CLASS 16 11-06-2021

**QUESTIONS**

👉 What is support vector machines with example?  
👉 What does Hyperplane in SVM mean?  
👉 Write some Support Vector Machine (SVM) Usecases  
👉 What is kernel in SVC?  
👉 What is List comprehension?  
👉 When do you need list comprehension in Python?  
👉 How does the condition work in Python list comprehension?

**ANSWERS**

1. **Support Vector Machine” (SVM)** is a supervised machine learning algorithm which can be used for both classification or regression challenges. However,  it is mostly used in classification problems. In the SVM algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiates the two classes very well.

2. A **hyperplane** in an n-dimensional Euclidean space is a flat, n-1 dimensional subset of that space that divides the space into two disconnected parts. In simple terms, **hyperplane** is a decision boundary that helps classifying data points.

3. We **use** **SVM** for identifying the classification of genes, patients on the basis of genes and other biological problems. Protein fold and remote homology detection. We use SVMs to recognize handwritten characters used widely.

4. “**Kernel**” is used due to set of mathematical functions used in Support Vector Machine provides the window to manipulate the data. So, Kernel Function generally transforms the training set of data so that a non-linear decision surface is able to transformed to a linear equation in a higher number of dimension spaces.

5. **List comprehension** is basically creating lists based on existing iterables. List comprehensions are relatively faster than for loops. Python list comprehension. So we iterate over an iterable and do something (optional!) with the items and then put them in a list.

6. **List comprehension** is an elegant way to define and create lists based on existing lists. List comprehension is generally more compact and faster than normal functions and loops for creating list. However, we should avoid writing very long list comprehensions in one line to ensure that code is user-friendly.

7. Given a **list comprehension** you can append one or more if **conditions** to filter values. [<expression> for <element> in <iterable> if <condition>] For each <element> in <iterable> ; if <condition> evaluates to True , add <expression> (usually a function of <element> ) to the returned list.