02 March 2019 08:57 AM

1. How is KNN different from k-means clustering?

**Answer:**

**KNN** is an algorithm that belongs to supervised learning domain, usable in both classification and regression problems but largely implemented in the classification scenarios. In it, we have a training dataset i.e - examples that are already classified into different categories.
And then we have a new record or element that have the same no of features as the training data. Then the KNN identifies the 'k' elements in the training dataset that are nearest or the most similar to the new record. And then the new record is assigned the class of the majority of the 'k' nearest neighbors. To sum up the working of KNN, its classification of new data work by identifying 'k' number of the nearest labeled data points.
Here 'k' is a parameter set by the user.

The main challenge for KNN is what is the optimal 'k' value is as deciding how many neighbors to use for KNN determines how well the model will generalize the future data. If 'k' is small we will have underfitting. If 'k' is large we will have overfitting.

One more important thing to keep in mind for KNN is that we do not build any model in it.

**k-means clustering** is an unsupervised learning algorithm that automatically divides the given unlabeled data into clusters (groups) of similar items without any prior training. To achieve this, k - means looks for 'k' clusters in the dataset. It begins by random initialization of centroids (a location that represents the center of cluster) then deciding for every data point what centroid is nearest to them. Then we calculate centroids for each cluster based on the existing members, thus providing us with new centroids.
Then we reassign each point to the closest cluster centroid. This process is run until convergence. The 'means' in K-means refer to the averaging of data i.e. the finding of the centroid.

One of the best ways for finding the 'k' parameter is using the Elbow method, in which we monitor the change of homogeneity within the clusters with different k values. It looks at the percentage of variance explained a function of the number of clusters, choosing 'k' in a way that adding another cluster does not give better modeling for the data.

**In the end**,we can say that KNN and k-mean clustering are both different algorithms for different domains carrying their own meaning of 'k'.
'k' stands for nearest neighbors in KNN while in k-means it gives the number of clusters.

2. What would be your approach to decompose a generic function into superposition of symmetric functions like given a smoothie, how the technique identifies the corresponding recipe for it?

**Answer:**

We have a defined systematic way that decomposes a generic function into a superposition of symmetric

functions in form of Fourier Transform. In a mathematical way it works by taking time base data, measuring every cycle by applying filters and then giving back the overall cycle(i.e. returning every detail found).

In the given context of a smoothie and how we can identify the recipe for it, Fourier transform can work as:

- Suppose we have different filters with us each one being a representation of a component of the smoothie. Let us say we have a filter for water, milk, chocolate, sugar and different fruits. One smoothie is poured through these individual filters then we get extracted details corresponding to every filter here, in this case, let's say in form of the amount of every ingredient present in the smoothie. We get the full recipe in the form a listing of the amount of every ingredient. Here we are using reverse engineering by taking out every ingredient to form the recipe.

Things that must be kept in mind are that:

1. Every filter that is used must be independent of each other.
2. Our collection of filter used must be able to catch every detail without leaving out a single thing.
3. The extracted components or in this case every ingredient must be combinable irrespective of the order to give us the same result.

3. What would be your strategy to handle a situation indicating an imbalanced dataset?

**Answer:**

On stumbling across an imbalanced dataset they are many ways in which we can resolve this issue. Some of the approaches that we can take to handle imbalanced data are:

1. Look out for more data for the given dataset - Being able to increase the size of our dataset might help us on giving a more balanced view of data classes by adding more data points to the minority class.

2. Data resampling approach: We can modify dataset for the predictive model to be more balanced through oversampling that is increasing data points of the minority class or through undersampling that is decreasing data points of majority class. In this domain rather than using conventional methods, we should use techniques like SMOTE and MSMOTE that that generate synthetic samples for us.

3. Trying a different performance metric: Accuracy is a very misleading metric when we are dealing with the imbalanced dataset, instead, we should use metrics that can paint us a better picture of the data classes. Some metrics that we can employ are Confusion Matrix, Precision, Recall, and F1 Score.

4. One of my favorite ways to approach the imbalanced datasets is to make use of ensemble techniques in which we look for ways to change our classification algorithms in a way, that they work more appropriately with imbalanced data. In this, we take a combination of predictions of new multi-stage classifiers created from the given instead of just relying on the single classifiers. Some ensemble techniques that we can use are Bootstrap Aggregating, Adaptive Boosting, Gradient Tree Boosting, XGBoost.

There is no single solution for building an accurate prediction model using imbalanced dataset, we have to try various methods to look out for the technique that is best for our dataset.

4. Suppose you are working on stock market prediction. You would like to predict whether or not a certain company will declare bankruptcy within the next 7 days (by training on data of similar companies that had

previously been at risk of bankruptcy). Would you treat this as a classification or a regression problem?

**Answer:**

This will be treated as a classification problem, as here we are trying to predict discrete valued outputs such as whether the company will 'declare bankruptcy' or company will ' not declare bankruptcy'.

5. What is deep learning, and how does it contrast with other machine learning algorithms?

**Answer:**

Deep Learning is a subset/part of a larger field of machine learning. The models in deep learning are deeply inspired by nervous systems of the human brain and use algorithms that try to mimic functions of the brain called artificial neural networks. Deep learning methods are differentiated from general machine learning algorithms in the respect that they make use of representation learning that is the learn how to present a problem while they are figuring how to solve the problem. Deep leaning is a representation of unsupervised algorithms that learn about the representation of data by using neural networks.

6. When should you use classification over regression?

**Answer:**

In a predictive modeling task, classification algorithms estimate the mapping function from input variables to the discrete output variables. Classification is to be used when we want our output to be as a discrete label. The most basic scenario we can take for use of classification is in email spam detection, as the output will fall under two category 'spam' or 'not-spam' we should use classification to for prediction. One thing to know is that classification can predict continuous values, in form of probability. These values can be used in selecting the label in terms of the higher probability values.

7. Using Python how do you find Rank, linear and tensor equations for an given array of elements? Explain your approach.

**Answer:**

In python, we can make use of numpy module on an array of elements to get the rank of the array. One way is use .ndim to get array no of axes i.e. equal to rank and other way is to use Numpy linear algebra method matrix_rank to get rank of a matrix, the function is **numpy.linalg.matrix_rank()**.

To get the linear equation of array of elements we can use **numpy.linalg.solve()** function where we can divide the array into an (n-1) columns and nth column if rank is n and (n-1) columns act as a coefficient matrix and nth column acts as ordinate and dependent variable values.

To get the solution of the tensor equation for given an array of the element we can use **numpy.linalg.tensorsolve()** function to get the solution for tensor equation.

8. What exactly do you know about Bias-Variance decomposition?

**Answer:**

In a prediction model of supervised learning, there are errors associated with predictions. Two of those errors are Bias and Variance.

Bias - It is the difference between the values predicted by our model and the ideal data relationship values, that we are trying to match. We can say that bias is the inability of the ML algorithm to capture the real relationships between the data of the datasets. High bias leads to underfitting.

Variance - It is the difference in the fits when different training data is used for modeling of a dataset. High variance leads to overfitting.

Bias-Variance decomposition is a tool that helps us to analyze a supervised learning algorithm. It enables us to observe that the MSE of a model consists of bias - that tells us about the accuracy of model and variance that tells us about the sensitivity of the algorithm. to change. From it, we can say that to have a model with a great performance we need to have optimal values of bias and variance such that MSE is minimized.

If we have to say in a nutshell Bias-Variance decomposition is:
Mean Square Error(MSE) = $Bias^2$ + Variance.

9. What is the best recommendation technique you have learnt and what type of recommendation technique helps to predict ratings?

**Answer:**

We can divide our techniques into two categories:

- Collaborative filtering methods - this relies on analysis of choices made by the user in their past and recommending things for them on the basis of similarity they have with other users that took the same decisions in the past. It enables content diversity.
- Content-based filtering methods - It works by reading the properties/attributes of user/item. It recommends thing similar to the one that user liked in the past.

But in modern scenarios, we prefer a hybrid system that combines content-based and collaborative filtering methods for increased effectiveness of recommendations.

The best recommendation that I can relate to is Matrix Factorization method - which decomposes user/data matrix into low dimensional factors and predict rating an item by multiplication of corresponding row and column of lower dimension matrices, along with Alternating Least Squares method for minimization of training loss instead of Stochastic Gradient Descent for minimization as ALS method give us benefits over SGD when dealing with implicit dataset in form of better model optimization.

The problem of predict ratings is solved by the use of Deep Recommendation, these are techniques that are making use of Deep Learning methods like Feed Forward nets in a recommender system for predicting ratings of items that users are yet to rate.

10. How can you assess a good logistic model?

**Answer:**

A logistic model can be assessed by checking the goodness of it's fit, by validating it's predicted values. The fit can be checked by using the following methods:

- Likelihood Ratio Test: A logistic model gives us a better fit if it shows an improvement over a model which has less number of predictors. In this test, we make a comparison of the likelihood of data in a full model and one in which we have fewer predictors.
- Hosmer - Lemeshow Test: It is applied on data after the observations have gone through divisions to form groups on the basis of similar predicted probabilities. Then it employs as Pearson chi-square test for calculating p-values against predicted probabilities of being in a subgroup and then tells about the goodness of fit on the basis of p-value which being large indicate a good fit.

The validation of predicted values can be done by comparing the predicted targeted value against original observed values by using the following methods:

- Classification Rate
- ROC Curve
- K - Fold Cross-Validation.

11. How to you read the text from an image? Explain?

**Answer:**

The process by which we extract text from an image and digital documents is termed as Optical Character Recognition(OCR).

One of the direct and simplest ways is using the pytesseract tool in python for reading the text by passing the image to its predefined functions.

Theoretically and from a general solution perspective, reading text from an image is a task divided into two stages,

1. Find out the appearances of the text in the image.
2. Then choose one of the approaches from the following for data extraction.

- Classic Computer Vision methods- This works by:
  - Use filters for differentiating characters from the background of the image.
  - Using contour detection for one by one character recognition.
  - Doing character identification by image classification.

- Using Deep Learning: After text detection, we can use standardized Deep Learning Methods like SSD, YOLO, Fast - RCNN and Mask RCNN for further detection.

- Using Specialized Deep Learning methods:
  - Efficient Accurate Scene Text detector(EAST) - It is a method limited to text detection which operates as an FCN, with great robustness.
  - Convolutional - Recurrent Neural Network(CRNN) - It approaches the problem by a three-layered process starting form Convolutional to Recurrent and ending at Transcription Layer where it makes use of probability to get meaningful data from the extracted sequence of characters.

Extracting text from an image can be done in various ways, but not a single one is the best in say.

12. What are all the options to convert speech to text? Explain and name few available tools to implement the same?

**Answer:**

Some of the algorithms and methods available to us are :

1. Hidden Markov models
2. Dynamic Time wrapping (DTW) - based speech recognition
3. Use of Deep feedforwarded and recurrent neural networks.

Some tools that are available for us for conversion of speech to text are:

- apiai - It is a python SDK that provides a user with features such as Speech Recognition, Voice Activity Detection, and NLP. It is now commercially available as Dialogflow with integration with Google Cloud Speech-to-Text

- AssemblyAI - It can be used for accurate recognition of speech in our application. It gives us output in a word by word manner.

- Google cloud speech - It is a package that focuses solely on the conversion of speech to text. Its reason for being one of the most powerful speech recognition systems is its use of powerful deep learning neural network algorithms.

- SpeechRecognition - This is one of my favorite libraries for STT as it provides support for several API and existing methods in both offline and online scenarios. It gives us support for-: Google Speech Recognition, Microsoft Bing Voice Recognition, CMU Sphinx, IBM Watson services.