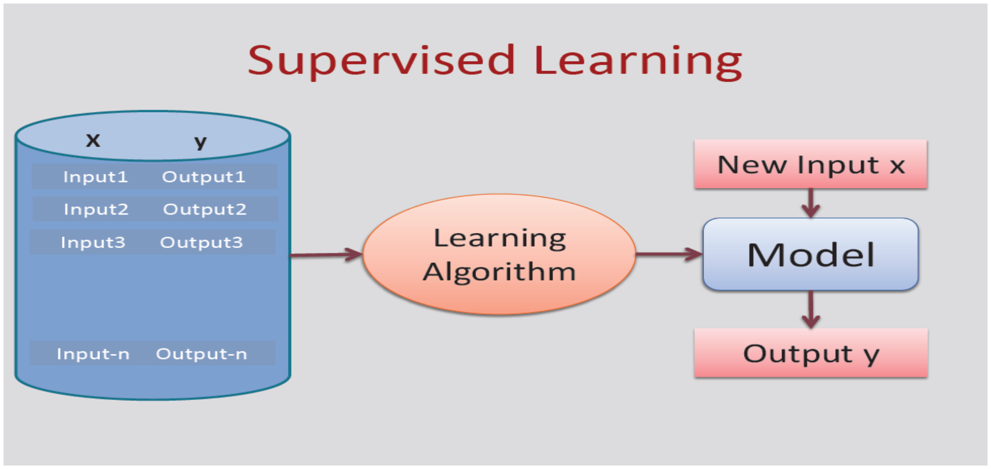
Q1. Difference between Supervised and Unsupervised Machine Learning.

• Supervised Learning

– X,y (pre-classified training examples)

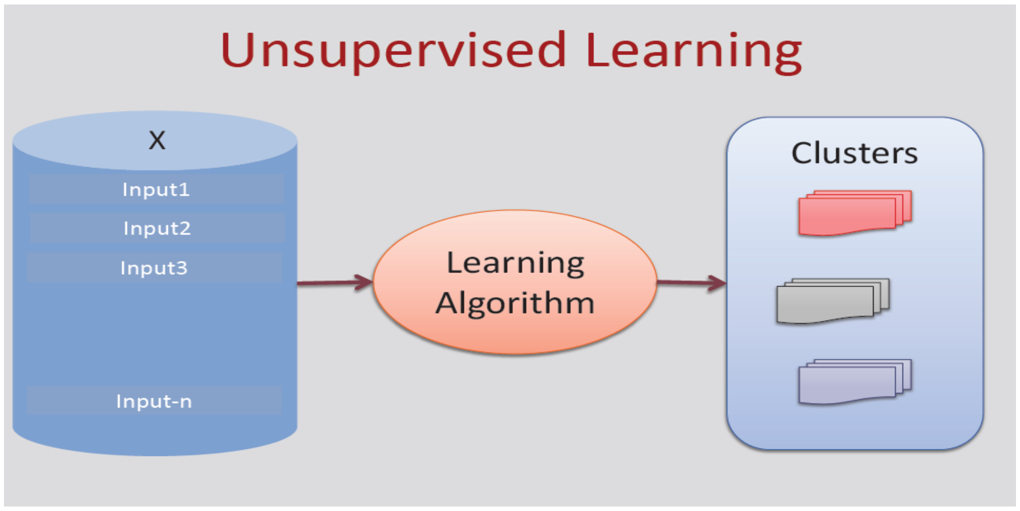
– Given an observation x, what is the best label for y?



• Unsupervised learning

– X

– Given a set of x's , cluster or summarize them



2. What is Parametric and Nonparametric Machine Learning Algorithms?

S.No.	Parametric Methods	Non-Parametric Methods
1.	Parametric Methods uses a fixed number of parameters to build the model.	Non-Parametric Methods use the flexible number of parameters to build the model.
2.	Parametric analysis is to test group means.	A non-parametric analysis is to test medians.
3.	It is applicable only for variables.	It is applicable for both – Variable and Attribute.
4.	It always considers strong assumptions about data.	It generally fewer assumptions about data.
5.	Parametric Methods require lesser data than Non-Parametric Methods.	Non-Parametric Methods requires much more data than Parametric Methods.
6.	Parametric methods assumed to be a normal distribution.	There is no assumed distribution in non-parametric methods.
7.	Parametric data handles – Intervals data or ratio data.	But non-parametric methods handle original data.
8.	Here when we use parametric methods then the result or outputs generated can be easily affected by outliers.	When we use non-parametric methods then the result or outputs generated cannot be seriously affected by outliers.
9.	Parametric Methods can perform well in many situations but its performance is at peak (top) when the spread of each group is different.	Similarly, Non-Parametric Methods can perform well in many situations but its performance is at peak (top) when the spread of each group is the same.

10.	Parametric methods have more statistical power than Non-Parametric methods.	Non-parametric methods have less statistical power than Parametric methods.
11.	As far as the computation is considered these methods are computationally faster than the Non-Parametric methods.	As far as the computation is considered these methods are computationally faster than the Parametric methods.
12.	Examples – Logistic Regression, Naïve Bayes Model, etc.	Examples – KNN, Decision Tree Model, etc

Q3 Explain the K Nearest Neighbor Algorithm in details

Training method:

Save the training examples

At prediction time:

Find the k training examples (x1,y1),…(xk,yk) that are closest to the test example x

Predict the most frequent class among those yi's.

The k-nearest-neighbor method was first described in the early 1950s. The method is labor intensive when given large training sets, and did not gain popularity until the 1960s when increased computing power became available. It has since been widely used in the area of pattern recognition. Nearest-neighbor classifiers are based on learning by analogy, that is, by comparing a given test tuplewith training tuples that are similar to it. The training tuples are described by n attributes. Each tuple represents a point in an n-dimensional space. In this way, all of the training tuples are stored in an n-dimensional pattern space. When given an unknown tuple, a k-nearest-neighbor classifier searches the pattern space for the k training tuples that are closest to the unknown tuple. These k training tuples are the k "nearest neighbors" of the unknown tuple. "Closeness" is defined in terms of a distance metric, such as Euclidean distance. The Euclidean distance between two points or tuples, say, X1 = (x11, x12, : : : , x1n) and X2 = (x21, x22, : : : , x2n), is

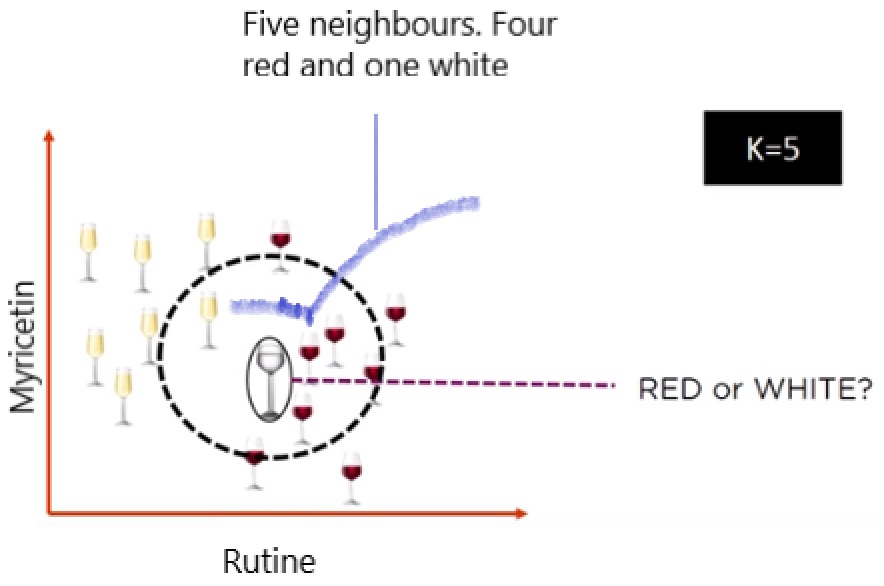
$$dist(X_1, X_2) = \sqrt{\sum_{i=1}^{n} (x_{1i} - x_{2i})^2}.$$

Q4. What is the difference between KNN and K-means

1. K-NN is a Supervised machine learning while K-means is an unsupervised machine learning.
2. K-NN is a classification or regression machine learning algorithm while K-means is a clustering machine learning algorithm.
3. K-NN is a lazy learner while K-Means is an eager learner. An eager learner has a model fitting that means a training step but a lazy learner does not have a training phase.
4. K-NN performs much better if all of the data have the same scale but this is not true for K-means.

Q5. What is the "K" in KNN algorithm?

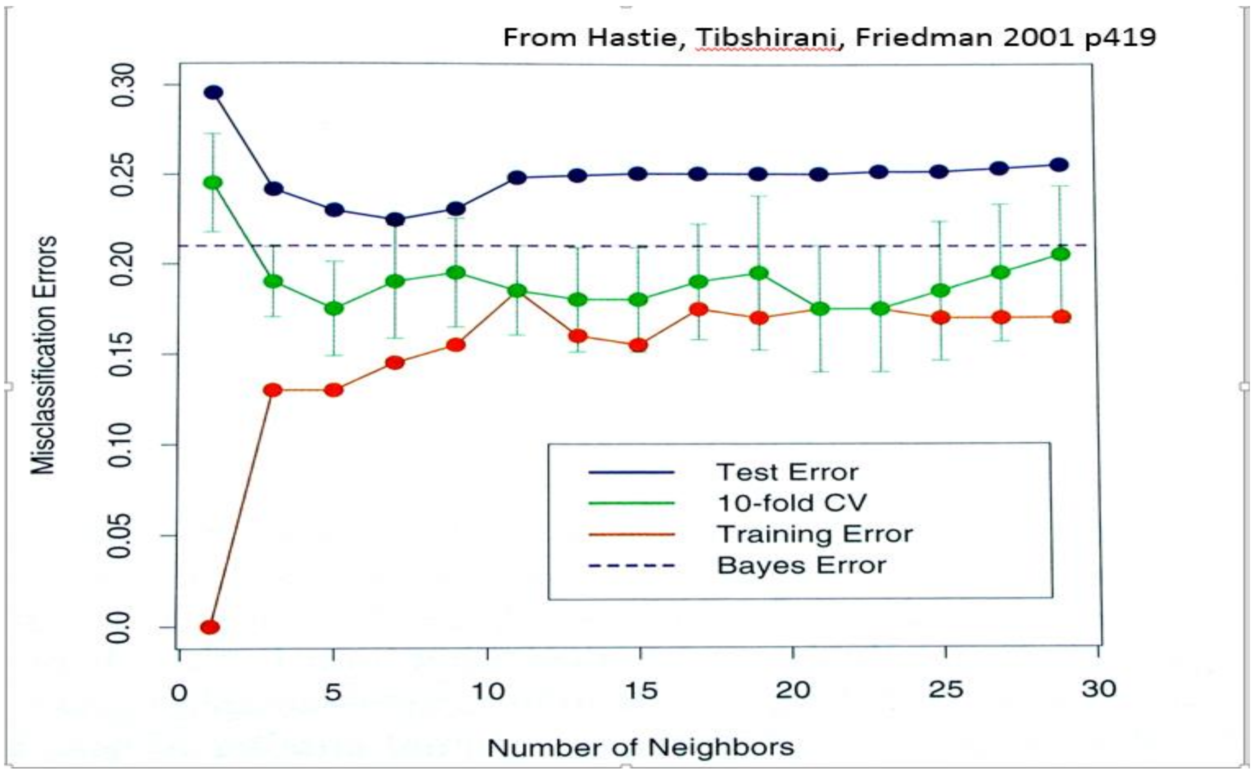
'k' in KNN is a parameter that refers to the number of nearest neighbours to include in the majority of the voting process. ... Let's say k = 5 and the new data point is classified by the majority of votes from its five neighbours and the new point would be classified as red since four out of five neighbours are red



Q6 How do we decide the value of "K" in KNN algorithm?

- Large k:
  - less sensitive to noise (particularly class noise)
  - better probability estimates for discrete classes
  - larger training sets allow larger values of k

- Small k:
  - captures fine structure of problem space better
  - may be necessary with small training sets
- Balance must be struck between large and small k
- As training set approaches infinity, and k grows large, kNN becomes Bayes optimal



Q7. Why is the odd value of "K" preferable in KNN algorithm?

K should be odd so that there are no ties in the voting.

Q8 What distance metrics can be used in KNN?

Following distance metrics is used in KNN

1. The most popular distance measure is Euclidean distance, which is defined as

$$d(i,j) = \sqrt{(x_{i1}-x_{j1})^2 + (x_{i2}-x_{j2})^2 + \cdots + (x_{in}-x_{jn})^2},$$

2. Another well-known metric is Manhattan (or city block) distance, defined as

$$d(i, j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \cdots + |x_{in} - x_{jn}|.$$

3. Minkowski distance is a generalization of both Euclidean distance and Manhattan distance. It is defined as.

$$d(i, j) = (|x_{i1} - x_{j1}|^p + |x_{i2} - x_{j2}|^p + \cdots + |x_{in} - x_{jn}|^p)^{1/p},$$

Q9. What is the difference between Euclidean Distance and Manhattan distance? What is the formula of Euclidean distance and Manhattan distance?

Manhattan distance is usually preferred over the more common Euclidean distance when there is high dimensionality in the data

Euclidean distance:

$$d(i, j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \cdots + (x_{in} - x_{jn})^2},$$

Manhattan (or city block) distance:

$$d(i, j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \cdots + |x_{in} - x_{jn}|.$$

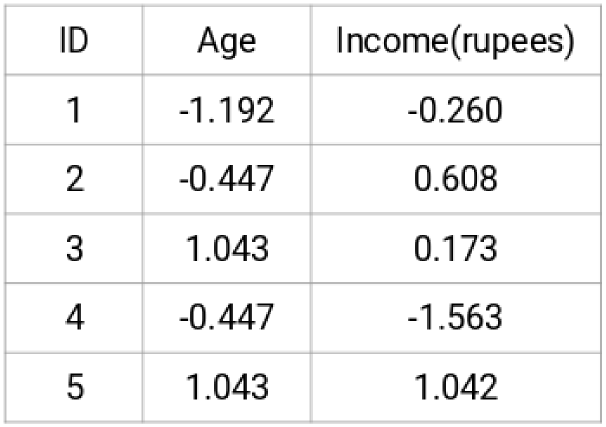
Q10 . Why do you need to scale your data for the k-NN algorithm

All such distance based algorithms are affected by the scale of the variables. Consider your data has an age variable which tells about the age of a person in years and an income variable which tells the monthly income of the person in rupees

ID	Age	Income(rupees)
1	25	80,000
2	30	100,000
3	40	90,000
4	30	50,000
5	40	110,000

:

Here the Age of the person ranges from 25 to 40 whereas the income variable ranges from 50,000 to 110,000. Let's now try to find the similarity between observation 1 and 2. The most common way is to calculate the Euclidean distance and remember that smaller this distance closer will be the points and hence they will be more similar to each other.
For now, we will be focusing on normalization. You can try min-max scaling as well. Let's see how normalization can bring down these variables to same scale and hence improve the performance of these distance based algorithms. If we normalize the above data, it will look like



ID	Age	Income(rupees)
1	-1.192	-0.260
2	-0.447	0.608
3	1.043	0.173
4	-0.447	-1.563
5	1.043	1.042

Q11 What are the Gradient Descent Based, Tree-Based, and distance-based algorithms?

Gradient Descent Based
1. Linear Regression
2. Logistic Regression

Tree-Based

1. Decision Tree
2. Random Forest
3. XGBoost

Distance-based algorithms?
1. KNN
2. K-Means
3. Support vector machines

Q12 What is Normalization

Normalization, where the attribute data are scaled so as to fall within a small specified range, such as -1: 0 to 1:0, or 0:0 to 1:0.

An attribute is normalized by scaling its values so that they fall within a small specified range, such as 0.0 to 1.0.Normalization is particularly useful for classification algorithms involving neural networks, or distance measurements such as nearest-neighbor classification and clustering. If using the neural network backpropagation algorithm for classification mining (Chapter 6), normalizing the input values for each attribute measured in the training tuples will help speed up the learning phase. For distance-based methods, normalization helps prevent attributes with initially large ranges (e.g., income) from outweighing attributes with initially smaller ranges (e.g., binary attributes). There are many methods for data normalization. We study three: min-max normalization, z-score normalization, and normalization by decimal scaling.

Min-max normalization performs a linear transformation on the original data. Suppose that min_A and max_A are the minimum and maximum values of an attribute, A. Min-max normalization maps a value, v, of A to v' in the range $[new\_min_A, new\_max_A]$ by computing

$$v' = \frac{v - min_A}{max_A - min_A}(new\_max_A - new\_min_A) + new\_min_A. \qquad (2.11)$$

Min-max normalization preserves the relationships among the original data values. It will encounter an "out-of-bounds" error if a future input case for normalization falls outside of the original data range for A.

Min-max normalization. Suppose that the minimum and maximum values for the attribute *income* are \$12,000 and \$98,000, respectively. We would like to map *income* to the range $[0.0, 1.0]$. By min-max normalization, a value of \$73,600 for *income* is transformed to $\frac{73{,}600-12{,}000}{98{,}000-12{,}000}(1.0-0)+0 = 0.716$. ■

In **z-score normalization** (or *zero-mean normalization*), the values for an attribute, A, are normalized based on the mean and standard deviation of A. A value, v, of A is normalized to v' by computing

$$v' = \frac{v - \bar{A}}{\sigma_A}, \tag{2.12}$$

where $\bar{A}$ and σ_A are the mean and standard deviation, respectively, of attribute A. This method of normalization is useful when the actual minimum and maximum of attribute A are unknown, or when there are outliers that dominate the min-max normalization.

z-score normalization Suppose that the mean and standard deviation of the values for the attribute *income* are \$54,000 and \$16,000, respectively. With z-score normalization, a value of \$73,600 for *income* is transformed to $\frac{73{,}600-54{,}000}{16{,}000} = 1.225$. ■

Normalization by decimal scaling normalizes by moving the decimal point of values of attribute A. The number of decimal points moved depends on the maximum absolute value of A. A value, v, of A is normalized to v' by computing

$$v' = \frac{v}{10^j}, \tag{2.13}$$

where j is the smallest integer such that $Max(|v'|) < 1$.

Q13 What is Standardization?

Standardization is another scaling technique where the values are centered around the mean with a unit standard deviation. This means that the mean of the attribute becomes zero and the resultant distribution has a unit standard deviation.

$$Z = \frac{x - \mu}{\sigma}$$

Z = standard score

x = observed value

μ = mean of the sample

σ = standard deviation of the sample

Q14 When Should You Use Normalization and Standardization?

Normalization is useful when your data has varying scales and the algorithm you are using does not make assumptions about the distribution of your data, such as k-nearest neighbors and artificial neural networks. Standardization assumes that your data has a Gaussian (bell curve) distribution.

Q15 Why is KNN algorithm called Lazy Learner?

K-NN is a lazy learner because it doesn't learn a discriminative function from the training data but “memorizes” the training dataset instead. For example, the logistic regression algorithm learns its model weights (parameters) during training time.

Q16 Why should we not use the KNN algorithm for large datasets?

It needs to store all the data and then makes decision only at run time. ... So if dataset is large, there will be a lot of processing which may adversely impact the performance of the algorithm.

Q17. What are the advantages and disadvantages of KNN algorithm?

Advantages of KNN

1. No Training Period: KNN is called Lazy Learner (Instance based learning). It does not learn anything in the training period. It does not derive any discriminative function from the training data. In other words, there is no training period for it. It stores the training dataset and learns from it only at the time of making real time predictions. This makes the KNN algorithm much faster than other algorithms that require training e.g. SVM, Linear Regression etc.

2. Since the KNN algorithm requires no training before making predictions, new data can be added seamlessly which will not impact the accuracy of the algorithm.

3. KNN is very easy to implement. There are only two parameters required to implement KNN i.e. the value of K and the distance function (e.g. Euclidean or Manhattan etc.)

Disadvantages of KNN

1. Does not work well with large dataset: In large datasets, the cost of calculating the distance between the new point and each existing points is huge which degrades the performance of the algorithm.

2. Does not work well with high dimensions: The KNN algorithm doesn't work well with high dimensional data because with large number of dimensions, it becomes difficult for the algorithm to calculate the distance in each dimension.

3. Need feature scaling: We need to do feature scaling (standardization and normalization) before applying KNN algorithm to any dataset. If we don't do so, KNN may generate wrong predictions.

4. Sensitive to noisy data, missing values and outliers: KNN is sensitive to noise in the dataset. We need to manually impute missing values and remove outliers.

Q18 What is the difference between Model Parameters VS HyperParameters?

Table of difference between Model Parameters and HyperParameters

PARAMETERS	HYPERPARAMETER
They are required for making predictions	They are required for estimating the model parameters
They are estimated by optimization algorithms(Gradient Descent, Adam, Adagrad)	They are estimated by hyperparameter tuning
They are not set manually	They are set manually

PARAMETERS	HYPERPARAMETER
The final parameters found after training will decide how the model will perform on unseen data	The choice of hyperparameters decide how efficient the training is. In gradient descent the learning rate decide how efficient and accurate the optimiz

Q19 What is Hyperparameter Tuning in Machine Learning?

A Machine Learning model is defined as a mathematical model with a number of parameters that need to be learned from the data. By training a model with existing data, we are able to fit the model parameters.

However, there is another kind of parameters, known as Hyperparameters, that cannot be directly learned from the regular training process. They are usually fixed before the actual training process begins. These parameters express important properties of the model such as its complexity or how fast it should learn.

Some examples of model hyperparameters include:

The penalty in Logistic Regression Classifier i.e. L1 or L2 regularization.

The learning rate for training a neural network.

The C and sigma hyperparameters for support vector machines.

The k in k-nearest neighbors.

Q22. What is cross-validation?

k-fold cross-validation

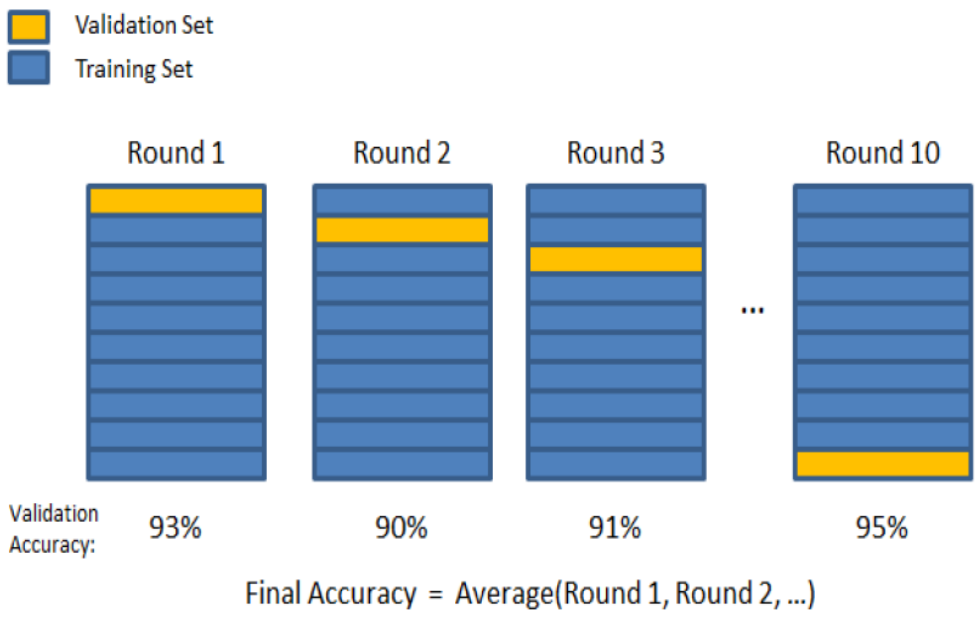
1. Split the data into k equal subsets

2. Perform k rounds of learning; on each round

– 1/k of the data is held out as a test set and

– the remaining examples are used as training data.

3. Compute the average test set score of the k rounds



Q23. What are GridSearchCV and RandomizedSearchCV, differences between them?

GridSearchCV

Cross-Validation permits us to evaluate and improve our model. But there is another interesting technique to improve and evaluate our model, this technique is called Grid Search.

Grid Search is an effective method for adjusting the parameters in supervised learning and improve the generalization performance of a model. With Grid Search, we try all possible combinations of the parameters of interest and find the best ones.

Scikit-learn provides the GridSeaechCV class. Obviously we first need to specify the parameters we want to search and then GridSearchCV will perform all the necessary model fits. For example, we can create the below dictionary that presents all the parameters that we want to search for our model.

```
parameters = {'C': [0.001, 0.01, 0.1, 1, 10, 100],
'gamma': [0.001, 0.01, 0.1, 1, 10, 100]}
```

Then we can instantiate the GridSearchCV class with the model SVC and apply 6 experiments with cross-validation. Of course, we need also to split our data into a training and test set, to avoid overfitting the parameters.

```
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC search = GridSearchCV(SVC(),
parameters, cv=5)X_train, X_test, y_train, y_test =
train_test_split(X, y, random_state=0)
```

Now we can fit the search object that we have created with our training data.

```
search.fit(X_train, y_train)
```

So the GridSearchCV object searches for the best parameters and automatically fits a new model on the whole training dataset.

24. What are the Applications of KNN?

Applications of KNN

Text mining

Agriculture

Finance

Medical

Facial recognition

Recommendation systems (Amazon, Hulu, Netflix, etc)