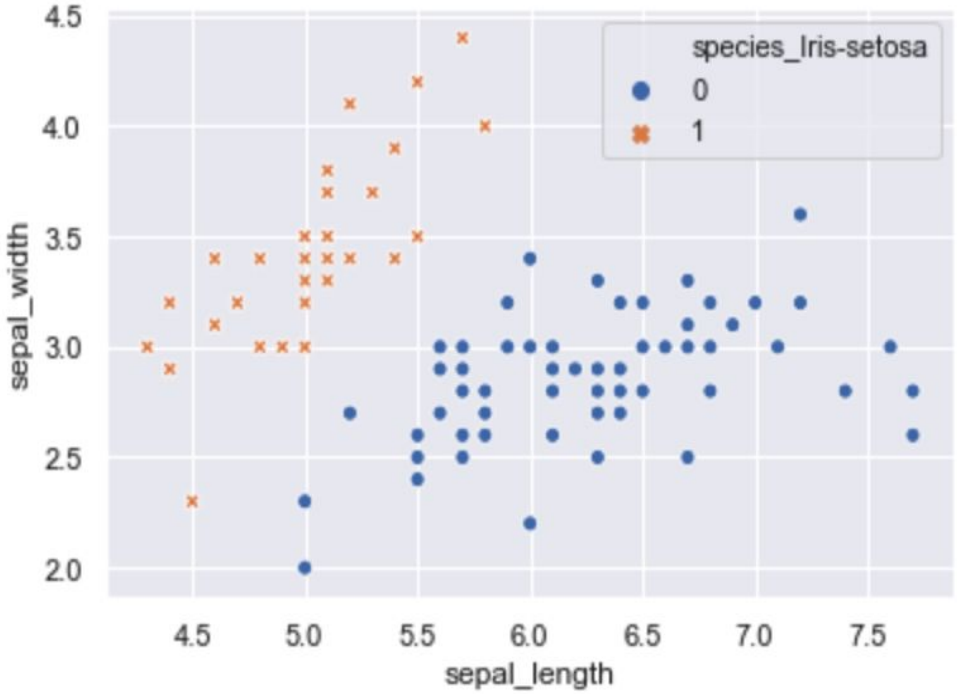
Binary classification means there are two classes to work with that relate to one another as true and false.

Examples Include

- Email Spam detection(spam or not)
- Churn prediction(churn or not)
- Conversion prediction(buy or not)

Algorithms for binary classification; **logistic regression, K-Nearest neighbours, Decision trees, SVM, Naive Bayes.**

Figure 1: Binary classification

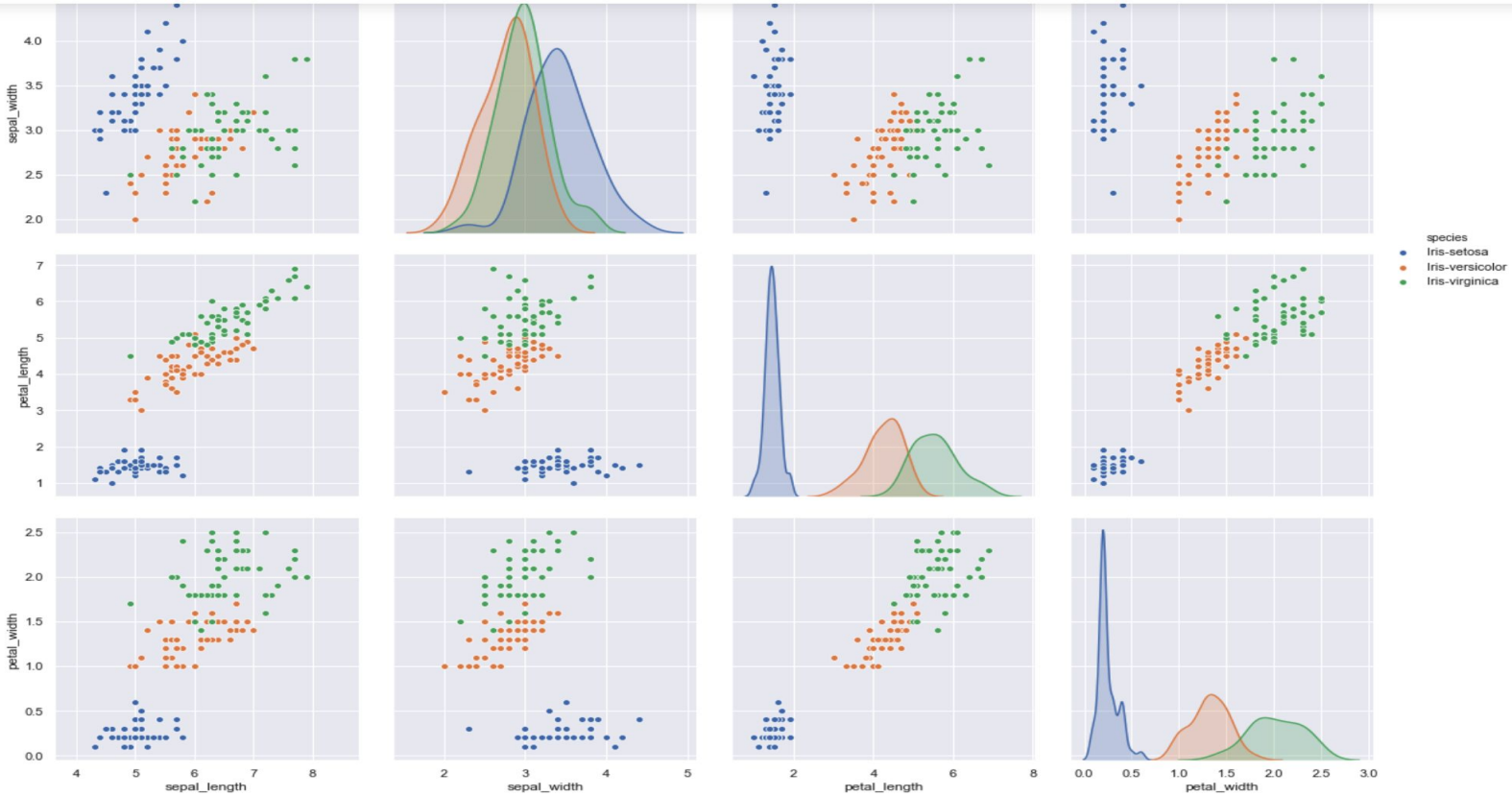


Multi-class classification

This works with more than two classes.

Algorithms; **K-Nearest neighbours, Decision trees, Naive Bayes, random forest, gradient boosting, artificial neural nets**

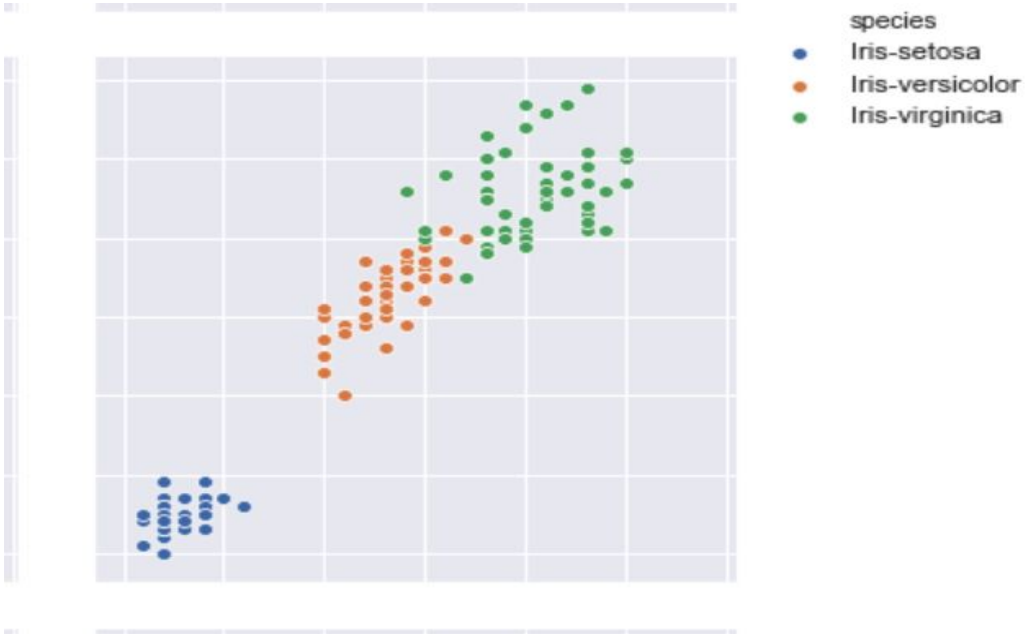
Figure 2: multi-class classification



Linear classifiers

Linear classifiers classify data into labels based on a linear combination of input features/characteristics. Therefore, these classifiers separate data using a line or plane or a hyperplane (a plane in more than 2 dimensions). They can only be used to classify data that is linearly separable. They can be modified to classify non-linearly separable data.

Figure 3: linearly separable vs non-linearly separable.



Linear classifier algorithms

- Perceptron - Perceptron is a linear binary classification algorithm. It is a single layer neural network. Its functioning is similar to the functioning of a single neuron in our brain. A neuron in our brain takes input signals from many other neurons and based on those inputs, it decides whether to fire or not
- Logistic regression - It takes a linear combination of features as input and outputs a number between 0 and 1
- SVM - Given some linearly separable data, we can have multiple hyperplanes that can act as a separation boundary as shown in figure 1. SVM chooses the "optimal" hyperplane amongst all candidate hyperplanes

Model evaluation

Model Evaluation is an integral part of the *model* development process. It helps to find the best *model* that represents our data and how well the chosen *model* will work in the future.

Model evaluation metrics include:

- Classification accuracy.
- Logarithmic loss.
- The area under the curve.
- Confusion matrix.
- F-measure (F-score).

Logarithmic Loss

This is a performance metric for evaluating the predictions of probabilities of membership to a given class.

$$LogarithmicLoss = \frac{-1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij} * \log(p_{ij})$$

Where y_ij, indicates indicates whether sample i belongs to class j or not and p_ij indicate the probability of sample i belonging to class j

Classification Accuracy

This is the number of correct predictions made as a ratio of all predictions made.

$$Accuracy = \frac{Number\ of\ Correct\ predictions}{Total\ number\ of\ predictions\ made}$$

Area under a curve

AUC of a classifier is equal to the probability that the classifier will rank a randomly chosen positive example higher than a randomly chosen negative example.

Area under the curve - True Positive Rate (Sensitivity)

True Positive Rate is defined as *TP/ (FN+TP)*. True Positive Rate corresponds to the proportion of positive data points that are correctly considered as positive, with respect to all positive data points.

$$TrueNegativeRate = \frac{TrueNegative}{TrueNegative + FalsePositive}$$

Area under the curve - True Negative Rate (Specificity)

True Negative Rate is defined as *TN / (FP+TN)*. False Positive Rate corresponds to the proportion of negative data points that are correctly considered as negative, with respect to all negative data points

$$TrueNegativeRate = \frac{TrueNegative}{TrueNegative + FalsePositive}$$

Area under the curve - False Positive Rate

False Positive Rate is defined as *FP / (FP+TN)*. False Positive Rate corresponds to the proportion of negative data points that are mistakenly considered as positive, with respect to all negative data points.

$$FalsePositiveRate = \frac{FalsePositive}{TrueNegative + FalsePositive}$$

Confusion Matrix

Confusion Matrix as the name suggests gives us a matrix as output and describes the complete performance of the model.

Accuracy for the matrix can be calculated by taking average of the values lying across the **“main diagonal”**

$$Accuracy = \frac{TruePositive + TrueNegative}{TotalSample}$$

Confusion Matrix - Example

n=165	Predicted: NO	Predicted: YES
Actual: NO	50	10
Actual: YES	5	100

$$\therefore Accuracy = \frac{100 + 50}{165} = 0.91$$

F1 Score

This is the measure of a test's accuracy that considers both the precision and the recall of the test to compute the score.

It is the Harmonic Mean between precision and recall.

$$F1 = 2 * \frac{1}{\frac{1}{precision} + \frac{1}{recall}}$$

F1 Score - Precision

Precision is the number of correct positive results divided by the number of positive results predicted by the classifier.

$$Precision = \frac{TruePositives}{TruePositives + FalsePositives}$$

F1 Score - Recall

Recall is the number of correct positive results divided by the number of *all* relevant samples (all samples that should have been identified as positive).

$$Precision = \frac{TruePositives}{TruePositives + FalseNegatives}$$

Resampling

Re-sampling is a series of methods used to reconstruct your sample data sets, including training sets and validation sets. It can provide more "useful" different sample sets for the learning process in some way.

Two commonly used resampling methods that you may encounter are k-fold cross-validation and the bootstrap.

- **Bootstrap.** Samples are drawn from the dataset with replacement (allowing the same sample to appear more than once in the sample), where those instances not drawn into the data sample may be used for the test set.
- **k-fold Cross-Validation.** A dataset is partitioned into k groups, where each group is given the opportunity of being used as a held out test set leaving the remaining groups as the training set.

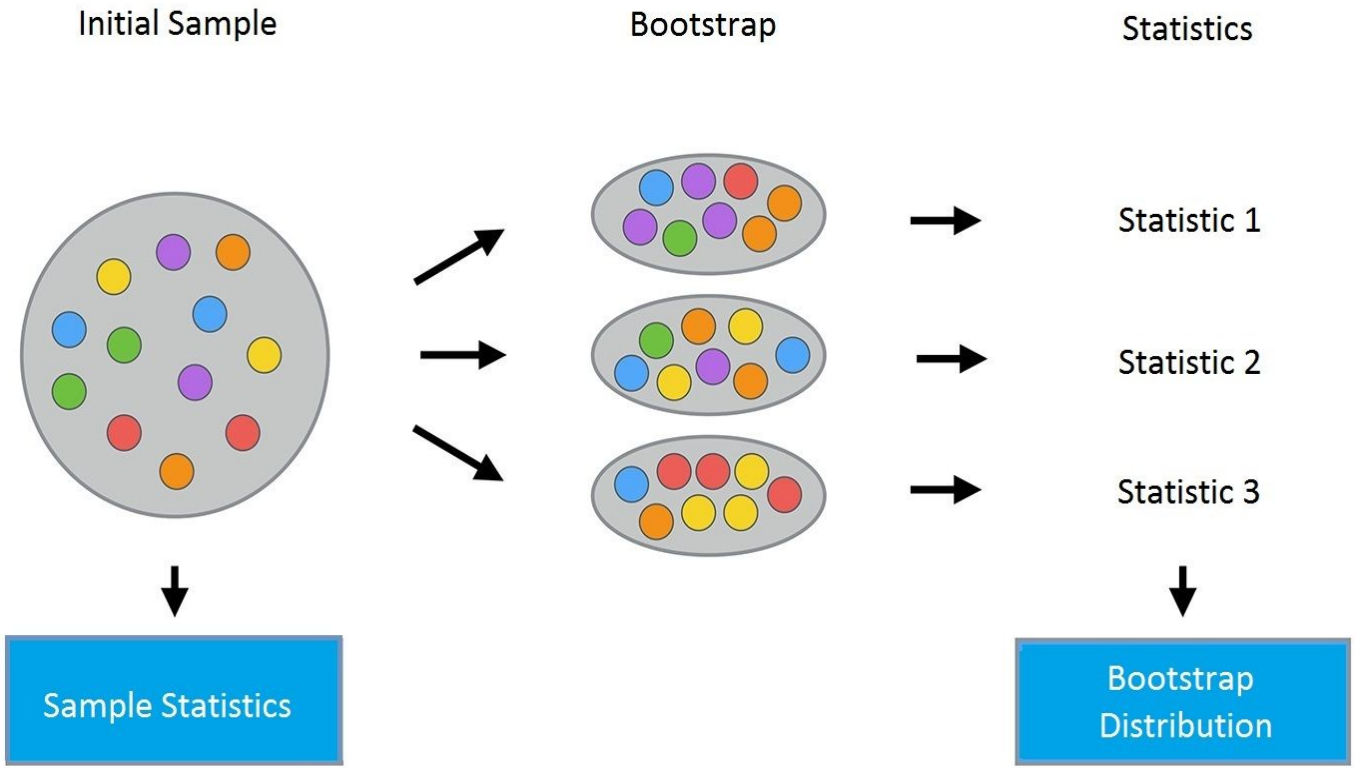
Resampling(Bootstrapping)

Bootstrap resampling was originally invented as a method for approximating the sampling distribution of statistics whose theoretical properties are intractable (Davison and Hinkley 1997).

Bootstrapping involves drawing sample data repeatedly with replacement from a data source to estimate a population parameter.

Bootstrap sampling is used in a machine learning ensemble algorithm called bootstrap aggregating (also called bagging). It helps in avoiding overfitting and improves the stability of machine learning algorithms.

Figure 4: Bootstrapping



Cross validation

Is used to estimate the performance of machine learning models

Types of cross-validation

The types of cross-validation are divided into 2 categories i.e. exhaustive and non-exhaustive. *Exhaustive* cross-validation methods and tests on all possible ways to divide the original sample into a training and a validation set while *non-exhaustive* cross-validation does not compute all ways of splitting the original data.

Cross Validation(Non Exhaustive methods)

- **Holdout method**

  This is a quite basic and simple approach in which we divide our entire dataset into two parts viz- training data and testing data. As the name, we train the model on training data and then evaluate the testing set. Usually, the size of training data is set more than twice that of testing data, so the data is split in the ratio of 70:30 or 80:20.

  In this approach, the data is first shuffled randomly before splitting. As the model is trained on a different combination of data points, the model can give different results every time we train it, and this can be a cause of instability. Also, we can never assure that the train set we picked is representative of the whole dataset.

Cross Validation(Non Exhaustive methods)

- **K-fold cross-validation**

  K-fold cross-validation is one way to improve the holdout method. A dataset is divided into k subsets and the holdout method is repeated k number of times.

  Let us go through this in steps:

  a. Randomly split your entire dataset into k number of folds (subsets)
  b. For each fold in your dataset, build your model on k – 1 folds of the dataset. Then, test the model to check the effectiveness for kth fold
  c. Repeat this until each of the k-folds has served as the test set
  d. The average of your k recorded accuracy is called the cross-validation accuracy and will serve as your performance metric for the model.

Cross Validation(Non Exhaustive methods)

- **Stratified K-fold validation**

  With k-folds validation, data is split into k number of folds and there is a chance that we might have unbalanced folds. Stratified K-fold validation solves this by stratifying the data which is accomplished by rearranging the data so as to ensure that each of the k subsets is a good representation of the original dataset.

Figure 5: Hold out method

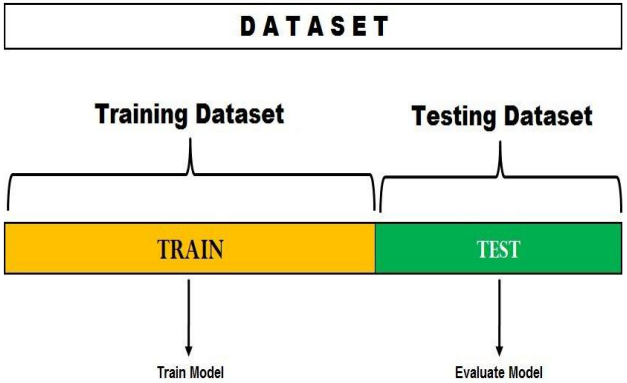
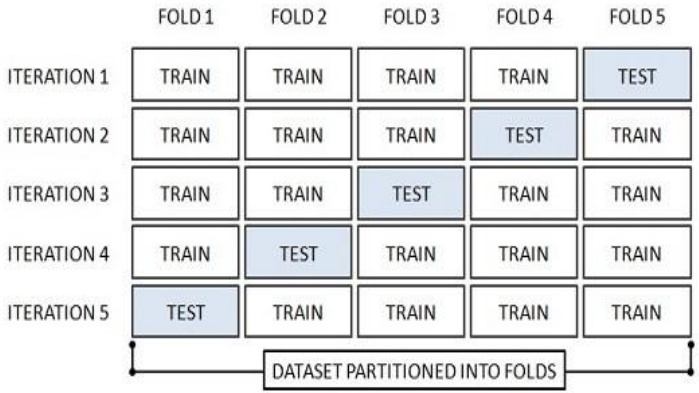


Figure 6: k-fold cross validation



Cross Validation(Exhaustive methods)

1. **Leave-P-Out cross-validation**

   Here we take p number of points out from the total number of data points in the dataset(n). We train the model on the remaining (n - p) points and test on the p points. We repeat the process for all possible combinations of p from the original data set.

2. **Leave-One-Out cross-validation**

   This is a simple variation of leave-p-out where the value of p is set as one.

Overfitting

This happens because your model is trying too hard to capture the noise in your training dataset. By noise, we mean the data points that don't really represent the true properties of your data, but random chance. Learning such data points, makes your model more flexible, at the risk of overfitting.

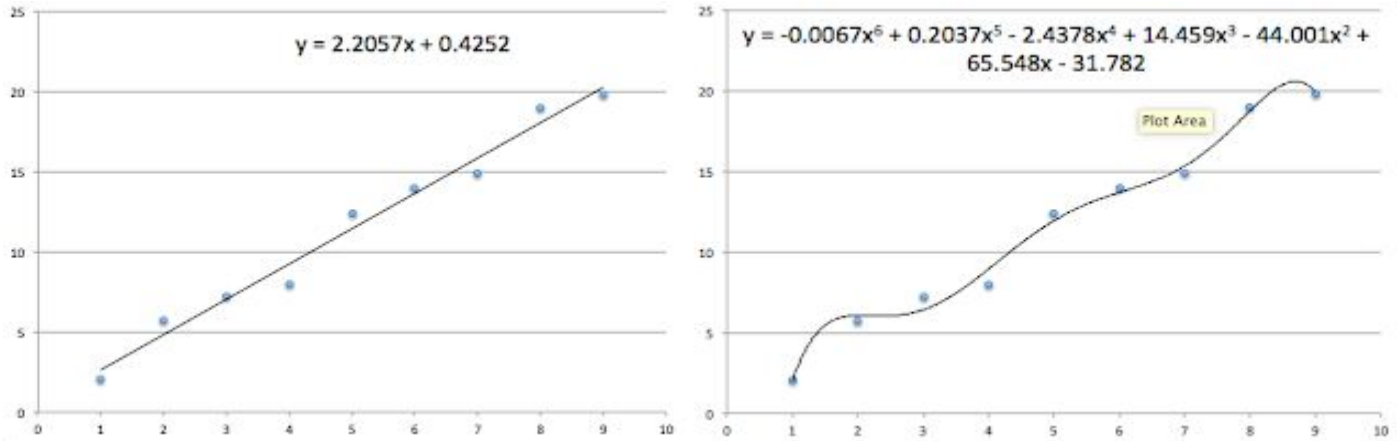
Regularization

The word regularize means to make things regular or acceptable.

Regularizations are techniques used to reduce the error by fitting a function appropriately on the given training set to avoid overfitting.

Principle of Occam's Razor states that given everything else is the same, a shorter/simpler explanation for the observed data should be preferred over a longer/complex one. In other words, this technique discourages learning a more complex or flexible model, so as to avoid the risk of overfitting.

Figure 4: Two hypotheses functions of different complexities explaining the same data



Occam's Razor states that the simpler hypothesis, which is a first-degree polynomial, describes the actual data better than the sixth order polynomial (Even though the accuracy of a sixth-order polynomial is better on the training data).

Source:
https://sites.google.com/site/machinelearningnotebook2/classification/binary-classification/overfitting-and-regularization

Bias-Variance Tradeoff

Bias is the difference between the Predicted Value and the Expected Value. To explain further, the model makes certain assumptions when it trains on the data provided. When it is introduced to the testing/validation data, these assumptions may not always be correct.

- Low Bias: An algorithm suggests fewer assumptions about the form of the target function. E.g Decision Trees, k-Nearest Neighbors and Support Vector Machines.
- High-Bias: An algorithm suggests more assumptions about the form of the target function. E.g Linear Regression, Linear Discriminant Analysis and Logistic Regression. High bias makes the algorithms fast at learning and easier to understand but generally less flexible. This leads to underfitting.

Variance is how much a target function will change if different training data was used. Machine learning algorithms that have a high variance are strongly influenced by the specifics of the training data. This means that the specifics of the training have influenced the number and types of parameters used to characterize the mapping function.

- Low Variance: Suggests small changes to the estimate of the target function with changes to the training dataset. E.g Linear Regression, Linear Discriminant Analysis and Logistic Regression.
- High Variance: Suggests large changes to the estimate of the target function with changes to the training dataset. E.g Decision Trees, k-Nearest Neighbors and Support Vector Machines. This leads to overfitting.

The goal of any supervised machine learning algorithm is to achieve low bias and low variance.

Sources

https://www.simplilearn.com/tutorials/machine-learning-tutorial/classification-in-machine-learning

https://serokell.io/blog/classification-algorithms

https://machinelearningmastery.com/types-of-classification-in-machine-learning/

https://monkeylearn.com/blog/classification-algorithms/

https://heartbeat.fritz.ai/introduction-to-machine-learning-model-evaluation-fa859e1b2d7f

https://www.analyticsvidhya.com/blog/2019/08/11-important-model-evaluation-error-metrics/

https://www.tmwr.org/resampling.html

https://sites.google.com/site/machinelearningnotebook2/classification/binary-classification/overfitting-and-regularization

https://www.cs.cmu.edu/~schneide/tut5/node42.html

Davison, A, and D Hinkley. 1997. *Bootstrap Methods and Their Application*. Vol. 1. Cambridge university press.

https://machinelearningmastery.com/gentle-introduction-to-the-bias-variance-trade-off-in-machine-learning/#:~:text=Bias%20is%20the%20simplifying%20assumptions,the%20bias%20and%20the%20variance.

https://www.analyticsvidhya.com/blog/2020/08/bias-and-variance-tradeoff-machine-learning/

Thank You.