**Q1**. I’ve a dataset with following columns (CustomerID, Education, Income, CreditScore, DebtToIncomeRatio, IsLoadApproved) in which CustomerID, Education, and IsLoanApproved are non-numeric.

Going by the data cleaning process, I’ve to find the z-score for numeric columns (Income, CreditScore, DebtToIncomeRatio) and form a data frame called df1.

Then I’ll perform the one-hot encoding on the categorical column on the columns named IsLoanApproved, and Education. Then form a data frame called df2.

I will then use the Index column from both the data frame to merge the df1, and df2 to form my final data frame for machine learning. But what about CustomerID column? Shall I treat it as Categorical Column and encode it?

**Q2**. You showed us a list of built-in algorithms that we could use to train the models. Is that good enough to all kind of problems in the world? Or there are times when you need to build your own algorithm as well?

**Q3**. Can you explain a probme statement where you have both numerical and categorical features? In such case, how do you choose what algorithm to use?

**Q4**. What are the accepted value of R2, RMSE, MSE to consider the model is good? Can you elaborate a bit more on how to read these values to draw our conclusion on the model?

Q5: How do we decide which algorithm to use?

Q6: I tried the K-means clustering example from the previous class. Every time, I train the model, I get a different labelling. There is no consistency.

kmeans\_model = KMeans(n\_clusters=K, init='k-means++')

kmeans\_model.fit(x)

y\_pred = kmeans\_model.predict(x)

print(y\_pred)

Q7. We plotted dendogram using 2 features. So, how to visualize how the distance calculated in the X-axis?

Q1. Do not we need scaling or encoding in unsupervised learnings? If no, why?

Q2. I used a dataset from [www.kaggle.com](http://www.kaggle.com), and my cluster is looking exactly same for both K-mean and HC. The Silhouette Score is almost the same. I do not see the benefit of HC over K-mean. Can you please share your dataset for me test them?

Q1. All the algorithms we are seeing or training, are they good enough to cater to all kind of real world problems? Or there are times, one need to build their own algorithm or mathematical model?

Q2. In PCA, your machine learning is based on PCA components, not on actual columns. So finally when the model is ready. How do you deploy it for inference? The user will send the actual column values to the model for prediction, so how do you translate the columns to PCs during the inference?

Q3. What if I am using OneHotEncoding? It increases the feature count. In that case, shall I also use PCA? Will this not completely change the essence of OneHotEncoding?