Principal Component Analysis

PCA is an unsupervised statistical technique that is used to reduce the dimensions of the dataset. ML models with higher dimensionality tend to fail when operating on a higher input dataset. PCA helps in identifying relationships among different variables & then coupling them. PCA works on some assumptions it helps developers maintain a standard.

Or

Principal Component Analysis or PCA is a simplest and very fundamental technique which is extensively used for dimensionality reduction of the large data set. Reducing the number of components or features costs some accuracy and on the other hand, it makes the large data set simpler, easy to explore and visualize. Also, it reduces the computational complexity of the model which makes machine learning algorithms run faster.

PCA transform the features into a new set of features by rotating the axes which are called PCs (Principal Components). The principal components would be equal to the number of original features in the given dataset.

The first principal component (PC1) contains the maximum variation which was present in earlier features, and this variation decreases as we move to the lower level. The final PC would have the least variation among all features and you will be able to reduce the dimensions of your feature set.

Assumptions:

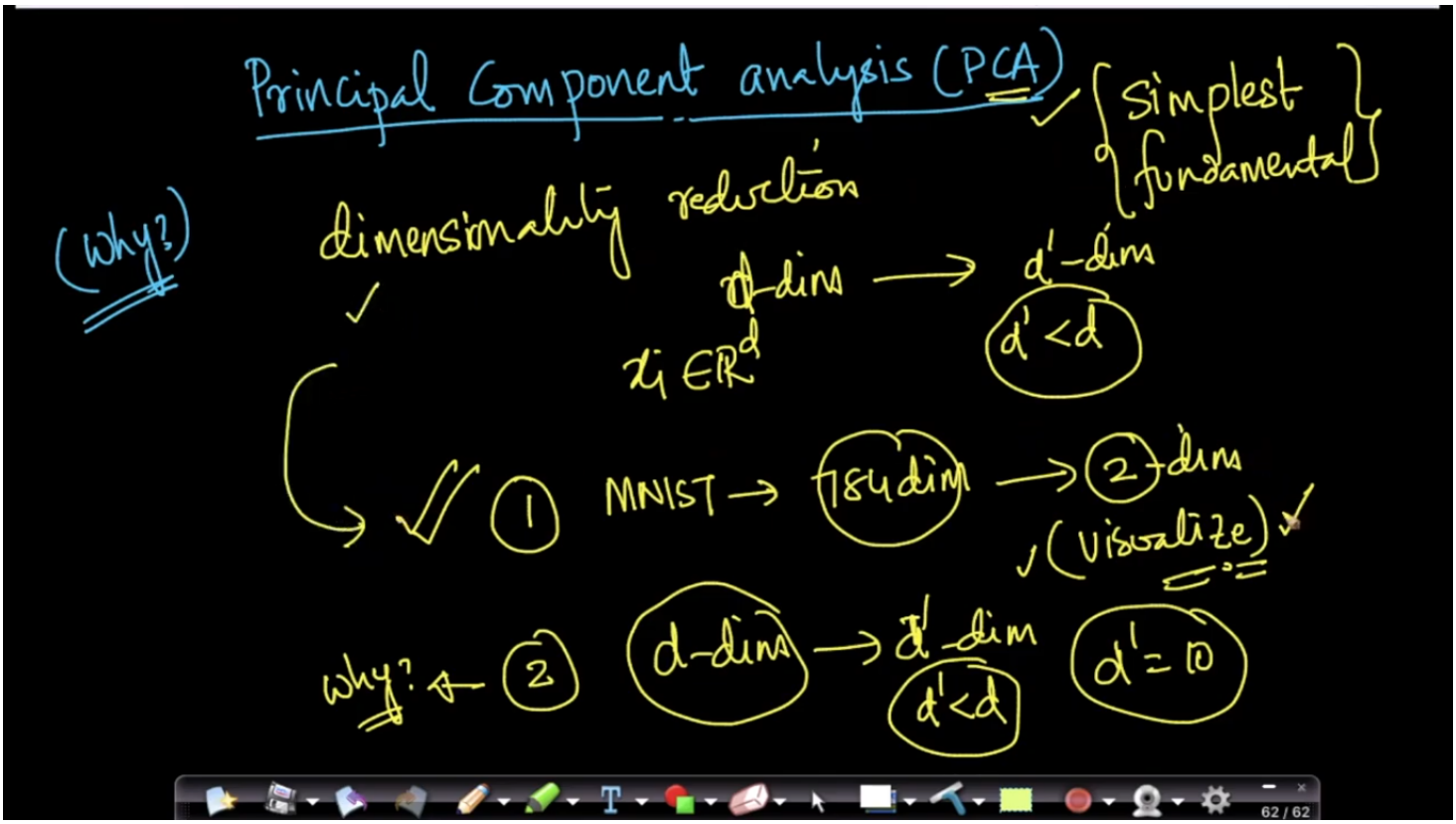
1. **Linearity**: There must be linearity in the data set, i.e. the variables combine in a linear manner to form the dataset. The variables exhibit relationships among themselves.

2. **Outliers** : outliers, should be less. More number of outliers will represent experimental errors and will degrade your ML model/algorithm.

3. The feature set must be correlated and the reduced feature set after applying PCA will represent the original data set but in an effective way with fewer dimensions.

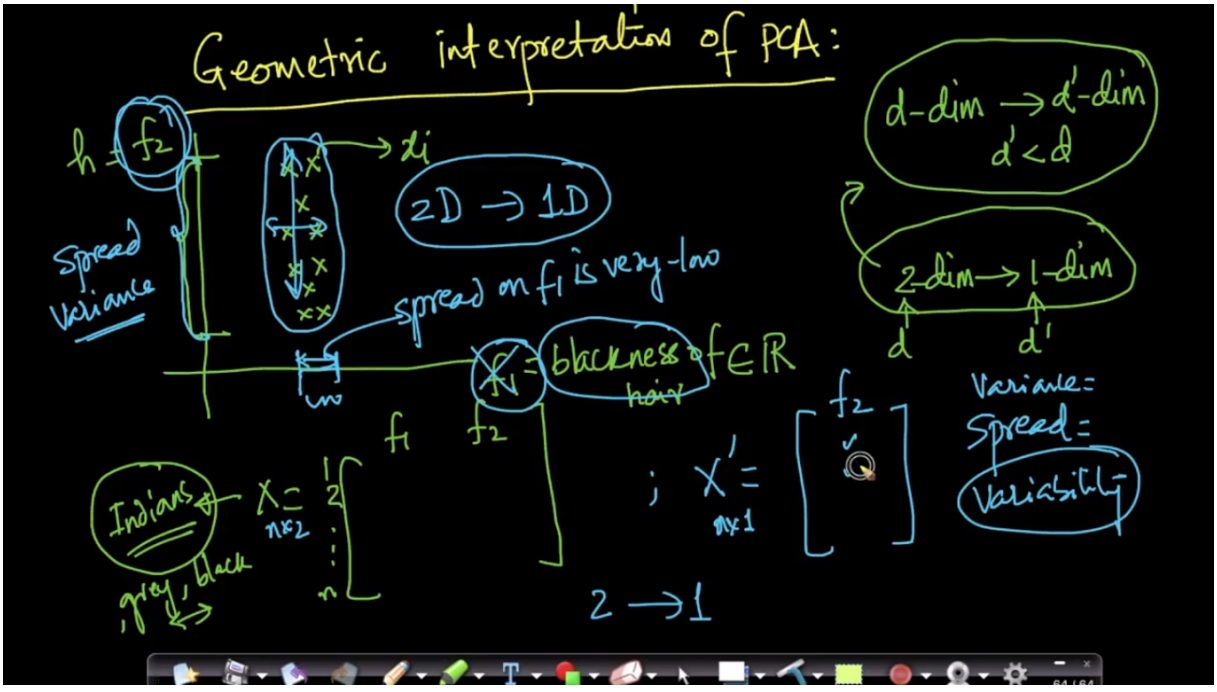
4. ***Large variance implies more structure:*** high variance axes are treated as principal components, while low variance axes are treated as noise and discarded.

Now for MNIST datdset it has 784 dimensions we can’t visualize it in 2D by using pair plot or scatter plot

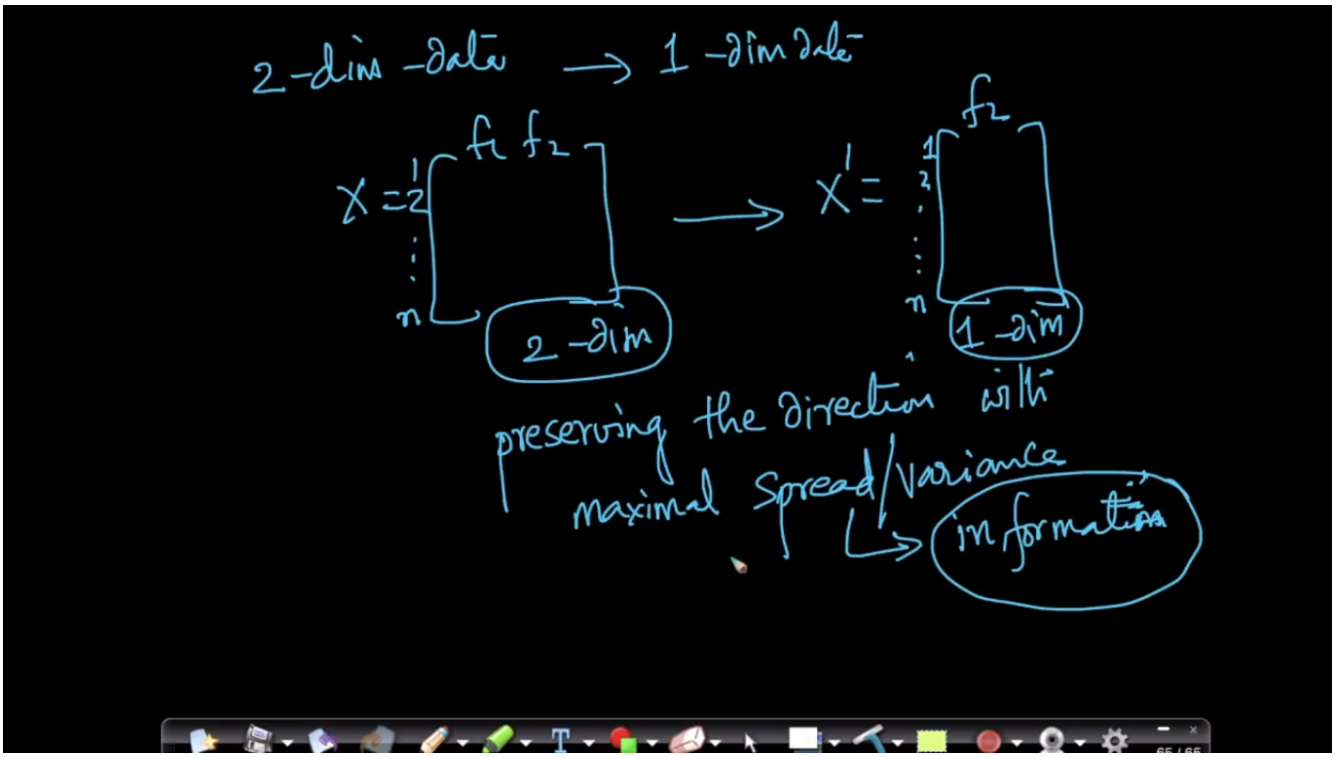


Geometric interpretation of of PCA:

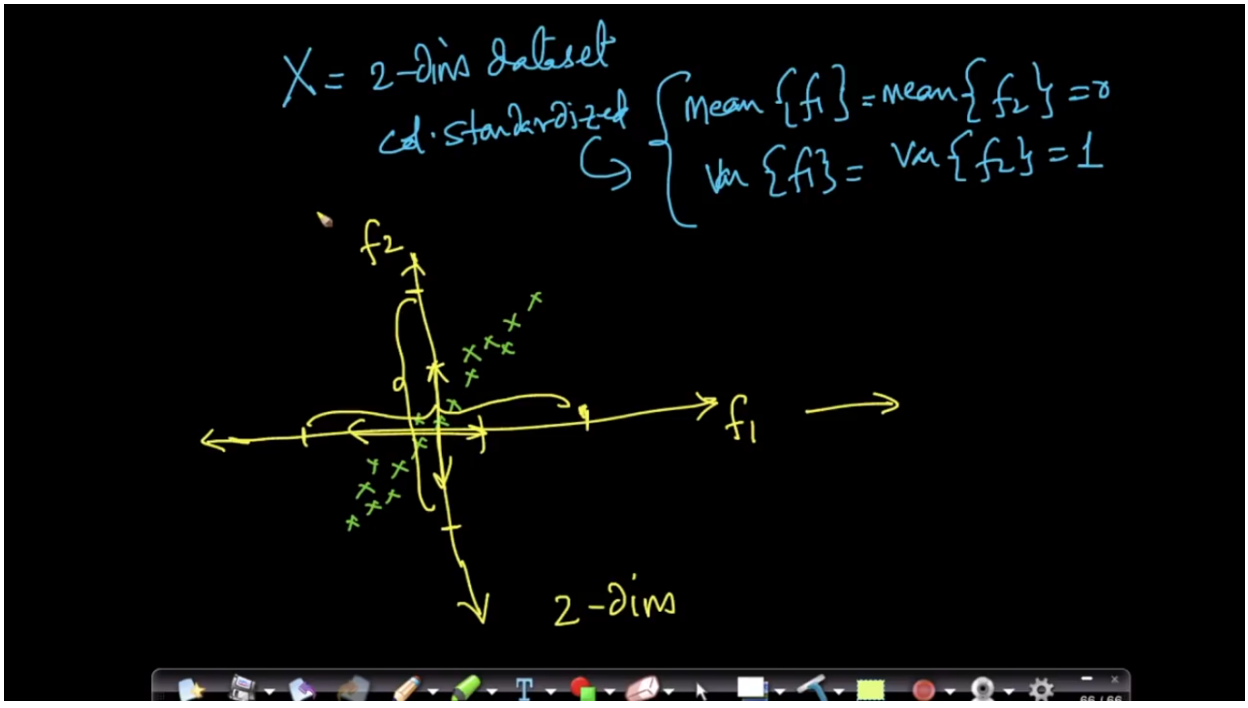
For visualization we convert d-dim. (Higher dim.) data to lower d' -dim. Data or we converting data 2D to 1D, Let say we have two feature f1 and f2 we plot f1 feature “blackness of hair” in x-axis and f2 feature is “height” in y-axis. And our dataset x is a datamatrix of this two f1 and f2 features from the plot we can say variance or spread of height or f2 feature is more compare to f1 feature now suppose we want to convert this 2D dataset into 1D or represent the x dataset into x' data matrix which contain only one feature or one vector, in that case we only keep one feature f2 and skip f1 feature bcz the spread is very low in f1 axis or we have to force to skip one feature between this two then we only skip f1 bcz by skiping this feature we lose very less information



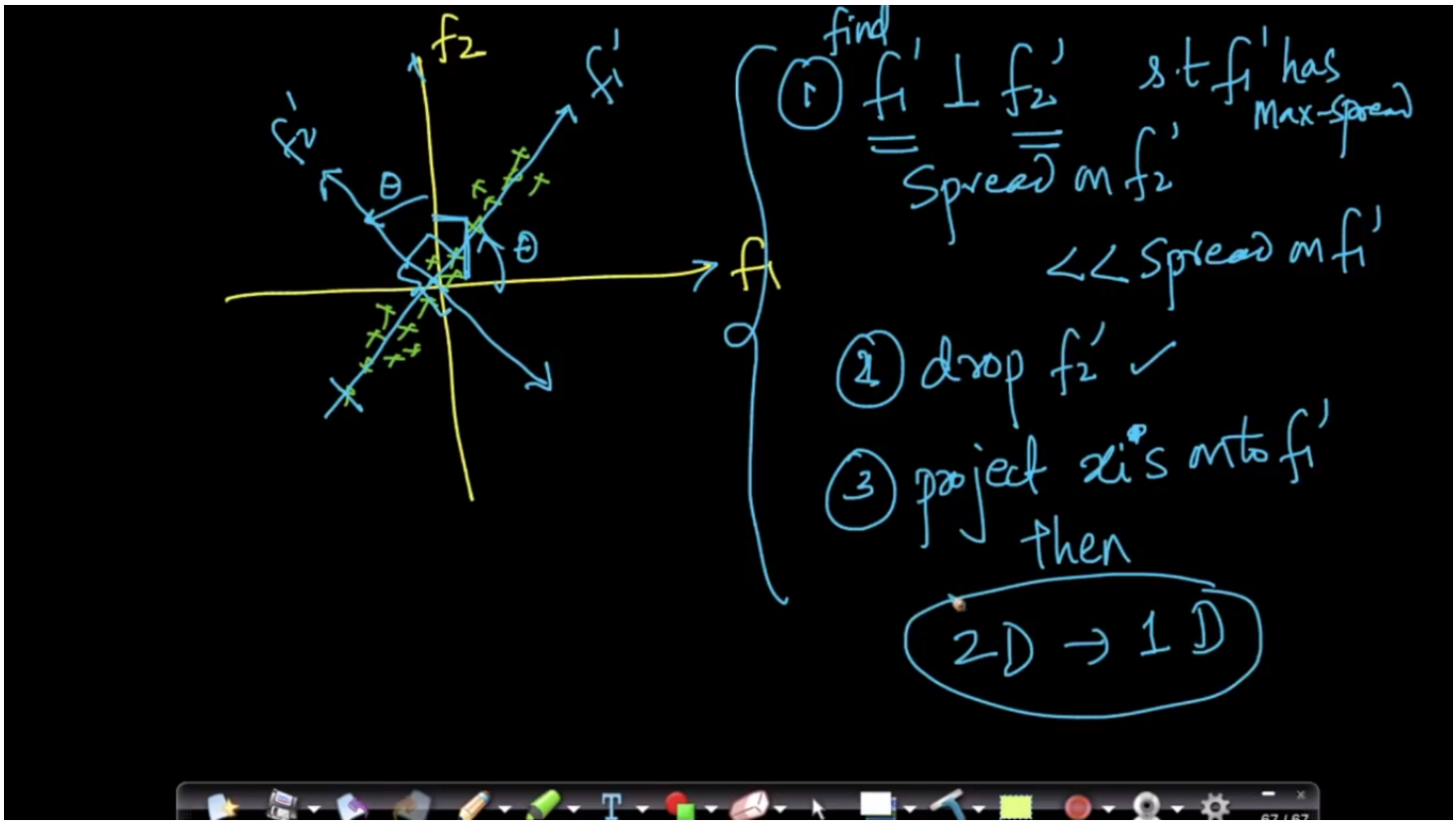
now we represent our data from 2D to 1D means convert the data x to x', so here we preserving **the direction** with maximal spread or maximal variance so more spread means more information so at the time of converting data high dimensiional to low dimensional we only preserve those features giving us more spread or more variance



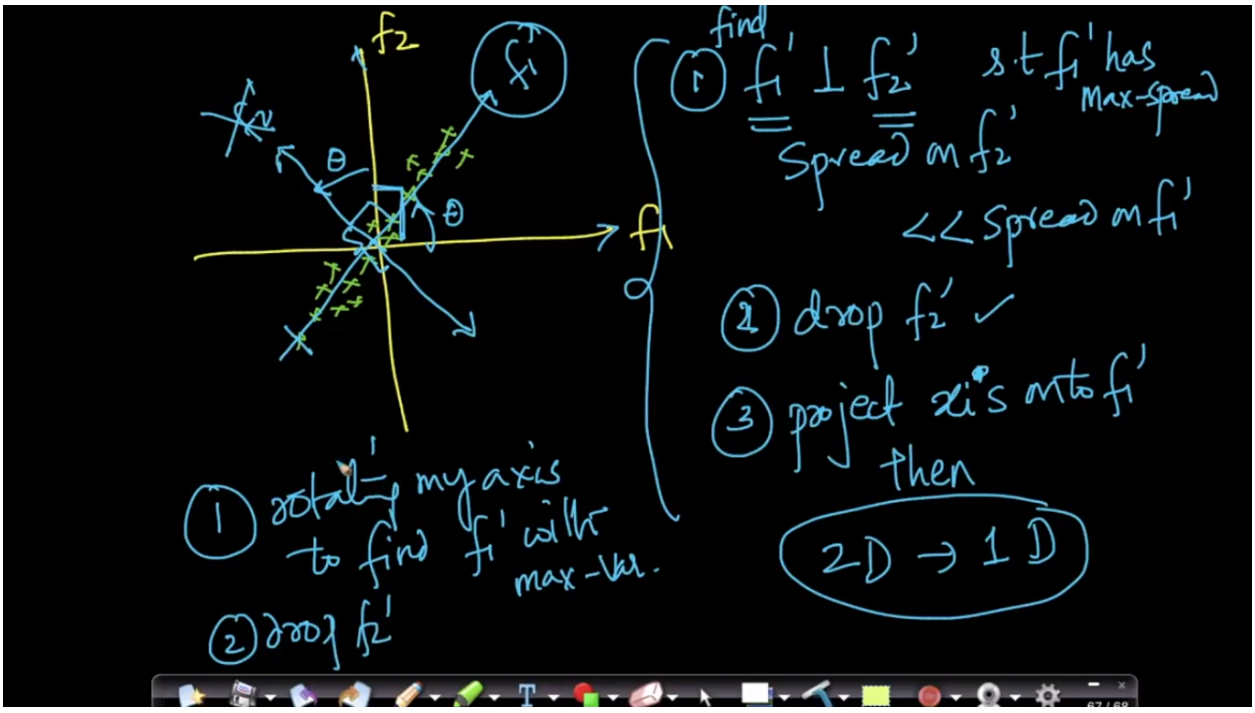
Now lets take another complex example our dataset x is 2 dimensional and dataset has been col. Standardize means mean of f1 and f2 is zero and varience of feature f1 and f2 is one and our dataset look like this and it is also a two 2d data and from the graph we can see sprade of data in f1 and f2 are same and also here we want to convert this 2d data into 1d



now we can see in f1' direction spread is very high and take one direction f2' perpendicular to f1' and spread in f2' is much much lesser than f1' and if we drop f2' and project all the point on f1' then we can easily convert this 2d to 1d , it is look like we rotating f1 with some angle θ and rotating f2 with same angle θ

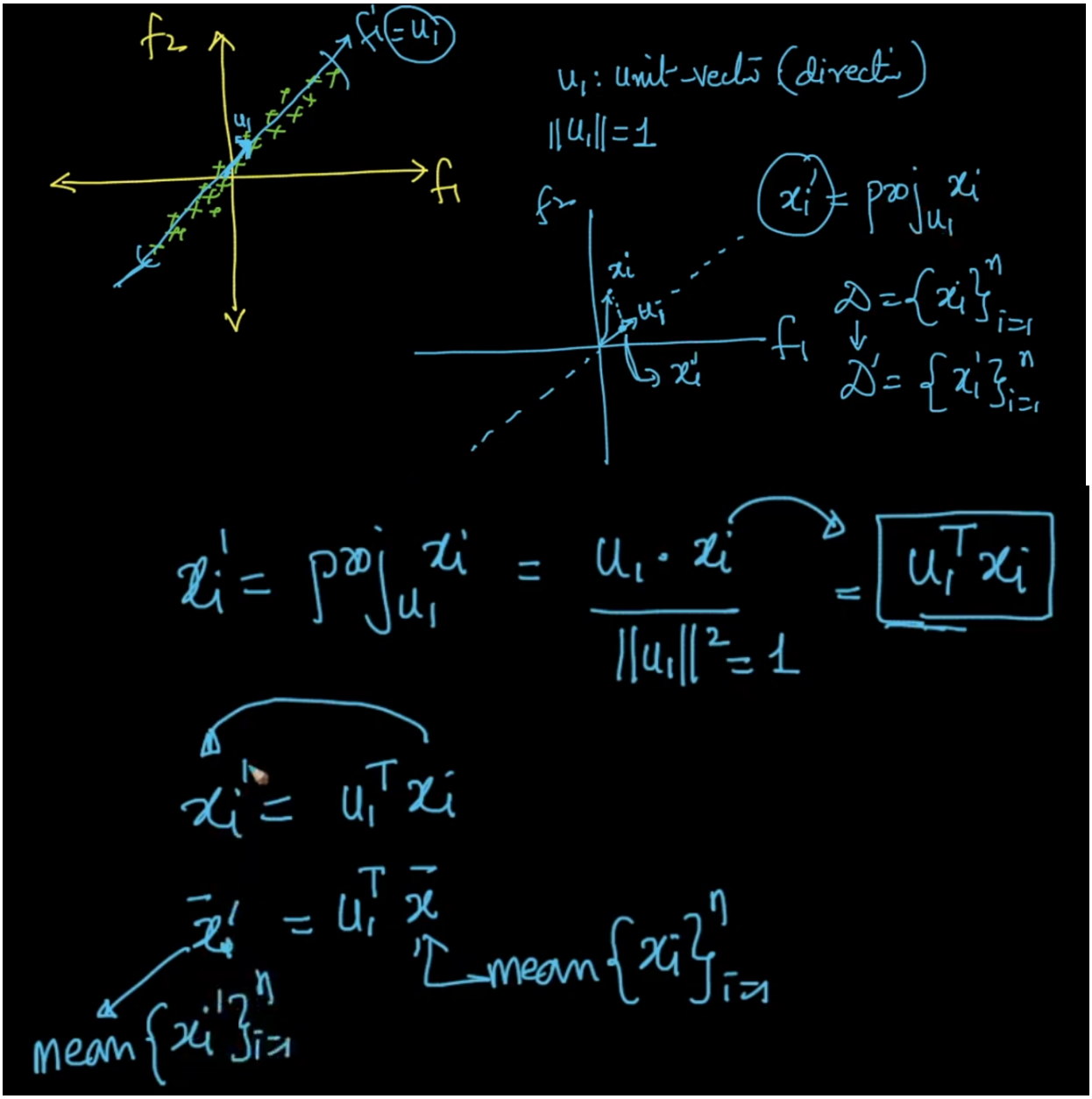


So here we want to find a direction f1' such that the variance of x_i projected onto f1' is maximal



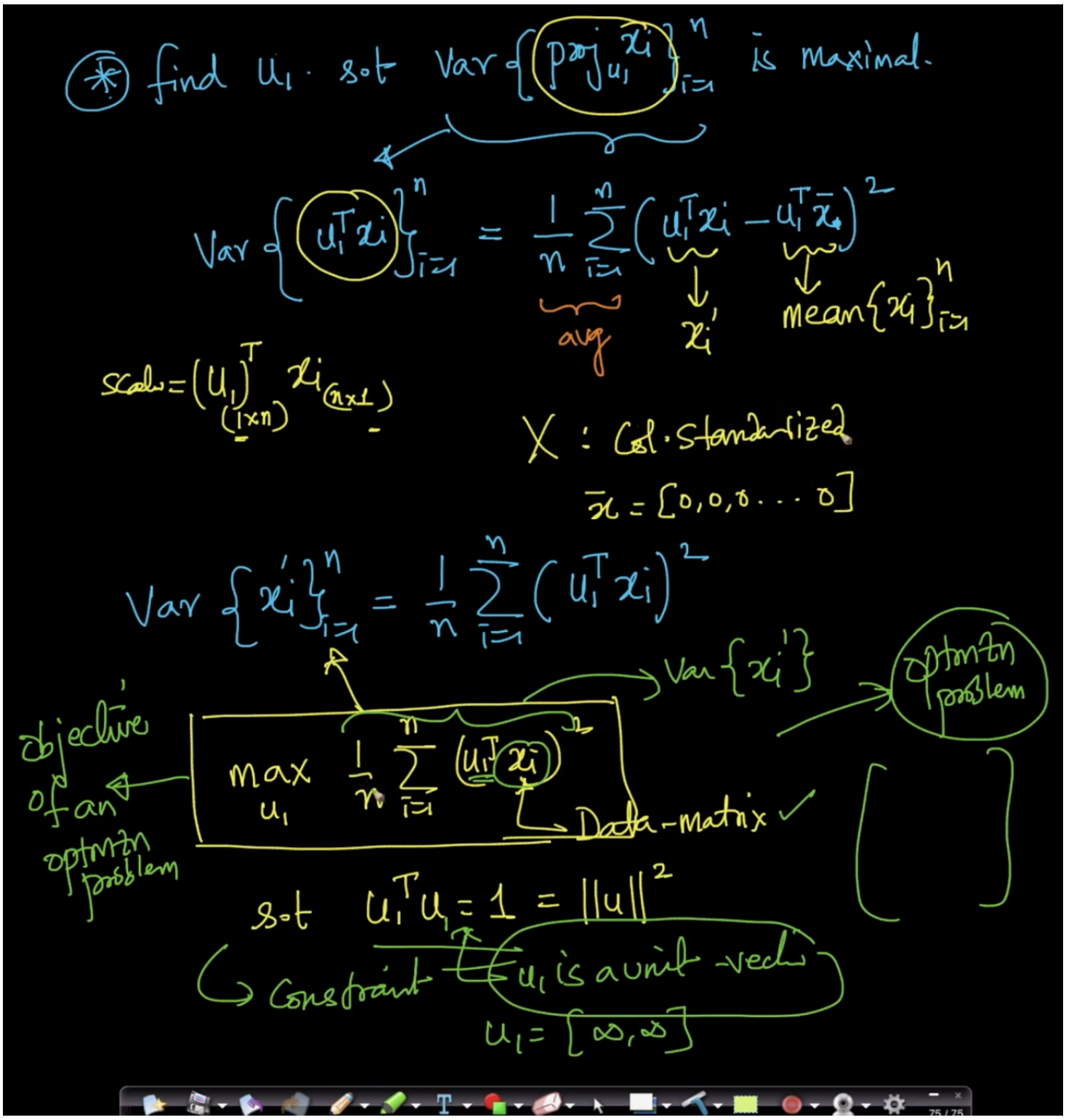
Mathmatically objective function of PCA

Let say we are assuming f1' = u1 & we are caring about finding direction of u1 bcz once we know the direction we can project the points on direction and we know we represent the direction using unit vector means the length of this unit vector(u1)=1

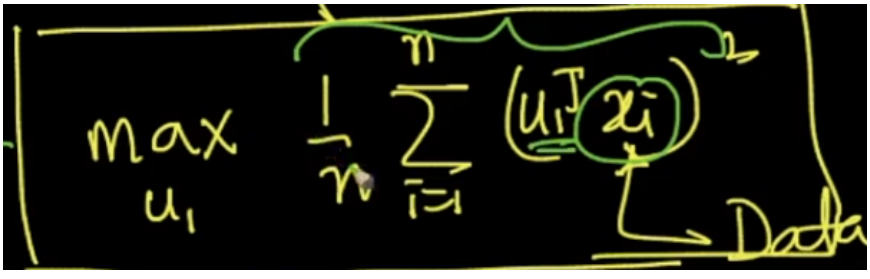


so given any point x_i we can convert into x_i' usinng u1 transpose and we can do same operation by using mean of x.

So now our task, for a point to find u1 such that.....

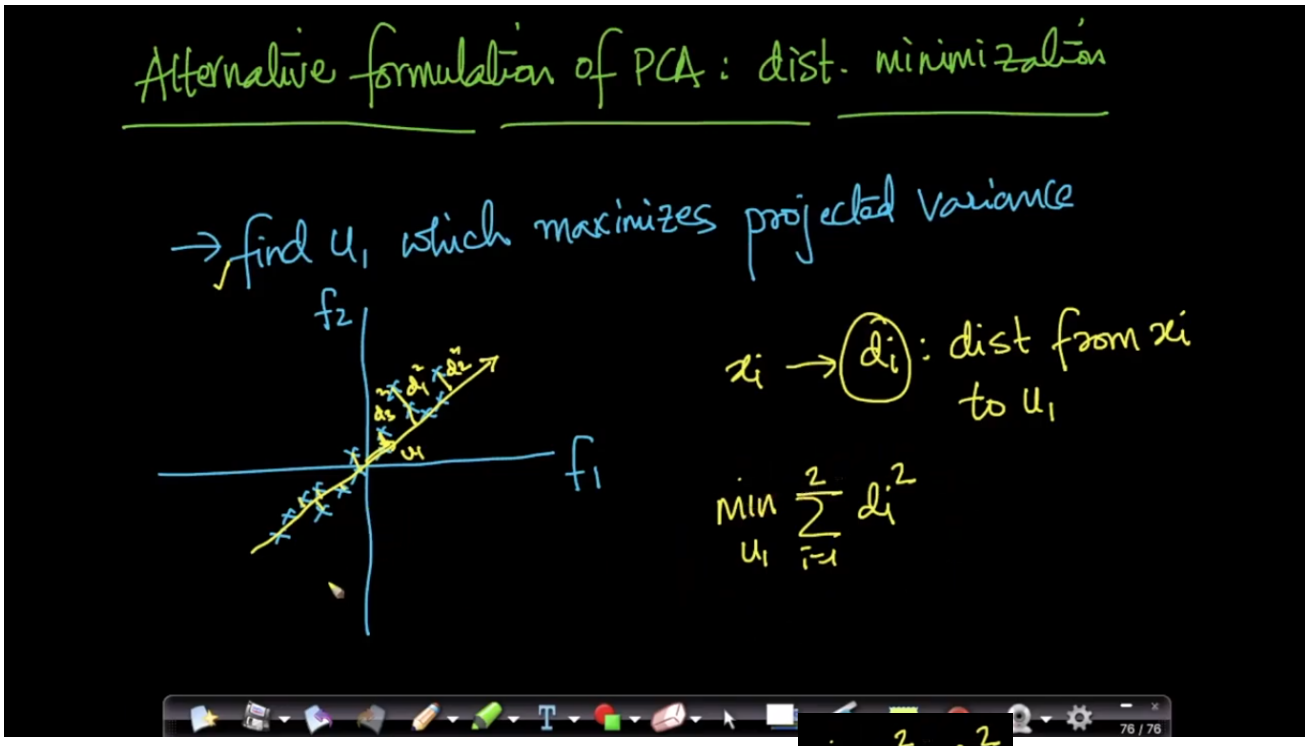


so our task is basically maximize the variance x_i' and we want to find out u1, u1 is the that direction that maximize the variance which is



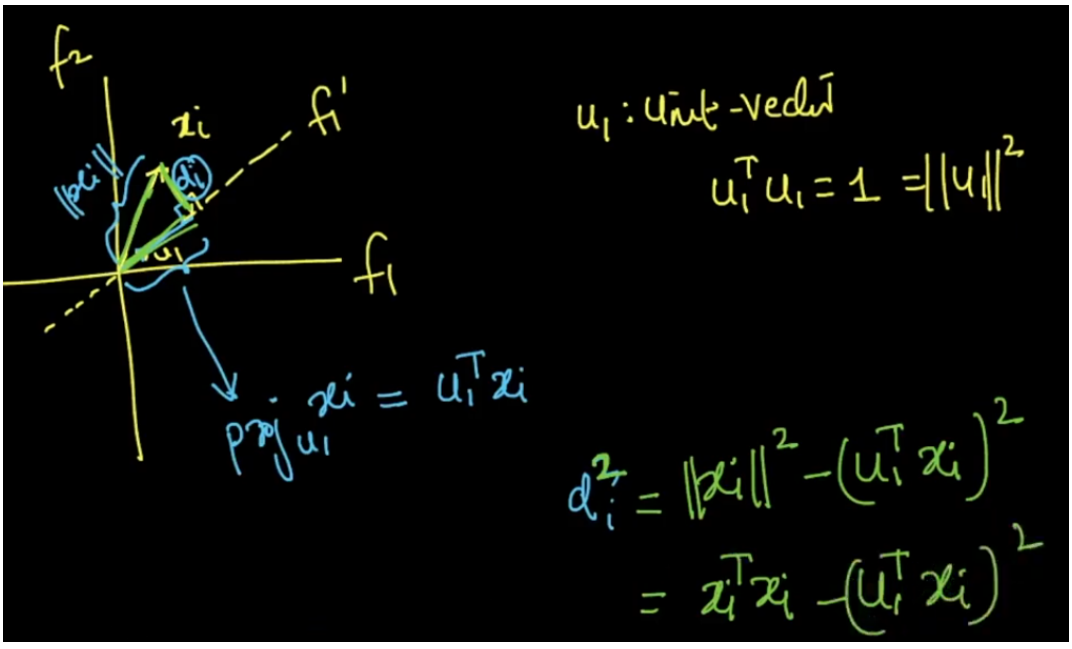
Alternative Formulation of PCA, distance minimization:

So in previously we saw the formula of maximizing the variance u1, so there is another alternative formula that is distance minimization. Means we take the distance of each points to this u1 line lets call this distance d1, d2, d3.....so for each point x_i we can find d_i which is distance from x_i to u1. So alternative formulation or optimization function we can write that we want to minimize d_i^2 we have to find u1 which minimizes the sumation of this distances

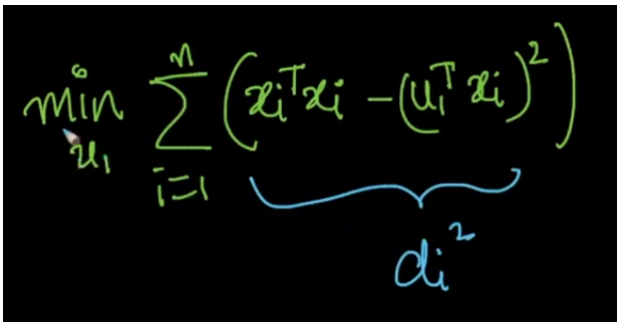


this is called distance minimization formula of PCA $\min_{u_1} \sum_{i=1}^{2} d_i^2$

so now we will see how to calculate distance square



so if we put d_i^2 in formula then our distance minimization formula will be



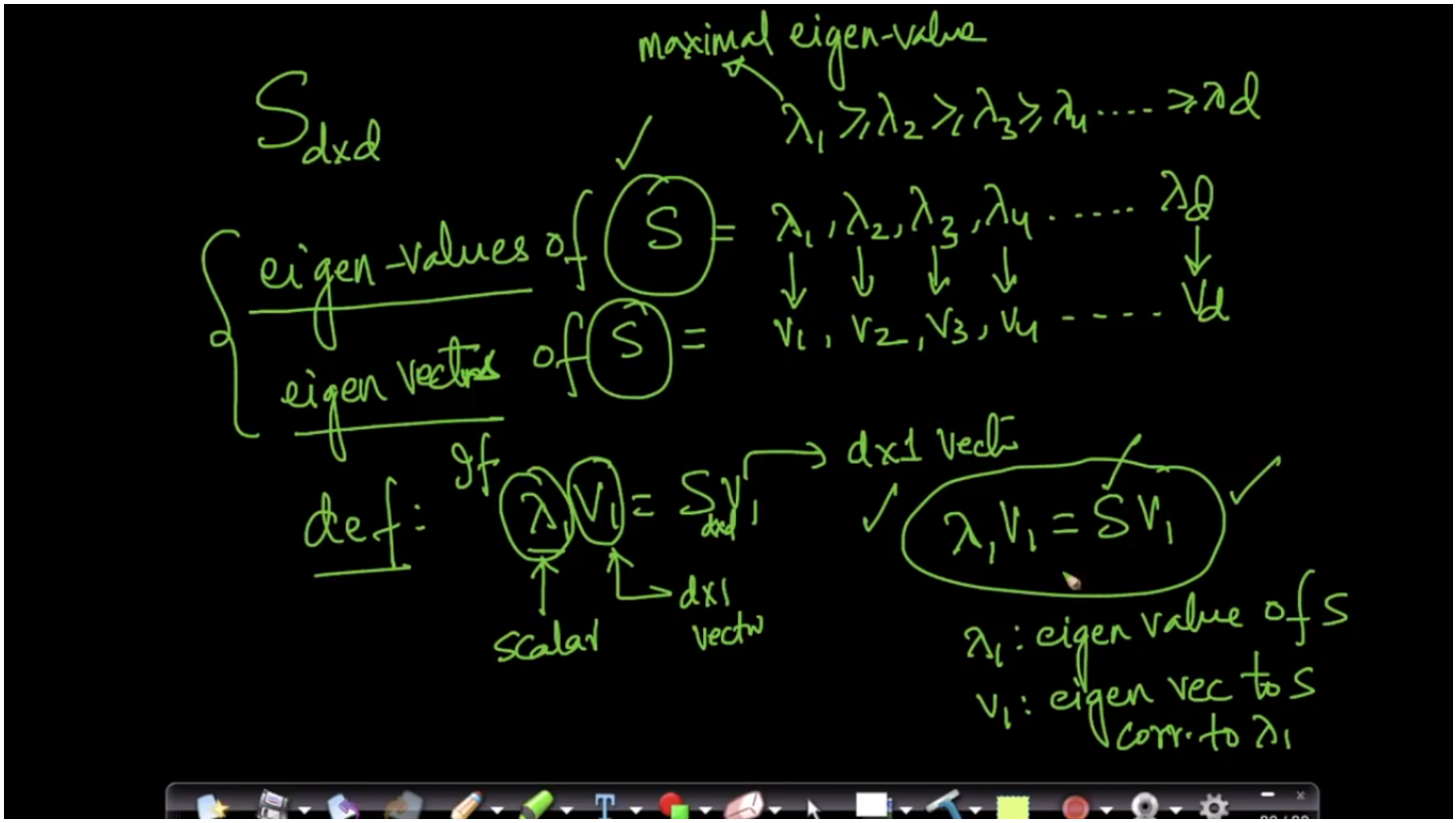
So we got two different optimization function for PCA one is maximization of varience and another is minimization of disatance, two formulas are different but we can achive same goal using one of them.

Steps Involved in PCA:

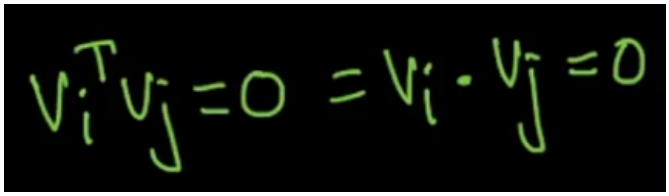
1. Standardize the data. (with mean =0 and variance = 1)
2. Compute the Covariance matrix(**S**) of dimensions.The off-diagonal elements in the covariance matrix will represent the covariance among each pair of variables and the diagonal elements will represent the variances of each variable/dimension.
3. Obtain the Eigenvectors and Eigenvalues from the covariance matrix.the eigenvalues of the covariance matrix which represents the variability in data on an orthogonal basis in the plot. You will also have to find eigenvectors of the covariance matrix which will represent the direction in which maximum variance among the data occurs.
4. Arrange the eigenvalues in an ascending/descending order and select the higher eigenvalues. You can choose how many eigenvalues you want to proceed with. You will lose some information while ignoring the smaller eigenvalues, but those minute values will not create enough impact on the final result.The selected higher eigenvalues will become the dimensions of your updated feature set.
5. Using the feature vector, we find the principal components of the dataset under analysis. We multiply the transpose of the feature vector with the transpose of the scaled matrix (a scaled version of data after normalisation) to obtain a matrix containing principal components.

Solution to our optimization Problem and Calculate eigen values and eigen vectors :

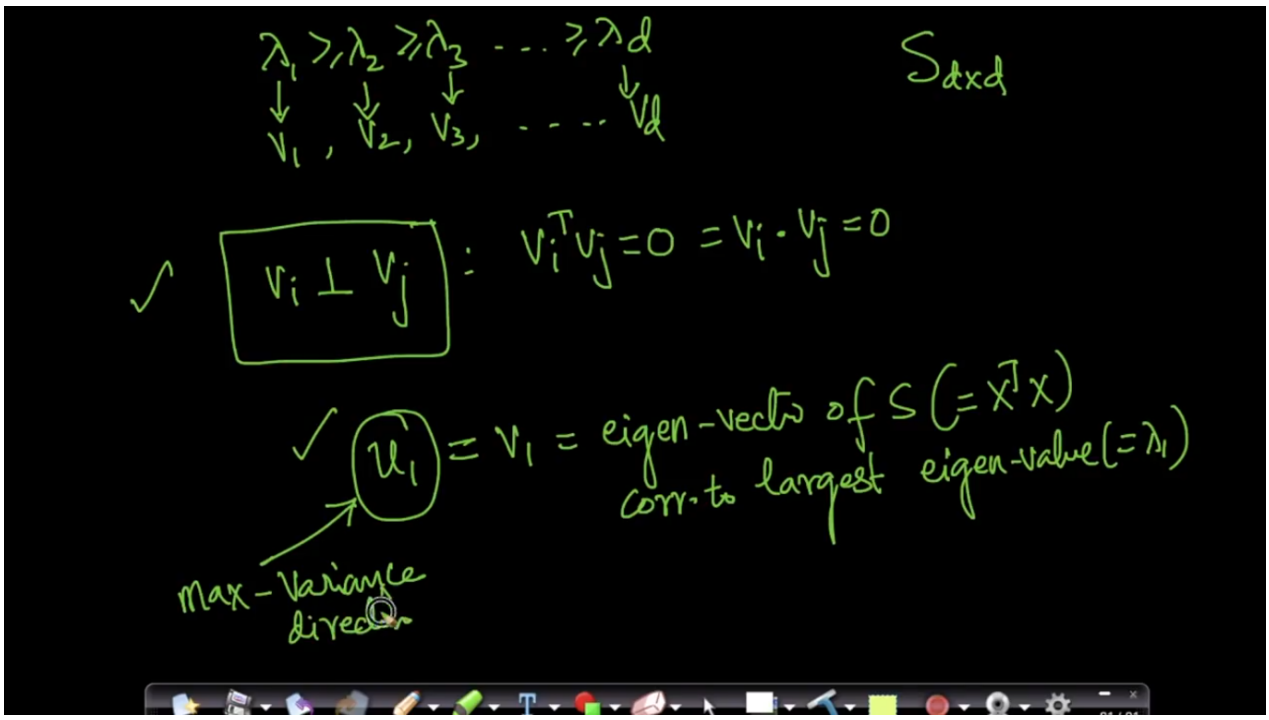
in d*d covariance matrix S there are eigen values like S =λ1,λ2,λ3.....λd and corresponding eigen vector is S =v1,v2,v3....vd and let also assume that λ1 >= λ2 >= λ3 >=.....>=λd so λ1 is maximal value, here λ is the scaller and v is vector matris if they satisfy this condition λ1v1=Sv1 then we can say λ1 is eigen values of co-variance matrix S and v1 is corresponding eigen vector of co-variance matrix S



also a good properties of eigen vector if λ1 >= λ2 >= λ3 >=.....>=λd and their coresponding vectors are v1,v2,v3....vd then we can say v1,v2,v3....Vi perpendicular to Vj means v1 perpendicular to v2 so on so fourth. And then it implies

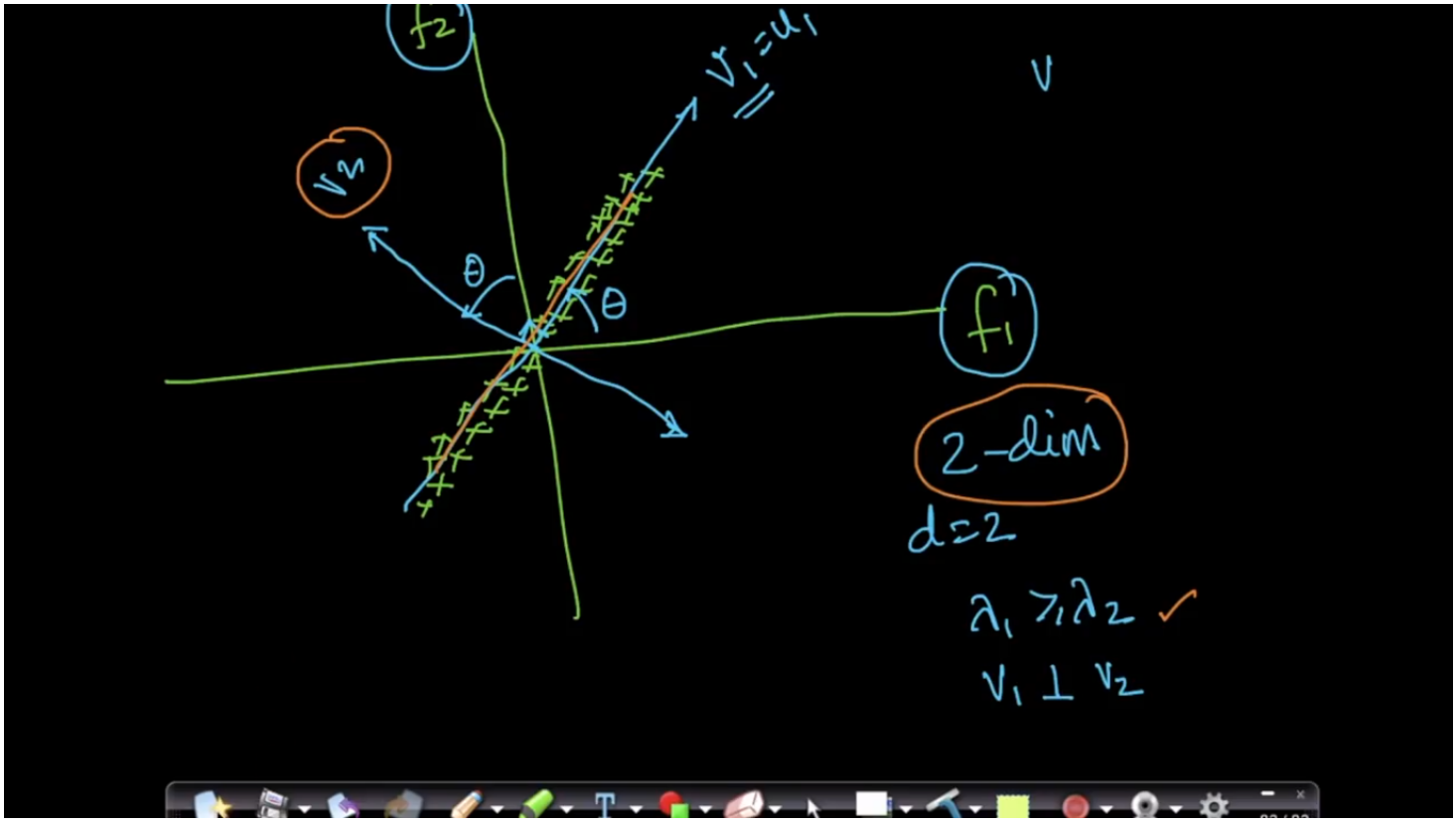


So basically u1 vector is nothibg but v1, u1 = v1 and this v1 is the eigen vector of S(co-variance matrix) corresponding to largest eigen vale which is λ1, so here u1 is the maximum variance of direction



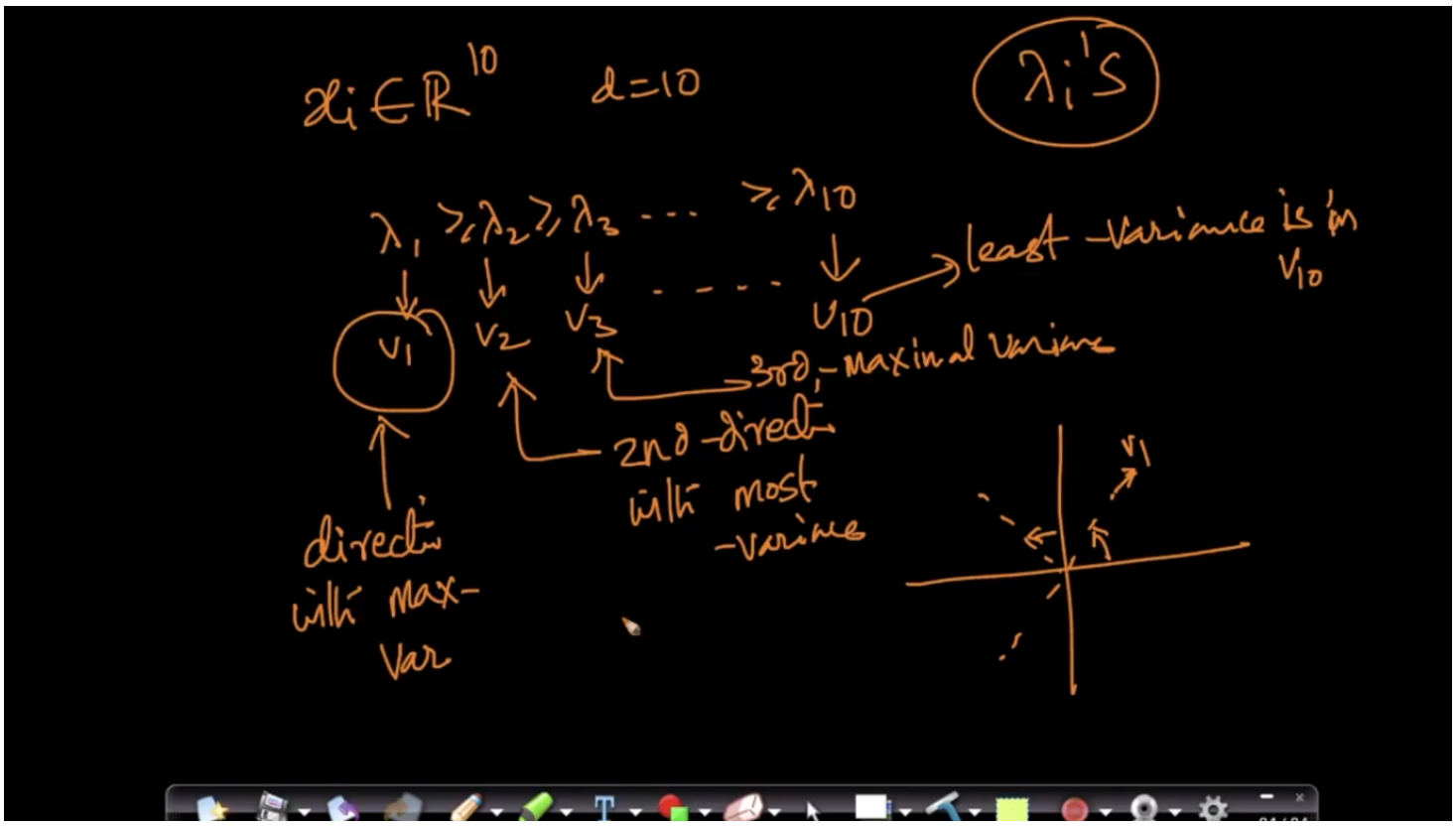
in python we can easily calculate eigen vales & eigen vectors.

Let say we have two dimensional data means two eigen vales & eigen vectors

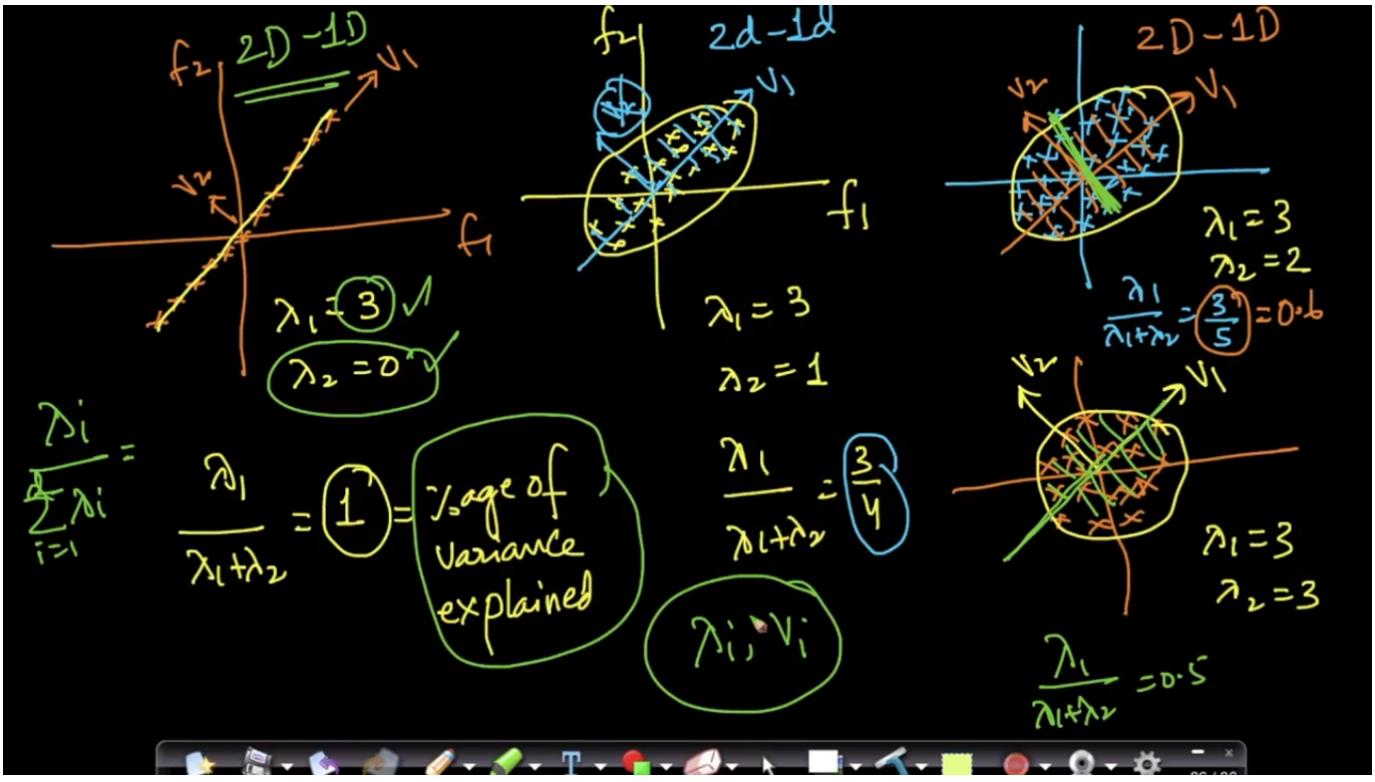


and v1 and v2 are perpendicular to each other .

Imagine we have a 10 dimensional data means d=10 so we have 10 eigen vales & coresponding 10 eigen vectors.like(v1,v2,v3,v4....v10) so v1 will be direction with maximum variance then v2 will be 2nd direction with most variance and so on so forth. And v10 will be least variance

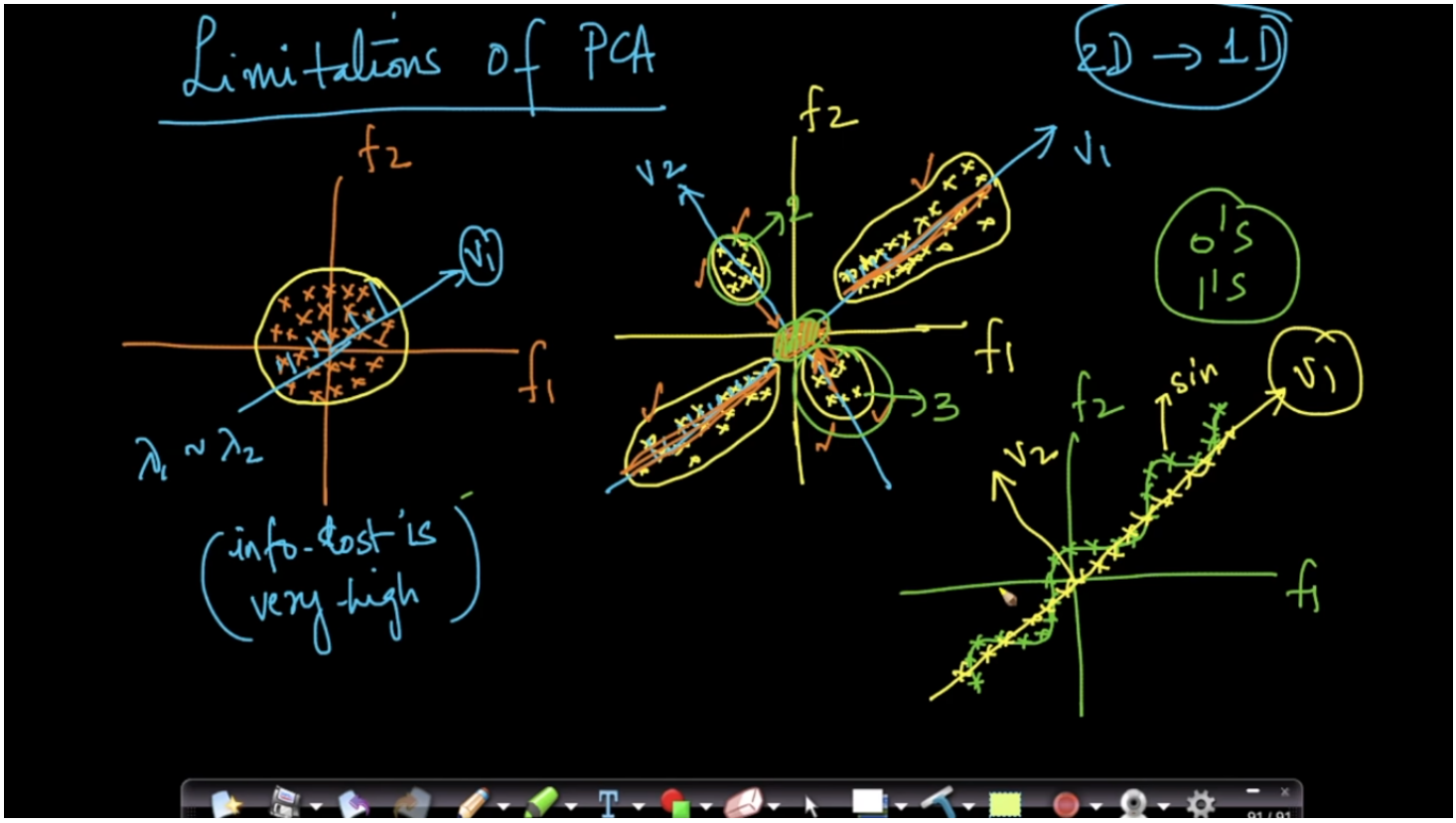


and if we plot this we will see how λ1 & λ2 will change for diifenet types of data



now if we want 2d to 1d conversation in first above plot we can see data almost lie on v1 in straghit line form and it covers 100% of variance its ratio says 1 means only 1 feature covers 100% variance, & next plot says ratio is 3/4 means if we skip f2 features then only 1 feature will covers 75% variance and so on so fourth.

Limitation of PCA:



i. from the avobe first plot we can see almost same data spread in two axis so if we convert this data 2d to 1d then no dobut we will definently lost almost 50% information.

ii. from the avobe 2nd plot we can see data is more spread on v1 axis and less spread in v2 axis so if we convert this data 2d to 1d then some percentage of data will be overlaping on one axis.

iii. from the avobe 3rd plot we can see data has nice sine structre if we convert this data 2d to 1d then then we will lose this nice data structre.

Applications of PCA:

• Neuroscience – Neuroscientists use PCA to identify any neuron or to map the brain structure during phase transitions.

• Finance – PCA is used in the finance sector for reducing the dimensionality of data to create fixed income portfolios. Many other facets of the finance sector involve PCA like forecasting returns, making asset allocation algorithms or equity algorithms, etc.

• Image Technology – PCA is also used for image compression or digital image processing. Each image can be represented via a matrix by plotting the intensity values of each pixel, and then we can apply PCA on it.

• Facial Recognition– PCA in facial recognition leads to the creation of eigenfaces which makes facial recognition more accurate.