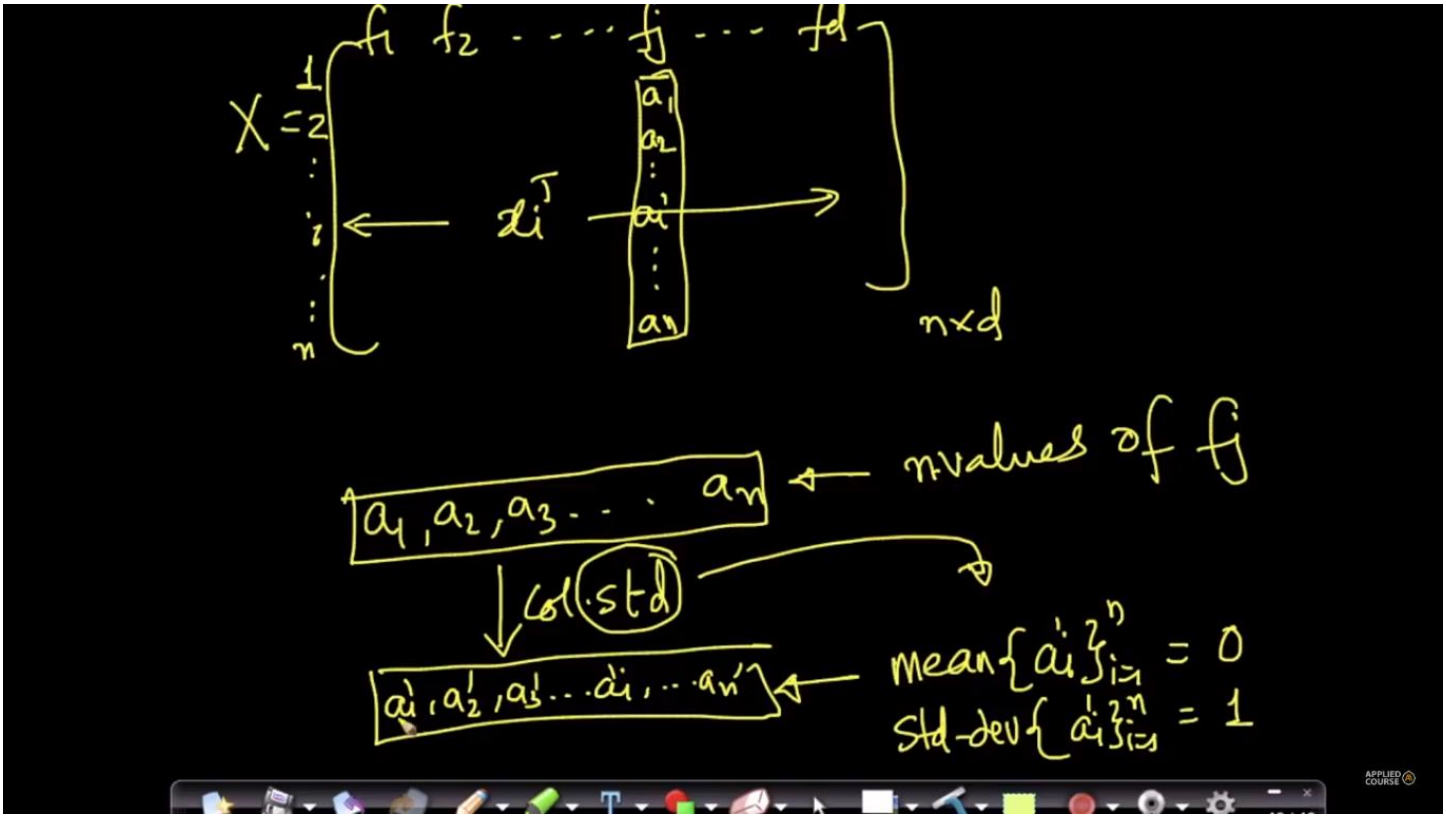
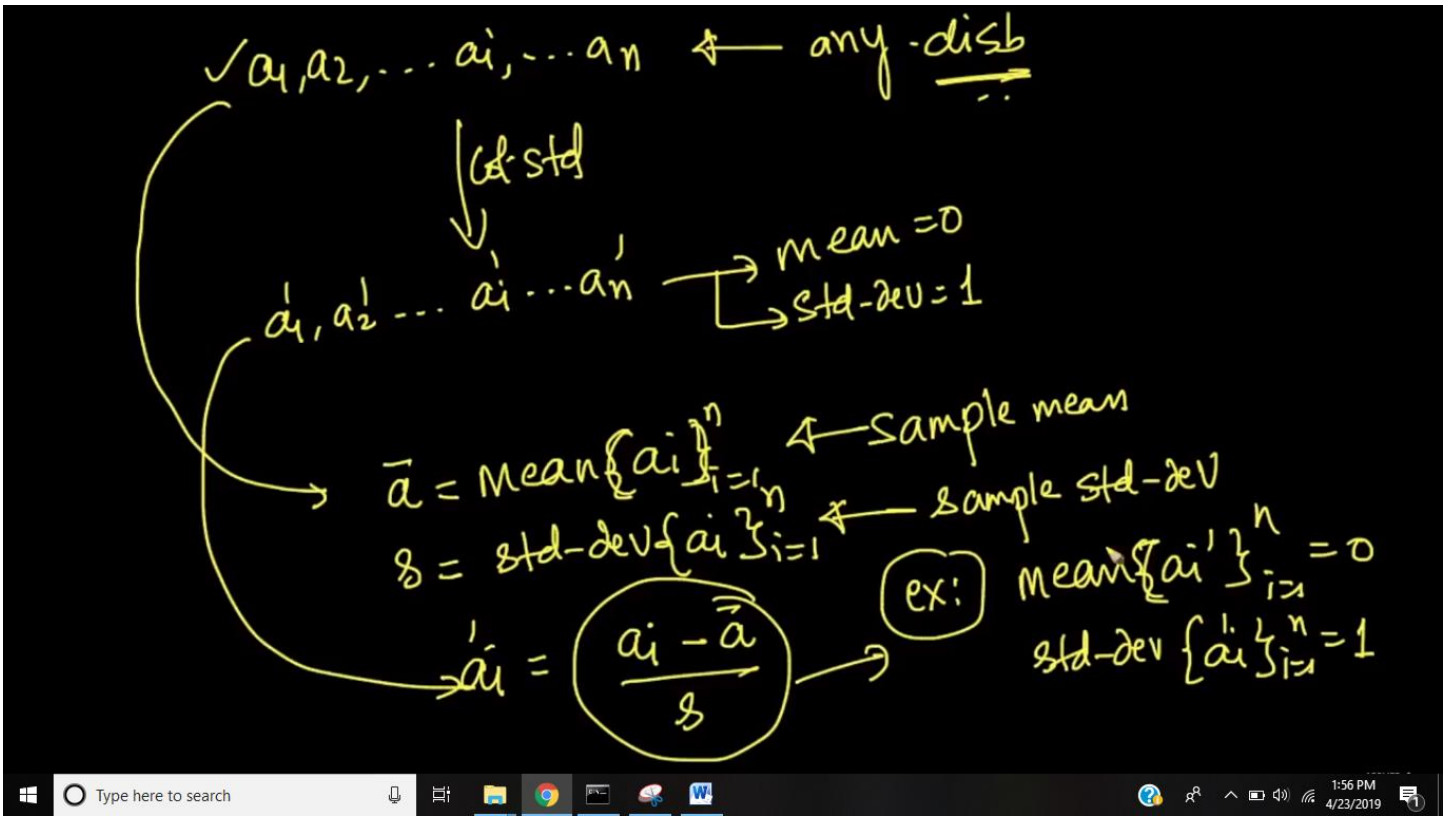
Column Standardization :

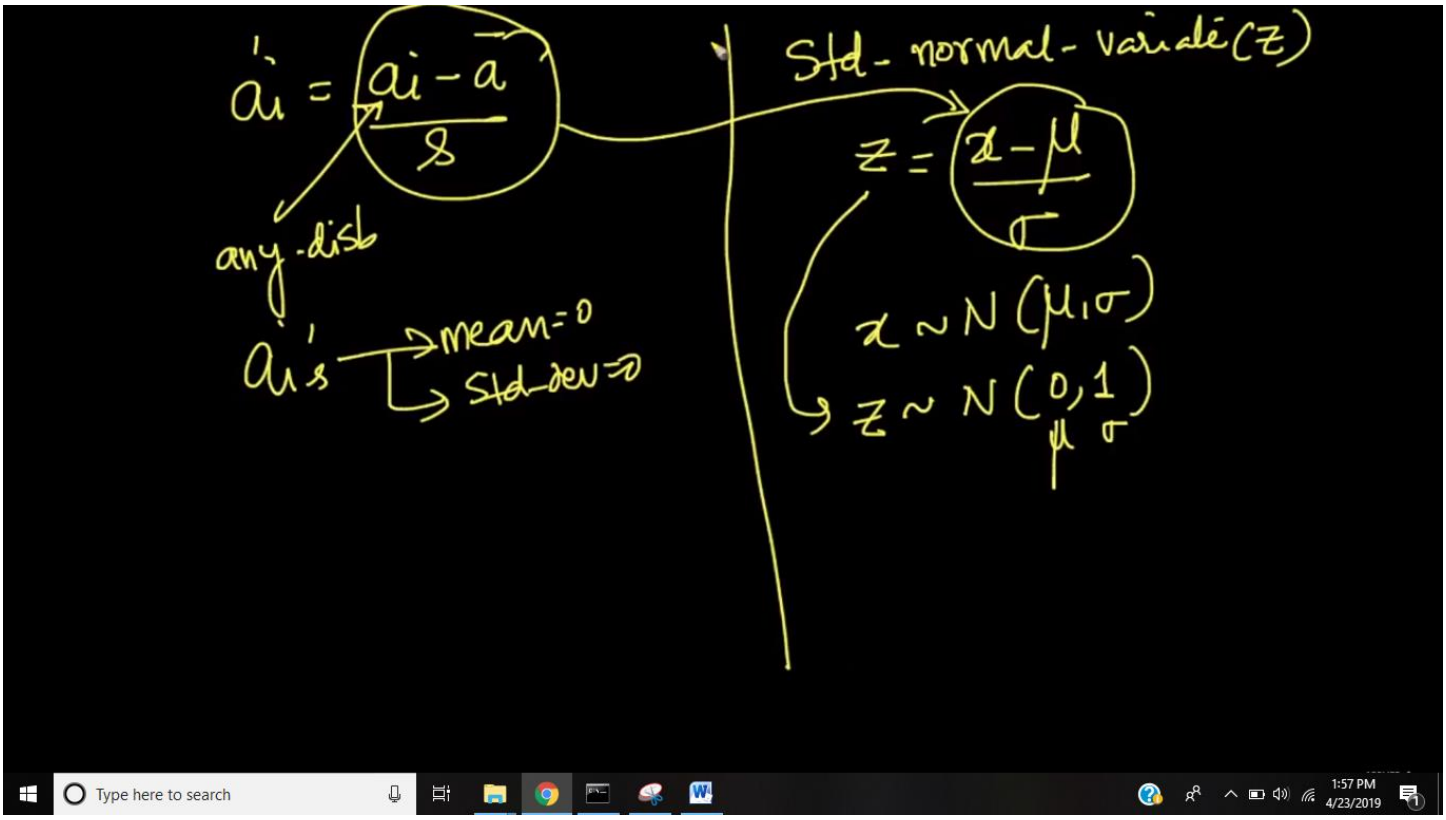
As shown below after applying column standardization to any feature vector then that feature vector will be converted into new vector whose mean = 0 and stand. Dev. = 1.



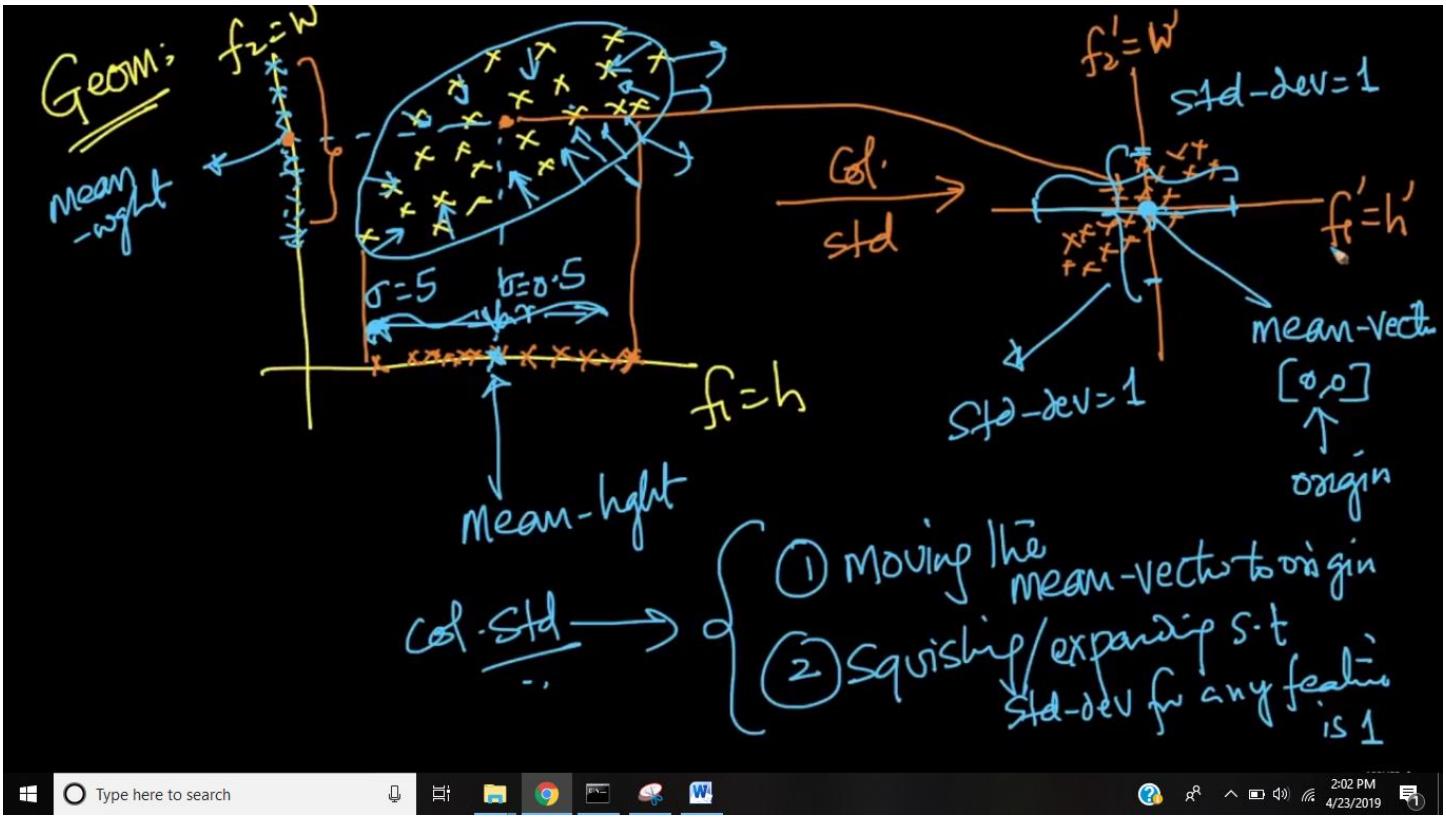
Here we calculate data point of new vector by using formulae as shown in below fig. which is same as standard normal variate.



Both standardization and standard normal variate is same.



Here we see geometrically before mean is at somewhere so first we bring the mean at 0 i.e moving the mean-vector to origin and then if data point have s.d more than 1 i.e they have more spread then we squish such that std. dev will come to 1 and similarly if datapoints are such that, they have sd less than 1 that mean they are cluttered in small space so we expand them such that they have SD 1.



Notes

- When to normalize the data and when to standardize the data depends purely on the context of the problem we are working and the scale of features that is required for that particular problem. If we want all the features to have values in the range [0,1], we go for normalization and if we want all the features with mean centred variance scaling, we go for standardization. It is recommended to go for Standardization because most of the models give better results when they are trained on standardized data over normalized data.

- Yes feature scaling is also a kind of fitting problem. We divide the dataset into train and test datasets. When we apply feature scaling, we create an instance of StandardScaler(module for standardization in scikit-learn) or MinMaxScaler(module for normalization in scikit-learn) and apply fit() method on the train data. It computes the parameters. Then the same instance is used to apply transform() on both train dataset and test dataset to transform the

values on to the scale which have the computed parameters.
Overfitting & Underfitting are related to hyper-parameter tuning and are no way related to standardization or normalization.