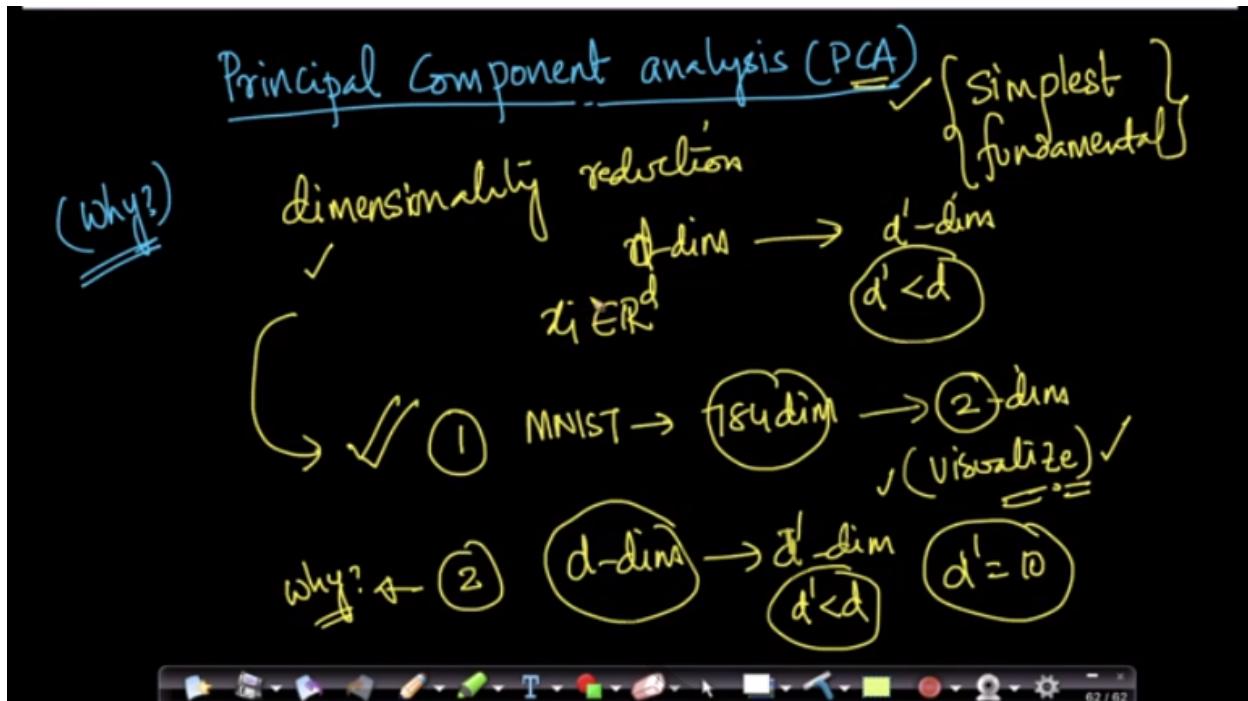


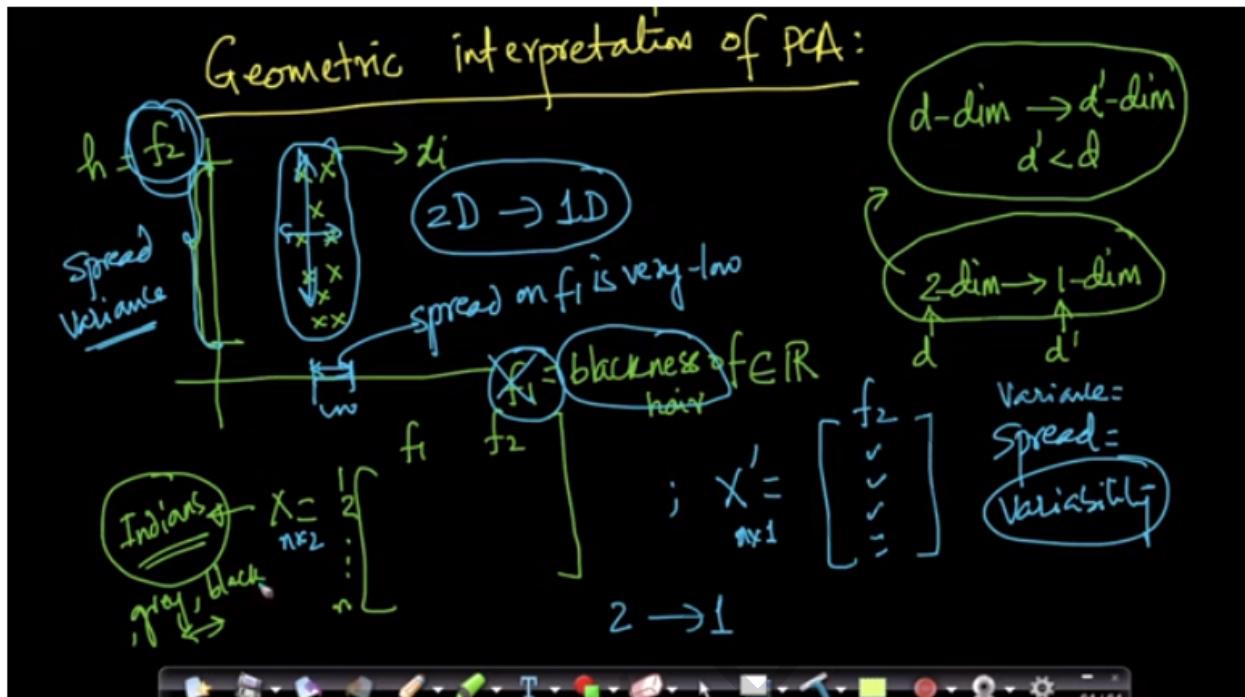
26.1 Why learn PCA?



Timestamp: 3:25

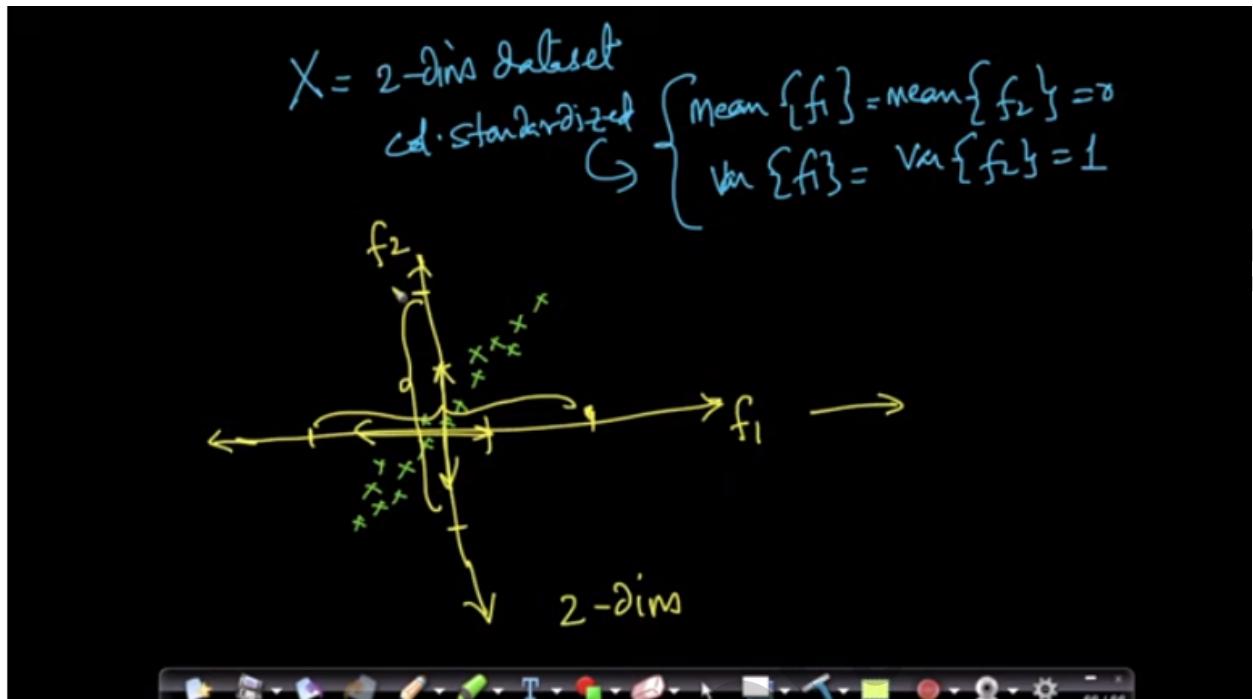
PCA is used for dimensionality reduction to convert data from d dimensions to d' dimensions where $d' < d$. It has applications to both visualize the data by converting it into 2-D or 3-D. Apart from visualization reducing the dimension of data using PCA has other applications which will be discussed in further chapters.

26.2 Geometric intuition of PCA



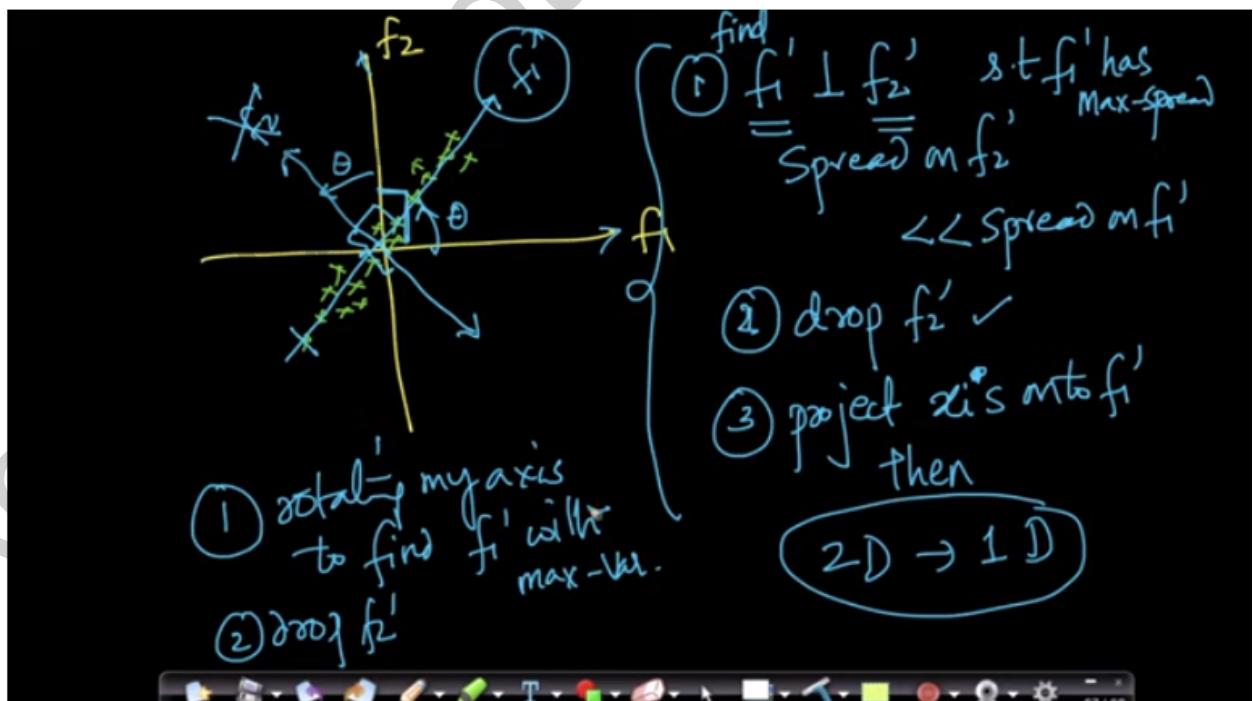
Timestamp: 5:24

As mentioned in the above figure, let's say that the feature f_2 has more spread than feature f_1 , if we have to reduce the dimensions from 2-D to 1-D then we can pick up the feature f_2 where the spread or variability of the data is much more. More the spread, the more we can visualize our data better. Hence we reduce our data which contains feature f_1 and f_2 to a data that contains only one feature f_2 since it has more spread and preserve the direction in which the data has more spread.



Timestamp: 9:16

Suppose we have a 2-D dimensional dataset which is column standardized and let us consider the data is spread more not along the direction f_1 or f_2 but rather as shown in the image. Then we cannot simply just drop f_1 or f_2 because the spread is not in either of f_1 or f_2 direction. For this we have to do something smart as below.



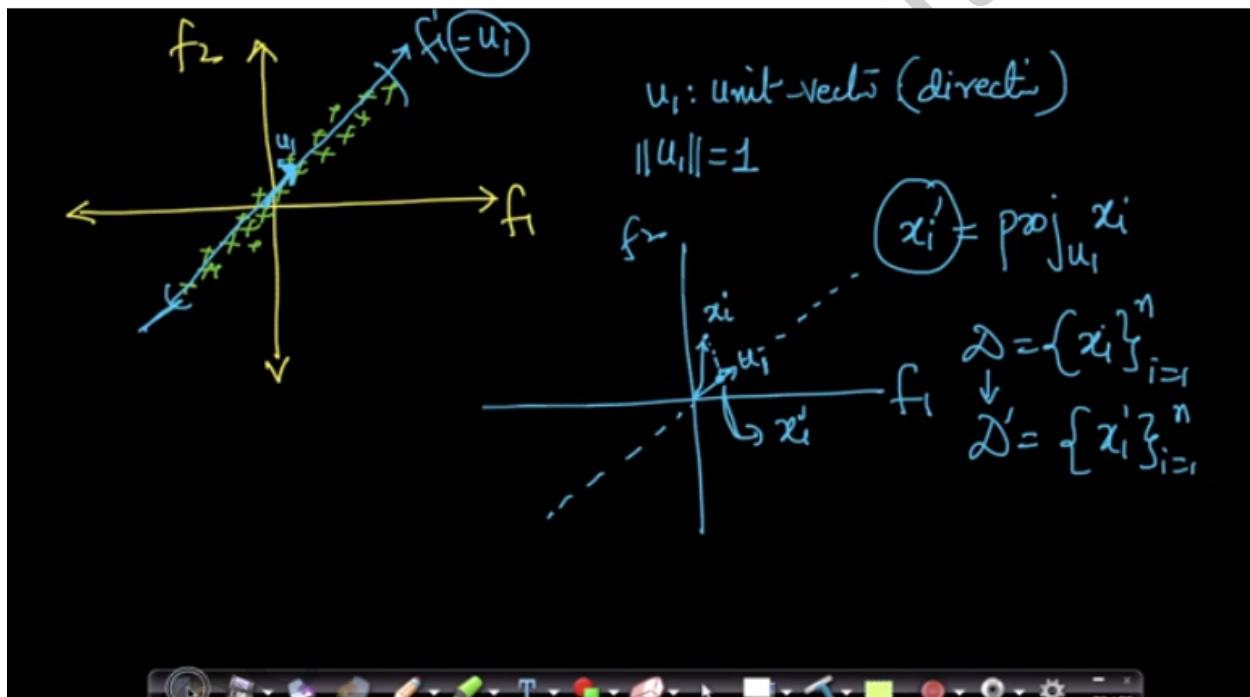
Timestamp: 13:43

As shown in the above figure, we will find f_1' and f_2' such that f_1' is perpendicular to f_2' such that the spread of f_1' is much greater than the spread of f_2' .

We will then drop f_2' and project our data to f_1' , thereby reducing data from 2-D to 1-D.

We can think of this as rotating the axis to find f_1' that has the max-variance and dropping f_2' .

26.3 Mathematical objective function of PCA



Timestamp: 2:34

As discussed before we want to find a direction where the spread of our data is more. Let us say u_1 is a unit vector in that direction. Then to get the new dataset we will project each of our points on u_1 to get the new dataset D' as shown in the image.

$$x'_i = \text{proj}_{u_1} x_i = \frac{u_1 \cdot x_i}{\|u_1\|^2=1} = u_1^T x_i$$

$$\bar{x}' = u_1^T \bar{x}$$

$$\text{mean}\{x'_i\}_{i=1}^n$$

$$\text{mean}\{x_i\}_{i=1}^n$$

Timestamp: 4:19

Since vector u_1 is a unit vector, the projection of point x_i on u_1 x'_i will be just $u_1^T x_i$, notice that mean of x'_i which is \bar{x}' , will be $u_1^T \bar{x}$. Meaning the mean of projections will be u_1^T multiplied by the mean of x_i .

④ find u_1 so that $\text{Var}\{\text{proj}_{u_1} x_i\}_{i=1}^n$ is maximal.

$$\text{Var}\{u_1^T x_i\}_{i=1}^n = \frac{1}{n} \sum_{i=1}^n (u_1^T x_i - \underbrace{u_1^T \bar{x}}_{\text{mean}\{x_i\}_{i=1}^n})^2$$

$$\text{scale} = (u_1^T)^T x_i \quad (\underbrace{1 \times n}_{\text{1x1}})$$

\bar{x} : Col. Standardized
 $\checkmark \bar{x} = [0, 0, 0, \dots, 0]$

Timestamp: 7:28

Our problem is to find u_1 such that the variance of projections along u_1 is maximal. So writing the variance of our projections, since we have column standardized the data, the mean of the original data points will be a zero vector. Hence our variance is reduced as below.

The image shows a handwritten derivation of the PCA optimization problem. At the top, the formula for variance is given: $\text{Var} \{x_i'\}_{i=1}^n = \frac{1}{n} \sum_{i=1}^n (u_1^T x_i)^2$. A green bracket groups the term $(u_1^T x_i)^2$ and is labeled "objective of an optim problem". Below this, the optimization problem is written as: $\max_{u_1} \frac{1}{n} \sum_{i=1}^n (u_1^T x_i)^2$, with a green bracket under the sum labeled "Data-matrix". To the right, a green circle contains the text "optim problem". Below the main equation, the constraint is given as: $\text{s.t. } u_1^T u_1 = 1 = \|u_1\|^2$. A green bracket groups $u_1^T u_1 = 1$ and $\|u_1\|^2$, with the label "constraint" above it. Below this, a green bracket groups $u_1^T u_1 = 1$ and $u_1 = [\infty, \infty]$, with the label "u_1 is a unit vec" above it. The bottom of the image shows a computer interface with a toolbar and a status bar indicating "76% 76".

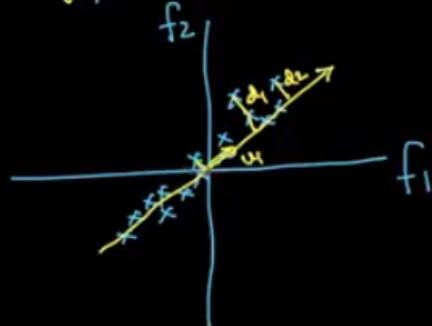
Timestamp: 11:23

So the variance of the projections is reduced to the sum mentioned in the above figure. So now we have our optimization problem for PCA is to find u_1 that maximizes the sum highlighted such that $u_1^T u_1 = 1$. This constraint of making u_1 a unit vector is important since otherwise any vector can satisfy the above equation by making its magnitude infinity. We will see how to solve these optimization problems in later chapters.

26.4 Alternative formulation of PCA: Distance minimization

Alternative formulation of PCA: dist. minimization

→ find u_1 which maximizes projected variance

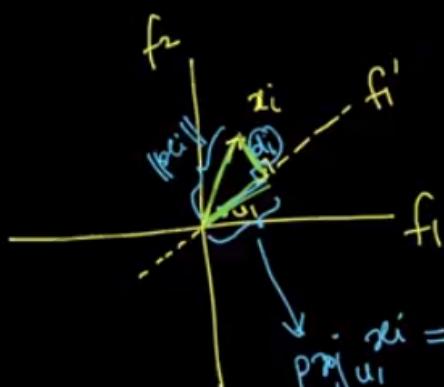


$x_i \rightarrow (\hat{d}_i)$: dist from x_i to u_1

$$\min_{u_1} \sum_{i=1}^n \hat{d}_i^2$$

Timestamp: 2:47

Distance minimization is an alternative formulation of PCA, in which instead of choosing u_1 that maximizes the variance of our projections, we want to choose u_1 that minimizes the distance from x_i to u_1 . Both are equivalent but just different formulations.



u_1 : Unit-Vektor

$$u_1^T u_1 = 1 = \|u_1\|^2$$

$$\begin{aligned} d_i^2 &= \|x_i\|^2 - (u_1^T x_i)^2 \\ &= x_i^T x_i - (u_1^T x_i)^2 \end{aligned}$$

Timestamp: 5:36

We find the distance d_i of our point x_i from the direction u_1 , using pythagoras theorem as above. So our final objective function would look like below.

The image shows a handwritten derivation on a blackboard. At the top left, a circle contains "dist min PCA". An arrow points from this to another circle containing "Min u_1 " with "i=1" and "n" above it. This leads to a diagram of a point x_i in a 2D plane, with its projection onto a line defined by $u_1^T x_i$. The distance d_i is shown as the hypotenuse of a right triangle, with the Pythagorean theorem formula $d_i^2 = x_i^T x_i - (u_1^T x_i)^2$ written next. Below this, a constraint $u_1^T u_1 = 1$ is given. To the right, a vector x_i is shown as a horizontal arrow with " x_i^T " at its tip. A bracket indicates that the entire row of terms is zero. At the bottom left, a circle contains "Max u_1 " with "s.t." and "1/n" above it, followed by a similar Pythagorean theorem formula. Another constraint $u_1^T u_1 = 1$ is shown. To the right, a circle contains "Variance maximization PCA". A blue line connects the two main equations, labeled with a checkmark and an equals sign. The bottom of the image shows a Mac OS X dock with various icons.

Timestamp: 9:42

So our objective function according to distance formulation looks like above, it can be proved that it will be equal to the variance maximization formulation of PCA. (the proof will be understood when optimization is taught later in the course).

26.5 Eigenvalues and Eigenvectors(PCA): Dimensionality reduction

Solution to our Optimization problems: λ_1, v_1

$X = \frac{1}{\sqrt{n}} \begin{bmatrix} 1 & 2 & 3 & \dots & d \\ \vdots & & & & \vdots \\ n & & & & \end{bmatrix}_{n \times d}$

Covariance matrix of $X = S$

$$S_{d \times d} = X^T X_{d \times n \quad n \times d}$$

Sq. Symm-matrix

eigen-values $(\lambda_1, \lambda_2, \dots, \lambda_d)$
eigen-vector (v_1, v_2, \dots, v_d)

Timestamp: 3:10

Solution to our optimization problem will be to find the eigenvalues and eigenvectors of the covariance matrix S of our data matrix X . (Correction in image $S=(X^T X)/n$) [$S=(X^T X)/(n-1)$ for unbiased estimate. What is meant by biased estimate and unbiased estimate will be covered in the upcoming live session]

$S_{d \times d}$

maximal eigen-value
 $\lambda_1 > \lambda_2 > \lambda_3 > \lambda_4 \dots \geq \lambda_d$

eigen-values of S = $\lambda_1, \lambda_2, \lambda_3, \lambda_4, \dots, \lambda_d$
eigen-vectors of S = $v_1, v_2, v_3, v_4, \dots, v_d$

def: If $\lambda_1 v_1 = S v_1$ \rightarrow $d \times 1$ vector

λ_1 : scalar v_1 : $d \times 1$ vector

$\lambda_1 v_1 = S v_1$

λ_1 : eigen value of S
 v_1 : eigen vector of S

Timestamp: 6:33

Finding eigenvalues and eigen vectors of any matrix S is pretty straightforward and we might have learnt earlier. We have to find eigen vectors and eigen values such that they satisfy the condition highlighted.

Handwritten notes on eigenvalues and eigenvectors of matrix $S_{d \times d}$. The notes show the relationship between eigenvalues $\lambda_1 > \lambda_2 > \lambda_3 > \dots > \lambda_d$ and eigenvectors $v_1, v_2, v_3, \dots, v_d$. It highlights the property that eigenvectors are perpendicular ($v_i \perp v_j$) with $v_i^T v_j = 0 = v_i \cdot v_j = 0$. A specific eigenvector $u_1 = v_1$ is identified as the eigenvector of $S (= X^T X)$ corresponding to the largest eigenvalue ($= \lambda_1$), labeled as the "max-variance direction".

Timestamp: 9:10

Eigenvectors have a property that every eigen vector is perpendicular to every other eigenvector. If we can find the eigenvectors and eigenvalues of the covariance matrix S , then we can easily get the direction u_1 which maximizes the variance as it is equal to the eigenvector of S that corresponds to the largest eigenvalue.

$$X = \begin{bmatrix} & \checkmark \\ & \checkmark \end{bmatrix}_{n \times d}$$

1 Col. Std of X is done

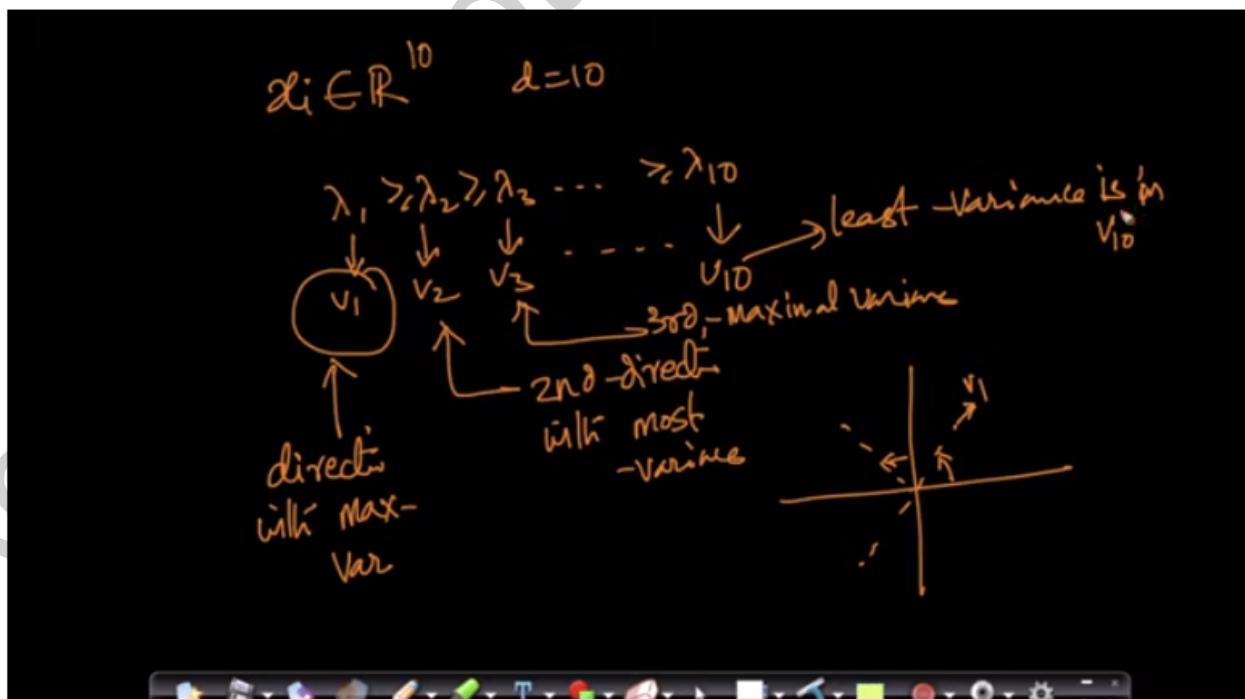
2 $S_{\text{cov}} = X^T X$

3 Eigen values & vectors of S_{cov}
 eigen(S)
 $\lambda_1 > \lambda_2 > \dots > \lambda_d$
 v_1, v_2, \dots, v_d

4 $u_1 = \underbrace{(v_1)}_{\text{why?}}$

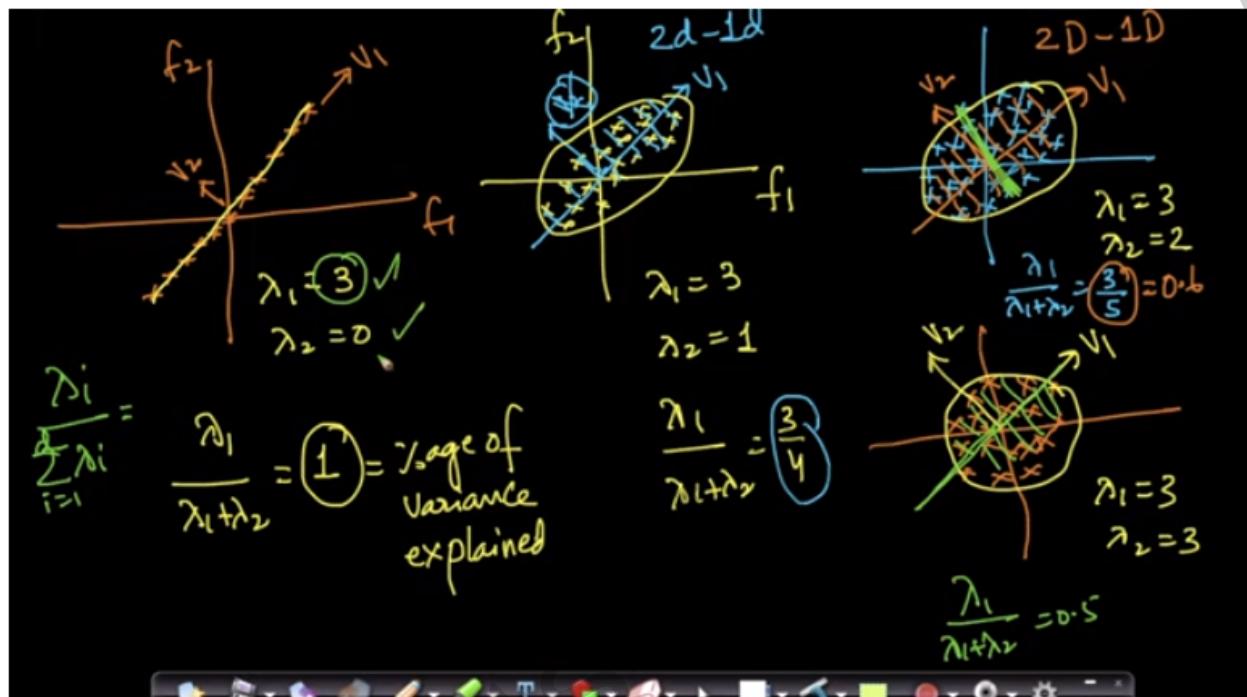
Timestamp: 11:12

So given a matrix X , in order to find u_1 , the max-variance direction, we need to ensure that X is column standardized, post this we can calculate the covariance matrix S . Following that you can just find the eigenvalues and eigenvectors of S and u_1 will just be the eigenvector with max eigenvalue.



Timestamp: 15:33

So if we sort of our eigenvectors of the covariance matrix S in the decreasing order of their corresponding eigenvalues, then v_1 represents the direction with max-variance , v_2 represents the direction with second max-variance and so on.



Timestamp: 22:36

While eigenvectors v_1,v_2,etc tell us the directions of most-variance, second most variance etc. $\lambda_1,\lambda_2,\text{etc}$ tell us the variance explained in that direction. As shown in the above figure $\lambda_1/(\lambda_1+\lambda_2)$ tells us the percentage of variance explained by the direction of v_1 .

26.6 PCA for Dimensionality Reduction and Visualization

$X = \begin{bmatrix} f_1 & f_2 & \dots & f_{10} \\ \vdots & \vdots & \ddots & \vdots \\ x_i^T & \rightarrow & \end{bmatrix}_{n \times 10}$

dim
 reduce
 (PCA)

$\xrightarrow{* \text{ visualize}} V_1 \quad V_2$

$X' = \begin{bmatrix} 1 & \vdots & n \\ x_i^T & \rightarrow & \end{bmatrix}_{n \times 2}$

$S = X^T X$
 $\text{eigen}(S) = \lambda_1 > \lambda_2 > \lambda_3 \dots > \lambda_{10}$

$x'_i = [x_i^T v_1, x_i^T v_2]$

Timestamp: 5:32

Suppose we have data X of 784 dimensions, for us to visualize this data, we can take the covariance matrix S and find the eigenvalues and eigenvectors of matrix S and find the eigenvectors v_1, v_2 that have top two eigenvalues and transform this data to X' by taking each x_i and transforming into x'_i by doing $x_i^T v_1$ and $x_i^T v_2$. Since we can visualize 2-D, we can visualize X' .

$$X = \begin{bmatrix} f_1 & f_2 & \dots & f_m \\ \vdots & \ddots & & \vdots \\ x_1^T & x_2^T & \dots & x_n^T \end{bmatrix}_{n \times m}$$

PCA

$$X' = \begin{bmatrix} v_1 & v_2 & \dots & v_{50} \end{bmatrix}_{n \times 50}$$

$$x_{ij}' = x_i^T v_j$$

$n \times 100$

$n \times 50$

$$S = X^T X$$

100×100

$$\lambda_1 > \lambda_2 > \dots > \lambda_{100}$$

$$v_1, v_2, \dots, v_{100}$$

Timestamp: 7:42

If we want to use PCA for just dimensionality reduction, to reduce our data from let's say 100 dimensions to 50 dimensions, then we can simply find the top 50 eigenvectors and find x_{ij}' by just doing $x_i^T v_j$.

$$x_i \in \mathbb{R}^{100 \times 1} \quad ; \quad x_i' \in \mathbb{R}^{d'}$$

$d \xrightarrow{\text{PCA}} d'$

100×1

\checkmark preserve variance

let $\frac{\text{variance of } x_i}{\sum_{i=1}^{100} \lambda_i} = 0.99$

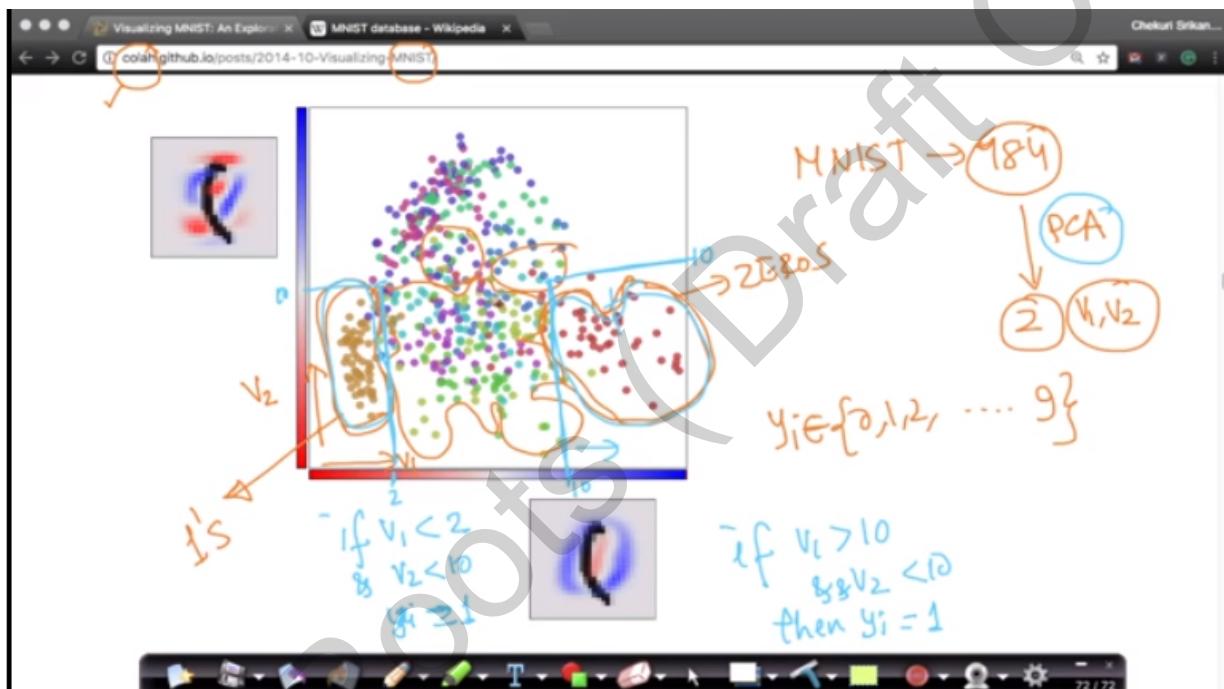
$d' < 100$

$d' = 51$

Timestamp: 9:42

Often it is the case that we don't pre-decide the dimensions to reduce the data to, often we need to preserve 99% of variance, etc. Then we just simply find the number of eigen vectors that needs to be taken to preserve at least 99% of variance and if 51 eigenvectors account for 99% of total variance then we just project our data onto this top 51 eigenvectors to get our dimension reduced data that has 99% of its variance preserved.

26.7 Visualize MNIST data

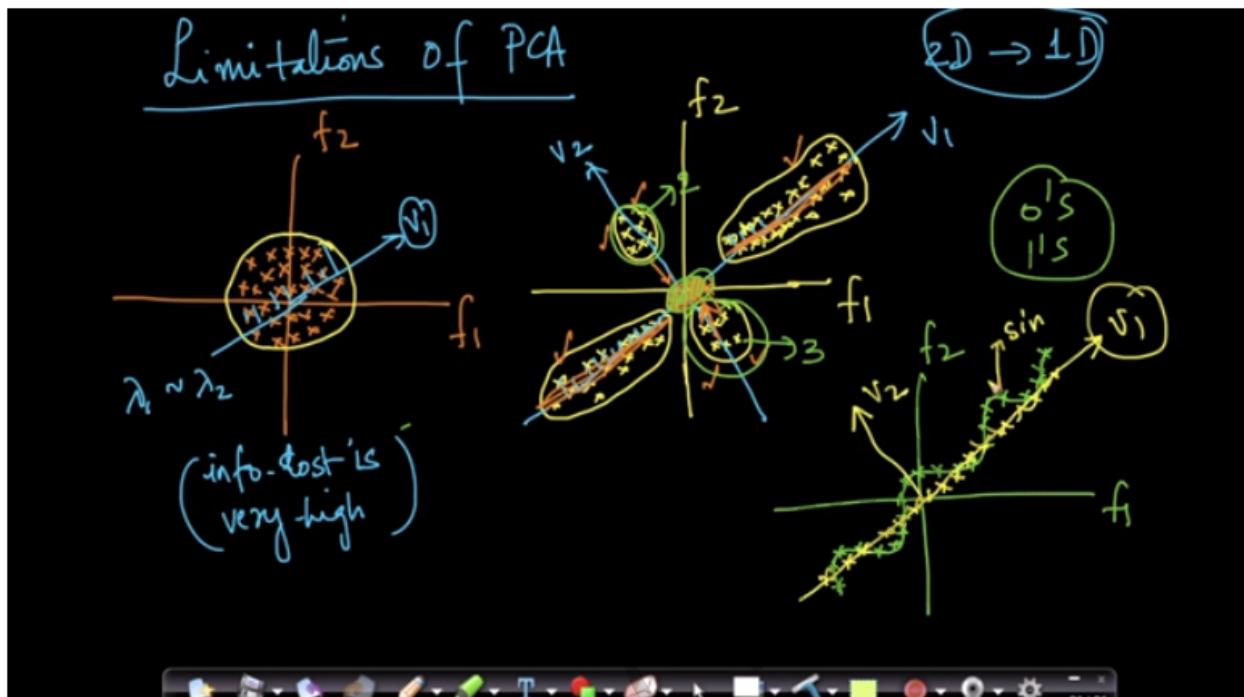


Minor Correction: @ 3:34min , it was mentioned as if $v_1 > 10$ and $v_2 < 10$, then $y_i = 1$. But it is actually:
if $v_1 > 10$ and $v_2 < 10$, then $y_i = 0$

Timestamp: 4:44

When PCA is applied on MNIST data then we get the above visualization. From the above visualization we can see that for some of the classes like zeros and ones, we can easily separate them from the rest of the classes using simple rules such as $v_1 > 10$ and $v_2 < 10$ then $y=0$, etc.

26.8 Limitations of PCA

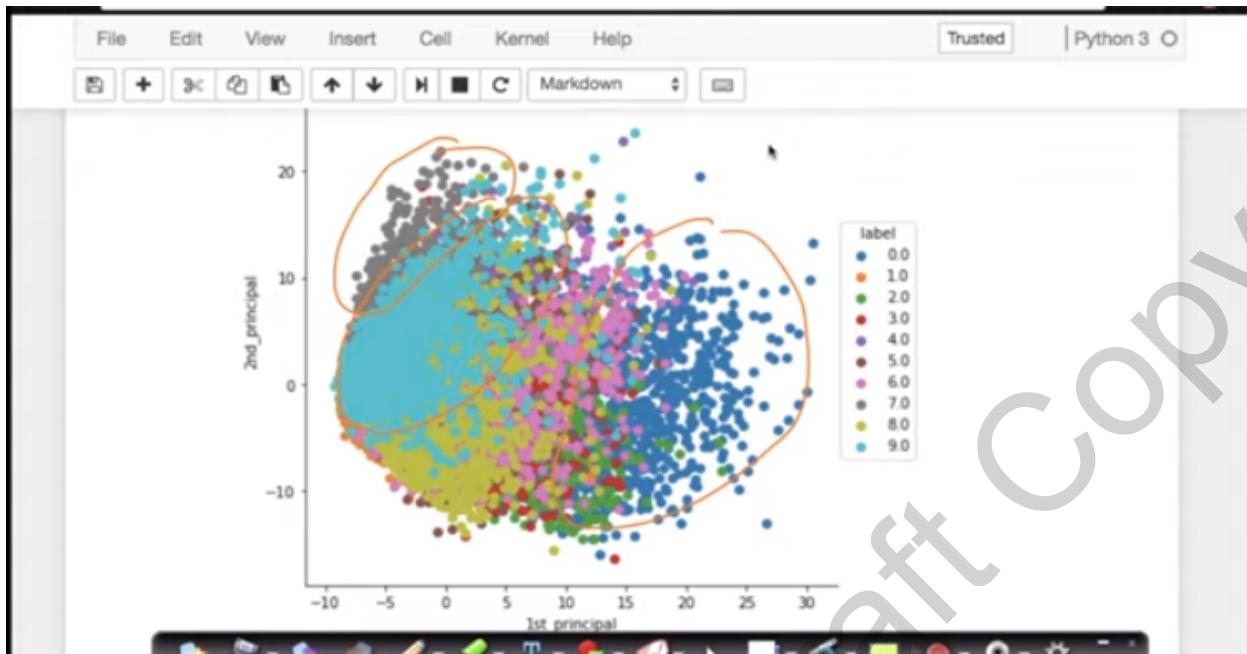


Timestamp: 4:47

There are certain limitations of PCA, as shown in the above images

- 1) Suppose if the original data is like a circle, then the variance is spread among all the directions, but if we reduce the data to lower dimensions, we lose most of the useful information.
- 2) As shown in the second image, when initially the clusters of points are well separated in the original space but when the dimensionality of the data is reduced then points which are well separated might be projected together and hence difficult to separate.
- 3) If the original data has nice shapes like sin wave, etc when dimension is reduced we lose this information.

26.9 PCA code example



Timestamp: 18:15

In this video, we have implemented the code with both our custom code and with PCA provided by scikit-learn. Please refer to the notebook link provided in the video description for the complete code and explanation via comments. From the above plot we can see that some of the classes are well separated and others are not well separated when we visualize the data generated by PCA.

A screenshot of a Jupyter Notebook titled "PCA code Example using visualization: Dimensionality reduction ...". The notebook shows two code cells:

```

In [25]: # initializing the pca
from sklearn import decomposition
pca = decomposition.PCA()

In [26]: # configuring the parameters
# the number of components = 2
pca.n_components = 2
pca_data = pca.fit_transform(sample_data)

# pca_reduced will contain the 2-d projects of simple data
print("shape of pca_reduced.shape = ", pca_data.shape)

```

The output of the second cell is:

```

shape of pca_reduced.shape = (15000, 2)

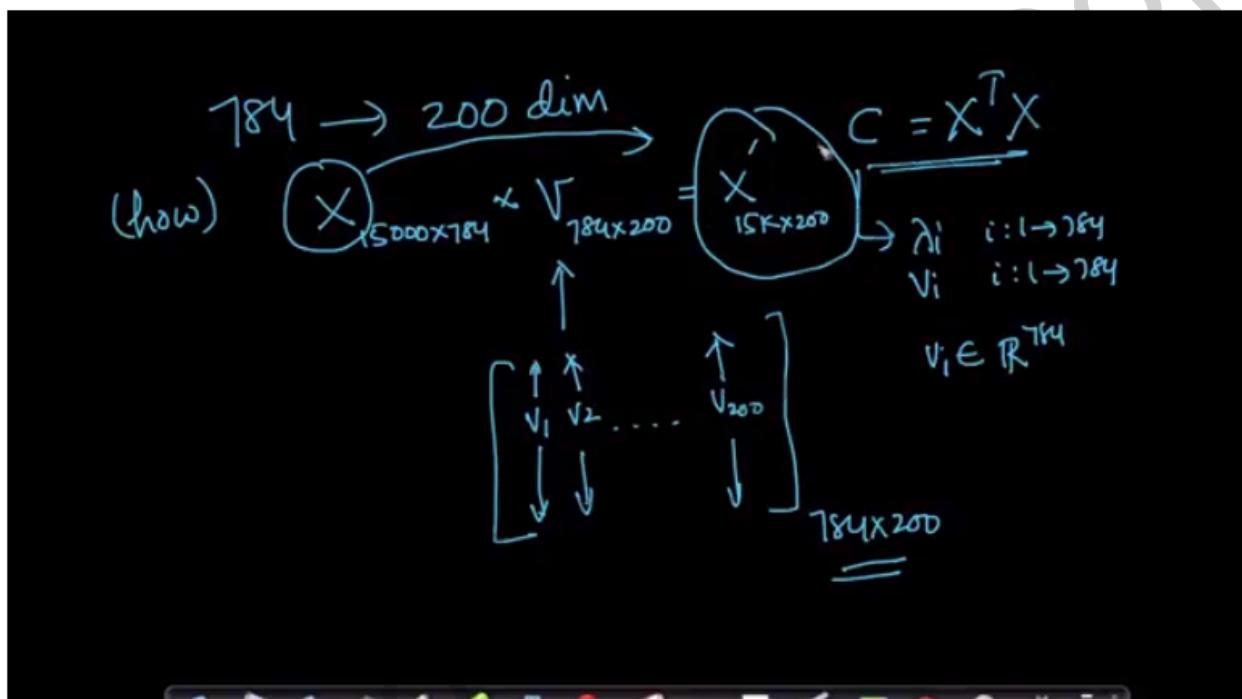
```

The video player interface at the bottom shows a play button, volume control, timestamp (18:46 / 18:59), and other standard video controls.

Timestamp: 18:46

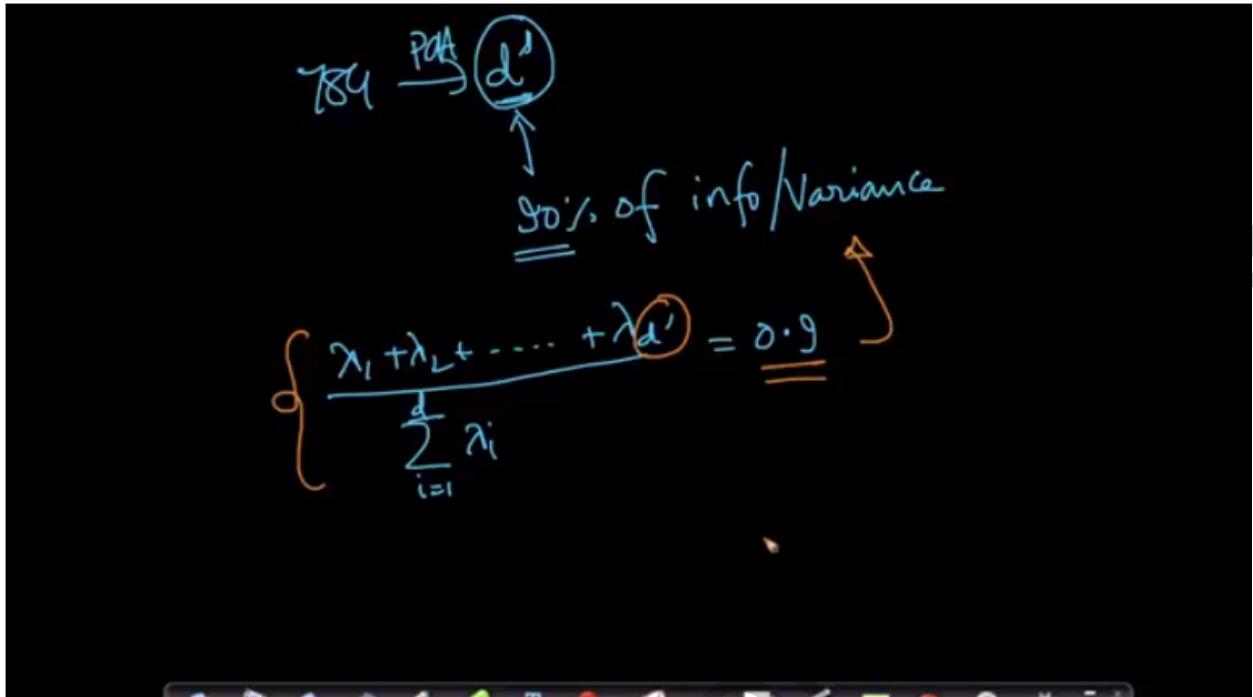
The above is the simple code of scikit learn PCA, using which we can convert the sample data of MNIST of 784 dimensions to 2 dimensions.

26.10 PCA for dimensionality reduction (not-visualization)



Timestamp: 4:21

Using PCA, to reduce the dimensions from 784 dimensions to 200 dimensions, we will just multiply the data X with the matrix of eigenvectors V to get the data of reduced dimension X' . Notice that our matrix of eigenvectors V is formed by placing the top 200 eigenvectors of the correlation matrix of X as a column vector.



Timestamp: 9:46

As discussed before, people often doesn't predecide d' but rather decide on the amount of information/variance of the data they want to preserve and find such d' that preserves 90% of variance of our data and reduce our data from d to d' dimensions.

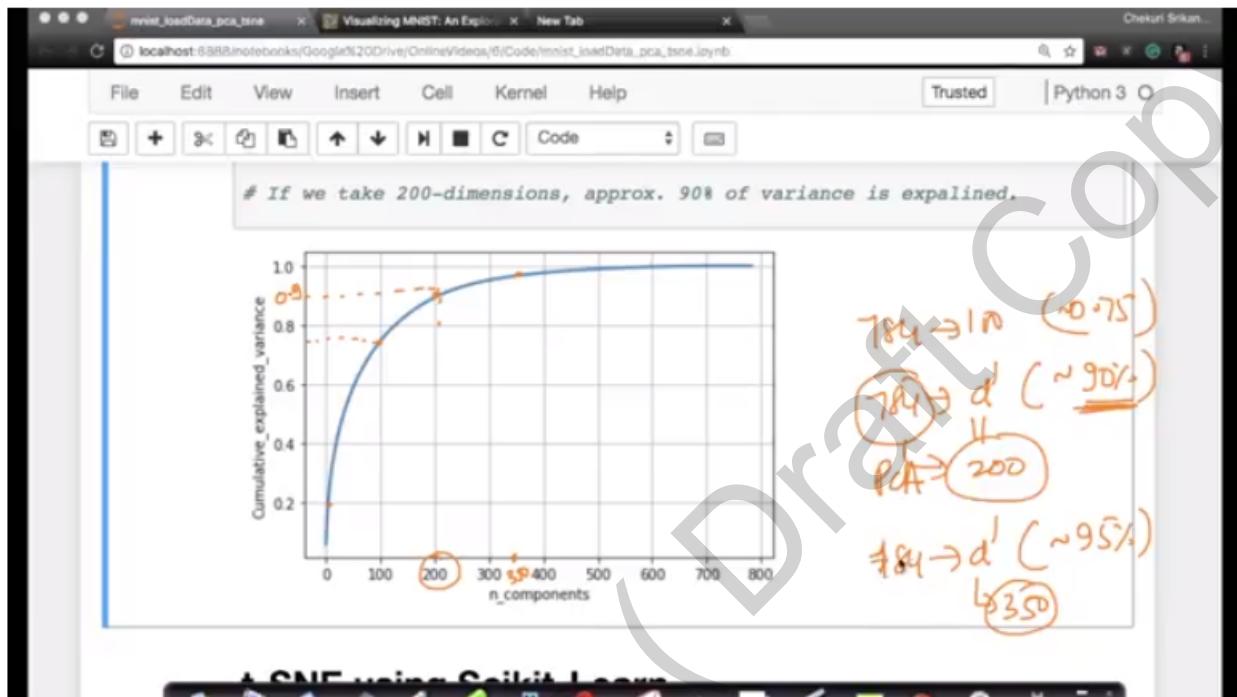
```
In [28]: # PCA for dimensionality reduction (non-visualization)
pca.n_components = 784
pca_data = pca.fit_transform(sample_data)
percentage_var_explained = pca.explained_variance_ / np.sum(pca.explained_v
cum_var_explained = np.cumsum(percentage_var_explained)

# Plot the PCA spectrum
plt.figure(1, figsize=(6, 4))

plt.clf()
plt.plot(cum_var_explained, linewidth=2)
plt.axis('tight')
plt.grid()
plt.xlabel('n_components')
plt.ylabel('Cumulative_explained_variance')
plt.show()
```

Timestamp: 12:31

The above is the code to plot the variance explained when we take different d' , the dimensions to reduce the data into.



Timestamp: 14:36

From the above plot, we can see that about 75% of variance can be explained if we reduce the data to 100 dimensions, similarly around 90% of variance can be explained when we choose d' to be 200 dimensions. Hence if we decide that it is ok to preserve 90% variance of our original data, we can then choose d' to be 200 and can reduce the data to 200 dimensions using PCA.