DATE
PAGE

Principal Component Analysis

lec1 To convert high dimensional dataset to smaller dimension.

for eg- MNIST dataset was 784 dimensions and we convert it to 2 dimensions.

The Principal Component Analysis is used there.

lec2 Geometric Intuition of PCA.

We have to convert d-dimensions to d' dimensions, where $d' < d$.

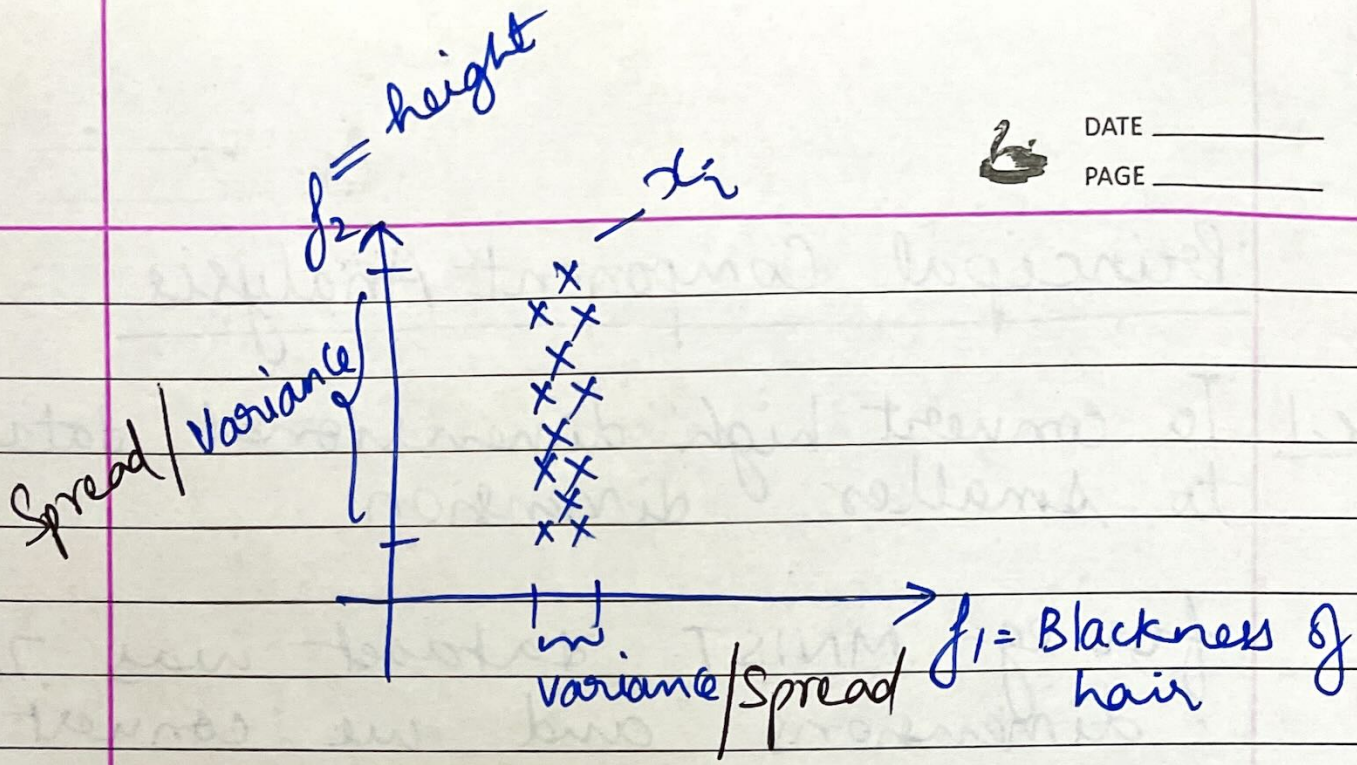
Say d = 2-dim and d' is 1dim.

Lets assume two features f_1 and f_2.

Where f_1 = Blackness of hair $\in \mathbb{R}$ and f_2 is Height of people.

The data matrix can be represented as

$$X = \begin{matrix} 1 \\ 2 \\ \vdots \\ n \end{matrix} \begin{bmatrix} f_1 & f_2 \\ & \\ & \end{bmatrix}$$



To reduce this 2 dim data to 1 dim we can skip f_1 (as low variance/pick) and keep f_2 as it has high variance and more information.

$$X' = \begin{matrix} 1 \\ 2 \\ \vdots \\ n \end{matrix} \begin{bmatrix} f_2 \\ \\ \\ \\ \end{bmatrix}$$

You lose less info in f_1 so skip f_1.
You lose more info in f_2 so don't skip f_2.

Preserving the direction with maximal spread/variance. More variance means more information.

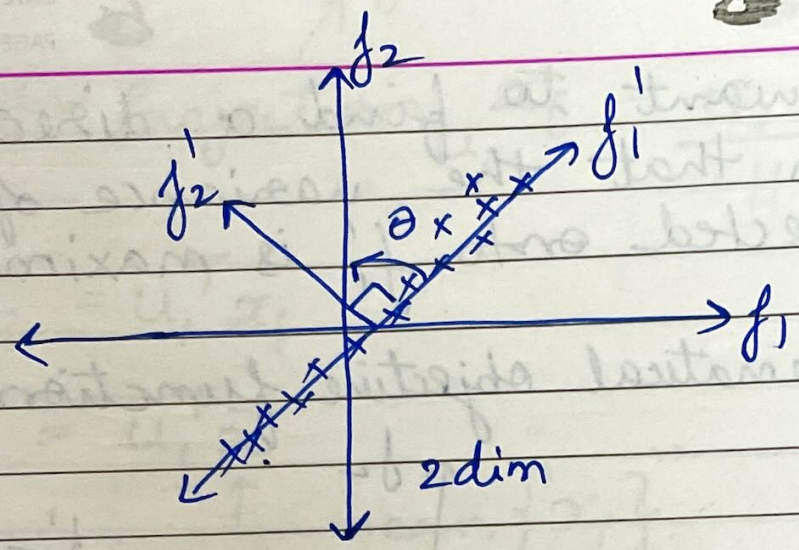
A different dataset now :-

Let X = 2 dim dataset.

It is column standardized which means

So, $\text{mean}\{f_1\} = \text{mean}\{f_2\} = 0$

$\text{Var}\{f_1\} = \text{Var}\{f_2\} = 1$



Here variance is like a line with datapoints.

(1) We draw a perpendicular to f_1' ie, f_2'. We try to rotate the f' axis so that the datapoints get to the maximum variance.

$\therefore$ Spread of mean of $f_2' \ll$ Spread of mean f_1'

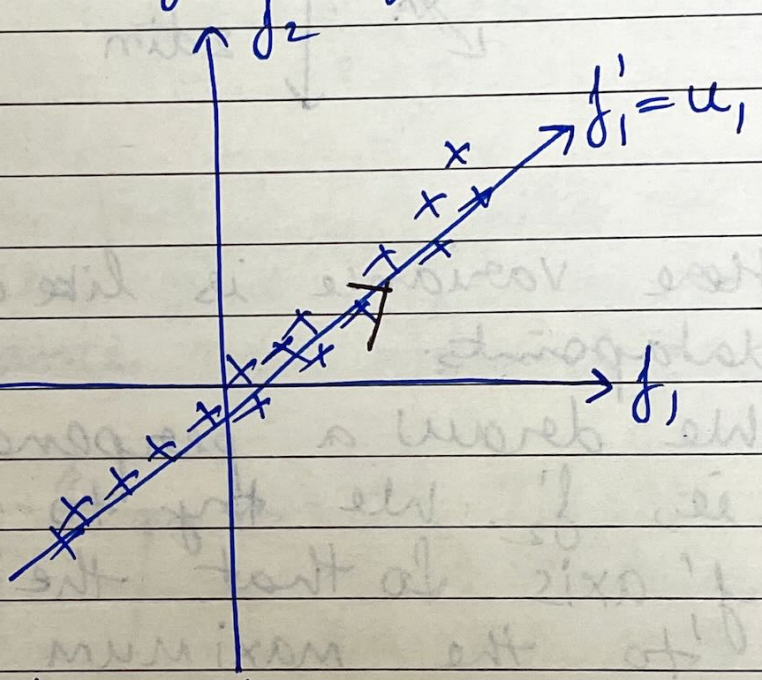
(2) Drop f_2'.

(3) Project x's ~~mean~~ onto f_1' then 2D to 1D.
such that f_1' has max variance.

DATE _____
PAGE _____

① we want to find a direction f_1' such that the variance of x_i's projected onto f_1' is maximum.

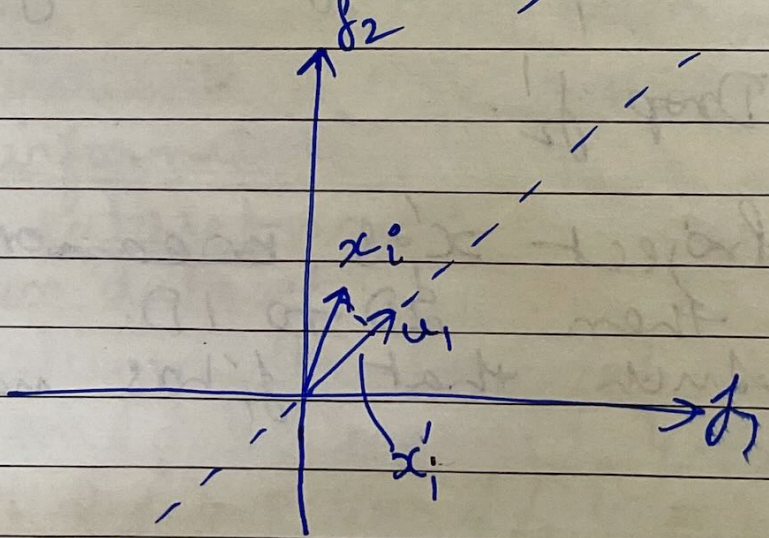
lec 3 Mathematical objective function of PCA.



Find direction of u_1

u_1 = unit vector (direction)

$\|u_1\| = 1$



projection of x_i onto u_1

DATE _____
PAGE _____

x_i' = Projection $_{u_1} x_i = \frac{u_1 \cdot x_i}{\|u_1\|^2 = 1} = u_1^T x_i$

$x_i' = u_1^T x_i$

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

$\bar{x}' \rightarrow \text{mean}\{x_i'\}_{i=1}^n$

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

~~u_1 is the~~

Problem: Find u_1 s.t. $\text{var}\{\text{proj}_{u_1} x_i\}_{i=1}^n$ is maximal

$$\text{variance}\{u_1^T x_i\}_{i=1}^n = \underbrace{\frac{1}{n}\sum_{i=1}^{n}}_{\text{avg.}} \left(\underbrace{u_1^T x_i}_{x_i'} - \underbrace{u_1^T \bar{x}}_{\text{mean of } x_i}\right)^2$$

minus

$\text{Scalar} = (u_1)^T x_{i(n\times 1)}$

$(1\times n)$

(1×1)

DATE _____
PAGE _____

Variance of $\{x_i\}_{i=1}^{n} = \frac{1}{n}\sum_{i=1}^{n} (u_1^T x_i)^2$

We want to maximize variance, ie

$$\max_{u_1} \frac{1}{n}\sum_{i=1}^{n} (u_1^T x_i)^2$$

objective of an optimization problem

x_i ↑ Data matrix

such that $u_1^T u_1 = 1 = \|u\|^2$

↳ constraint

u_1 is a unit vector

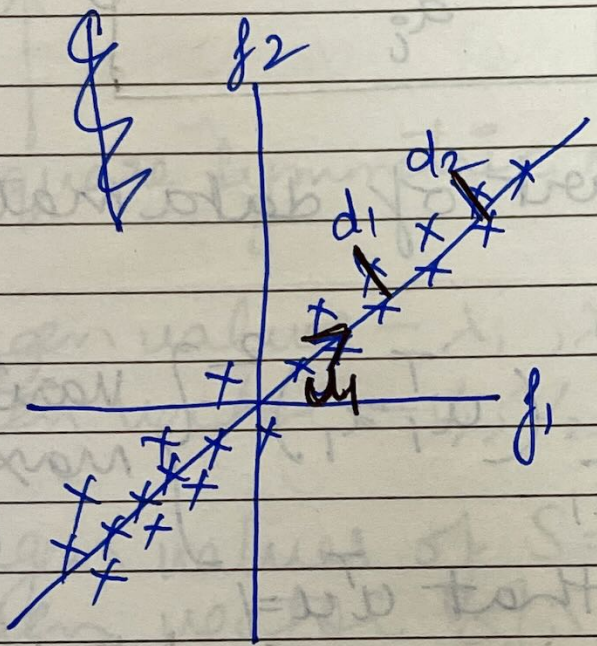
$u_1 = [\infty, \infty]$.

Find u_1 which maximizes projected variance.

DATE
PAGE

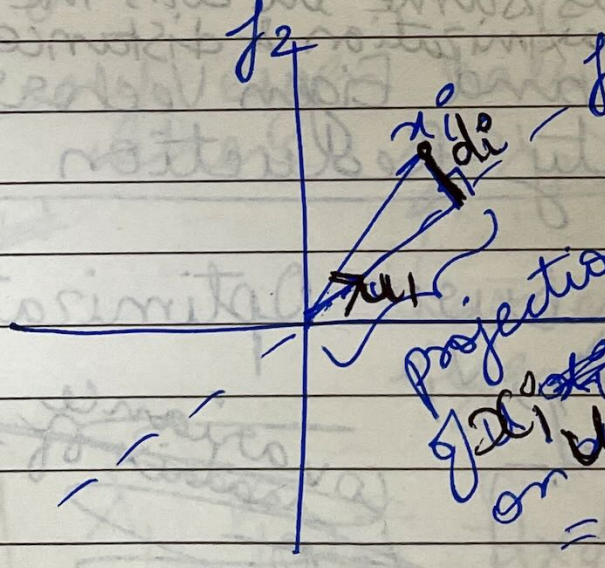
lec 4 Alternative Formulation of PCA: Minimization Distance

:→ Lets assume our points



For each point x_i we get d_i which is distance from x_i to u_1

$$\min_{u_1} \sum_{i=1}^{2} d_i^2$$



Assumptions —
u_1 = unit vector
$u_1^T u_1 = 1 = \|u_1\|^2$

$$d_i^2 = \|x_i\|^2 - (u_1^T x_i)^2$$
$$= x_i^T x_i - (u_1^T x_i)^2$$

objective function.

optimization problem for distance

$$\min_{u_1} \sum_{i=1}^{n} \left(x_i^T x_i - (u_1^T x_i)^2 \right) \quad \text{s.t} \; u^T u = 1$$

$\left(x_i^T x_i - (u_1^T x_i)^2 \right) = d_i^2$

constraint

distance min PCA

x_i are part of data matrix

We have done this

$$\max_{u_1} \frac{1}{n} \sum_{i=1}^{n} (u_1^T x_i)^2 \quad \text{such that } u^T u = 1$$

variance maximization PCA

Constraint $u^T u$ is same in both the Optimization of variance maximization & distance minimization

lec 5 Eigen Values and Eigen Vectors (PCA) Dimensionality Reduction.

Solution to our Optimization problems: λ_i, V_i $1, 2, \ldots, d$

Data matrix

columns standardized $\mu = 0$, $\sigma^2 = 1$

X: $1, 2, \ldots, n$ rows

$n \times d$

covariance matrix of X

$S = X^T X$

$d \times d$ $(d \times n)(n \times d)$

d features in terms of X

DATE
PAGE

Covariance matrix of $X = S$

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

↑ Square Symmetric matrix

eigen values $- \lambda_1, \lambda_2 \lambda_3 \ldots \lambda_d$
eigen vector $- V_1, V_2, V_3 \ldots V_d$

eigen values of $S = \lambda_1, \lambda_2 \ldots \lambda_d$
eigen vector of $S = V_1, V_2 \ldots V_d$

Every eigen value has an eigen vector
where $\lambda_1 > \lambda_2 > \lambda_3 > \lambda_4 \ldots \lambda_d$
↑ maximal eigen values

def of eigen values & vectors

$\lambda_1 \; V_1 = S_{d \times d} V_1$ — $d \times 1$ vector

λ_1 — Scalar
V_1 — $d \times 1$ column vector

then λ_1 = eigen value and V_1 is corresponding to eigen vector

numpy gives eigen values of an Matrix

DATE
PAGE

Property of Eigen vectors

$\rightarrow V_i \perp V_j$

which means $V_i^T V_j = 0$ or $V_i \cdot V_j = 0$

(dot product)

$u_1 = V_1$ = eigen vector of $S (= X^T X)$ corresponding to largest eigen value (λ_1)

maximal variance direction

$X = [\quad]_{n \times d}$

Steps to find u_1

1. Do Column Standardization of X
2. $S = \frac{1}{n}(X^T X)$
3. Compute Eigen Values & Eigen Vectors of S. Transpose the vector. DO (V_i^T matmul X_i^T)
4. $u_1 = V_1$ eigen vector

You will get 2-D data.

Use for dimensionality reduction as variance (max)

Blue points → To find variance (max)
Black points → Dim. Reduction.

For d dim data, we have 'd' eigen values and corresponding 'd' eigen vectors

DATE
PAGE

$x_i \in \mathbb{R}^{10}$ If, $d = 10$

$\lambda_1 \geq \lambda_2 \geq \lambda_3 \cdots \geq \lambda_{10}$

v_1 v_2 v_3 v_{10}

v_1 ↑ direction with max variance

v_2 ↑ 2nd direction with max var.

v_3 ↑ 3rd direc with most var

v_{10} ↑ least most var. direc

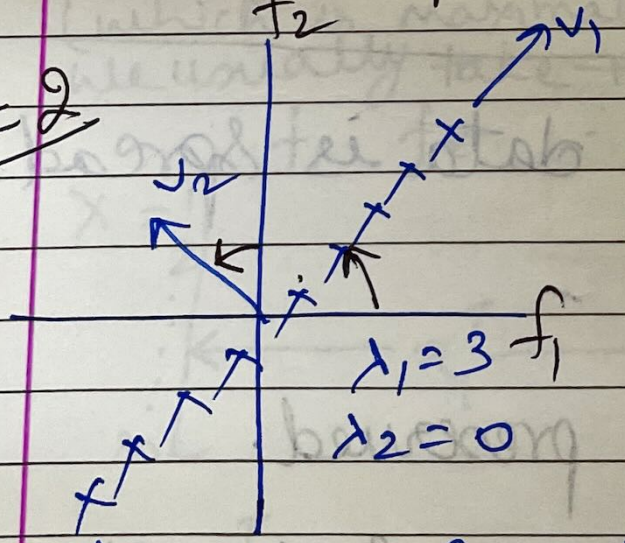
$\sum_{i=1}^{d} \lambda_i$

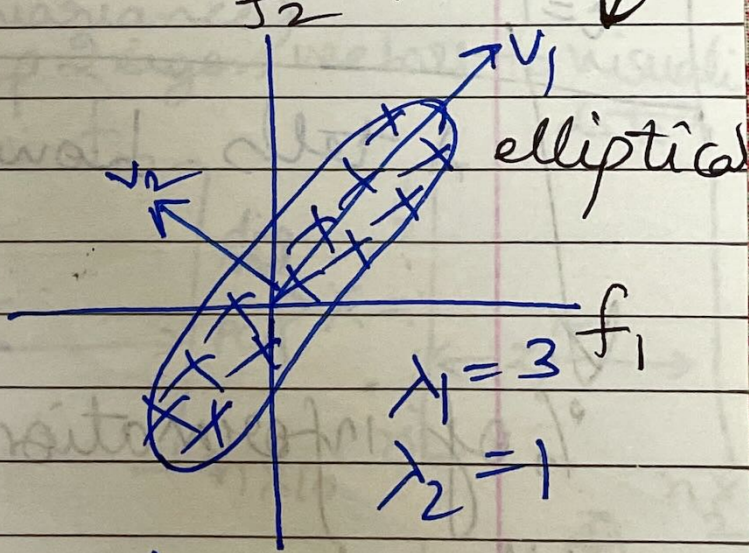
Geometric explanation of Eigenvalues & Eigen vectors

Linear spread

Some spread

d=2



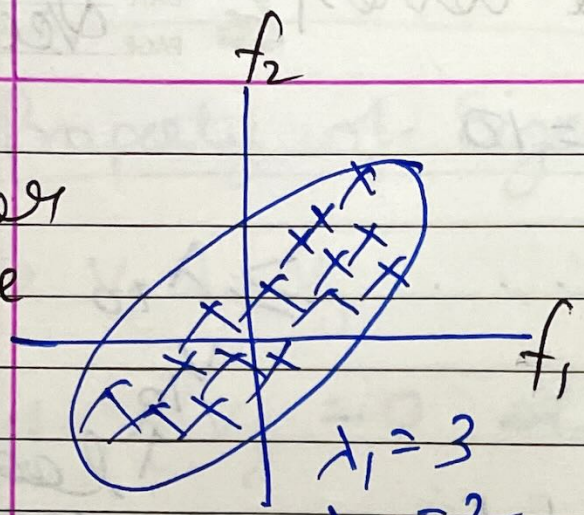


$\frac{\lambda_1}{\lambda_1 + \lambda_2}$ = 1% of variance explained

$\frac{\lambda_1}{\lambda_1 + \lambda_2} = \frac{3}{4}$ % of variance explained

Circular Variance

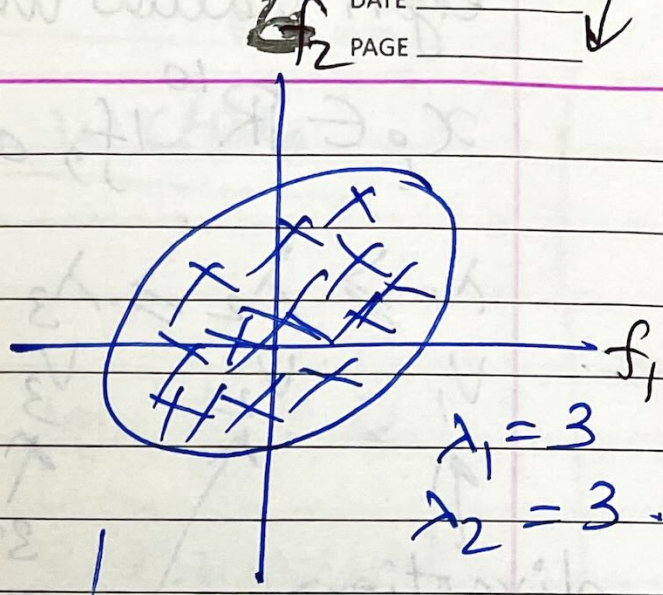
DATE
PAGE



Thicker ellipse

$\lambda_1 = 3$
$\lambda_2 = 2$

$\frac{\lambda_1}{\lambda_1 + \lambda_2} = \frac{3}{5} = 0.6$



$\lambda_1 = 3$
$\lambda_2 = 3$.

$\frac{\lambda_1}{\lambda_1 + \lambda_2} = 0.5$

$\frac{\lambda_i}{\sum_{i=1}^{d} \lambda_i}$ = Percentage of variance explained.

$i = 1$

λ tells how data is spread.

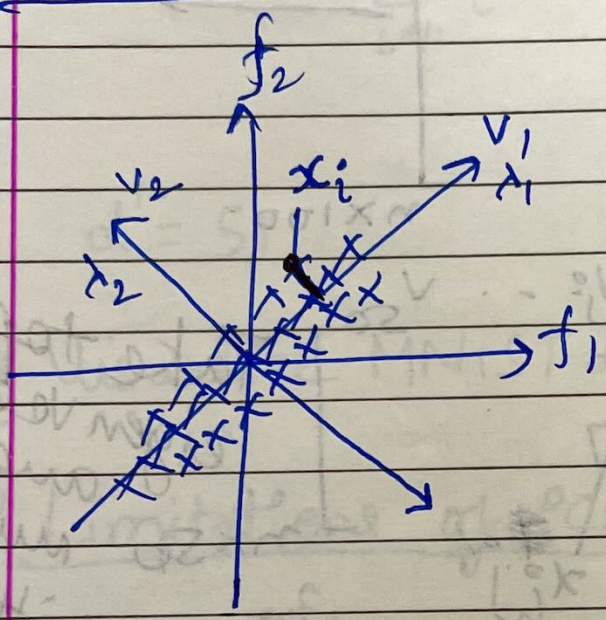
% of information preserved.

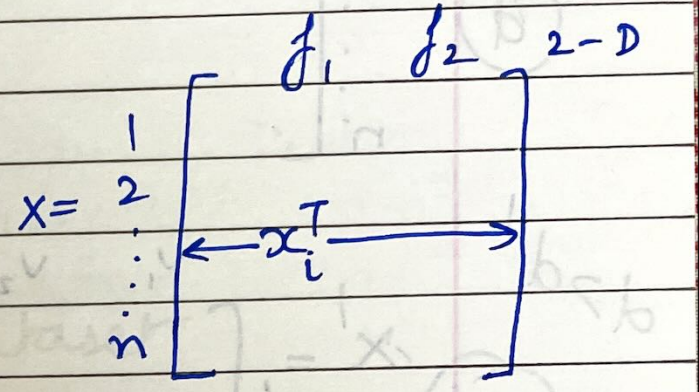
λ tells whether variance is spread on eigen value λ no. of axes.

e

DATE
PAGE

lec6 PCA for dimensionality reduction and visualization

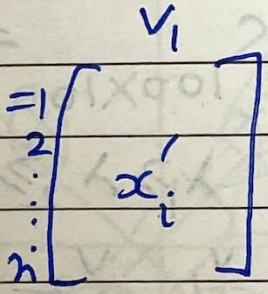




1-D

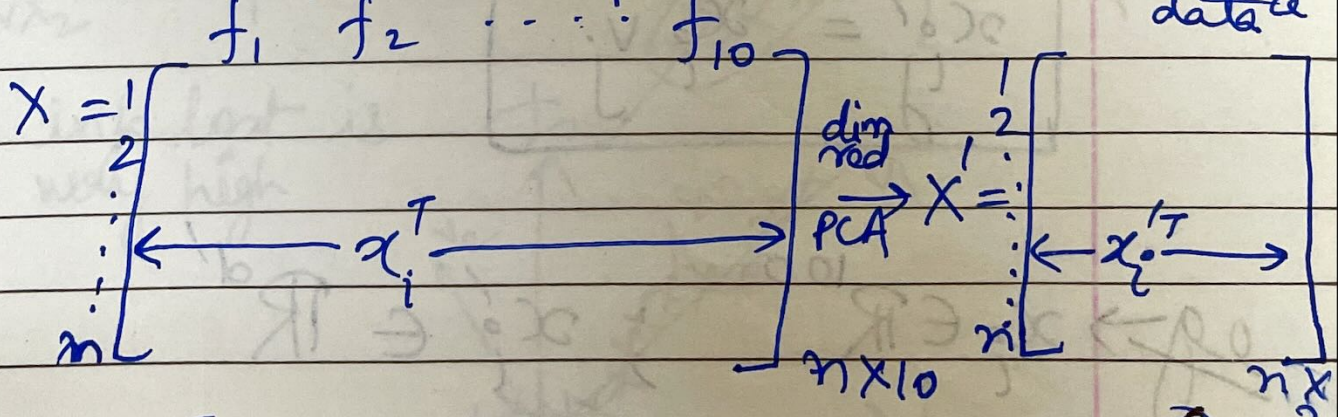
$S = X^T X$ (Covariance Matrix)

$x_i' = x_i^T v_1$ ← For v_1 to maximize variance



By projecting x_i' on v_1 (which is maximal variance)

We usually take top 2 eigen vectors to visualize data



$S = X^T X$

eigen(S) = $\lambda_1 \geq \lambda_2 \geq \lambda_3 \cdots \geq \lambda_{10}$

v_1 v_2 v_3 v_{10}

$x_i' = [x_i^T v_1 \quad x_i^T v_2]$

f_1 f_2

DATE _____
PAGE _____

f_1 f_2 ... f_{100}

$X = \begin{bmatrix} 1 \\ 2 \\ \vdots \\ n \end{bmatrix}$ ④ $\xrightarrow{PCA}$ $n \times 100$

$d > d'$

$X' = \begin{bmatrix} x_1' \end{bmatrix}$ ⑤ $n \times 50$

v_1 v_2 v_j ... v_{50}

Take top 50 eigenvectors and multiply with x_{ij} you get top 50 features

x_{ij}'

$S_{100 \times 100} = X^T X$

$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_{100}$

v_1 v_2 ... v_{100}

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

eg → $x_i \in \mathbb{R}^{100}$; $x_i' \in \mathbb{R}^{d'}$

$d' < 100$

Preserve 99% of the variance

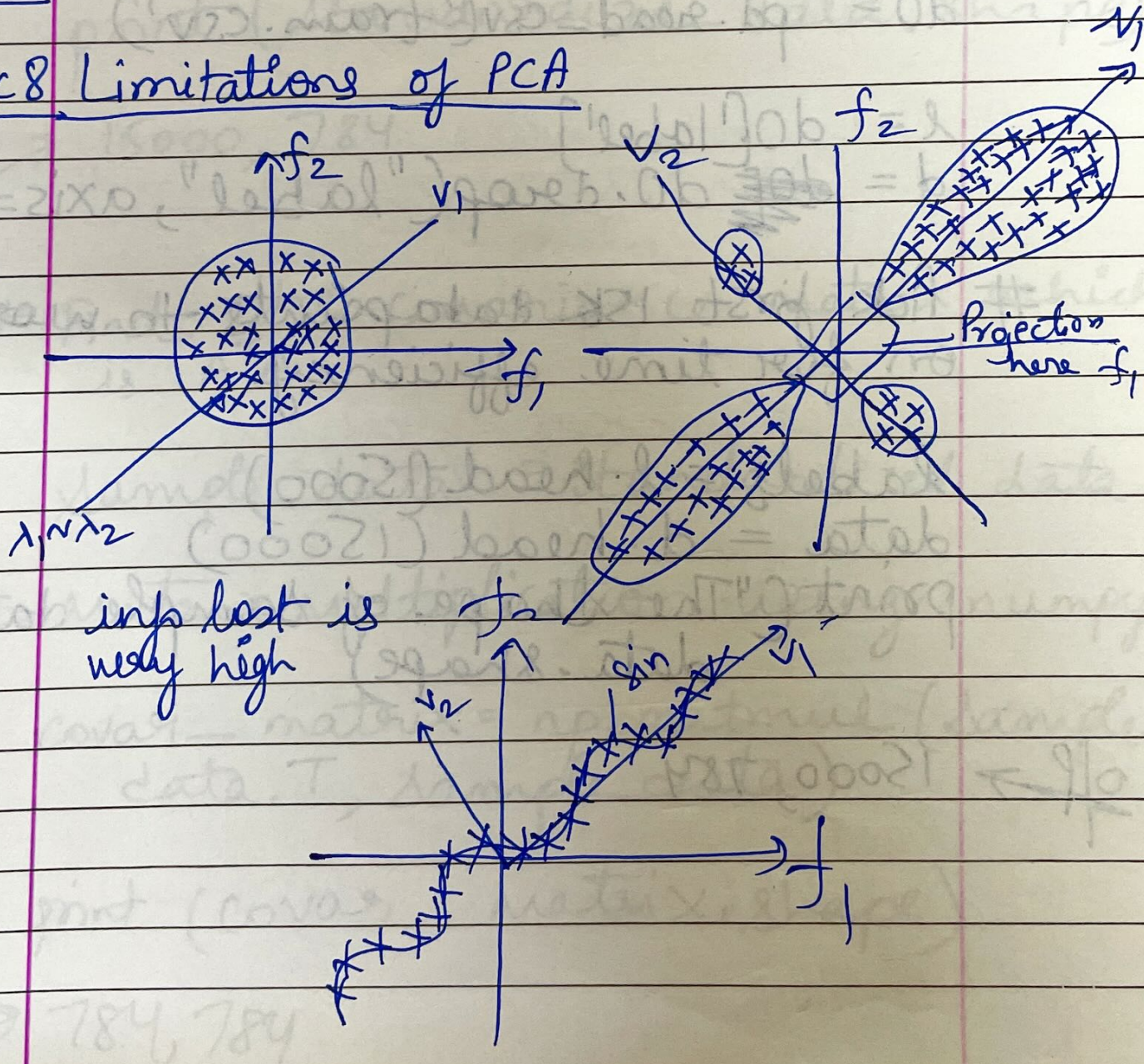
DATE
PAGE

Let $\frac{\lambda_1 + \lambda_2 + \lambda_3 + \cdots \lambda_{51}}{\sum_{i=1}^{100} \lambda_i} = 0.99$

$d' = 51$

lec7 Visualize MNIST dataset

lec8 Limitations of PCA



DATE
PAGE

lec9 PCA code example

Imports.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

d0 = pd.read_csv('train.csv')

l = d0['label']
d = d0.drop("label", axis=1)

# Pick first 15K datapoints to work on for time efficiency.

labels = l.head(15000)
data = d.head(15000)
print("The shape of sample data",
      data.shape)
```

o/p → 15000, 784

DATE
PAGE

Data preprocessing: Standardizing the data

```
from sklearn.preprocessing import StandardScaler

standardized_data = StandardScaler().fit_transform(data)
print(standardized_data.shape)
```

o/p → 15000, 784

Find the covariance matrix which is A^TA

```
sample_data = standardized_data
```

Matrix multiplication using numpy

```
covar_matrix = np.matmul(sample_data.T, sample_data)
print(covar_matrix.shape)
```

o/p → 784, 784

DATE
PAGE

Finding the top 2 eigen values and corresponding eigen vectors for projecting onto 2-Dim space.

The parameter 'eigvals' is defined (low value to high value) eigh function will return the eigen values in ascending order.

This code generates only top 2 (782, 783) eigenvalues.

```
from scipy.linalg import eigh
values, vectors = eigh(covar_matrix
                       eigvals=(782, 783))
print(vectors.shape)
```

op → (784, 2)

```
vectors = vectors.T
print(vectors.shape)
```

op → (2, 784)

DATE
PAGE

Projecting the original data sample on the plane formed by two principal eigen vectors by vector-vector multiplication

```
import matplotlib.pyplot as plt
new_coordinates = np.matmul(vectors, sample_data.T)
```

of → 2,784 × 784,15000 = 2,15000

```
import pandas as pd

# appending label to the 2D projected data
new_coordinates = np.vstack((new_coordinate, labels)).T

# Creating a new data frame for plotting the labelled points:
dataframe = pd.DataFrame(data = new_coordinates, columns =("1st principal", "2nd principal", "label"))
print(dataframe.head())
```

DATE
PAGE

plotting the 2d data points with seaborn.

```
import seaborn as sns
sns.FacetGrid(dataframe, hue="label
size=6).map(plt.scatter,"1stprincipal",
"2ndprincipal").add_legend()
plt.show()
```

PCA using Scikit Learn.

initializing the PCA

```
from sklearn import decomposition
pca = decomposition.PCA()
```

Configuring the parameters the no. of parameters =2

```
pca.n_components = 2
pca_data = pca.fit_transform(
                    sample_data)
```

DATE
PAGE

pca reduced will contain the 2-d projects of simple data

```
print(pca_data.shape)
"
o/p => 15000, 2
```

attaching the label for each 2-D data point

```
pca_data = np.vstack((pca_data.T, labels)).T
```

Creating a new data frame which help us in plotting the result data

```
pca_df = pd.DataFrame(data=pca_data
columns =("1st principal", "2nd principal",
"label"))

sns.FacetGrid(pca_df, hue="label",
size=6).map(plt.scatter, "1stprincipal",
"2nd principal").add_legend()
plt.show()
```

DATE _____
PAGE _____

lec10 PCA for dimensionality reduction not for ~~code~~ visualization.

```
# initializing the PCA
from sklearn import decomposition
pca = decomposition.PCA()
# Configuring the no. of parameters = 784.
pca.n_components = 784
pca_data = pca.fit_transform(sample_data)

percentage_var_explained = pca.explained_variance_ / np.sum(pca.explained_variance_)
cum_var_explained = np.cumsum(percentage_var_explained)
```

DATE
PAGE

Plot the PCA spectrum