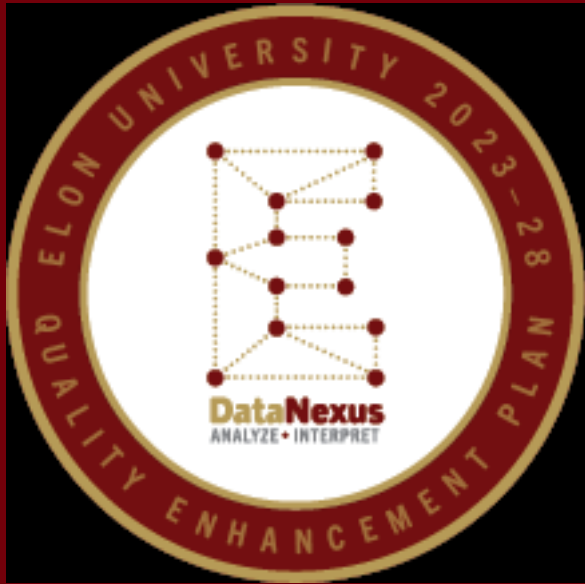


INTRO TO PYTHON 4

Be sure to get an account at

<https://research.google.com/colaboratory/>

DataNexus
ANALYZE • INTERPRET



DOWNLOAD THE DATA FOR THIS SESSION

Before we begin download
`cinderella_tales.csv` from our website





Files



- ..
- sample_data
- cinderella.csv



+ Code + Text

RAM
Disk Colab AI

```
[2] from numpy import linalg
from sklearn.linear_model import LogisticRegression
import statistics as st
import matplotlib.pyplot as plt
from scipy import stats
import numpy.polynomial
```


/usr/local/lib/python...
warnings.warn("Set

We will next Create a dataframe
Create a Code Cell
Type `df=pd.read_csv(" ")`

Uploading a File for Creating a Scatterplot

```
df=pd.read_csv("/content/gunlaws.csv")
```

Copy your file path here
Don't forget to run the cell



Click on the 3 dots next to the cinderella.csv file to Copy the Path



Files



- ..
- sample_data
- cinderella.csv

+ Code + Text

```
import pip
import pandas as pd
import numpy as np
import scipy
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
df=pd.read_csv("/content/cinderella.csv")
print(df)
data=np.array(df)
```

Binary data, each record (Cinderella story) has many attributes.

Focusing on projection called Principal Component Analysis (PCA)

	Unnamed: 0	Rhodopis - Egypt	\
0	Name means ash	0	
1	Herroine is persecuted	1	
2	Slavery	1	
3	Picking of lentils or beans	0	
4	Weaving/spinning	0	
5	Cannabolism	0	
6	Cruel Stepmother or Stepsisters	0	
7	Herroine kills stepmother, sister	0	
8	Herroine arranges marriage to stepmother	0	
9	Weak father or absent	0	
10	Magical animal	1	
11	Magical fairy (female or male)	0	





Files

{x}

sample data

+ Code + Text

0s

```
import pip
import pandas as pd
import numpy as np
import scipy
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
df=pd.read_csv("/content/cinderella.csv")
data=np.delete(df,0,1)
gramian=np.dot(np.transpose(data),data)
```

You might just want to see the relationship between the tales (your records) in this case take a look at the Gramian.

Note the tales are in the columns here.

If your records are in the rows:
`np.dot(data,np.transpose(data))`

Files

sample_data

cinderella.csv

Print the gramian

```
import pip
import pandas as pd
import numpy as np
import scipy
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
df=pd.read_csv("/content/cinderella.csv")
data=np.delete(df,0,1)
gramian=np.dot(np.transpose(data),data)
print(gramian)
```

The j^{th} number in the i^{th} row tells you how many attributes that tale i and tale j have in common.

How many attributes do the 1st tale and the 2nd tale have in common?

The i^{th} number in the i^{th} row tells you how many attributes that tale has.

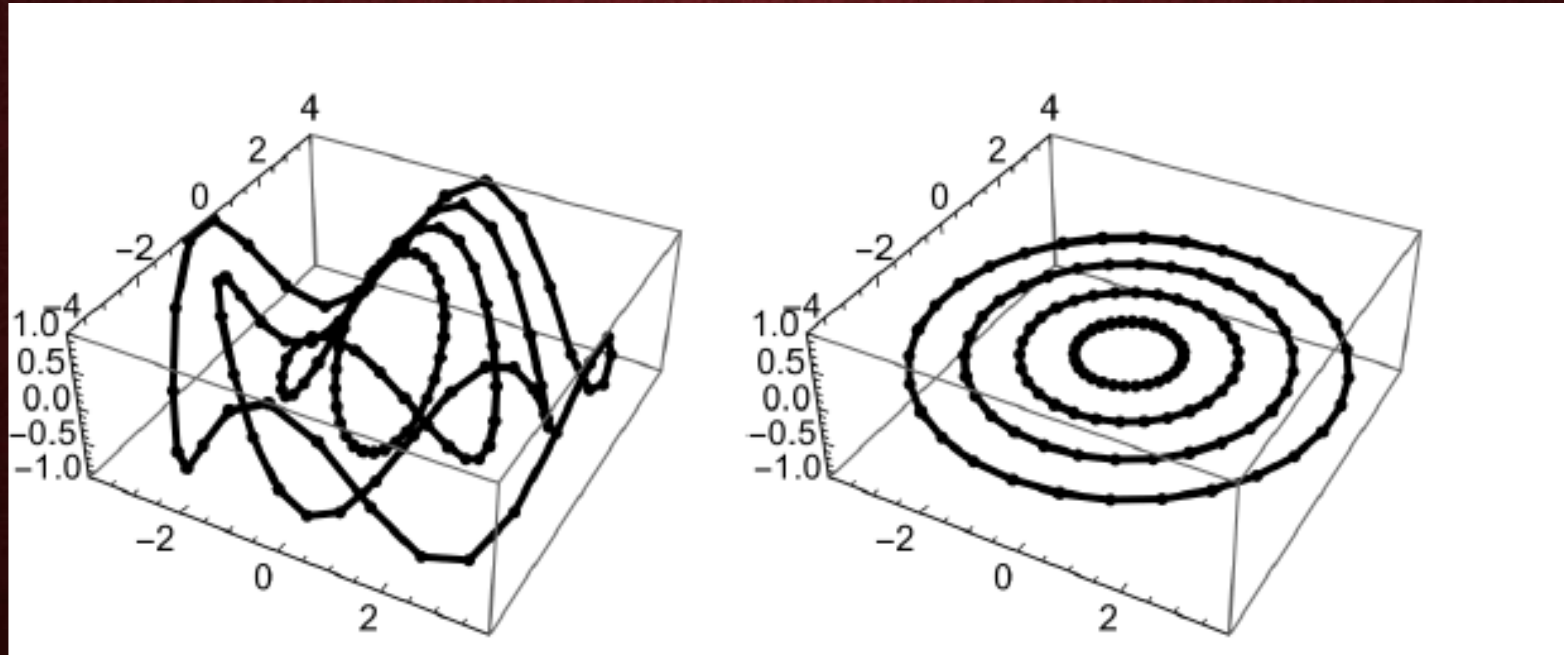
How many attributes does the 1st tale have?

```
[[5 3 4 3 4 3 4 4 3 4 4 4 3 3 4 4 4 3 4 5
 [3 9 4 4 6 4 6 5 6 6 5 6 6 6 5 5 5 5 4 5
 [4 4 15 8 10 9 6 10 5 7 6 5 5 7 8 6 8 8
 [3 4 8 10 7 9 6 8 6 6 4 5 6 8 6 6 7 8 7
 [4 6 10 7 18 7 11 9 9 9 7 9 9 10 13 10 1
 8]
 [3 4 9 9 7 11 5 8 5 6 4 4 5 7 6 5 6 7 7
 [4 6 6 6 11 5 12 7 8 9 8 10 7 9 10 11 9
 [4 5 10 8 9 8 7 14 7 6 5 6 7 7 7 7 8 7 8 7 7 7 8 10 4 9 4 8 5 10 10]
 [3 6 5 6 9 5 8 7 12 7 6 7 9 10 8 8 9 9 8 9 5 5 6 6 4 5 6 6 4 8 5]
 [4 6 7 6 9 6 9 6 7 11 8 9 7 9 9 8 7 8 8 8 5 7 9 7 4 5 4 7 4 9 7]
 [4 5 6 4 7 4 8 5 6 8 9 8 6 7 8 8 6 6 7 8 4 6 7 6 4 4 4 5 3 7 6]
```

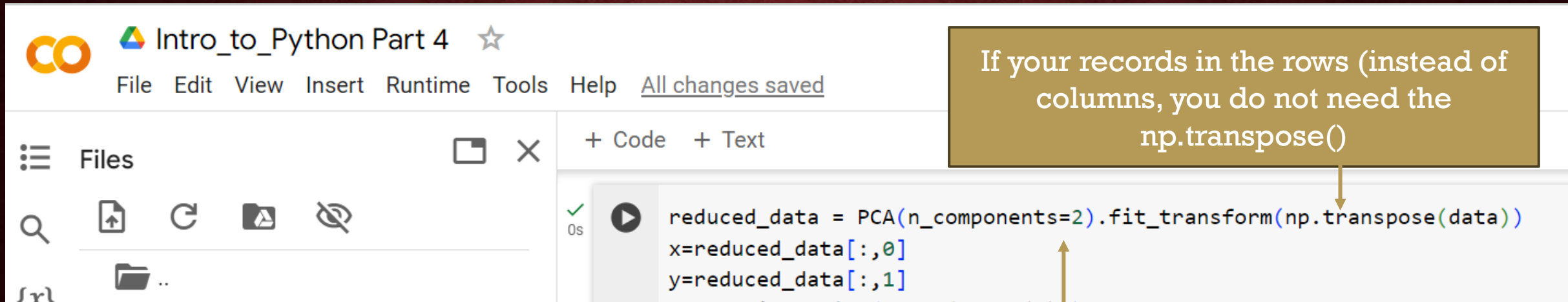
**WE NEXT FOCUS ON PROJECTION
WHAT IS A PROJECTION?**



WHAT IS A PROJECTION?



PRINCIPAL COMPONENT ANALYSIS



The screenshot shows a code editor window titled "Intro_to_Python Part 4". The code in the editor is:

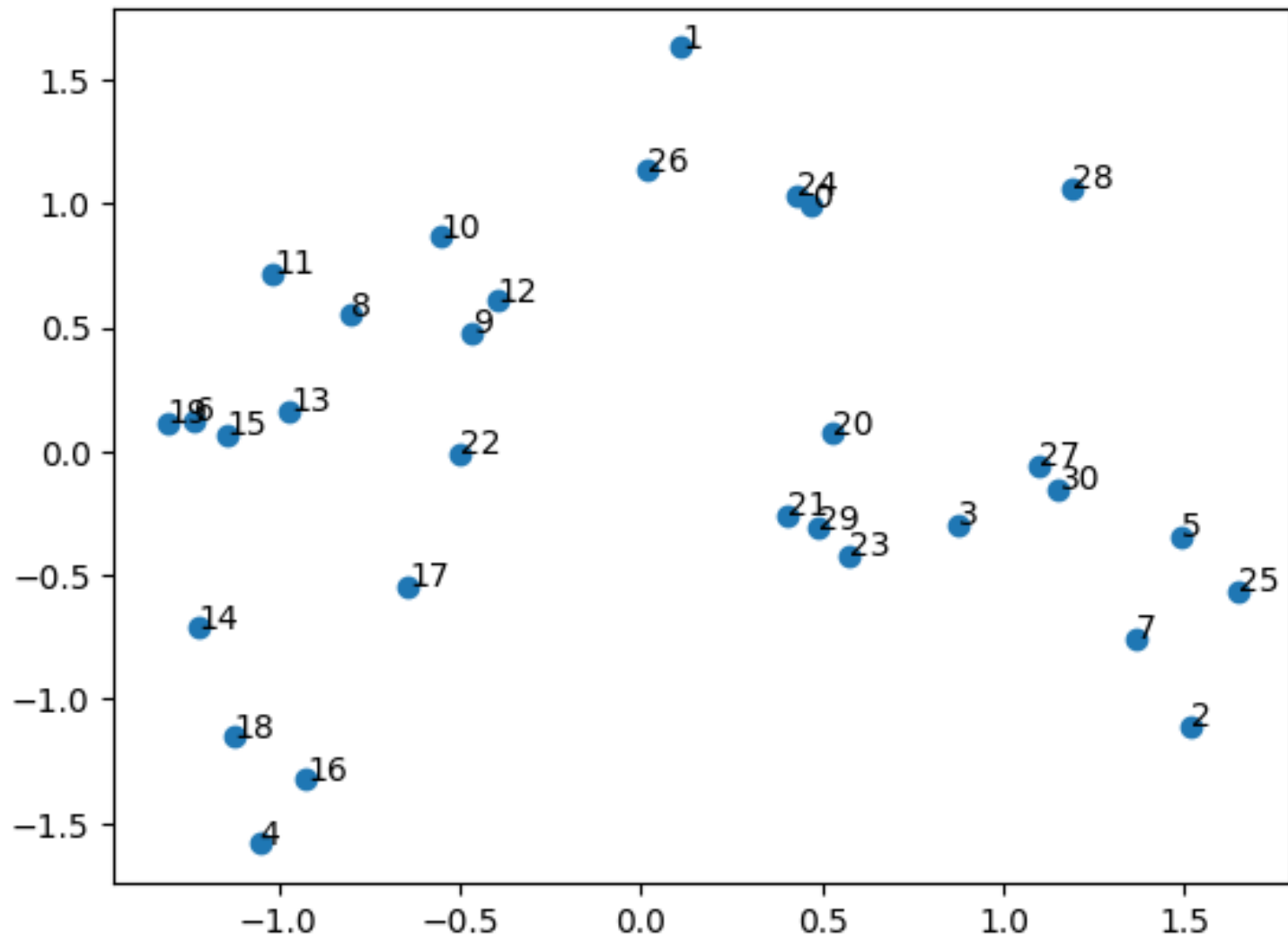
```
reduced_data = PCA(n_components=2).fit_transform(np.transpose(data))
x=reduced_data[:,0]
y=reduced_data[:,1]
```

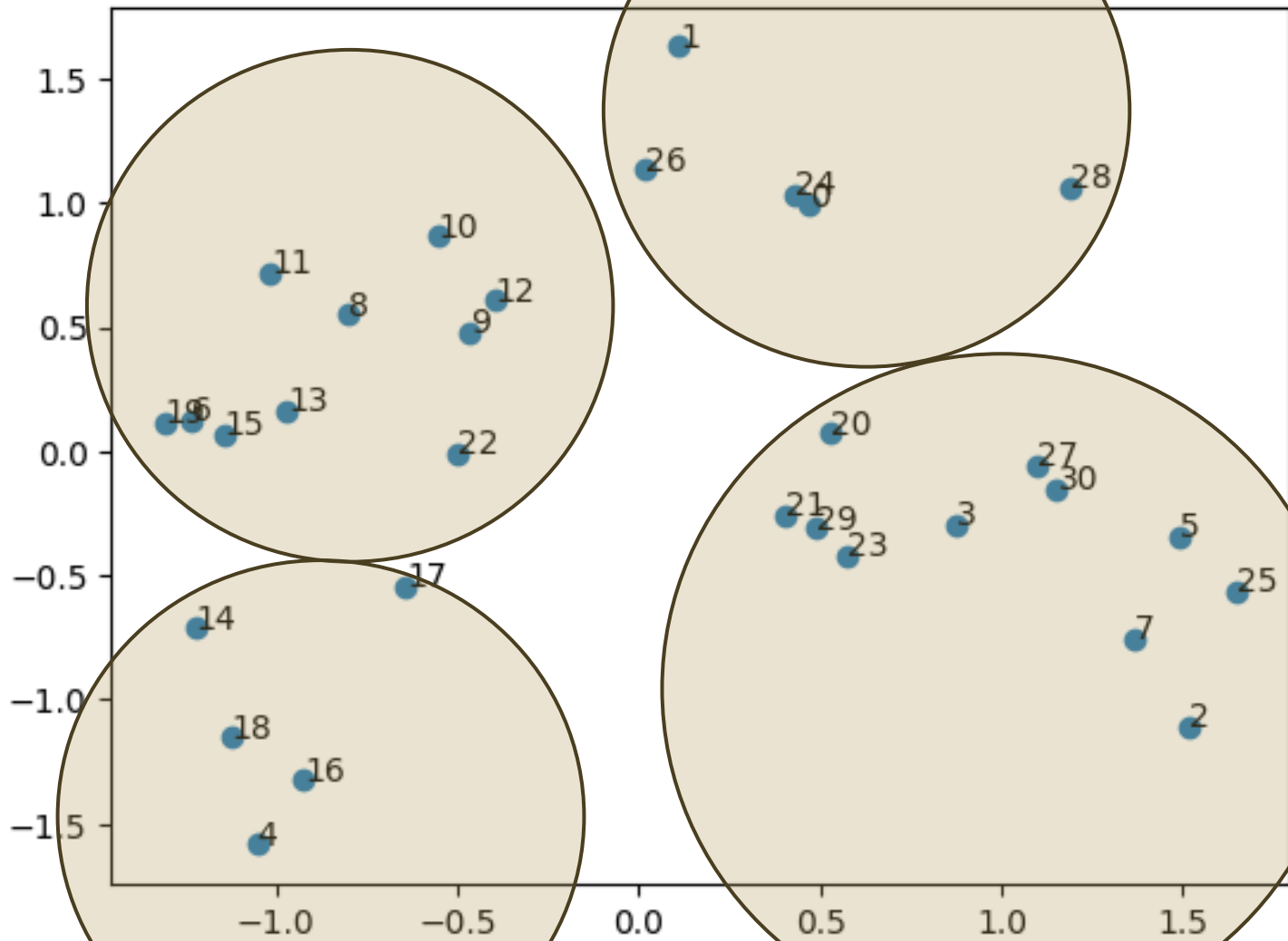
Annotations include:

- A box at the top right: "If your records in the rows (instead of columns, you do not need the np.transpose())" with an arrow pointing to the `np.transpose(data)` in the code.
- A box at the bottom right: "Tells PCA what dimension to project to." with an arrow pointing to the `n_components=2` in the code.

TO DO: Create a scatter plot with this new x and y and label the Cinderella tales 0 through 30.

Tells PCA what dimension to project to.





You can use PCA with non-binary data but standardize it first.

Contact Data Nexus if you have questions about this



Files



- ..
- sample_data
- cinderella.csv

Two options are k-means++ or random for choosing the centroids of the regions

Sets the number of regions in your clustering.

Number of iterations of kmeans with different centroids

Creates a x and y range just 1 less than the actual mins and maxs, you can choose other values.

```
kmeans = KMeans(init="k-means++", n_clusters=5, n_init=4)
kmeans.fit(reduced_data)

# Step size of the mesh. Decrease to increase the quality
h = 0.02 # point in the mesh [x_min, x_max]x[y_min, y_max].

# Plot the decision boundary. For that, we will assign a color to each
x_min, x_max = reduced_data[:, 0].min() - 1, reduced_data[:, 0].max() + 1
y_min, y_max = reduced_data[:, 1].min() - 1, reduced_data[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))
```

Kmeans clustering uses an average value to separate your regions

TO DO: Use this code to create a grid of .01 step size and 4 clusters.

Creates a grid within your range of step size h=0.02



Files  

{x}

- ..
- sample_data
- cinderella.csv

TO DO: Create a Kmeans clustering with 5 clusters, choose a new color scheme and put annotations/labels on the plot

+ Code + Text

```
# Obtain labels for each point in mesh. Use last trained model.
Z = kmeans.predict(np.c_[xx.ravel(), yy.ravel()])

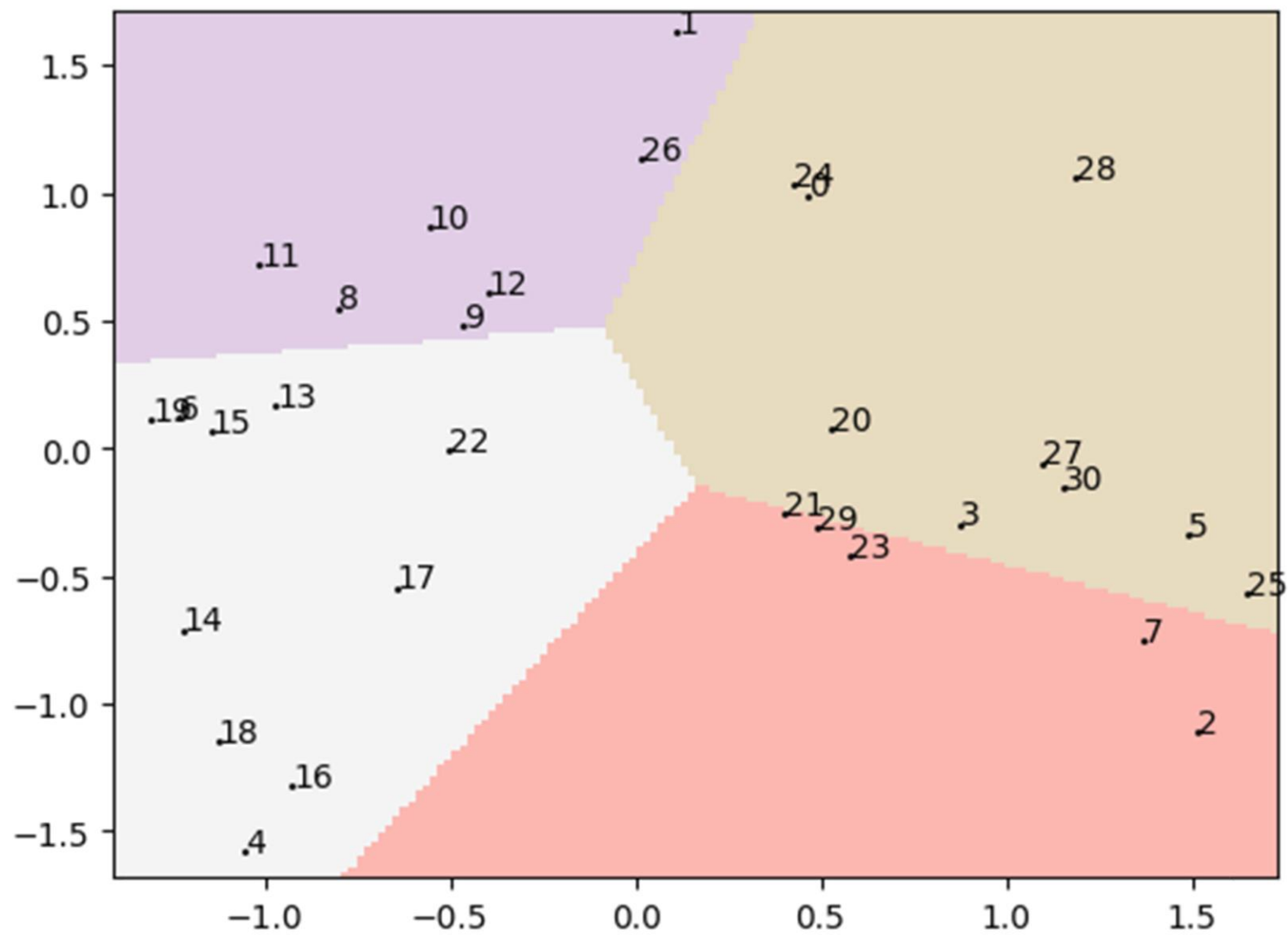
# Put the result into a color plot
Z = Z.reshape(xx.shape)

plt.imshow(
    Z,
    interpolation="nearest",
    extent=(xx.min(), xx.max(), yy.min(), yy.max()),
    cmap=plt.cm.Pastel1,
    aspect="auto",
)
```

Creates a grid within your range of step size h=0.02

Check Video 3 for the website to choose your color scheme

```
plt.plot(reduced_data[:, 0], reduced_data[:, 1], "k.", markersize=2)
plt.show()
```



You might also look into Decision Trees

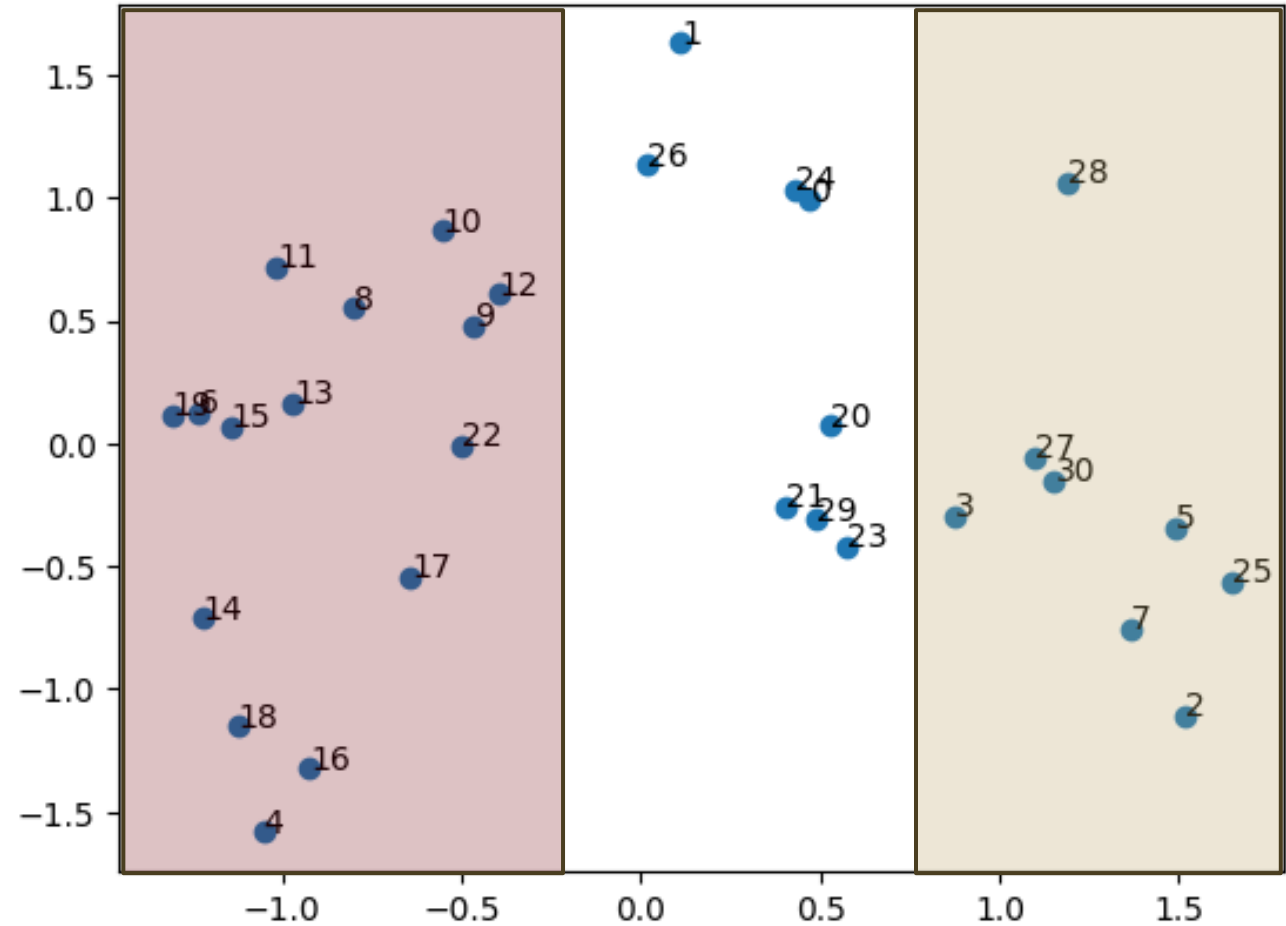
Reduced
Data

$-1.5 < X < -.25$
Group 1

$-.25 < X < .75$
Group 2

$.75 < X < 2$
Group 3

Example



You might also look into Decision Trees

Reduced
Data

$-1.5 < X < .75$

$.75 < X < 2$
Group 3

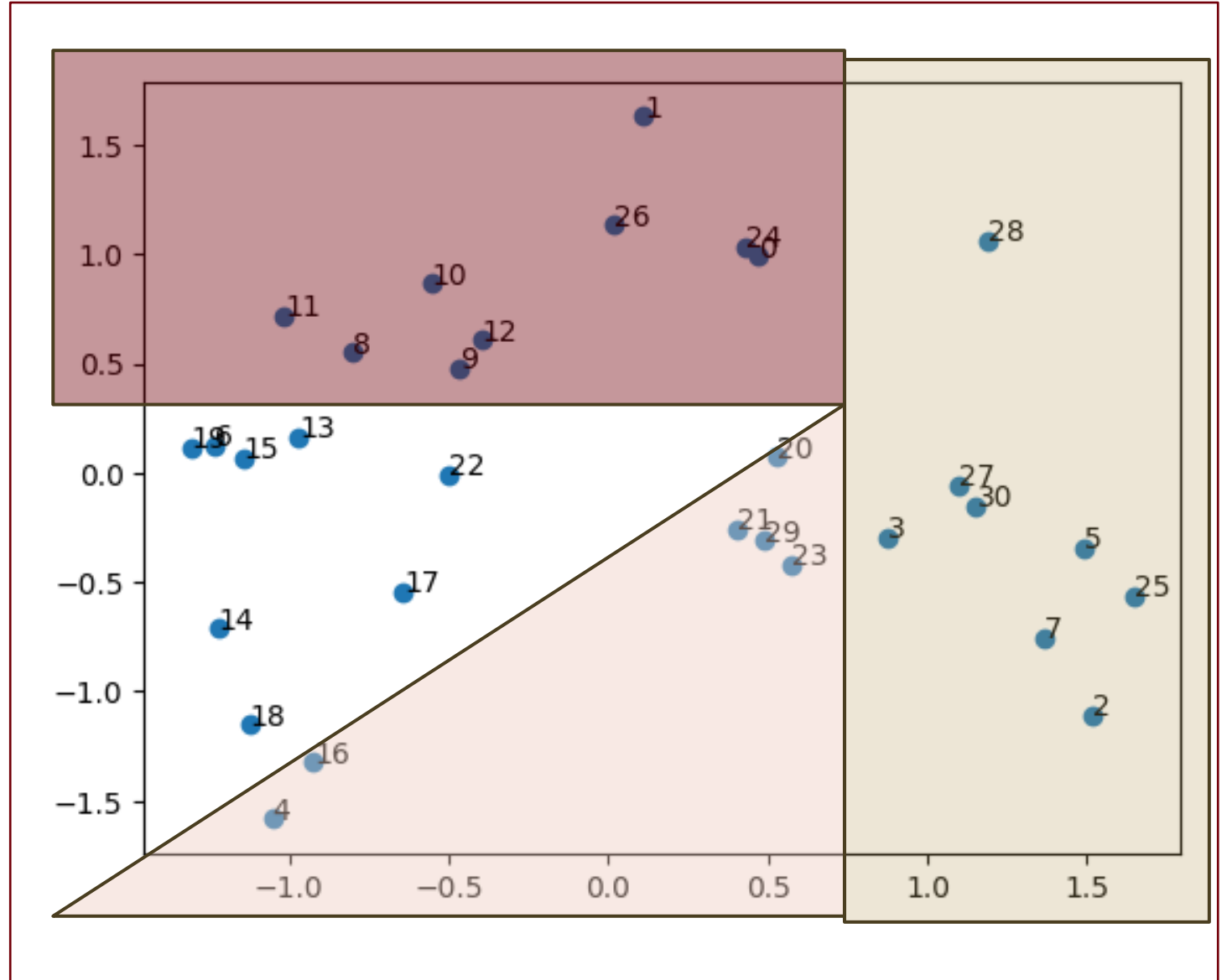
$-2 < Y < .25$

$.25 < Y < 1.5$
Group 4

Y is above
Line 1 and
below $Y = .25$
Group 2

Y is below
Line 1
Group 1

Example



IF YOU WANT TO LEARN
MORE ABOUT ANY OF
THESE TOPICS CONSIDER

CSC 1100

CSC 4422

MTH 2300

STS 2300

VISIT DATA NEXUS IN
INNOVATION HALL 108C
FOR OTHER
SUGGESTIONS

