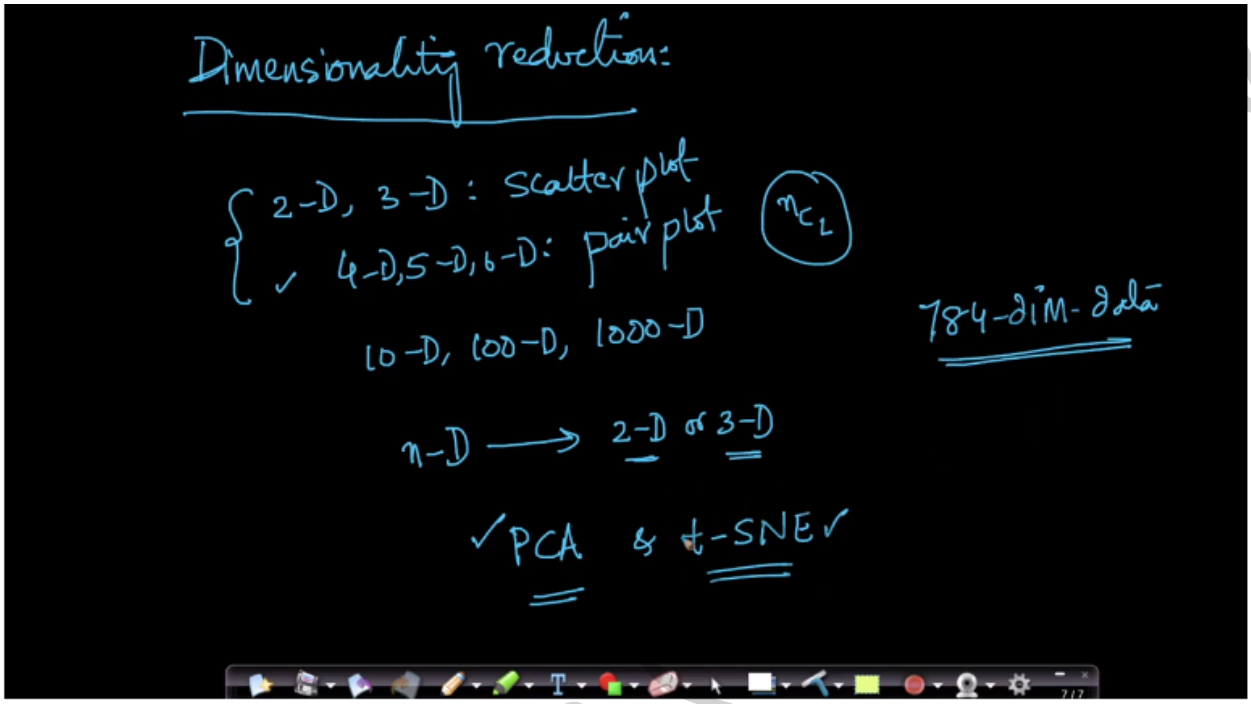
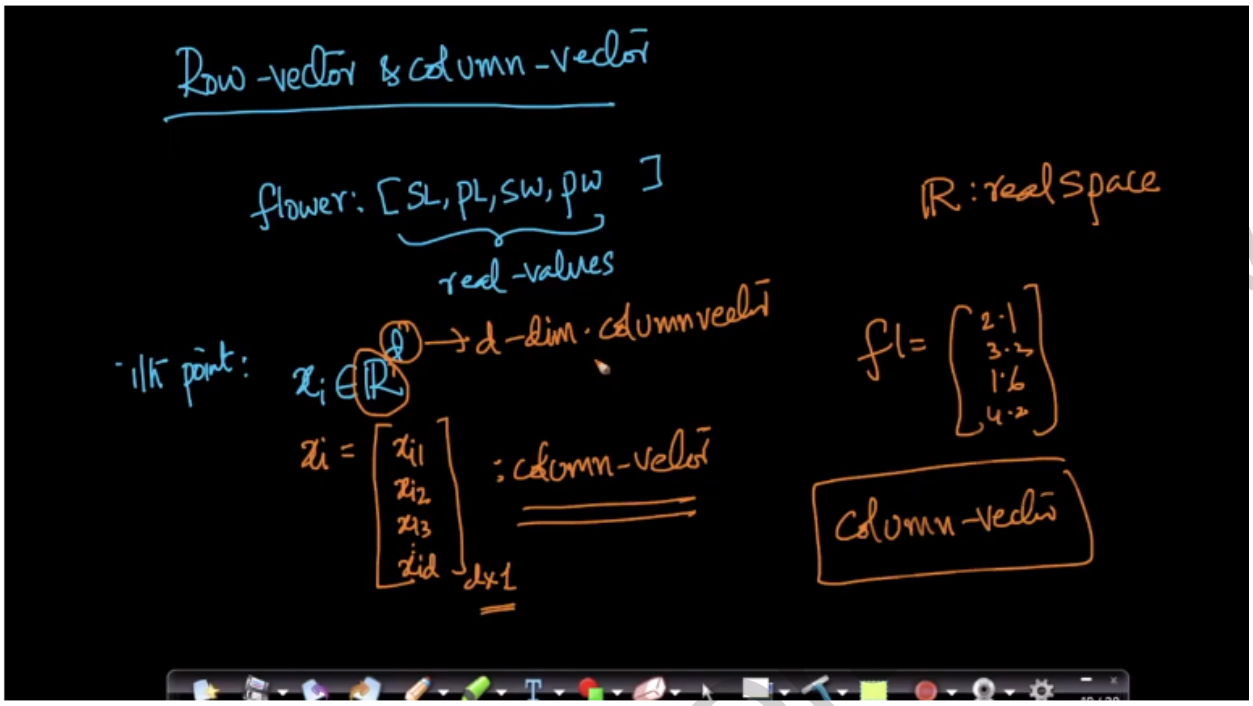
25.1 What is Dimensionality Reduction?



Timestamp: 2:11

We have seen that 2-D and 3-D data can be visualized using scatter plots, but as the number of dimensions increases we need nc2 number of pair plots, which can be difficult to visualize.Hence we use dimensionality reduction techniques like PCA and t-SNE using which we can reduce the dimensionality of the data from n-D to 2-D and 3-D, which we can visualize easily.

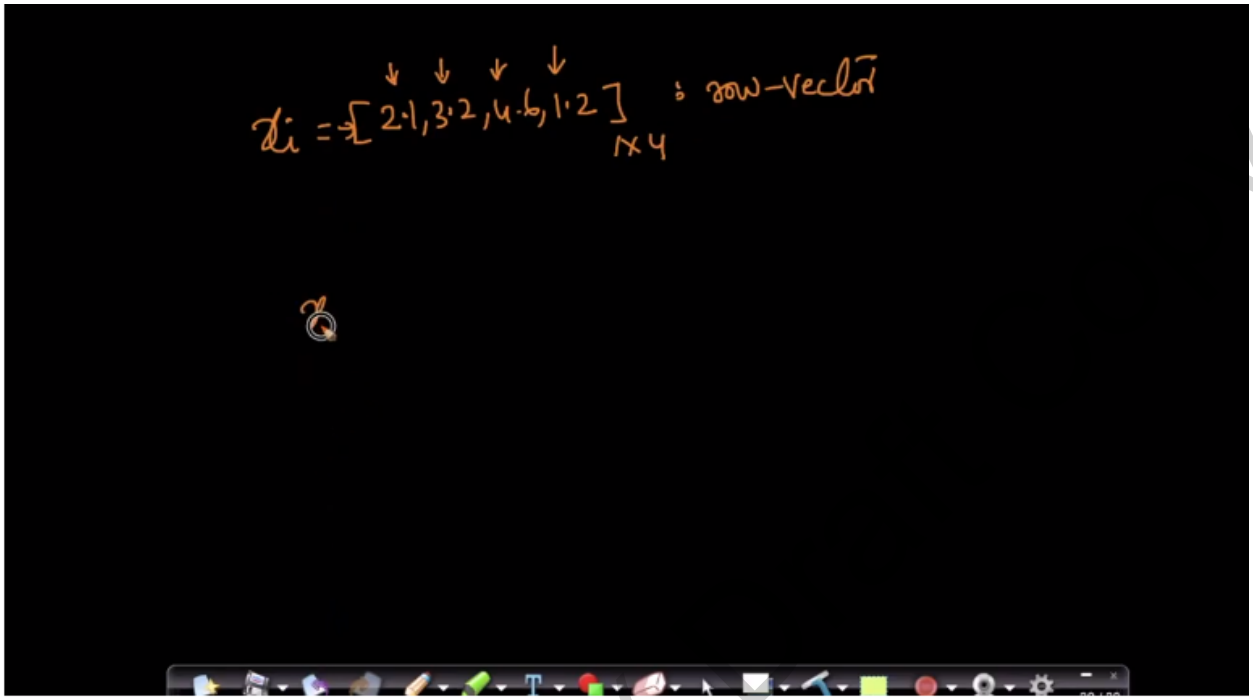
25.2 Row vector and Column vector?



Timestamp: 4:06

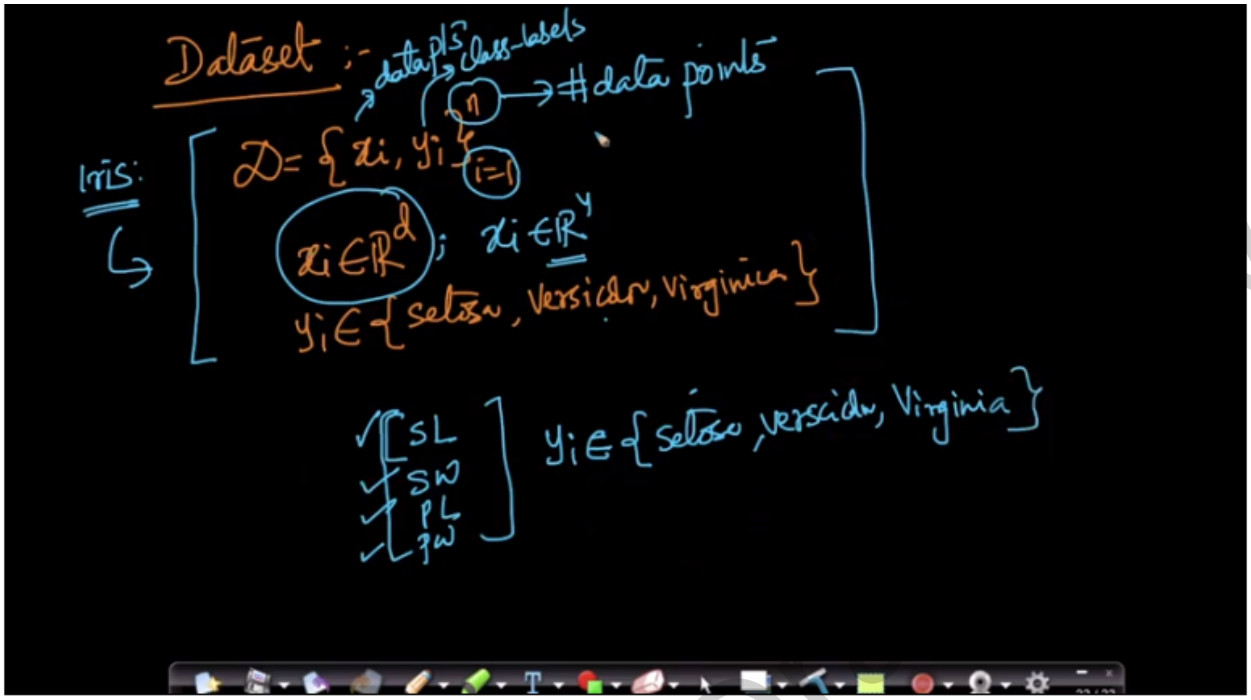
An ith datapoint xi is often represented either as a column vector or a row vector. If not mentioned explicitly then it is assumed to be a column vector. A column vector looks like as mentioned in the above figure. A d dimensional vector that takes real values is often mentioned as xi belonging to R^d where R stands for real-values.

A row vector representation of xi looks as below.



Timestamp: 3:30

25.3 How to represent a data set?

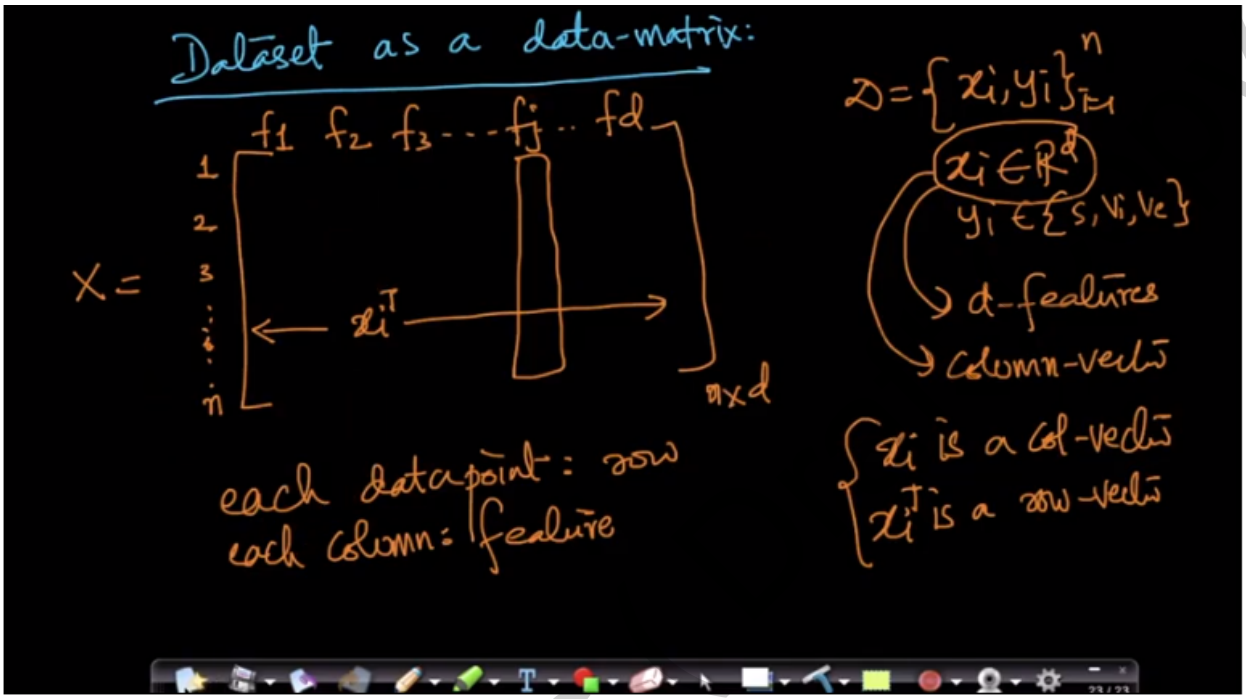


Timestamp: 3:09

There are many ways to represent a dataset but one such way to represent a dataset for the example of iris dataset is as above. A dataset is represented as D = {xi,yi} i from 1 to n, where xi represents the datapoint and yi represent the corresponding label.

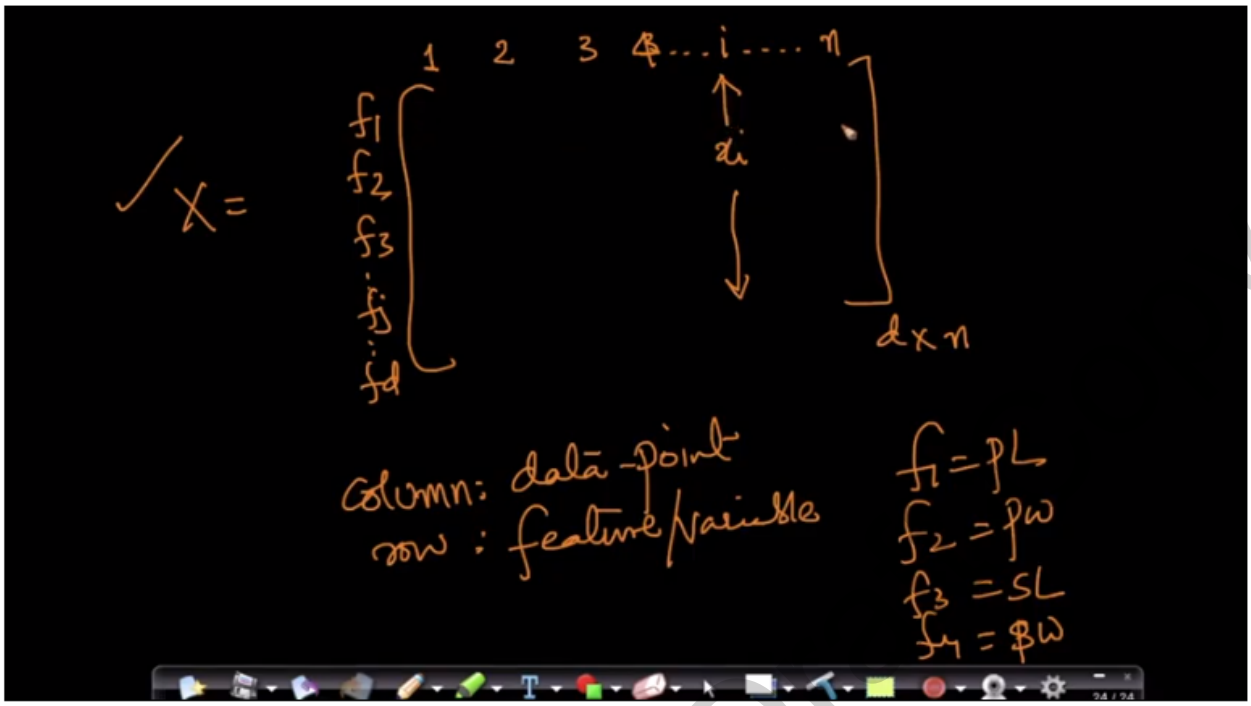
Xi is represented as belonging to R^4 as it contains 4 real valued features and yi can take any value from {setosa,versica,virginica} the three classes or labels in the iris dataset.

25.4 How to represent a data set as a Matrix?



Timestamp: 2:09

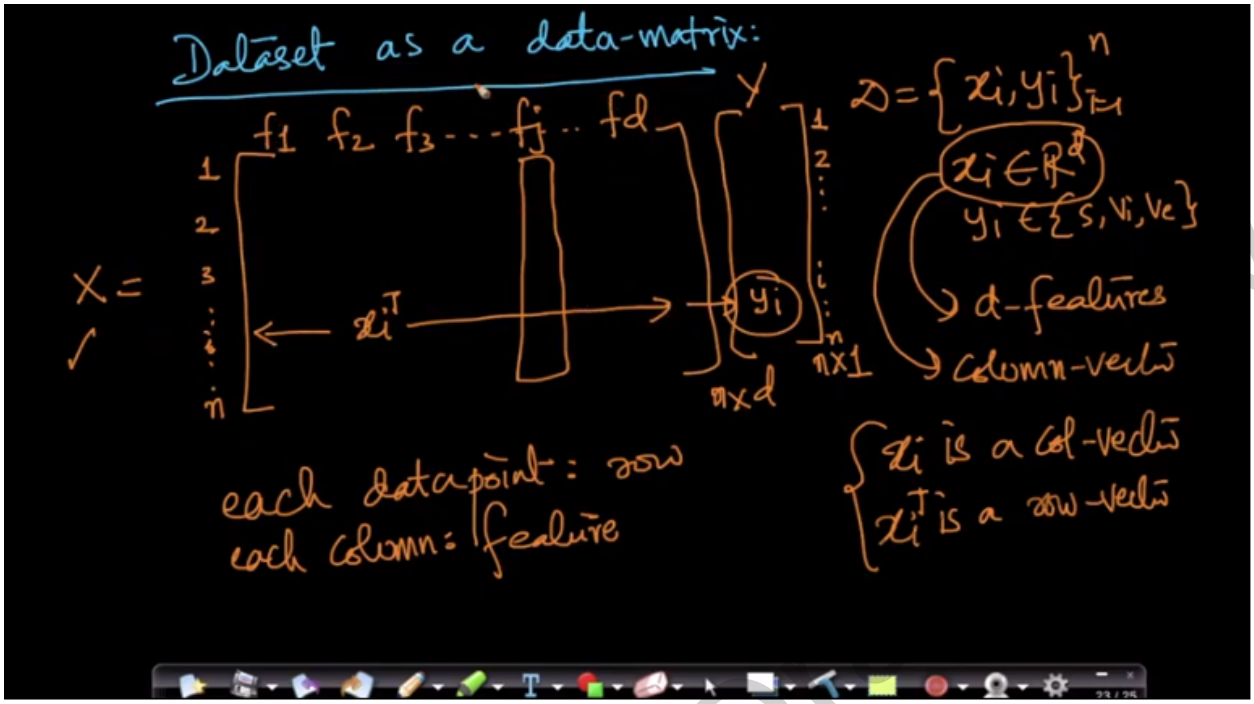
There are two ways to represent our data matrix X, the first way is to represent in such a way that each row represents a datapoint and each column represents a feature. Notice that since xi is a column vector, in order for us to represent it as a row vector, we are using xi^T.



Timestamp: 3:56

The other way is to just represent it as above, where each column represents a data point and each row represents a feature. This data matrix X is just a transpose of the data matrix X in the first approach.

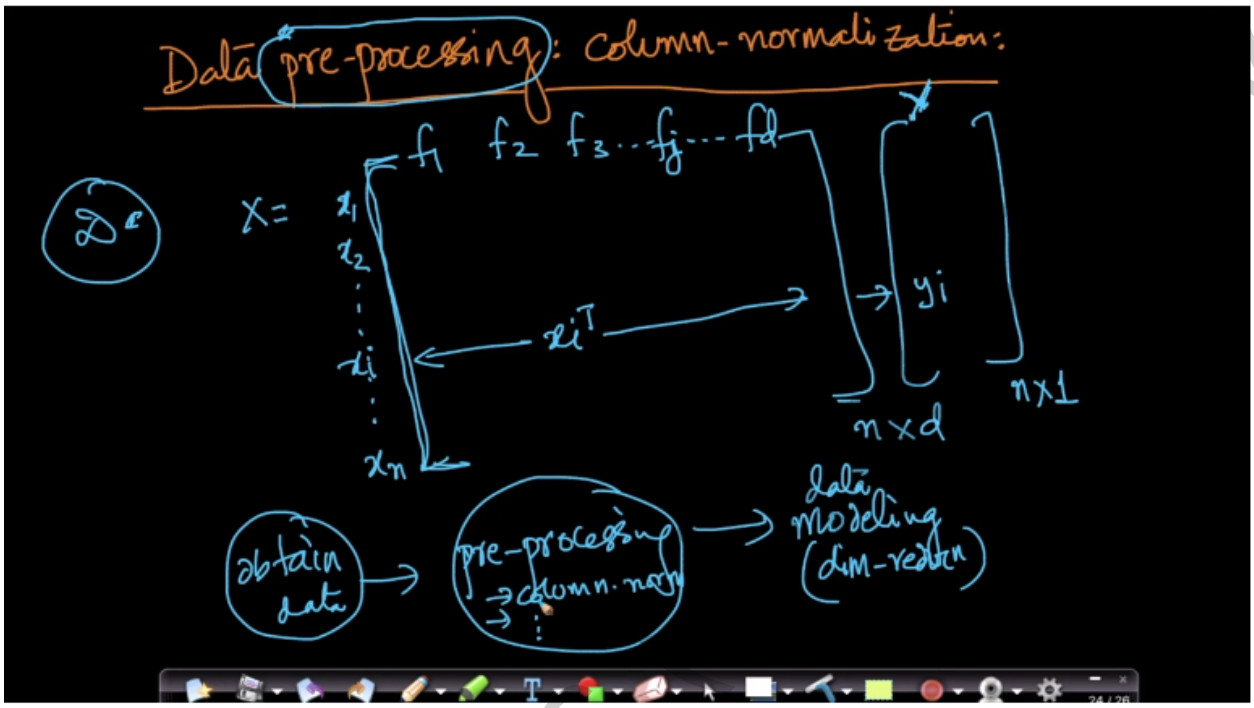
The most common approach is the first one, where each row represents a datapoint and each column represents a feature. The entire dataset now looks like below using the first approach.



TImestamp: 6:35

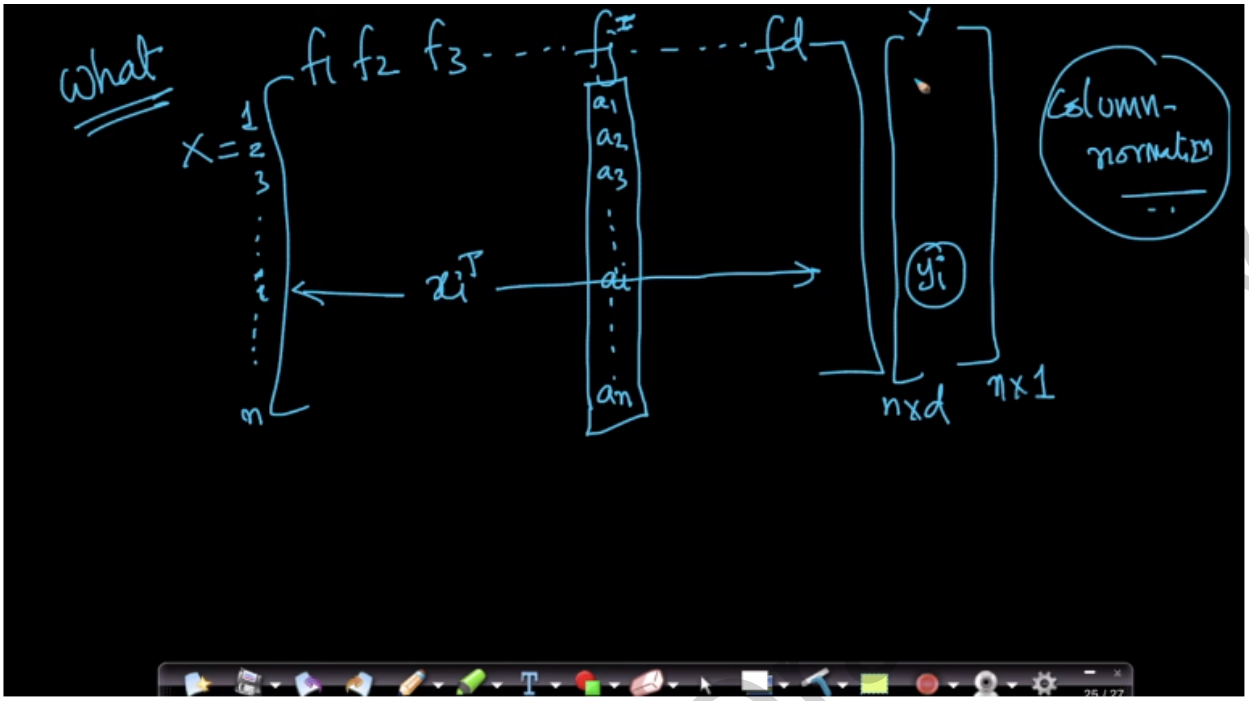
Notice that Y is represented as a column matrix where each row yi represents the label for the corresponding datapoint xi in X.

25.5 Data Preprocessing: Feature Normalization



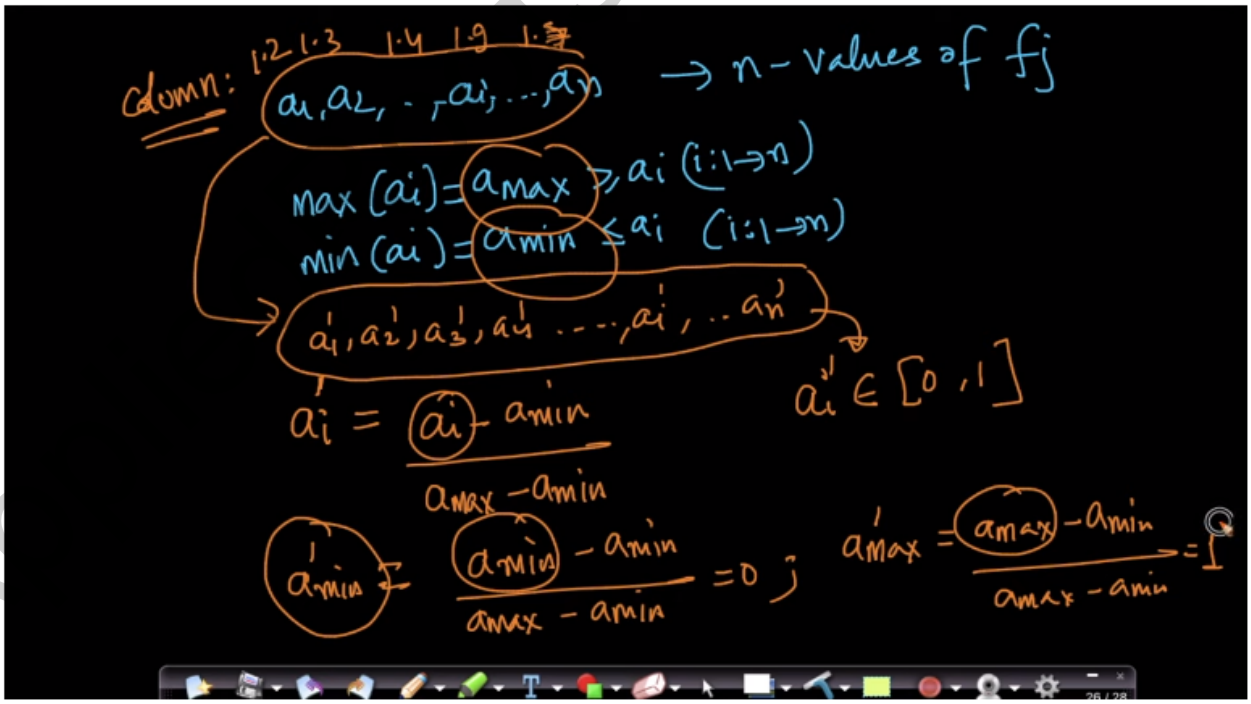
Timestamp: 3:10

After obtaining the data, we need to preprocessing of the data such that our dimensionality reduction algorithms can work better. One such preprocessing is column normalization.



Timestamp: 5:26

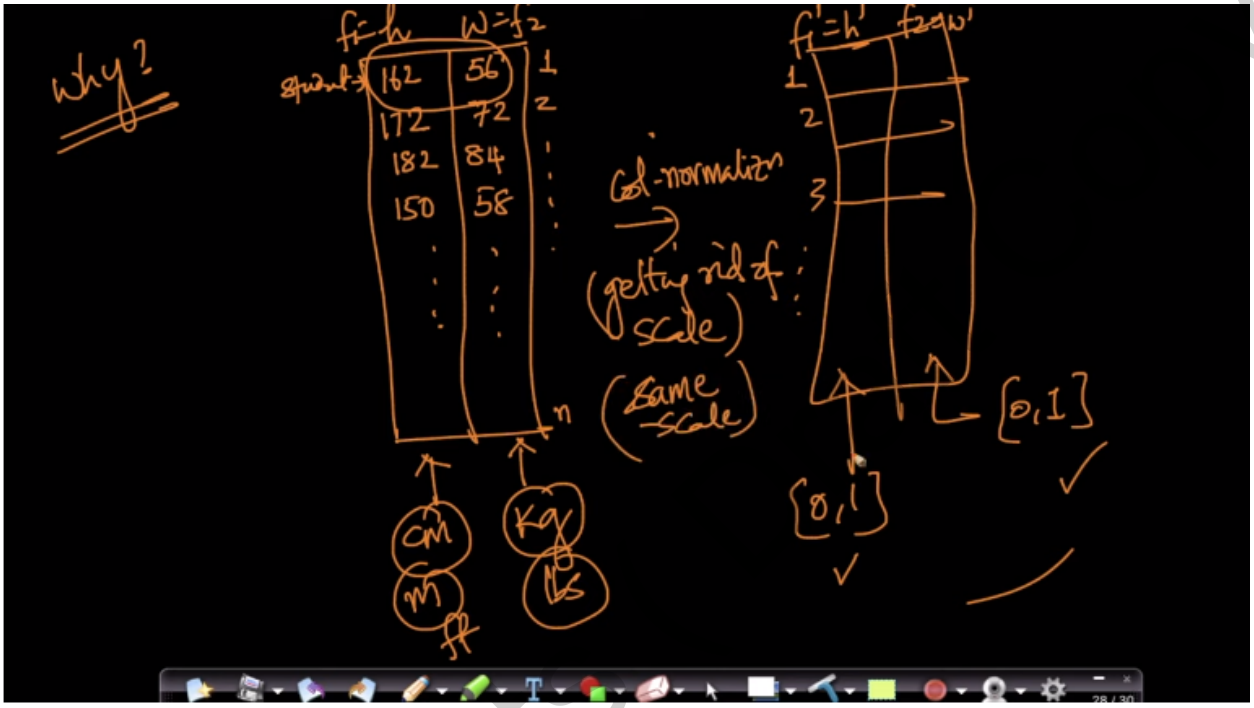
For column normalization ,we pick each feature fj, lets say it contains numbers a1,a2…..an as above then we do column normalization as follows.



Timestamp: 9:30

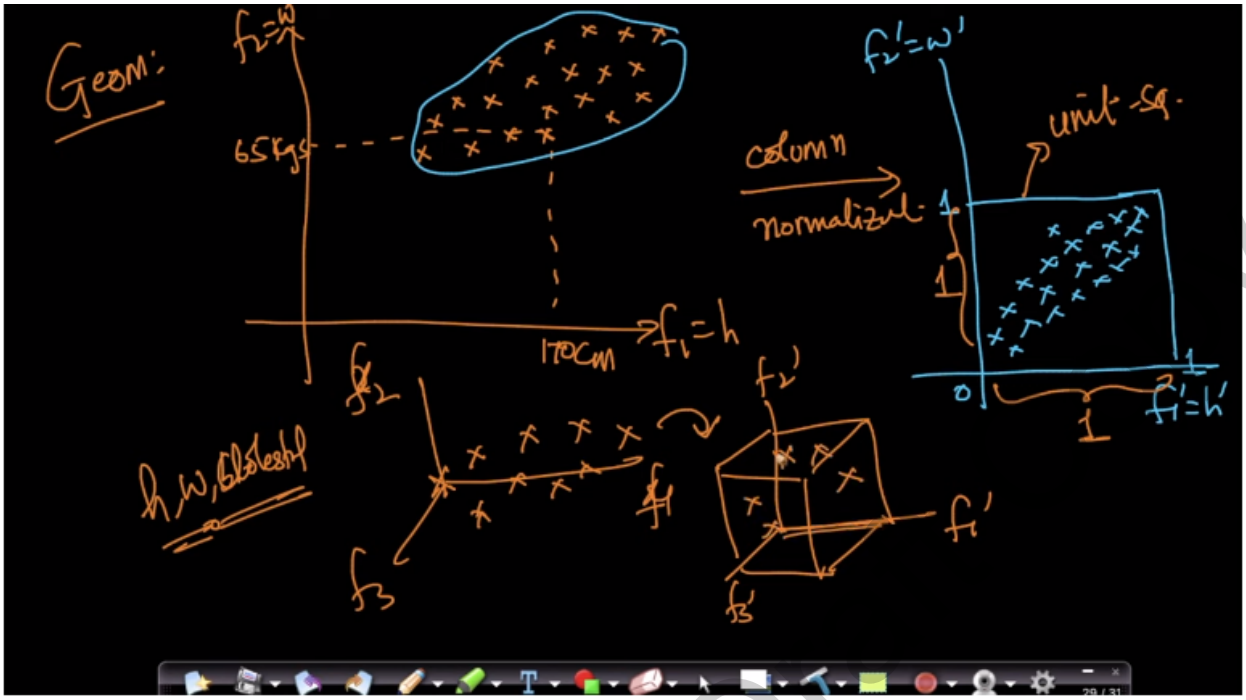
As shown in the above figure, with column normalization what we are doing is, from a1,a2,...an we are getting a1',a2',.....an' as below, such that each ai' now belongs to [0,1]

$ai' = (ai - a_{min})/(a_{max} - a_{min})$



Timestamp: 14:32

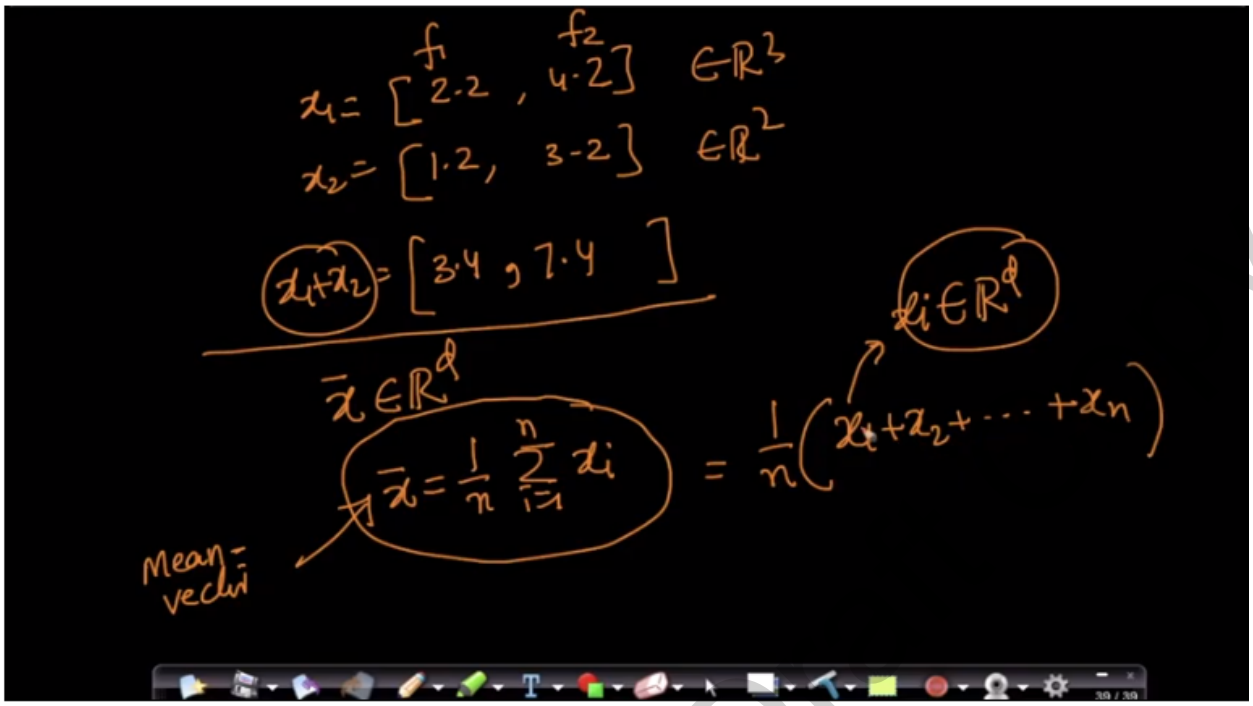
Since features like height, weight etc can be measured in different scales such as cm,m,etc, we are doing column normalization to get rid of the problems that come up with the scale(which will be discussed later) and we are converting all the features into the same scale using column normalization.



TImestamp: 17:45

Geometrically, through column normalization, we are squishing data points that are spread over the entire space into a unit-square when it is 2-D data or into unit-cube when it is 3-D data and n-D unit hyper cube when the data is of n-D data.

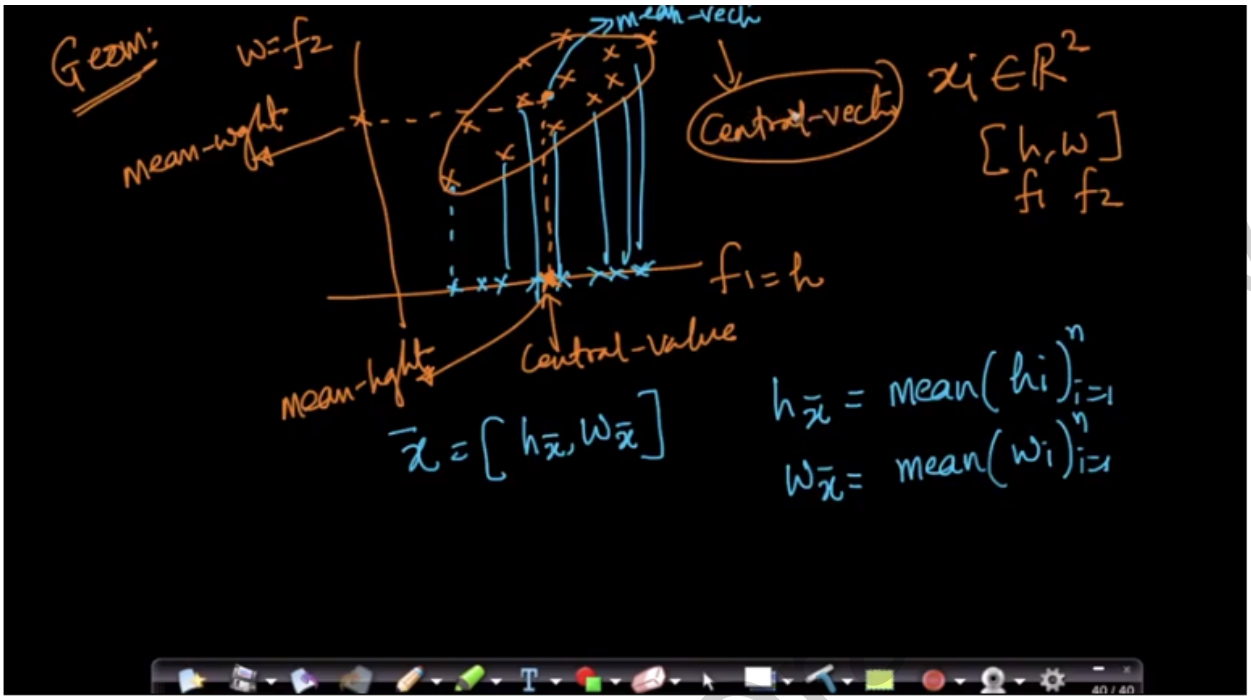
25.6 Mean of a Data Matrix



Timestamp: 2:31

As shown in the above picture, the sum of two vectors x1 and x2 is just a vector where each element is the sum of corresponding elements in the vectors x1 and x2.

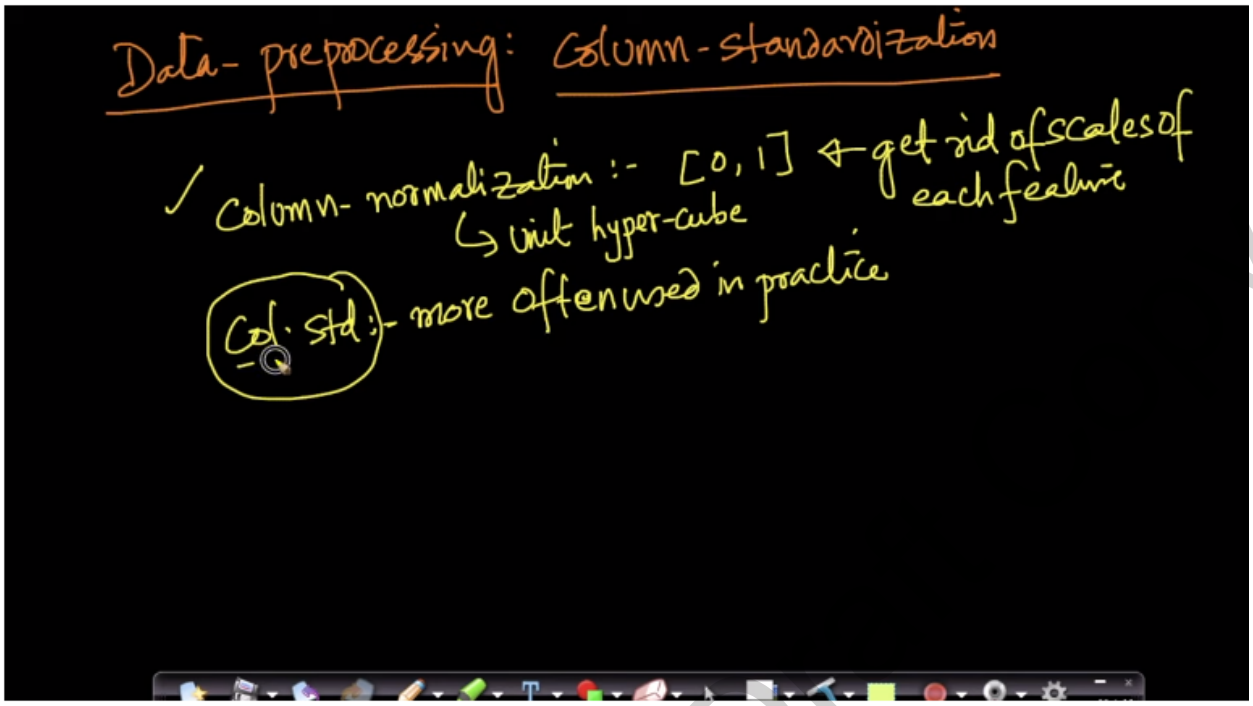
The mean vector of n data points, is the sum of n datapoints divided by n as shown above.



Timestamp: 5:47

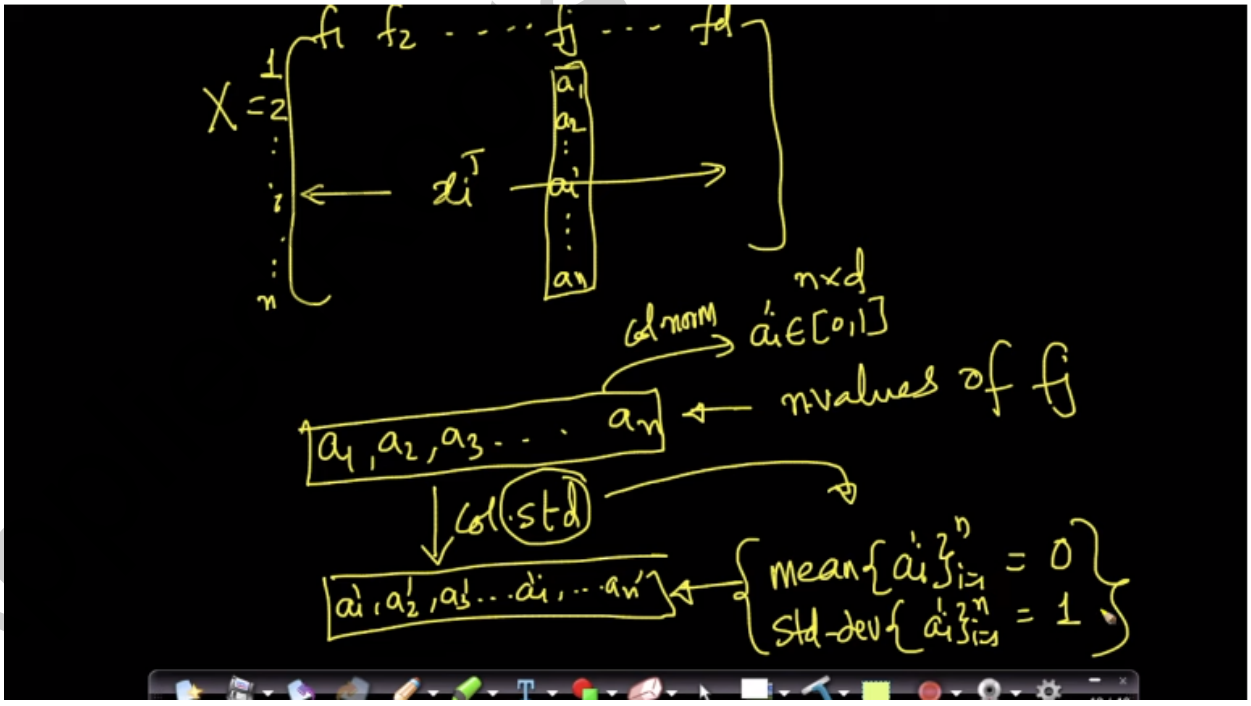
Geometrically, mean vector is like a central value of the data.

25.7 Data Preprocessing: Column Standardization



Timestamp: 1:11

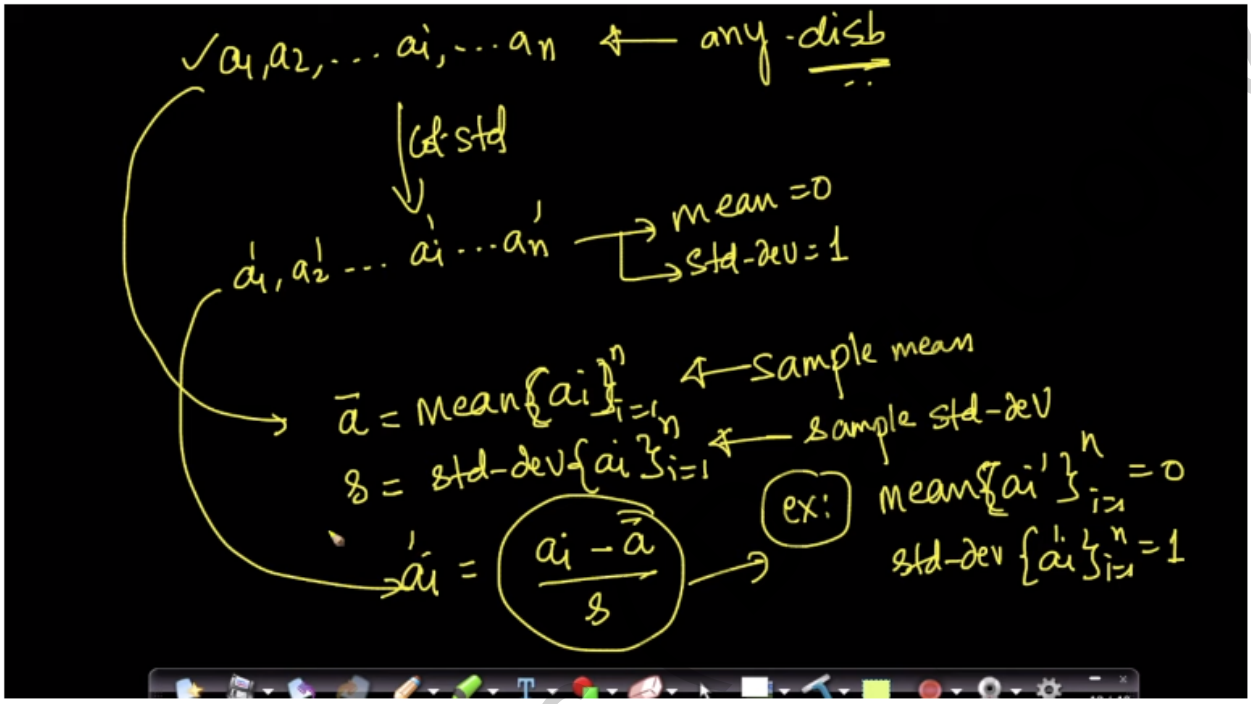
Column standardization is a similar preprocessing technique similar to column normalization and it is used more often in practice.



Timestamp: 4:26

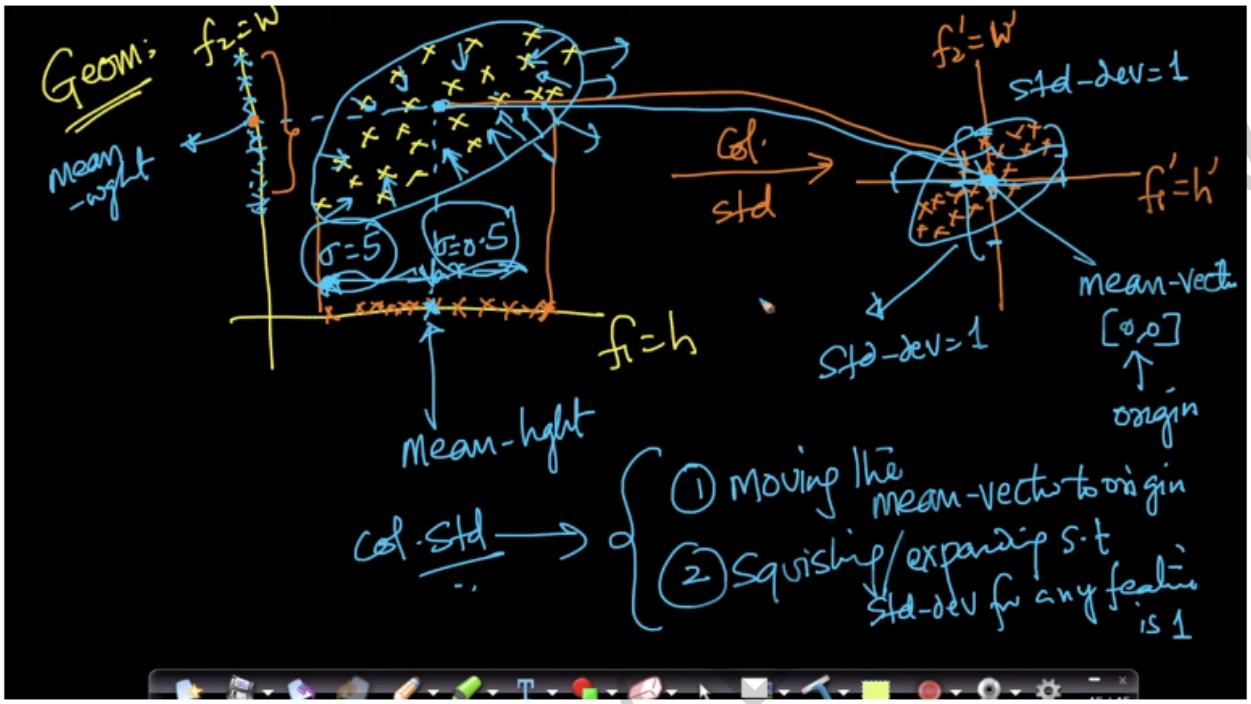
Given a data matrix X, for each feature containing n values a1,a2,....an ,we convert them into a1',a2',.....an' such that the mean of ai's is 0 and std of ai' is 1.

We do it, by getting ai' from ai as below



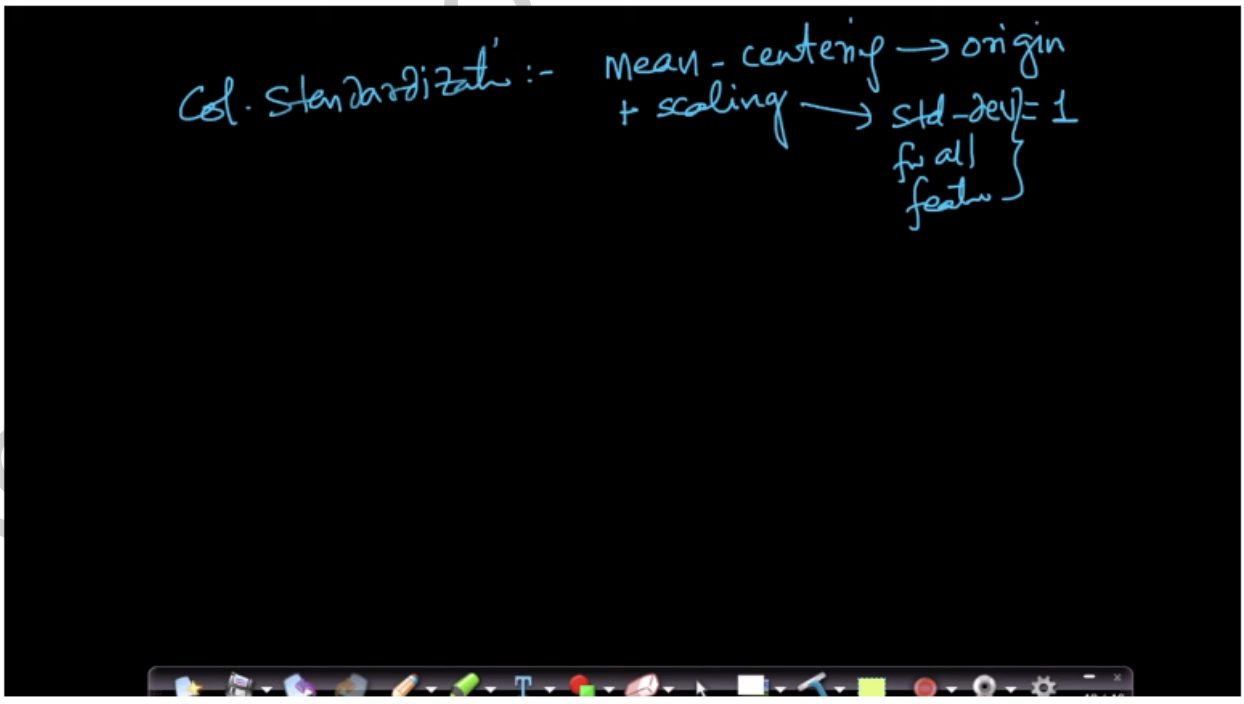
Timestamp: 6:57

We subtract the mean of ai and divide by the standard deviation of ais from each ai to get ai'.



Timestamp: 14:01

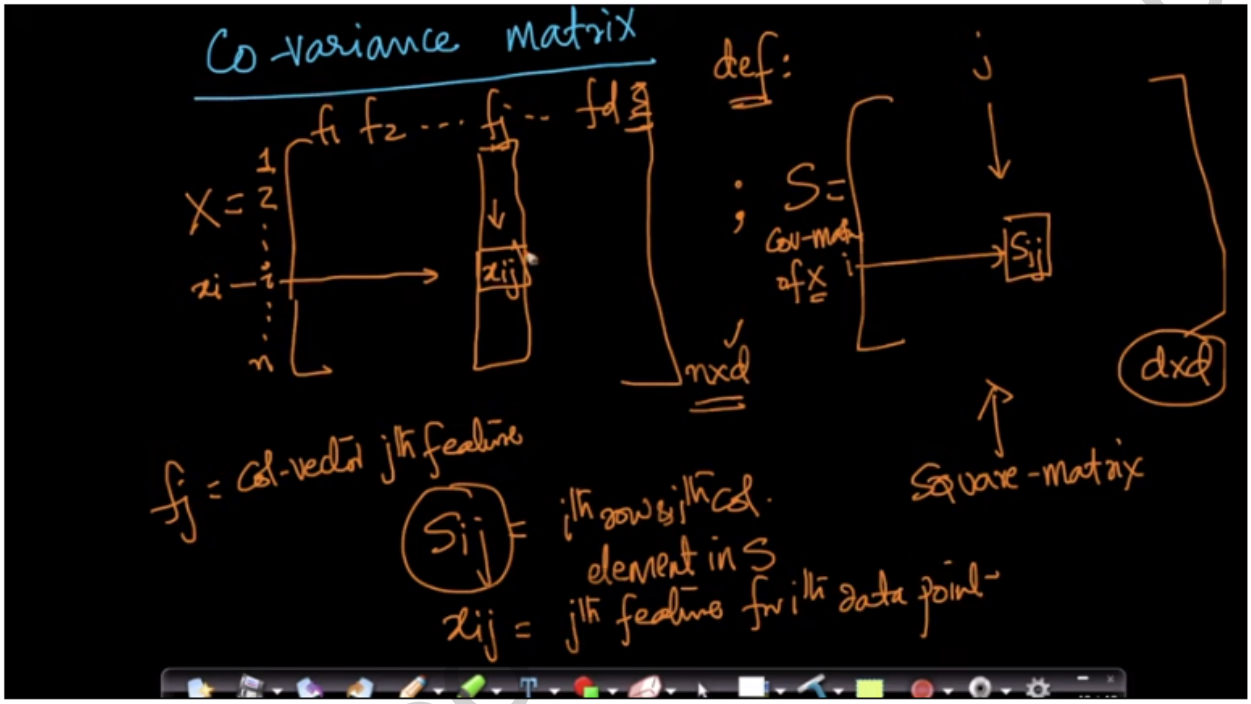
Geometrically, with column standardization we are moving the mean vector of our data to origin and squishing or expanding our data such that the standard deviation for any feature is 1.



Timestamp: 14:50

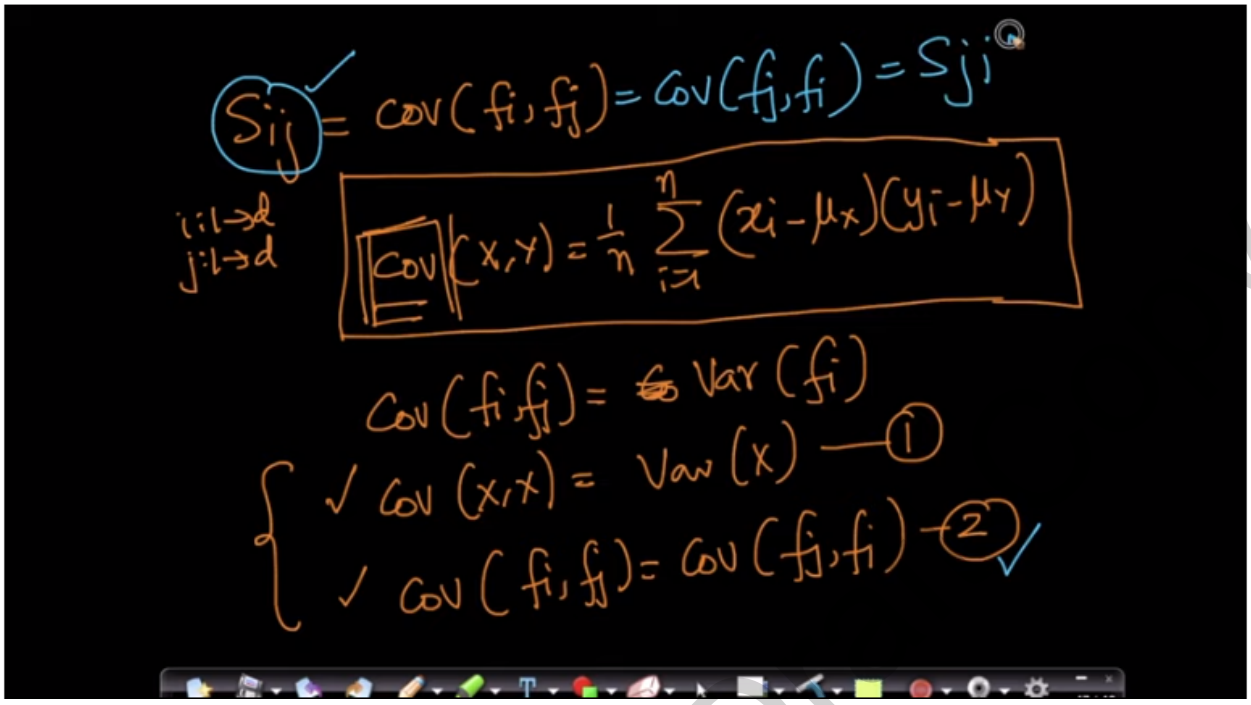
Basically, column standardization is considered as mean centering (such that mean vector is origin) plus scaling (such that standard deviation is 1 for each feature).

25.8 Co-variance of a Data Matrix



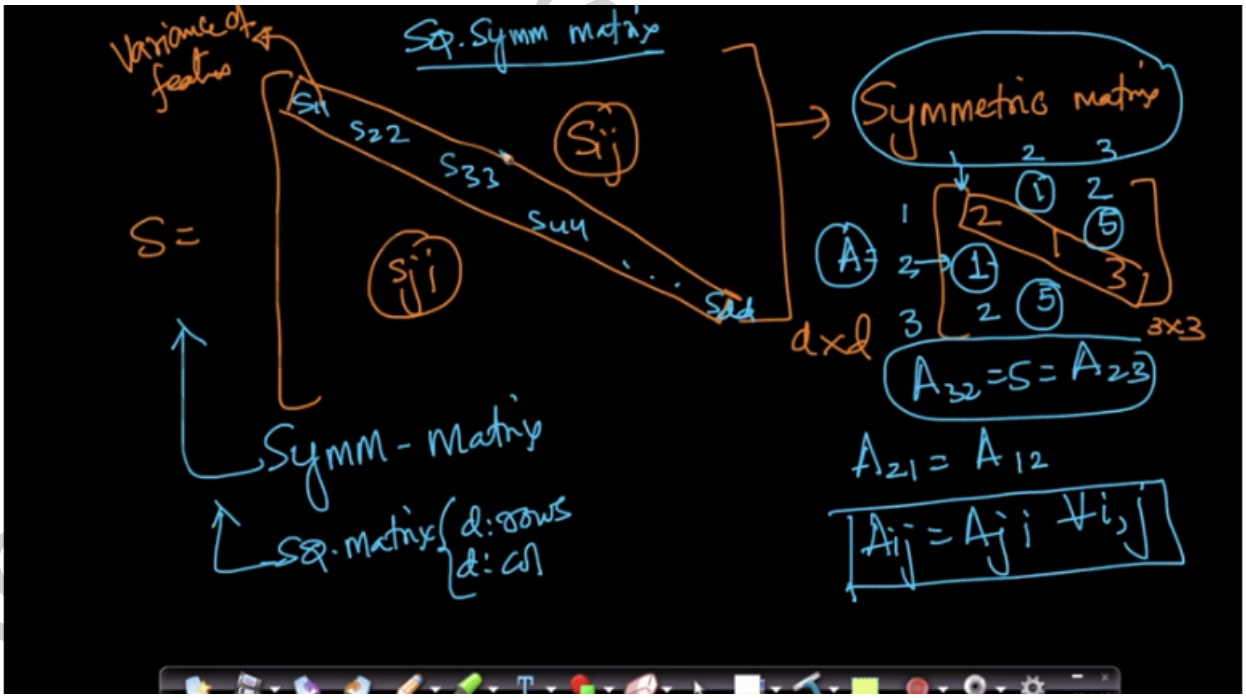
Timestamp: 4:00

We define the covariance matrix of our data S as a square matrix of dxd shape, where d is the number of features of our data. Each element Sij which is the element of ith row and jth column is defined as the covariance of ith feature and jth feature as below.



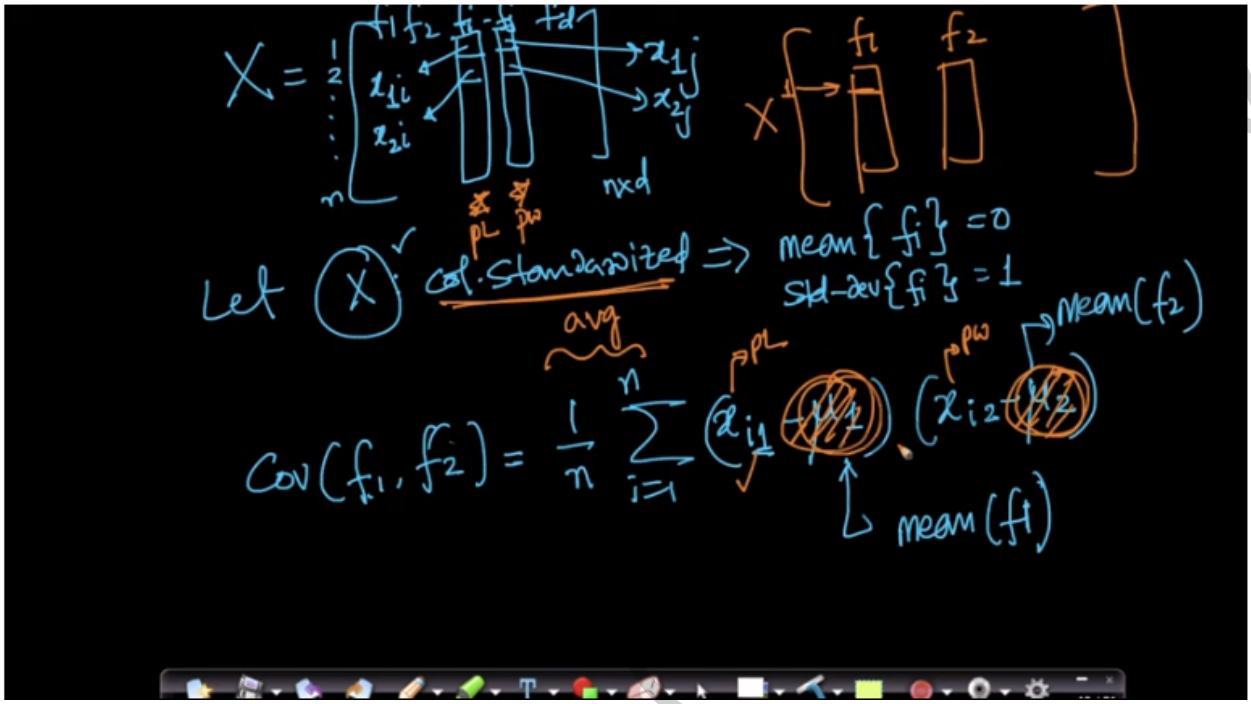
Timestamp: 9:27

Notice that due to how covariance is defined, cov(fi,fj)=cov(fj,fi) hence sij=sji.



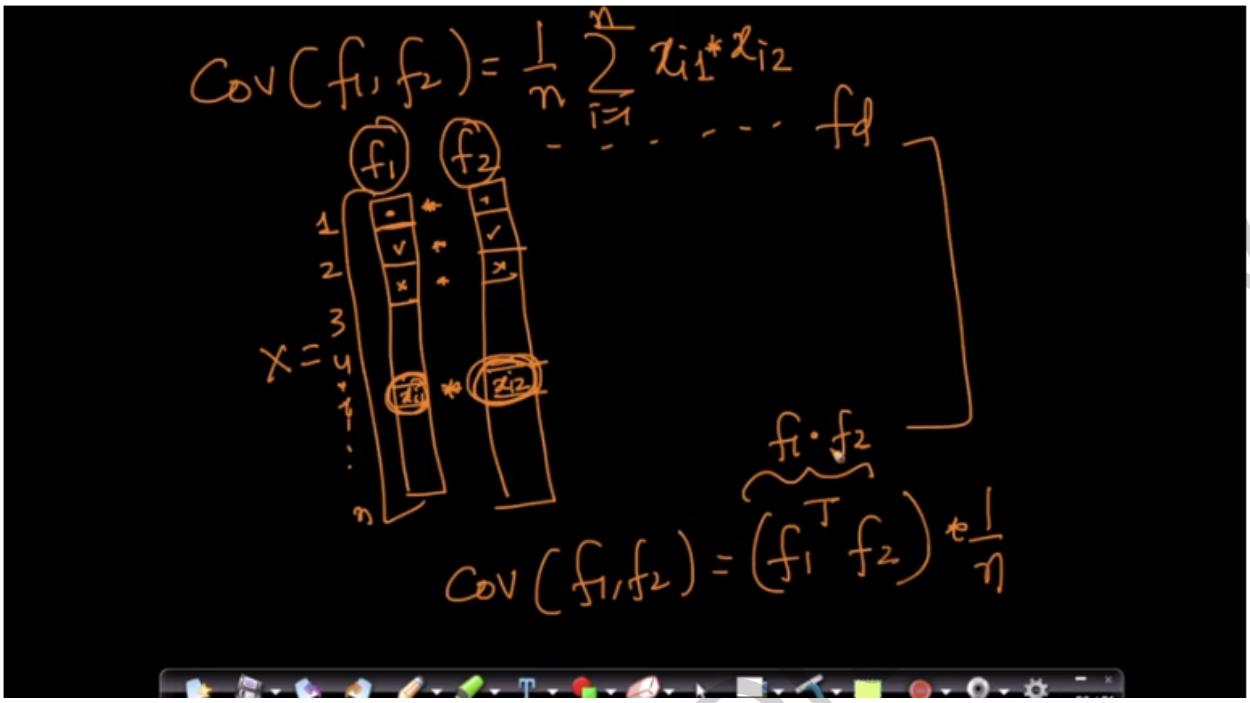
TImestamp: 10:15

As shown above, the covariance matrix S is a square symmetric matrix, as it satisfies all the rules of a square matrix where the number of rows must be equal to the number of columns and that of a symmetric matrix where Sij=Sji.



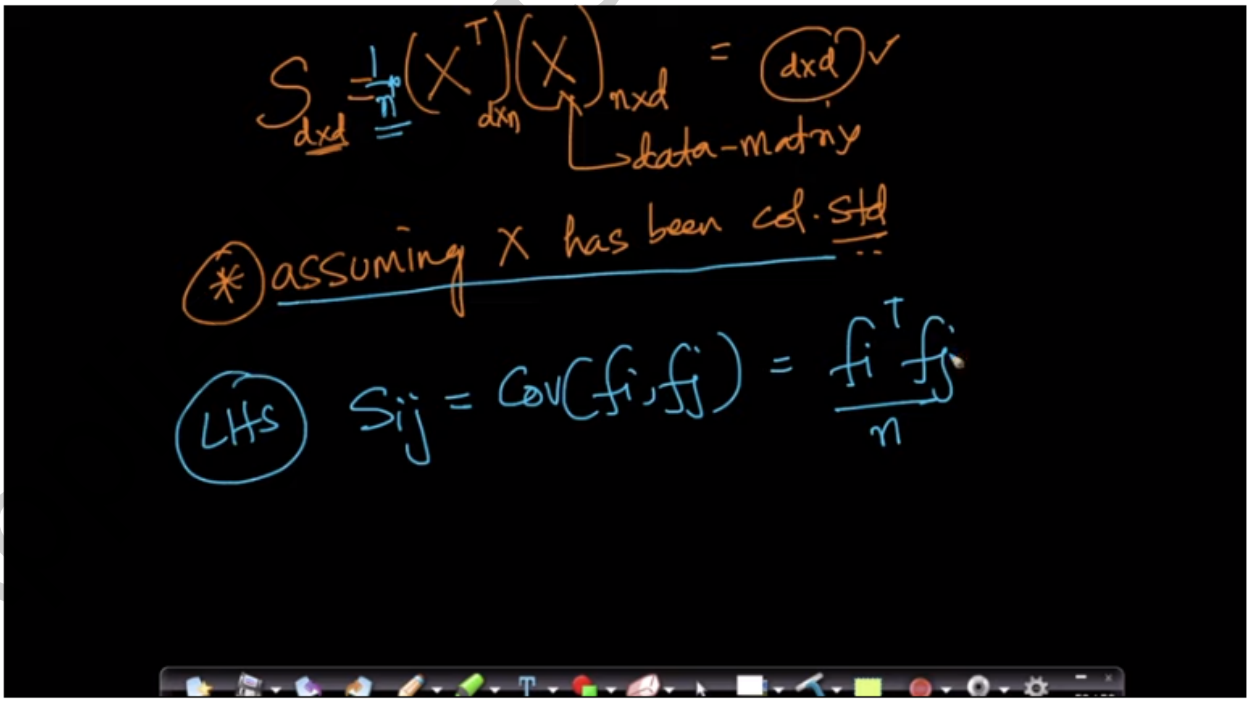
Timestamp: 14:47

If our data matrix X is column standardized then the mean of each feature is 0 and the standard deviation of each feature is 1, hence we are left with cov(f1,f2) as below



Timestamp: 17:19

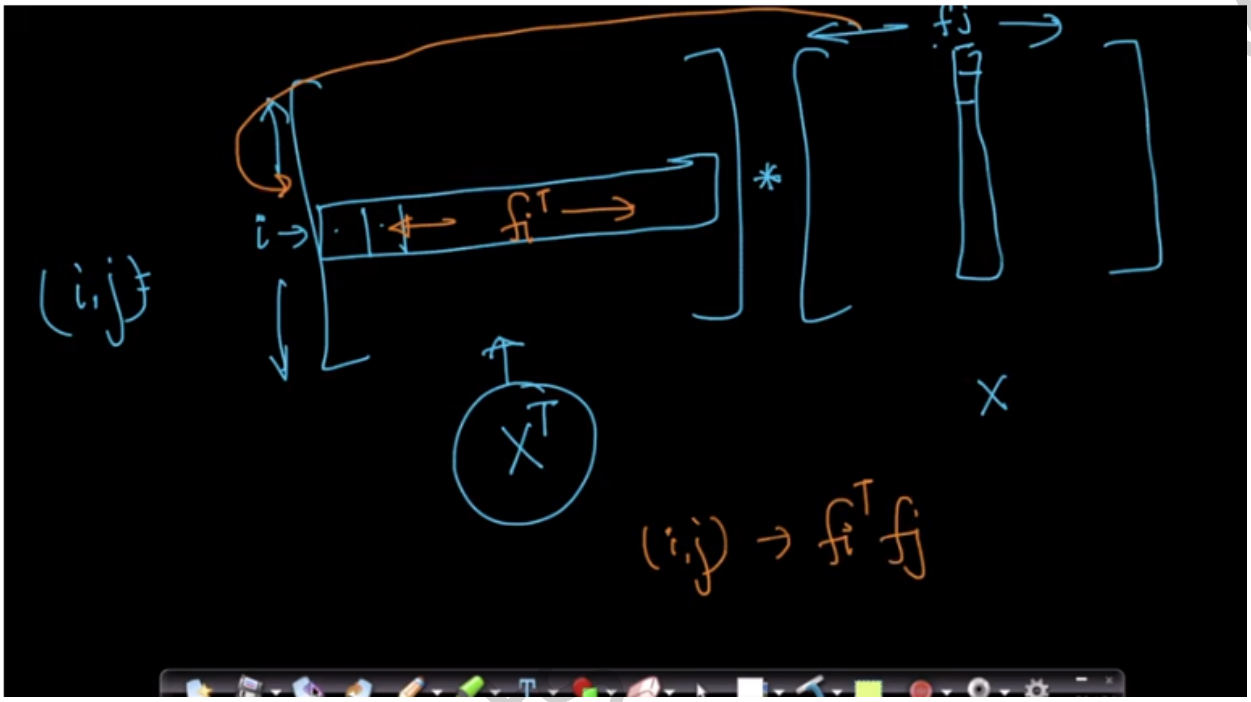
So Cov(f1,f2) would just be the summation as above. Cov(f1,f2) can also be represented as $(f1^{T}f2)/n$.



Timestamp: 19:40

Since each element Sij of covariance matrix S is the covariance of feature fi and fj,and the cov(fi,fj)=(fi^Tfj)/n, therefore Sij=(fi^Tfj)/n.

We can prove that S = (X^TX)/n, by using the below figure.

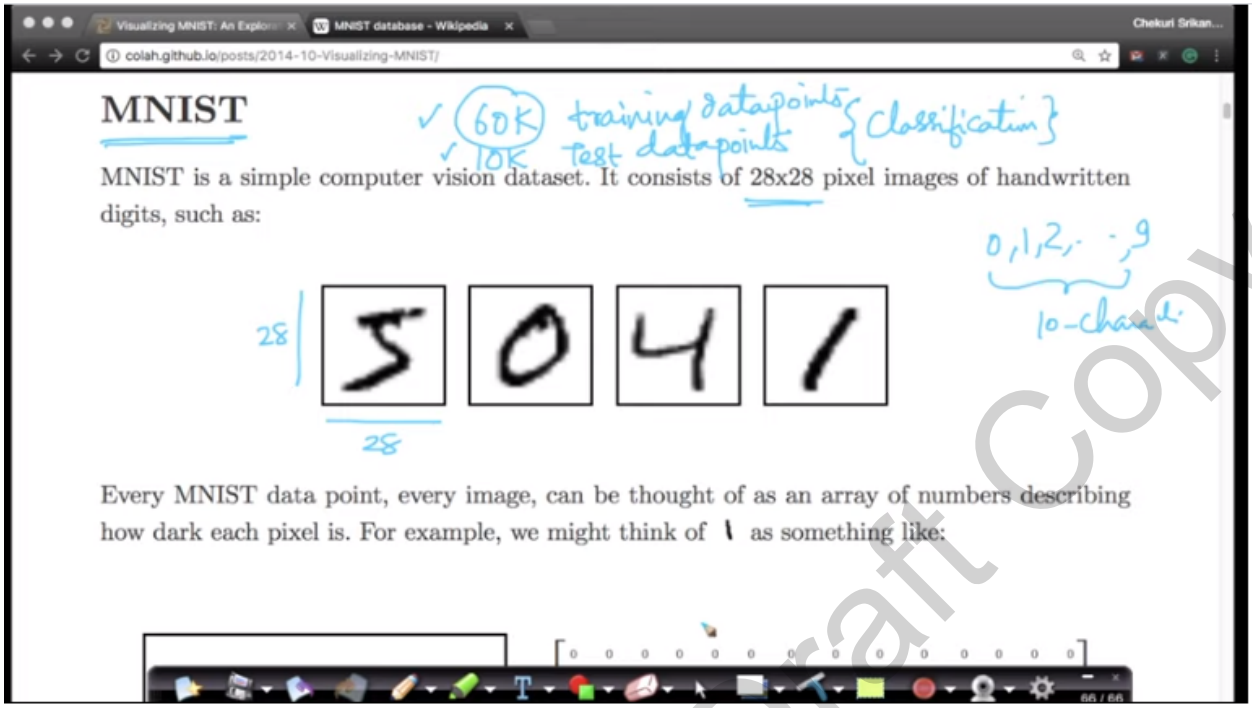


Timestamp: 22:24

From the above figure we can see the (i,j) the element of X^TX is fi^Tfj hence the (i,j) the element of (X^TX)/n will be (fi^Tfj)/n which is how we defined Sij.

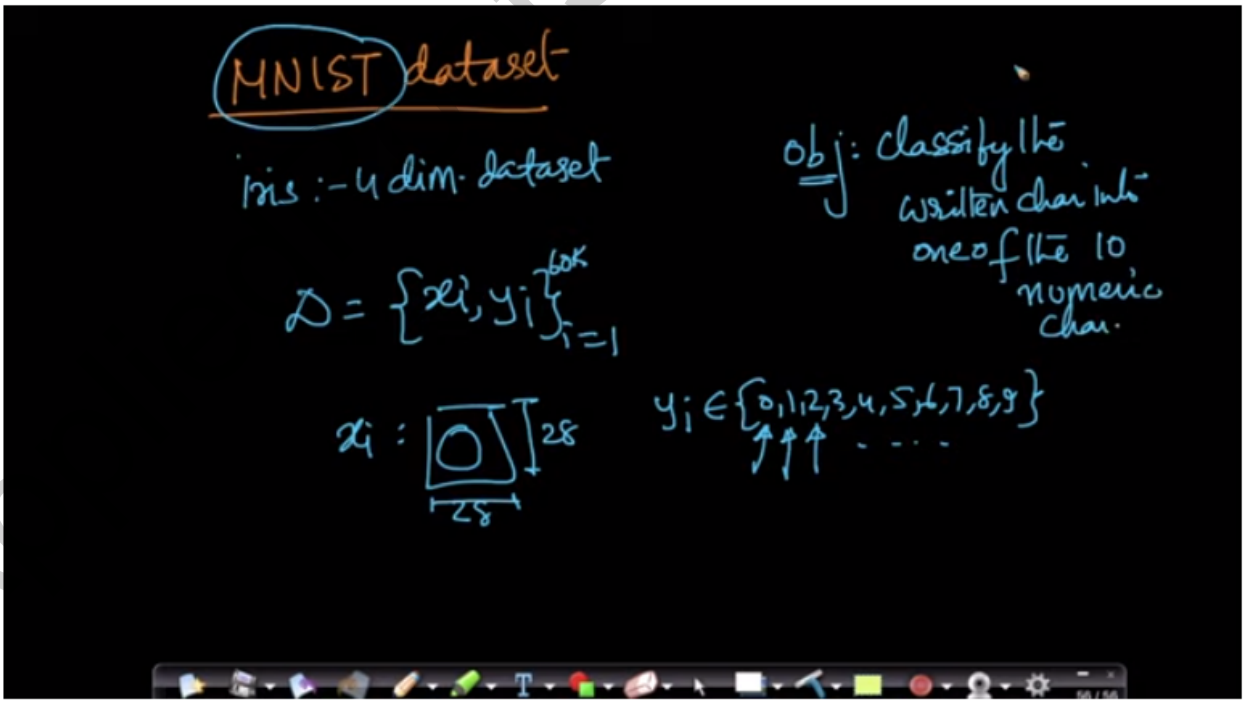
Hence S = (X^TX)/n if X has been column standardized.

25.9 MNIST dataset



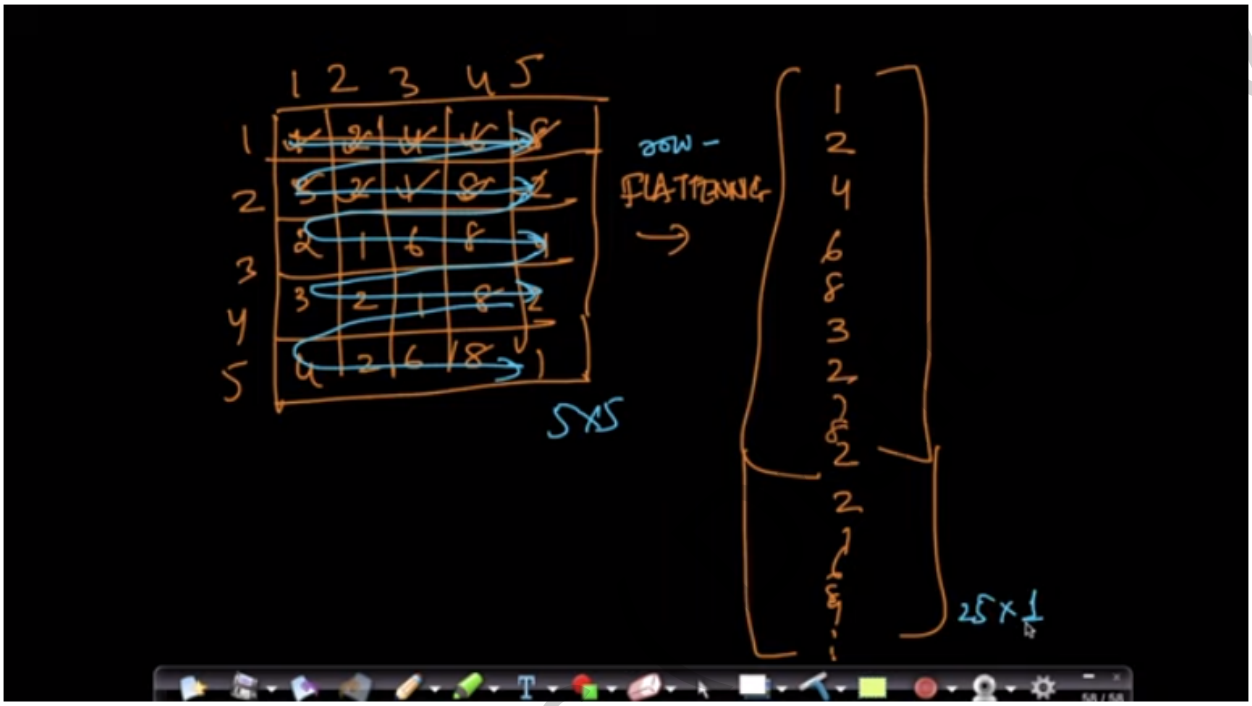
Timestamp: 3:43

MNIST dataset is a computer vision dataset, where we are given 28x28 pixel images containing hand written digits and their corresponding labels.



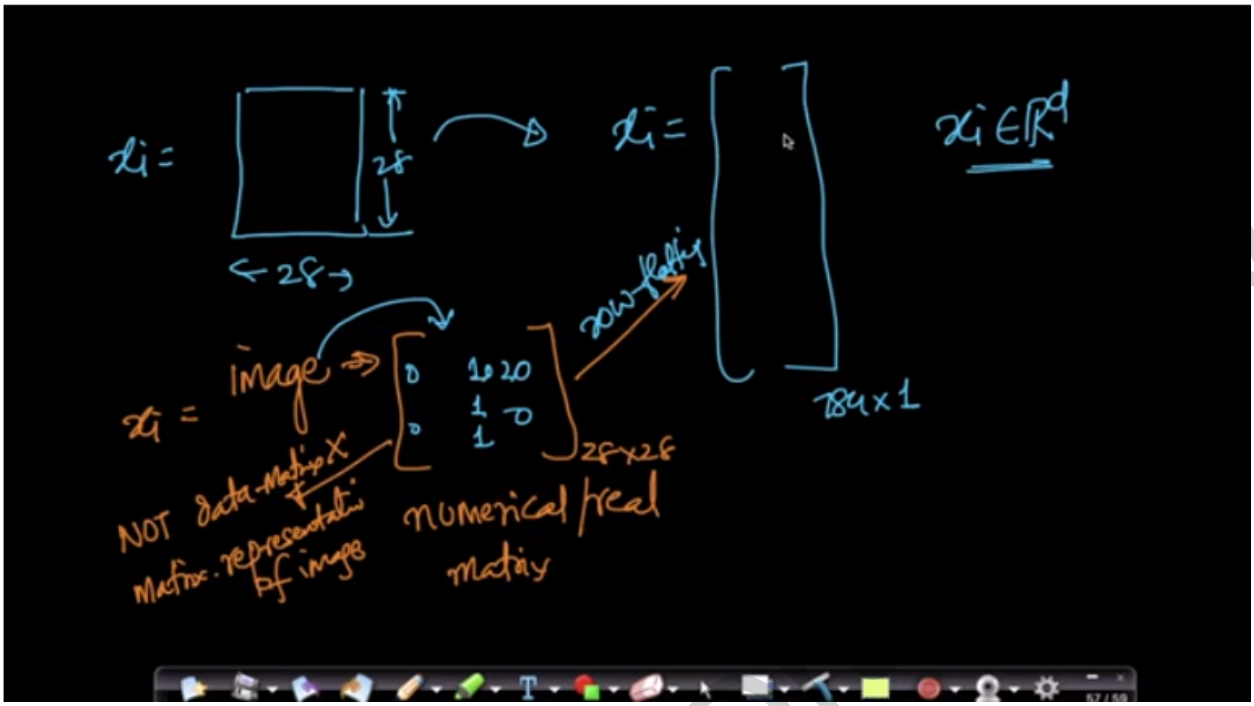
TImestamp: 5:30

The dataset can be represented as D= {xi,yi} where i is from 1 to n, where each xi is an image containing 784 pixels which are represented by real values (1 indicating black and 0 indicating white and values between 0 and 1 representing shades of grey) and yi is the corresponding numeric value of that image between 0 and 9.



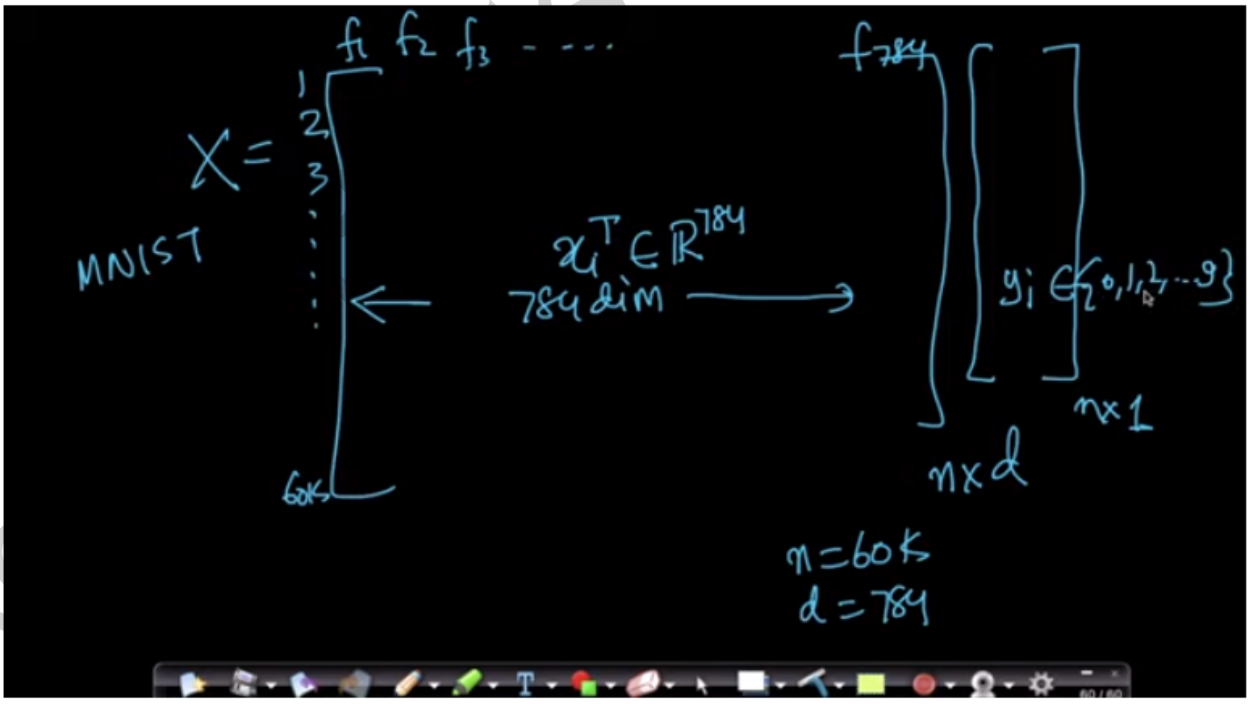
Timestamp: 10:24

Above is an example how we can do row-flattening to convert a matrix of shape 5x5 to 25x1 by first adding all the elements of the first row, followed by all elements of second row and so on.



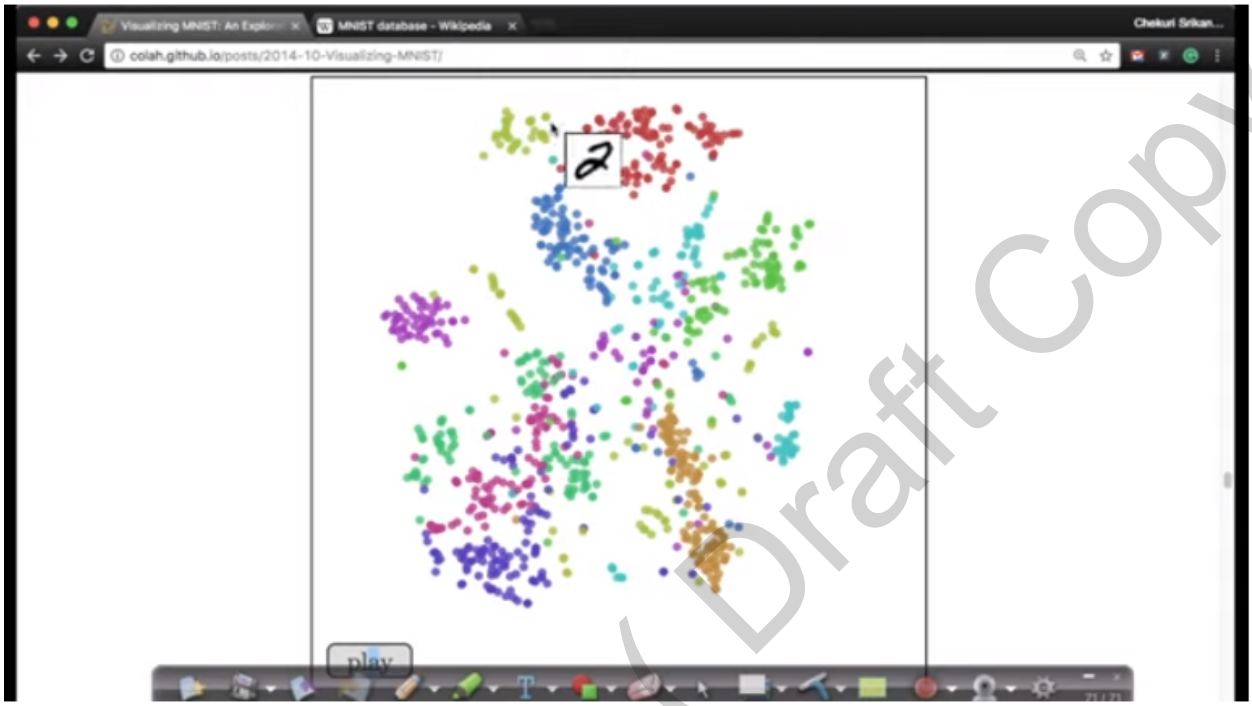
Timestamp: 10:24

So for each data point xi, if we do row-flattening we get a column vector of 784x1 dimension.



Timestamp: 13:06

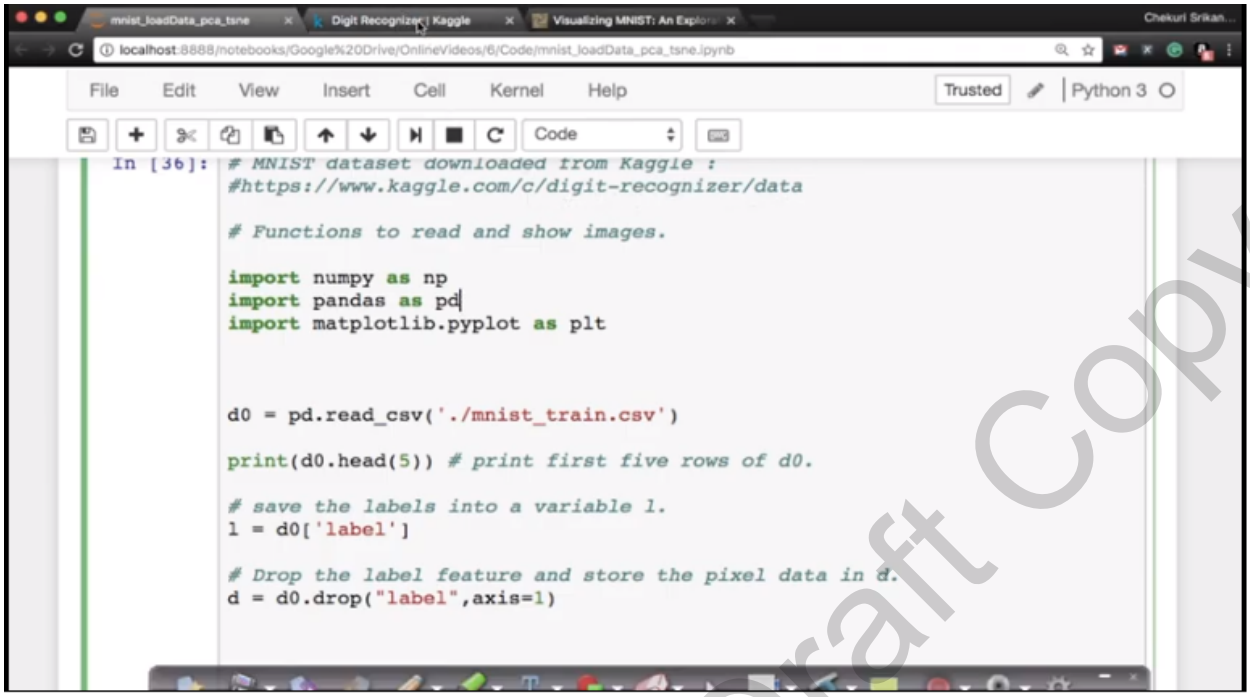
So our dataset will be represent as above, where X is the data matrix of dimensions 60K x 784 since there are 60K training data points and 784 features for each data point and yi contains the label for the corresponding datapoint xi and belongs to [0,9].



Timestamp: 18:51

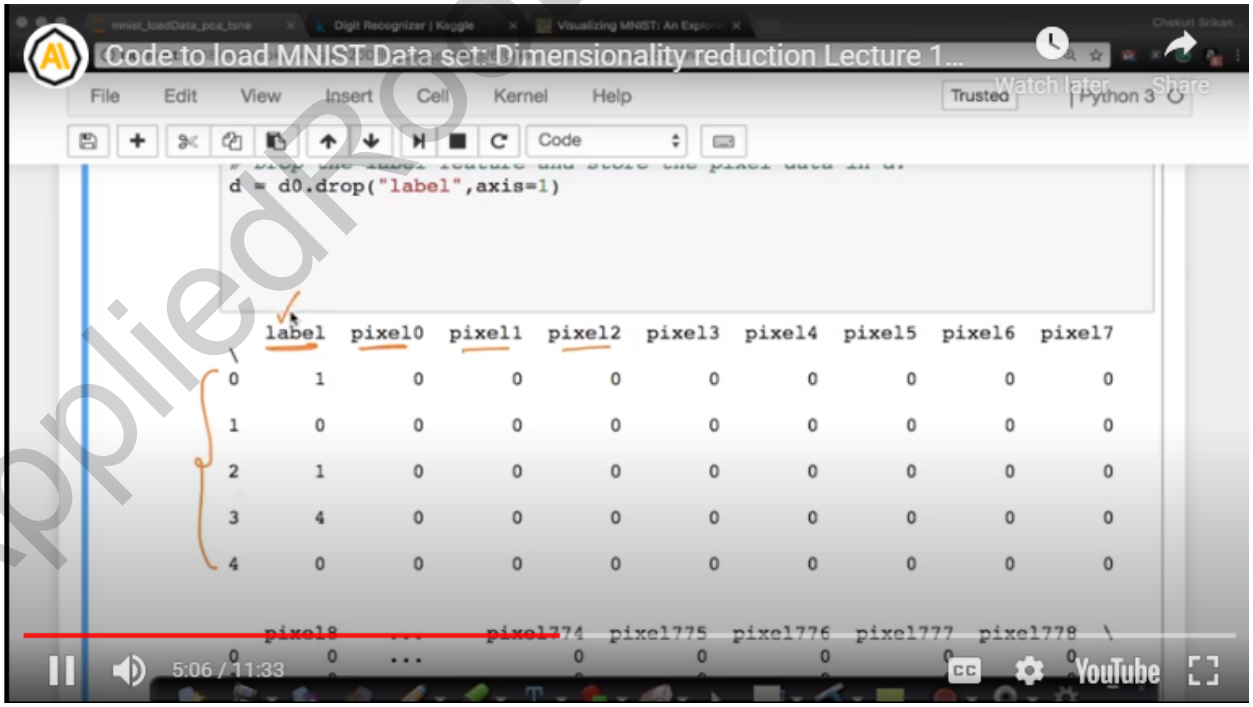
Since we have 784 dimensional data which is difficult to visualize, we use dimensionality reduction technique called t-SNE and above are the results(we will see how to apply t-SNE in later chapters). Notice that each point represents each datapoint in the reduced dimension and the color represents the class-label of the datapoint.

25.10 Code to load MNIST dataset

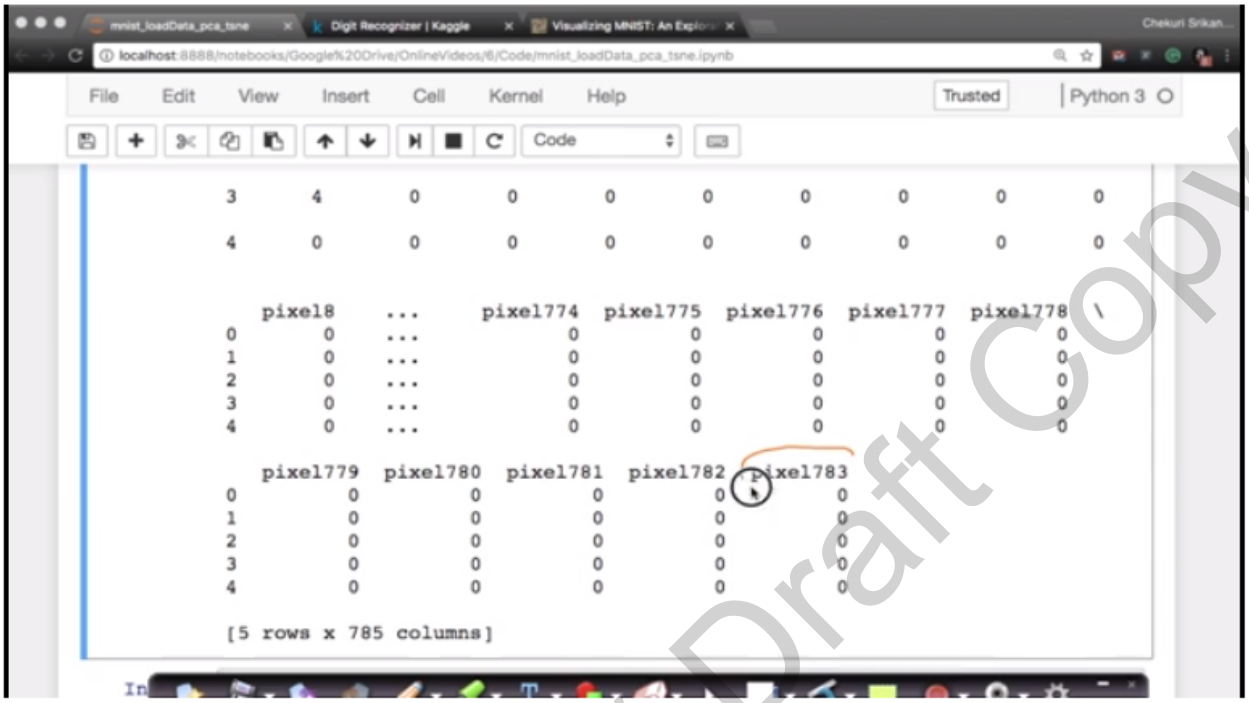


Timestep: 1:42

Downloaded the data from the link mentioned, renamed the train csv as mnist_train.csv, imported the necessary libraries and storing the label data and pixel data into l and d respectively.

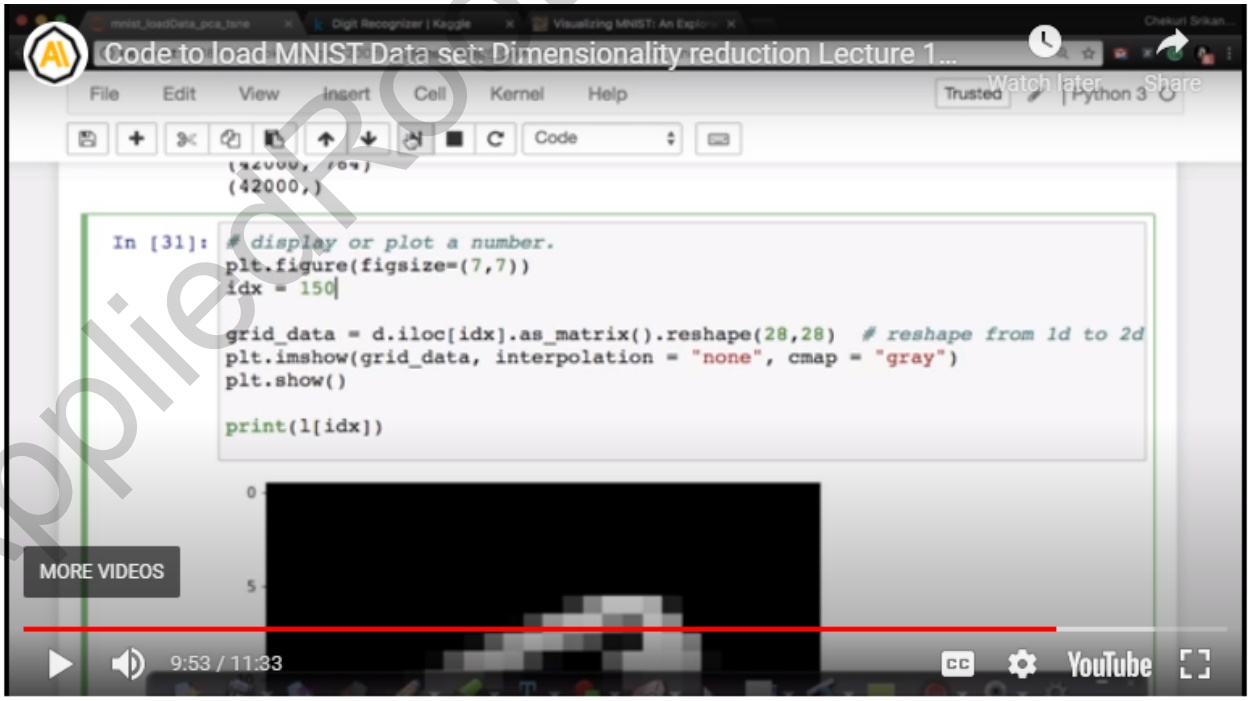


Timestamp: 5:06



Timestamp: 5:30

As discussed in our data, we have the label column and pixel data from pixel 0 to pixel 783.



Timestamp: 9:51

Using the above code, we plot images for us to visualize. The plots look like below.

TImestamp: 9:55