1. Recognize the differences between supervised, semi-supervised, and unsupervised learning.

Supervised machine learning algorithms are trained on datasets that include labels added by a machine learning engineer or data scientist that guide the algorithm to understand which features are important to the problem at hand

* too much labeling can impose human biases on the model.
* Some popular examples of supervised machine learning algorithms are:
  + Linear regression for regression problems.
  + Random forest for classification and regression problems.
  + Support vector machines for classification problems
* Supervised learning problems further grouped into regression and classification problems

.Unsupervised machine learning algorithms, on the other hand, are trained on unlabeled data and must determine feature importance on their own based on inherent patterns in the data

* The most basic disadvantage of any Unsupervised Learning is that it’s application spectrum is limited.
* Some popular examples of unsupervised learning algorithms are:
  + k-means for clustering problems.
  + Apriori algorithm for association rule learning problems.
* Unsupervised learning problems can be further grouped into clustering and association problems.

Semi supervised :algorithm is trained upon a combination of labeled and unlabeled data. Typically, this combination will contain a very small amount of labeled data and a very large amount of unlabeled data

* Practical applications of Semi-Supervised Learning:-
  + **Speech Analysis**: Since labeling of audio files is a very intensive task, Semi-Supervised learning is a very natural approach to solve this problem.
  + **Internet Content Classification**: Labeling each webpage is an impractical and unfeasible process and thus uses Semi-Supervised learning algorithms.
  + **Protein Sequence Classification**: Since DNA strands are typically very large in size, the rise of Semi-Supervised learning has been imminent in this field.

2. Describe in detail any five examples of classification problems.

Ans:

### 1 - Email Spam

The goal is to predict whether an email is a *spam* and should be delivered to the Junk folder.

There are more than one method of identifying a mail as a spam. A simple method is discussed.

The raw data comprises only the text part but ignores all images. Text is a simple sequence of words which is the input (X). The goal is to predict the binary response Y: spam or not.

The first step is to process the raw data into a vector, which can be done in several ways. The method followed here is based on the relative frequencies of most common words and punctuation marks in e-mail messages. A set of 57 such words and punctuation marks are pre-selected by researchers. This is where domain knowledge plays an important role.

Given these 57 most commonly occurring words and punctuation marks, then, in every e-mail message we would compute a relative frequency for each word, i.e., the percentage of times this word appears with respect to the total number of words in the email message.

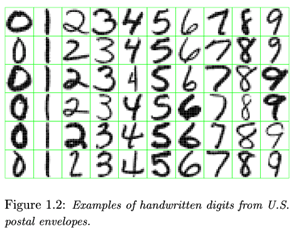
In the current example, 4601 email messages were considered in the training sample. These e-mail messages were identified as either a good e-mail or spam after reading the emails and assuming implicitly that human decision is perfect (an arguable point!). Relative frequency of the 57 most commonly used words and punctuation based on this set of emails was constructed. This is an example of supervised learning as in the training data the response Y is known.

In the future when a new email message is received, the algorithm will analyze the text sequence and compute the relative frequency for these 57 identified words. This is the new input vector to be classified into spam or not through the learning algorithm.

### 2 - Handwritten Digit Recognition

The goal is to identify images of single digits 0 - 9 correctly.

The raw data comprises images that are scaled segments from five-digit ZIP codes. In the diagram below every green box is one image. The original images are very small, containing only 16 × 16 pixels. For convenience the images below are enlarged, hence the pixelation or 'boxiness' of the numbers.



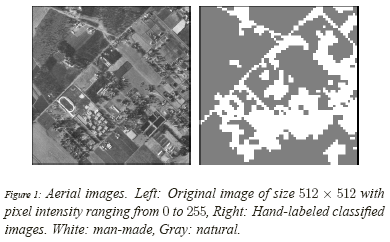
Every image is to be identified as 0 or 1 or 2 ... or 9. Since the numbers are handwritten, the task is not trivial. For instance, a '5' sometimes can very much look like a '6', and '7' is sometimes confused with '1'.

To the computer, an image is a matrix, and every pixel in the image corresponds to one entry in the matrix. Every entry is an integer ranging from a pixel intensity of 0 (black) to 255 (white). Hence the raw data can be submitted to the computer directly without any feature extraction. The image matrix was scanned row by row and then arranged into a large 256-dimensional vector. This is used as the input to train the classifier. Note that this is also a supervised learning algorithm where Y, the response, is multi-level and can take 10 values.

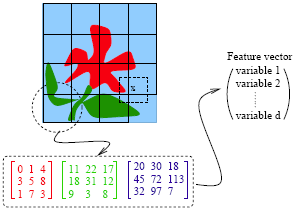
### 3 - Image segmentation

Here is a more complex example of an image processing problem. The satellite images are to be identified into man-made or natural regions. For instance, in the aerial images shown below, buildings are labeled as man-made, and the vegetation areas are labeled as natural.

These grayscale images are much larger than the previous example. These images are 512 × 512 pixels and again because these are grayscale images we can present pixel intensity with numbers 0 to 255.



In the previous example of hand-written image identification, because of the small size of the images, no feature extraction was done. However in this problem feature extraction is necessary. A standard method of feature extraction in an image processing problem is to divide images into blocks of pixels or to form a neighborhood around each pixel. As is shown in the following diagram, after dividing the images into blocks of pixels or forming a neighborhood around each pixel, each block may be described by several features. As we have seen in the previous example, grayscale images can be represented by one matrix. Every entry in a greyscale image is an integer ranging from a pixel intensity of 0 (black) to 255 (white). Color images are represented by values of RGB (red, green and blue). Color images, therefore, are represented by 3 such matrices as seen below.



For each block, a few features (or statistics) may be computed using the color vectors for the pixels in the block. This set forms a feature vector for every block.

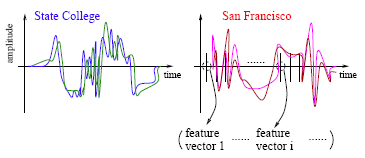
Examples of features:

* Average of R, G and B values for pixels in one block
* Variance of the brightness of the pixels (brightness is the average of RGB color values). Small variance indicates the block is visually smooth.

The feature vectors for the blocks sometimes are treated as independent samples from an unknown distribution. Ignoring f the spatial dependence among feature vectors results in performance loss. To make the learning algorithm efficient the spatial dependence needs to be exploited. Only then the accuracy in classification will improve.

### 4 - Speech Recognition

Another interesting example of data mining deals with speech recognition. For instance, if you call the University Park Airport, the system might ask you your flight number, or your origin and destination cities. The system does a very good job recognizing city names. This is a classification problem, in which each city name is a class. The number of classes is very big but finite.



The raw data involves voice amplitude sampled at discrete time points (a time sequence), which may be represented in the waveforms as shown above. In speech recognition, a very popular method is the *Hidden Markov Model*.

At every time point, one or more features, such as frequencies, are computed. The speech signal essentially becomes a sequence of frequency vectors. This sequence is assumed to be an instance of a hidden Markov model (HMM). An HMM can be estimated using multiple sample sequences under the same class (e.g., city name).

##### *Hidden Markov Model (HMM) Methodology:*

HMM captures the time dependence of the feature vectors. The HMM has unspecified parameters that need to be estimated. Based on the sample sequences, model estimation takes place and an HMM is obtained. This HMM is like a mathematical signature for each word. Each city name, for example, will have a different signature. In the diagram above the signatures corresponding to State College and San Francisco are compared. It is possible that several models are constructed for one word or phrase. For instance, there may be a model for a female voice as opposed to another for a male voice.

When a customer calls in for information and utters origin or destination city pairs, the system computes the likelihood of what the customer uttered under possibly thousands of models. The system finds the HMM that yields the maximum likelihood and identifies the word as the one associated with that HMM.

### 5 - DNA Expression Microarray

Our goal here is to identify disease or tissue types based on the gene expression levels.

For each sample taken from a tissue of a particular disease type, the expression levels of a very large collection of genes are measured. The input data goes through a data cleaning process. Data cleaning may include but is certainly not limited to, normalization, elimination of noise and perhaps log-scale transformations. A large volume of literature exists on the topic of cleaning microarray data.

In the example considered 96 samples were taken from 9 classes or types of tissues. It was expensive to collect the tissue samples, at least in the early days. Therefore, the sample size is often small but the dimensionality of data is very high. Every sample is measured on 4026 genes. very often microarray data analysis has its own challenges with a small number of observations and very large number of features from each observation.

### 6 - DNA Sequence Classification

Each genome is made up of DNA sequences and each DNA segment has specific biological functions. However there are DNA segments which are non-coding, i.e. they do not have any biological function (or their functionalities are not yet known). One problem in DNA sequencing is to label the sampled segments as coding or non-coding (with a biological function or without).

The raw DNA data comprises sequences of letters, e.g., A, C, G, T for each of the DNA sequences. One method of classification assumes the sequences to be realizations of random processes. Different random processes are assumed for different classes of sequences.

3. Describe each phase of the classification process in detail.

Ans:

**Data acquisition and segmentation: Acquiring the data from source**

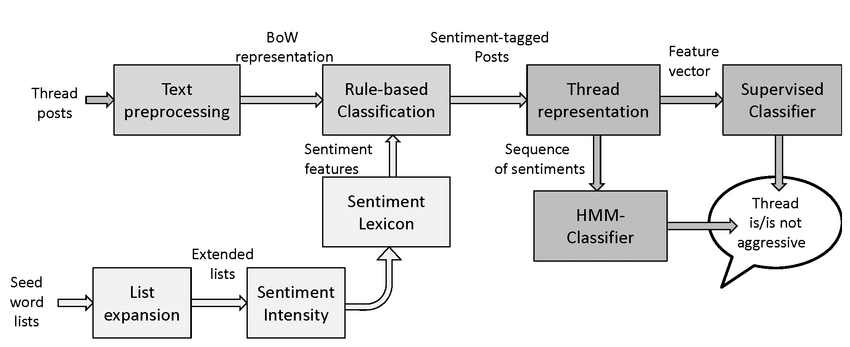
**Data preprocessing:**

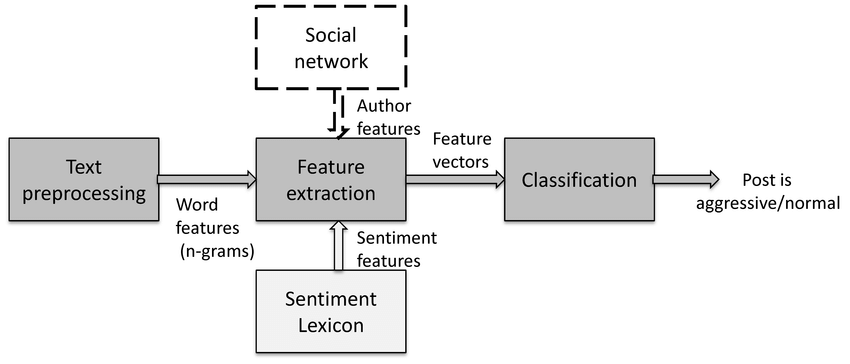
**Learning Step (Training Phase)**: Construction of Classification Model   
Different Algorithms are used to build a classifier by making the model learn using the training set available. The model has to be trained for the prediction of accurate results.

**Classification Step**: Model used to predict class labels and testing the constructed model on test data and hence estimate the accuracy of the classification rules.



The proposed pipeline for detecting an aggressive behaviour in the text threads:



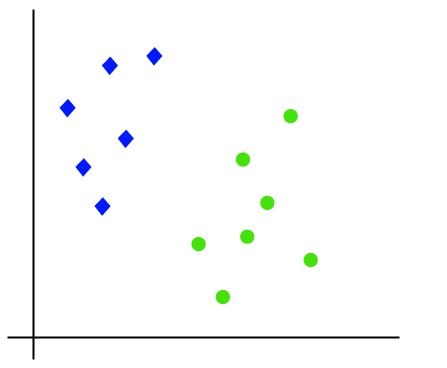


**4. Go through the SVM model in depth using various scenarios.**

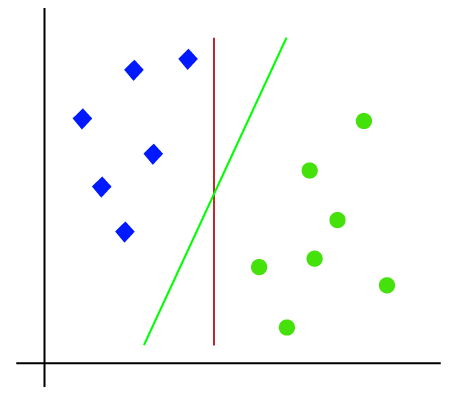
**Ans:**

**Linear SVM:**

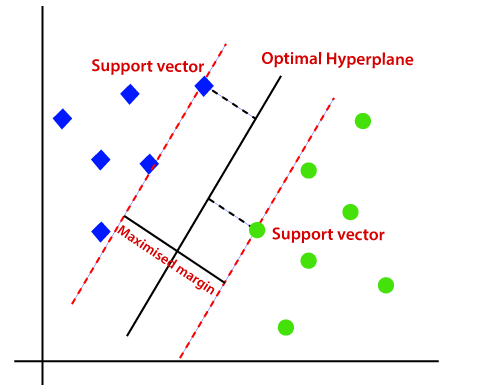
Scenario: Suppose we have a dataset that has two tags (green and blue), and the dataset has two features x1 and x2. We want a classifier that can classify the pair(x1, x2) of coordinates in either green or blue. Consider the below image:



So as it is 2-d space so by just using a straight line, we can easily separate these two classes. But there can be multiple lines that can separate these classes. Consider the below image:

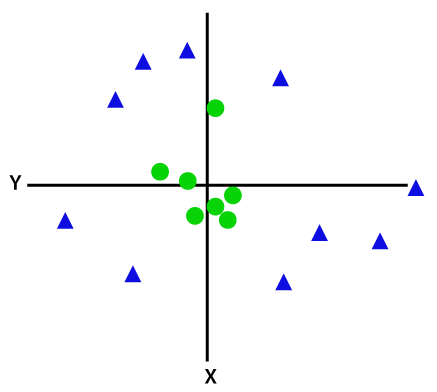


Hence, the SVM algorithm helps to find the best line or decision boundary; this best boundary or region is called a **hyperplane**. SVM algorithm finds the closest point of the lines from both the classes. These points are called support vectors. The distance between the vectors and the hyperplane is called the margin. And the goal of SVM is to maximize this margin. The **hyperplane** with maximum margin is called the **optimal hyperplane**.



**Non-Linear SVM:**

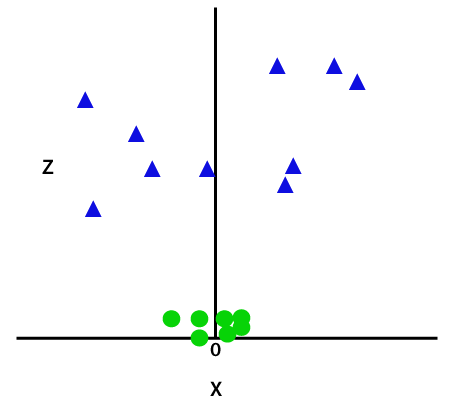
Consider the below image:



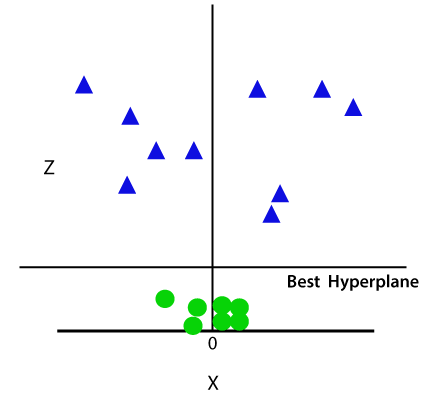
So to separate these data points, we need to add one more dimension. For linear data, we have used two dimensions x and y, so for non-linear data, we will add a third dimension z. It can be calculated as

z=x2 +y2

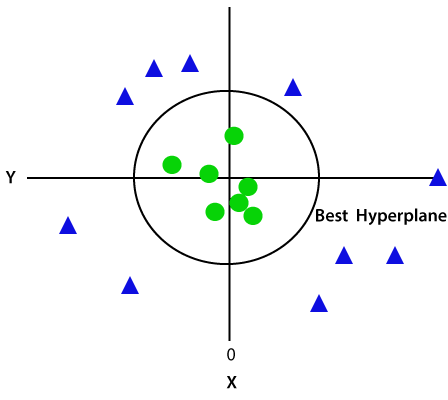
By adding the third dimension, the sample space will become as below image:



So now, SVM will divide the datasets into classes in the following way. Consider the below image:



Since we are in 3-d Space, hence it is looking like a plane parallel to the x-axis. If we convert it in 2d space with z=1, then it will become a



Hence we get a circumference of radius 1 in case of non-linear data.

5. What are some of the benefits and drawbacks of SVM?

Ans:

**Advantages of support vector machine:**

* Support vector machine works comparably well when there is an understandable margin of dissociation between classes.
* It is more productive in high-dimensional spaces.
* It is effective in instances where the number of dimensions is larger than the number of specimens.
* Support vector machine is comparably memory systematic
* Handling high-dimensional data: SVMs are effective in handling high-dimensional data, which is common in many applications such as image and text classification.
* Handling small datasets: SVMs can perform well with small datasets, as they only require a small number of support vectors to define the boundary.
* Modeling non-linear decision boundaries: SVMs can model non-linear decision boundaries by using the kernel trick, which maps the data into a higher-dimensional space where the data becomes linearly separable.
* Robustness to noise: SVMs are robust to noise in the data, as the decision boundary is determined by the support vectors, which are the closest data points to the boundary.
* Generalization: SVMs have good generalization performance, which means that they are able to classify new, unseen data well.
* Versatility: SVMs can be used for both classification and regression tasks, and it can be applied to a wide range of applications such as natural language processing, computer vision and bioinformatics.
* Sparse solution: SVMs have sparse solutions, which means that they only use a subset of the training data to make predictions. This makes the algorithm more efficient and less prone to overfitting.
* Regularization: SVMs can be regularized, which means that the algorithm can be modified to avoid overfitting.
* advantages of SVM and support vector regression include that they can be used to avoid the difficulties of using linear functions in the high-dimensional feature space, and the optimization problem is transformed into dual convex quadratic programs

**Disadvantages of support vector machine:**

* Support vector machine algorithm is not acceptable for large data sets.
* It does not execute very well when the data set has more sound i.e. target classes are overlapping.
* In cases where the number of properties for each data point outstrips the number of training data specimens, the support vector machine will underperform.
* As the support vector classifier works by placing data points, above and below the classifying hyperplane there is no probabilistic clarification for the classification.
* Computationally expensive: SVMs can be computationally expensive for large datasets, as the algorithm requires solving a quadratic optimization problem.
* Choice of kernel: The choice of kernel can greatly affect the performance of an SVM, and it can be difficult to determine the best kernel for a given dataset.
* Sensitivity to the choice of parameters: SVMs can be sensitive to the choice of parameters, such as the regularization parameter, and it can be difficult to determine the optimal parameter values for a given dataset.
* Memory-intensive: SVMs can be memory-intensive, as the algorithm requires storing the kernel matrix, which can be large for large datasets.
* Limited to two-class problems: SVMs are primarily used for two-class problems, although multi-class problems can be solved by using one-versus-one or one-versus-all strategies.
* Lack of probabilistic interpretation: SVMs do not provide a probabilistic interpretation of the decision boundary, which can be a disadvantage in some applications.
* Not suitable for large datasets with many features: SVMs can be very slow and can consume a lot of memory when the dataset has many features.
* Not suitable for datasets with missing values: SVMs requires complete datasets, with no missing values, it can not handle missing values.

**6. Go over the kNN model in depth.**

Ans: kNN is the simplest machine learning algorithm to understand and also to explain.

* It is a versatile algorithm i.e. useful for both classification and regression.
* One big advantage is that kNN has no pre assumptions about the data. “ Let the data speak for itself ”.

**Definition:**

kNN is ***non-parametric***, ***instance based***, ***lazy algorithm*** and used in the supervised setting.

## Non-parametric :

It means that algorithm has no pre assumptions about the functional form of the model, to avoid mismodeling .

## Instance based :

It means that our algorithm does not explicitly learn a model.

Instead, it memorize the training instances which are subsequently used as “knowledge” for the prediction.

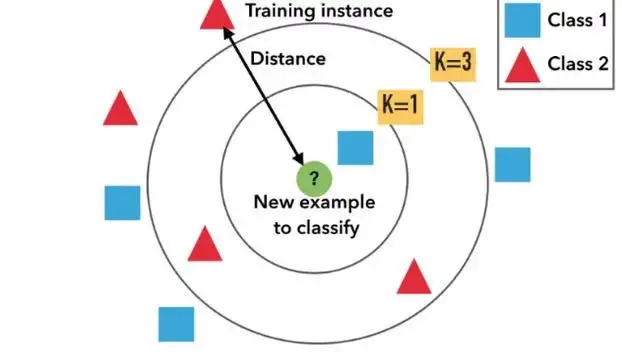
## Lazy algorithm :

It means that it does not use the training data for the Generalization i.e. these algorithm has no explicit training phase or it is minimal. Training is very fast.

**kNN Algorithm for Classification :**

Training element {xi, yi} , Testing point(x)

1. Compute the Distance D(x,xi) to every training element xi.
2. Select k closest instance xi1,xi2,…….., xik and their labels yi1, yi2 …, yik.
3. Output the class y\* which is most frequent in yi1,yi2 ……yik.



Have you pay attention to the Distance D(x, xi) ,

## Euclidian (Numerical Attributes):



1. It is most popular and widely used distance measure.
2. It is valid for continuous variables.
3. One drawback is that , it is sensitive to the outliers(single extreme difference attribute).

## Hamming (Categorical Attributes) :



1. It is used to calculate distance between binary vectors.
2. It is only valid for discrete variables.

Other distance measures can be used : Manhattan, KL divergence ,Custom Distance measures (tf.idf for text).

## Significant of “k” :

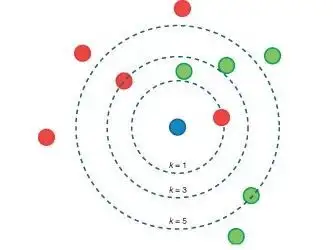
* Value of k has strong effect on kNN performance.
* k act as controller to decide the shape of decision boundary.
* Large value of k has following properties:

\_ \_ \_ 1. Smoother decision boundary .

\_ \_ \_ 2. It provide more voters for prediction, it implies less affect from outliers.

\_ \_ \_ 3. As a result has Lower Variance and High Bias.

To know more about Bias and Variance.



* Small value of k has following disadvantages:

\_ \_1. We found Unstable decision boundary.

\_ \_2. Small change in training set implies large change in classification accuracy.

## How to Select k :

* The simplest solution is Cross Validation.
* Best method is to try many k values and use Cross-Validation to see which k value is giving the best result.

## Cons of kNN Algorithm :

* It is computationally expensive algorithm.(As it store all training data)
* High memory requirement.
* Prediction is Slow for large n (data observations).

**7. Discuss the kNN algorithm's error rate and validation error.**

**Ans:**

**1.Training error Rate 2.Validation Error Rate**

|  |  |
| --- | --- |

If we observe the training error rate graph it can be seen that error increases for increasing value of K,also error is zero for K=1.This is because the closest point to any training data point is itself.Hence the prediction is always accurate with K=1.

If the validation error curve would have been similar, our choice of K would have been 1.

By observing validation error rate we can interpret that At K=1, we were over fitting the boundaries. In the Validation graph Error rate initially decreases and reaches a minimum. After the minimum point, it then increases with increasing K. This value of K where error reaches minima should be used for all predictions.

8. For kNN, talk about how to measure the difference between the test and training results.

Ans:

KNN algorithm can also be used for regression problems.The only difference will be using averages of nearest neighbors rather than voting from nearest neighbors.

The most popular distance metrics used are :

### Measures for numerical data

* Euclidean Distance(most popular)
* Manhattan Distance
* Chebyshev Distance
* Euclidean Distance :

9. Create the kNN algorithm.

Ans:

Steps:

1. Determine the number of nearest neighbours (K values).

2. Compute the distance between test sample and all the training samples.

3. Sort the distance and determine nearest neighbours based on the K-th minimum distance.

4. Assemble the categories of the nearest neighbours.

5. Utilise simple majority of the category of nearest neighbours as the prediction value of the new data object.

# Import necessary modules

from sklearn.neighbors import KNeighborsClassifier

from sklearn.model\_selection import train\_test\_split

from sklearn.datasets import load\_iris

# Loading data

irisData = load\_iris()

# Create feature and target arrays

X = irisData.data

y = irisData.target

# Split into training and test set

X\_train, X\_test, y\_train, y\_test = train\_test\_split(

X, y, test\_size = 0.2, random\_state=42)

knn = KNeighborsClassifier(n\_neighbors=7)

knn.fit(X\_train, y\_train)

# Predict on dataset which model has not seen before

print(knn.predict(X\_test))

In the example shown above following steps are performed:

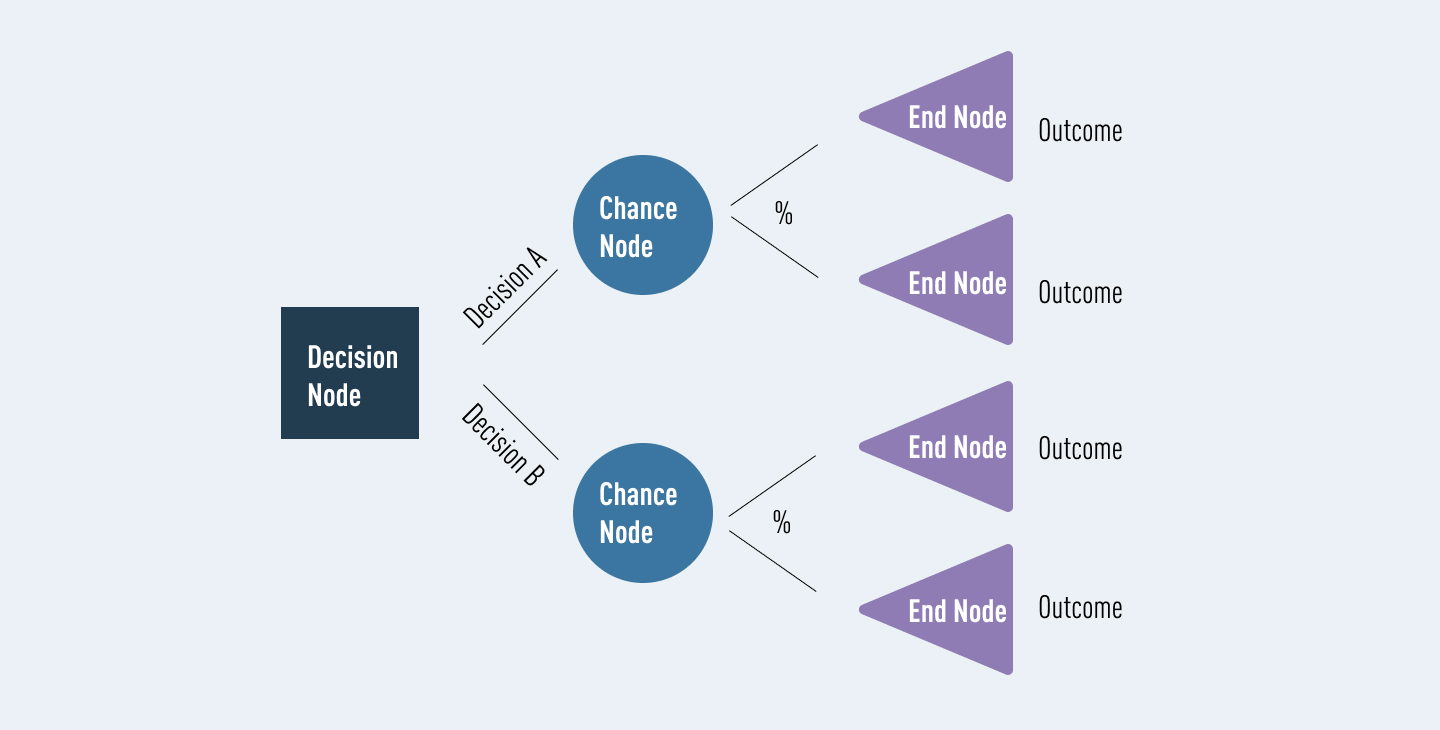
1. The k-nearest neighbor algorithm is imported from the scikit-learn package.
2. Create feature and target variables.
3. Split data into training and test data.
4. Generate a k-NN model using neighbors value.
5. Train or fit the data into the model.
6. Predict the future.

10. What is a decision tree, exactly? What are the various kinds of nodes? Explain all in depth.

Ans: In its simplest form, a decision tree is a type of flowchart that shows a clear pathway to a decision. In terms of data analytics, it is a type of algorithm that includes conditional ‘control’ statements to classify data. A decision tree starts at a single point (or ‘node’) which then branches (or ‘splits’) in two or more directions. Each branch offers different possible outcomes, incorporating a variety of decisions and chance events until a final outcome is achieved.

Decision trees ultimately consist of just three key parts, or ‘nodes’:

* Decision nodes: Representing a decision (typically shown with a square)
* Chance nodes: Representing probability or uncertainty (typically denoted by a circle)
* End nodes: Representing an outcome (typically shown with a triangle)



Some other terms you might come across will include:

* Root nodes

In the diagram above, the blue decision node is what we call a ‘root node.’ This is always the first node in the path. It is the node from which all other decision, chance, and end nodes eventually branch.

* Leaf nodes

In the diagram above, the lilac end nodes are what we call ‘leaf nodes.’ These show the end of a decision path (or outcome). You can always identify a leaf node because it doesn’t split, or branch any further. Just like a real leaf!

* Internal nodes

Between the root node and the leaf nodes, we can have any number of internal nodes. These can include decisions and chance nodes (for simplicity, this diagram only uses chance nodes). It’s easy to identify an internal node—each one has branches of its own while also connecting to a previous node.

11. Describe the different ways to scan a decision tree.

12. Describe in depth the decision tree algorithm.

Ans: Tree depth is **a measure of how many splits a tree can make before coming to a prediction**. This process could be continued further with more splitting until the tree is as pure as possible. The problem with many repetitions of this process is that this can lead to a very deep classification tree with many nodes.

In general, the depth of the tree may not depend on the number of features, but it may just depend on the labels or training examples (which is a degenerate case), although, in most cases, it will also depend on the number of features, because each node represents a split of the training examples based on some condition that needs to be true for some feature (e.g. the height of the people must be less than 150cm).

The depth of the tree depends on how your algorithm builds the tree. For a fixed dataset

D, there're many algorithms, such as ID3, C4.5, CART, etc. (and their variants) to build your tree. For the most part, these algorithms recursively partition the dataset, so it's never possible to get a tree larger than D. Large/deep trees are also prone to overfitting and are computationally expensive, so these algorithms typically prune the tree so it's much smaller than D.

13. In a decision tree, what is inductive bias? What would you do to stop overfitting?

Ans:

In the case of decision trees, the depth of the trees is the inductive bias. If the depth of the tree is too low, then there is too much generalisation in the model.

Trees that place high information gain attributes close to the root

**Avoiding overfitting**

– How can we avoid overfitting?

• Stop growing before it reaches the point where it perfectly classifies the training data

• Grow full tree, then post-prune

– How to select the best tree?

• Measure performance statistically over training data

• Measure performance over separate validation data set

• MDL: minimize the complexity for encoding the training examples and the decision trees

14.Explain advantages and disadvantages of using a decision tree?

### Advantages of decision trees

* Good for interpreting data in a highly visual way.
* Good for handling a combination of numerical and non-numerical data.
* Easy to define rules, e.g. ‘yes, no, if, then, else…’
* Requires minimal preparation or data cleaning before use.
* Great way to choose between best, worst, and likely case scenarios.
* Can be easily combined with other decision-making techniques.

### Disadvantages of decision trees

* Overfitting (where a model interprets meaning from irrelevant data) can become a problem if a decision tree’s design is too complex.
* They are not well-suited to continuous variables (i.e. variables which can have more than one value, or a spectrum of values).
* In predictive analysis, calculations can quickly grow cumbersome, especially when a decision path includes many chance variables.
* When using an imbalanced dataset (i.e. where one class of data dominates over another) it is easy for outcomes to be biased in favor of the dominant class.
* Generally, decision trees provide lower prediction accuracy compared to other predictive algorithms.

15. Describe in depth the problems that are suitable for decision tree learning.

They’re commonly used by data analysts to carry out predictive analysis (e.g. to develop operations strategies in businesses). They’re also a popular tool for machine learning and artificial intelligence, where they’re used as training algorithms for supervised learning (i.e. categorizing data based on different tests, such as ‘yes’ or ‘no’ classifiers.)

Broadly, decision trees are used in a wide range of industries, to solve many types of problems. Because of their flexibility, they’re used in sectors from technology and health to financial planning. Examples include:

* A technology business evaluating expansion opportunities based on analysis of past sales data.
* A toy company deciding where to target its limited advertising budget, based on what demographic data suggests customers are likely to buy.
* Banks and mortgage providers using historical data to predict how likely it is that a borrower will default on their payments.
* Emergency room triage might use decision trees to prioritize patient care (based on factors such as age, gender, symptoms, etc.)
* Automated telephone systems guiding you to the outcome you need, e.g. ‘For option A, press 1. For option B, press 2’, and so on.

16. Describe in depth the random forest model. What distinguishes a random forest?

Ans: Random forests are **a way of averaging multiple deep decision trees, trained on different parts of the same training set, with the goal of reducing the variance**.

Forests are like the pulling together of decision tree algorithm efforts. Taking the teamwork of many trees thus improving the performance of a single random tree.

This comes at the expense of a small increase in the bias and some loss of interpretability, but generally greatly boosts the performance in the final model.

Random forests or random decision forests is an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time.

For classification tasks, the output of the random forest is the class selected by most trees.

For regression tasks, the mean or average prediction of the individual trees is returned

Random decision forests correct for decision trees' habit of overfitting to their training set.Random forests generally outperform decision trees, but their accuracy is lower than gradient boosted trees.However, data characteristics can affect their performance.The proper introduction of random forests was made in a paper by Leo Breiman.This paper describes a method of building a forest of uncorrelated trees using a CART like procedure, combined with randomized node optimization and bagging. In addition, this paper combines several ingredients, some previously known and some novel, which form the basis of the modern practice of random forests, in particular:

* Using out-of-bag error as an estimate of the generalization error.
* Measuring variable importance through permutation.

The random forest algorithm is an extension of the bagging method as it utilizes both bagging and feature randomness to create an uncorrelated forest of decision trees. Feature randomness, also known as feature bagging or “the random subspace method” generates a random subset of features, which ensures low correlation among decision trees. This is a key difference between decision trees and random forests. While decision trees consider all the possible feature splits, random forests only select a subset of those features.

17. In a random forest, talk about OOB error and variable value.

Ans:

**Out-of-bag** (**OOB**) **error**, also called **out-of-bag estimate**, is a method of measuring the [prediction error](https://en.wikipedia.org/wiki/Prediction_error) of [random forests](https://en.wikipedia.org/wiki/Random_forest), [boosted decision trees](https://en.wikipedia.org/wiki/Gradient_boosting), and other [machine learning](https://en.wikipedia.org/wiki/Machine_learning) models utilizing [bootstrap aggregating](https://en.wikipedia.org/wiki/Bootstrap_aggregating) (bagging). Bagging uses subsampling with replacement to create training samples for the model to learn from.

OOB error is the mean prediction error on each training sample *xi*, using only the trees that did not have *xi* in their bootstrap sample.

### Variable importance

Random forests can be used to rank the importance of variables in a regression or classification problem in a natural way. The following technique was described in Breiman's original paper and is implemented in the [R](https://en.wikipedia.org/wiki/R_(programming_language)) package *randomForest*.

The first step in measuring the variable importance in a data set D={(Xi,Yi)}ni=1  is to fit a random forest to the data. During the fitting process the [out-of-bag error](https://en.wikipedia.org/wiki/Out-of-bag_error) for each data point is recorded and averaged over the forest (errors on an independent test set can be substituted if bagging is not used during training).

To measure the importance of the j-th feature after training, the values of the j-th feature are permuted among the training data and the out-of-bag error is again computed on this perturbed data set. The importance score for the j-th feature is computed by averaging the difference in out-of-bag error before and after the permutation over all trees. The score is normalized by the standard deviation of these differences.

Features which produce large values for this score are ranked as more important than features which produce small values. The statistical definition of the variable importance measure was given and analyzed by Zhu *et al.*

This method of determining variable importance has some drawbacks. For data including categorical variables with different number of levels, random forests are biased in favor of those attributes with more levels. Methods such as [partial permutations](https://en.wikipedia.org/wiki/Partial_permutation) and growing unbiased trees can be used to solve the problem. If the data contain groups of correlated features of similar relevance for the output, then smaller groups are favored over larger groups