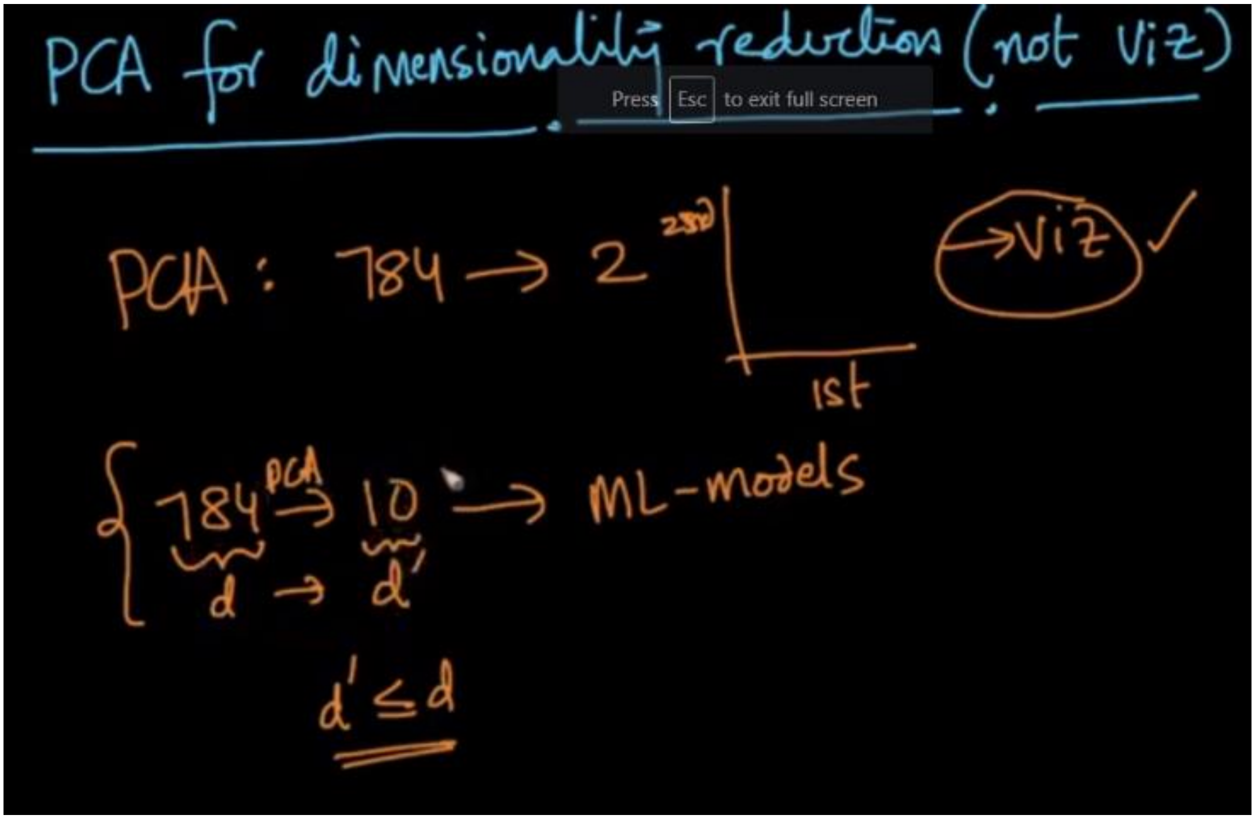
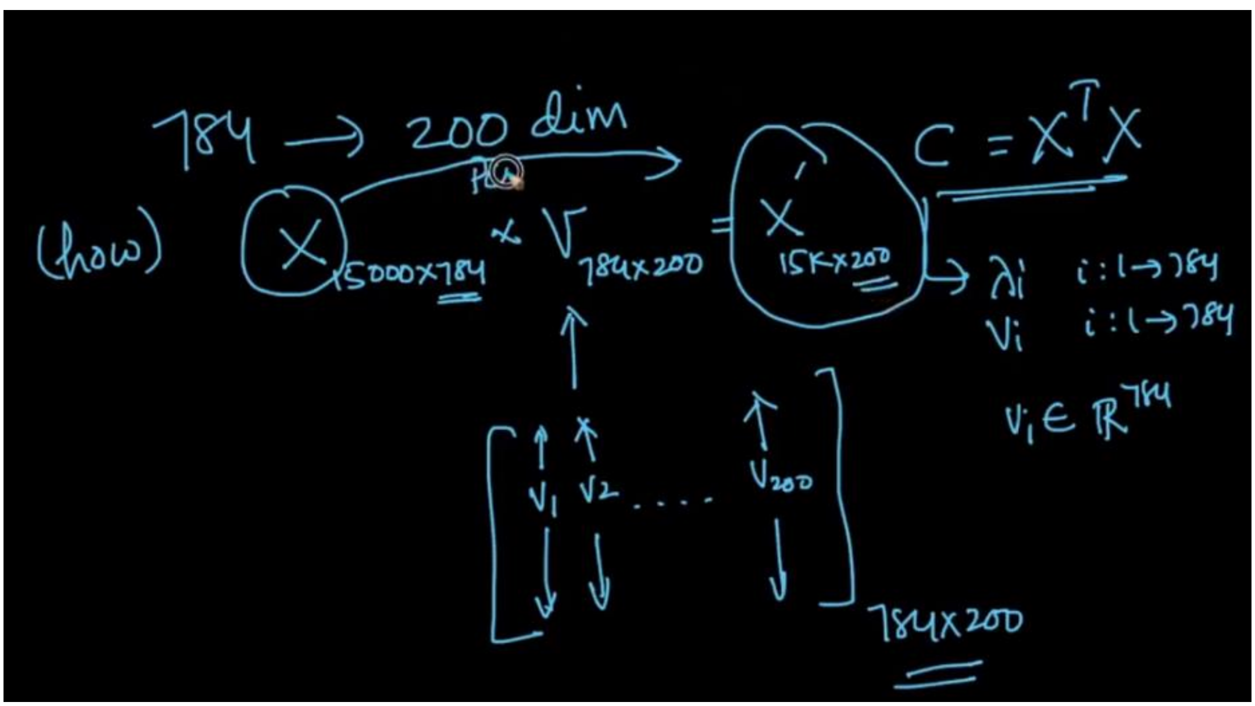
This can be says as a summary of PCA, how we do it for dimensionality reduction.

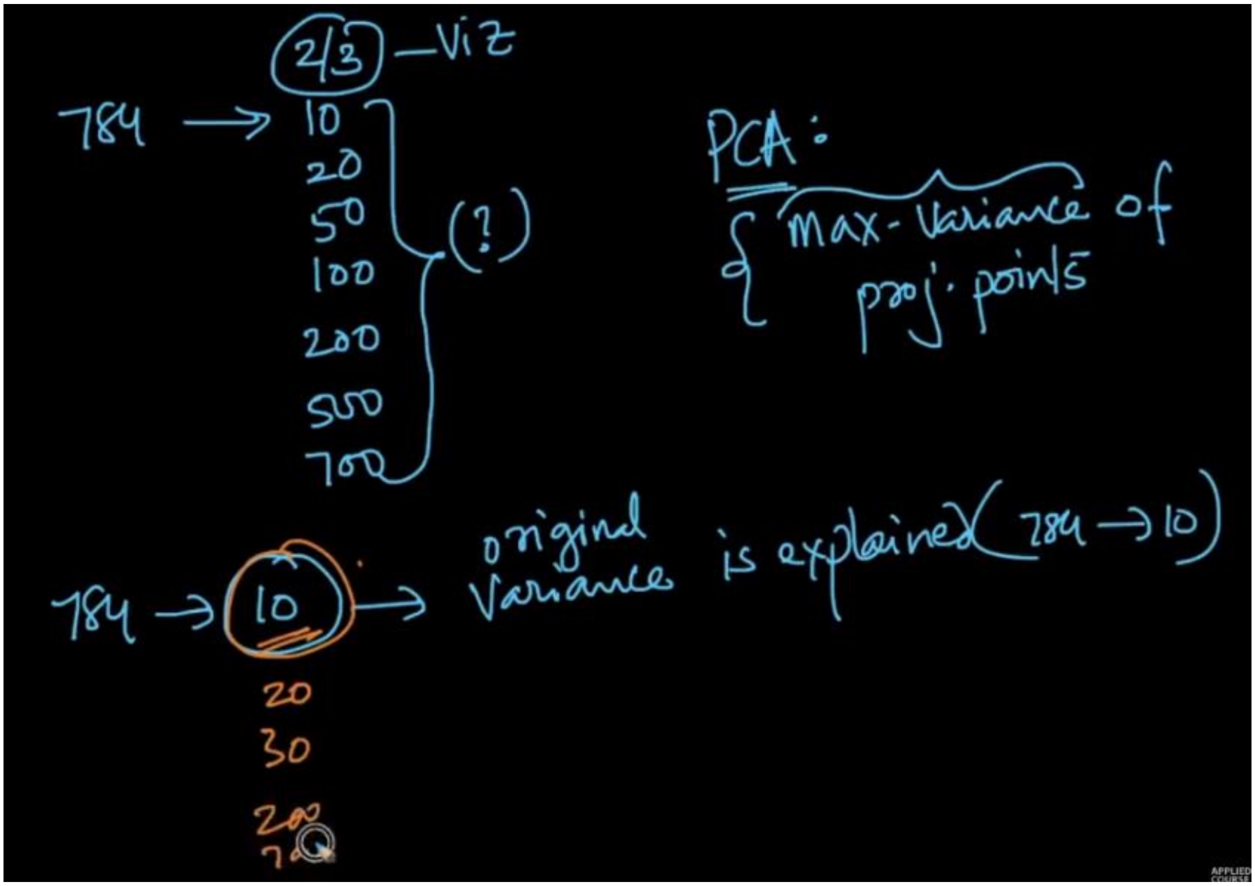
Our ultimate aim is to obtain features **d’** such that **d’ <= d**, where d is original no. of features.



Let’s say for mnist where we are obtaining 200 features from 784 features, here we have X of dimension 15000 * 784 and the eigen vectors matrix of dimension 784 * 200, which results the final matrix X’ of dimension 15k * 200



So our ultimate aim is to find the no of features for which the information retain percentage Is what we want which may b 75%, 99% and how it’s calculated is given in below image.



PCA:

$$C_{784 \times 784} = X^T X$$

$$\downarrow$$

$$\lambda_i, V_i$$

$$\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_{784}$$

784 → 10 dim

$$\text{percentage of Variance explained in 10-dim} = \frac{\overbrace{\lambda_1 + \lambda_2 + \dots + \lambda_{10}}}{\sum_{i=1}^{784} \lambda_i} = 0.2$$

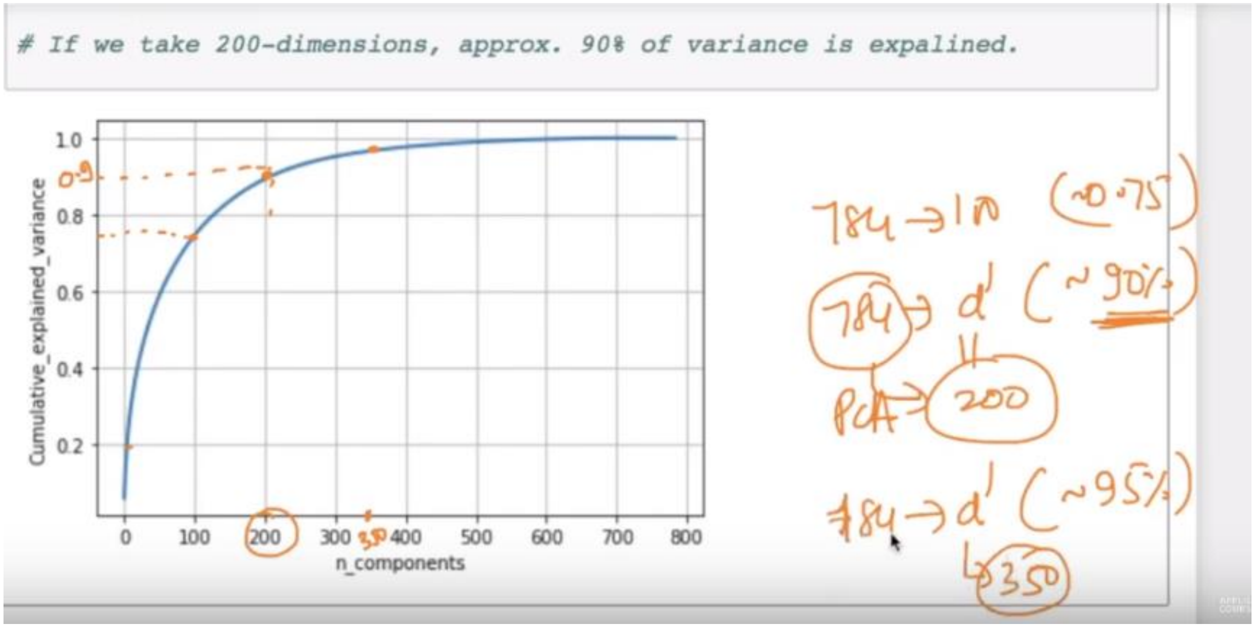
20% of the total variance in 784-dim is explained in 10-dim

```
# PCA for dimensionality redcution (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()
```

$$\frac{\lambda_i}{\sum \lambda_i} \quad \forall i$$

$$\frac{\lambda_1}{\sum \lambda_i}, \frac{\lambda_2}{\sum \lambda_i}, \frac{\lambda_3}{\sum \lambda_i} \dots$$

$$\frac{\lambda_1}{\sum \lambda_i}, \frac{\lambda_1 + \lambda_2}{\sum \lambda_i}, \frac{\lambda_1 + \lambda_2 + \lambda_3}{\sum \lambda_i} \dots$$



As we can see for 100 components we have 75%, for 200 we have 90% and for 300 components we have 95%.