Out[3]:	#features in the iris dataset iris.feature_names			
	sepal width (cm),			
Tn [4]:	<pre>'petal length (cm)', 'petal width (cm)']</pre>			
In [4]:	iris.target_names			
Out[4]: In [5]:		virginica'], dtype='	<010.)	
	<pre>X = iris.data y = iris.target</pre>			
In [6]:	#create dataframe using X,y df = pd.DataFrame(X,columns=iris	s,feature names)		
	#check header of the created dat df.head()	_		
Out[6]:	sepal length (cm) sepal width (cm) per	tal length (cm) petal widt	h (cm)	
	0       5.1       3.5         1       4.9       3.0	1.4 1.4	0.2	
	<b>2</b> 4.7 3.2 <b>3</b> 4.6 3.1	1.3 1.5	0.2	
	<b>4</b> 5.0 3.6	1.4	0.2	
In [7]:	<pre>#append target into dataframe df['target'] = iris.target</pre>			
In [8]:	<pre>df.columns=['sepal_length', 'sep df.head()</pre>	pal_width', 'petal_l	enth', 'petal_width', 'ta	arget']
Out[8]:	sepal_length sepal_width petal_lenth	petal_width target		
	0       5.1       3.5       1.4         1       4.9       3.0       1.4	0.2 0		
	24.73.21.334.63.11.5	0.2 0		
	<b>4</b> 5.0 3.6 1.4	0.2 0		
In [9]:	45 t	ngth, y = df.sepal_w	idth, hue = df.target, st	<pre>tyle = df.target);</pre>
	target 0			
	# 3.5 -	****		
	2.5 - * * * * * * * * * * * * * * * * * *			
	4.5 5.0 5.5 6.0 6.5 sepal_length	7.0 7.5 8.0		
In [10]:	Standardize the Data			
	<pre>x_temp = df.iloc[:, 0:4].values y_temp = df.target.values x temp = StandardScaler().fit tr</pre>	cansform(x temp)		
	2D representation of the dat	:a		
In [11]:	·			
	<pre>pl.figure() for i, c, label in zip(target_ic     pl.scatter(x_temp[iris.target])</pre>	ds, colors, iris.tar		
	<pre>c=c, label=label) pl.legend() pl.show()</pre>			
	setosa versicolor	••		
	virginica			
	0 -			
	-1 -			
	-2 $-1$ 0 1	2		
	The three different types of Iris are still clu  Compute the Eigenvectors a			
In [12]:		np.T)		
	Covariance matrix: [[ 1.00671141 -0.11835884 0.87]	_		
	[-0.11835884 1.00671141 -0.4313 [ 0.87760447 -0.43131554 1.0067 [ 0.82343066 -0.36858315 0.9693	31554 -0.36858315] 71141 0.96932762]		
In [13]:	<pre>eigen_values, eigen_vectors = nr print("Eigenvectors:\n", eigen_v</pre>		nce_matrix)	
	<pre>print("Eigenvalues:\n", eigen_va Eigenvectors:</pre>			
	[[ 0.52106591 -0.37741762 -0.719			
	[-0.26934744 -0.92329566 0.2443 [ 0.5804131 -0.02449161 0.1423 [ 0.56485654 -0.06694199 0.6343	12637 -0.80144925]		
	-	12637 -0.80144925] 27274 0.52359713]]		
	[ 0.5804131 -0.02449161 0.1423	12637 -0.80144925] 27274 0.52359713]] 82 0.02085386]	and visualization	
In [14]:	[ 0.5804131 -0.02449161 0.1423 [ 0.56485654 -0.06694199 0.6342 Eigenvalues: [2.93808505 0.9201649 0.1477418 PCA for transformation  #Apply PCA to transform iris data pca2 = PCA(n_components=2)	into 2D data a		
In [14]:	[ 0.5804131 -0.02449161 0.1423 [ 0.56485654 -0.06694199 0.6342 Eigenvalues: [2.93808505 0.9201649 0.1477418 PCA for transformation  #Apply PCA to transform iris dat pca2 = PCA (n_components=2) principalComponents2 = pca2.fit_	into 2D data a  taset into 2D for vi  transform(X)  principalCompone	suallization nts2, columns = ['princip	oal component 1', 'principal compone
<pre>In [14]: Out[14]:</pre>	[ 0.5804131 -0.02449161 0.1423 [ 0.56485654 -0.06694199 0.6342]  Eigenvalues: [2.93808505 0.9201649 0.1477418]  PCA for transformation  #Apply PCA to transform iris data pca2 = PCA(n_components=2) principalComponents2 = pca2.fit_principalDf2 = pd.DataFrame(data finalDf2 = pd.concat([principalIfinalDf2.head(5)	12637 -0.80144925] 27274 0.52359713]]  82 0.02085386]  into 2D data a  taset into 2D for vi  transform(X) a = principalCompone 0f2, df[['target']]]	suallization nts2, columns = ['princip	oal component 1', 'principal compone
	[ 0.5804131 -0.02449161 0.1423 [ 0.56485654 -0.06694199 0.6342]  Eigenvalues: [2.93808505 0.9201649 0.1477418]  PCA for transformation  #Apply PCA to transform iris data pca2 = PCA (n_components=2) principalComponents2 = pca2.fit_principalDf2 = pd.DataFrame (data finalDf2 = pd.concat([principalIfinalDf2.head(5)])  principal component 1 principal components	12637 -0.80144925] 27274 0.52359713]]  82 0.02085386]  into 2D data a  taset into 2D for vi  transform(X) a = principalCompone 0f2, df[['target']]]	suallization nts2, columns = ['princip	oal component 1', 'principal compone
	[ 0.5804131 -0.02449161 0.1423 [ 0.56485654 -0.06694199 0.6342]  Eigenvalues: [2.93808505 0.9201649 0.1477418]  PCA for transformation  #Apply PCA to transform iris data pca2 = PCA (n_components=2) principalComponents2 = pca2.fit_principalDf2 = pd.DataFrame (data finalDf2 = pd.concat([principalIf finalDf2.head(5)]  principal component 1 principal components	into 2D data a  taset into 2D for vi  transform(X)  a = principalCompone  Df2, df[['target']]]  ment 2 target  B19397 0	suallization nts2, columns = ['princip	oal component 1', 'principal compone
Out[14]:	[ 0.5804131 -0.02449161 0.1423 [ 0.56485654 -0.06694199 0.6342 Eigenvalues: [2.93808505 0.9201649 0.1477418]  PCA for transformation  #Apply PCA to transform iris data pca2 = PCA (n_components=2) principalComponents2 = pca2.fit_principalDf2 = pd.DataFrame (data finalDf2 = pd.concat([principalIfinalDf2.head(5)]  principal component 1 principal components	into 2D data a  taset into 2D for vi  transform(X)  a = principalCompone  Df2, df[['target']]]  ment 2 target  819397 0  177001 0  144949 0	suallization nts2, columns = ['princip	oal component 1', 'principal compone
	[ 0.5804131 -0.02449161  0.1422	into 2D data a  into 2D data a  taset into 2D for vi  transform(X)  a = principalCompone  of 2, df[['target']]]  nent 2 target  177001 0  144949 0  1818299 0  18 the model transform  transformed to 2D:  the data transformed	med from 4D to 2D  \n{pca2.components_}\n")  to 2D: pca2.explaine	
Out[14]:	[ 0.5804131 -0.02449161 0.142: [ 0.56485654 -0.06694199 0.6342]  Eigenvalues: [ 2.93808505 0.9201649 0.1477418]  PCA for transformation  #Apply PCA to transform iris data pca2 = PCA (n_components=2) principalComponents2 = pca2.fit_principalDf2 = pd.DataFrame (data finalDf2 = pd.concat([principalIfinalDf2.head(5))  principal component 1 principal compo  1	into 2D data a  into 2D data a  taset into 2D for vi  transform(X)  a = principalCompone  of 2, df[['target']]]  nent 2 target  319397	med from 4D to 2D  \n{pca2.components_}\n")  to 2D: pca2.explaine	
Out[14]:	[ 0.5804131	12637 -0.80144925] 27274 0.52359713]]  82 0.02085386]  into 2D data a  taset into 2D for vi  transform(X) a = principalCompone. 0f2, df[['target']]]  nent 2 target  819397 0  177001 0  144949 0  818299 0  826755 0  f the model transformed to 2D: che data	med from 4D to 2D  \n{pca2.components_}\n")  to 2D: pca2.explaine	
Out[14]:	[ 0.5804131	12637 -0.80144925] 27274 0.52359713]]  82 0.02085386]  into 2D data a  taset into 2D for vi  transform(X) a = principalCompone. 0f2, df[['target']]]  nent 2 target  819397 0  177001 0  144949 0  818299 0  826755 0  f the model transformed to 2D: transformed to 2D: the data transformed to 2D:	med from 4D to 2D  \n{pca2.components_}\n")  to 2D: pca2.explaine	
Out[14]:	[ 0.5804131	12637 -0.80144925] 27274 0.52359713]]  82 0.02085386]  into 2D data a  taset into 2D for vi  transform(X) a = principalCompone. 0f2, df[['target']]]  nent 2 target  819397 0  177001 0  144949 0  818299 0  826755 0  f the model transformed transformed to 2D: che data transformed to 2D:	med from 4D to 2D  \n{pca2.components_}\n")  to 2D: pca2.explaine	
Out[14]:	[ 0.5804131	12637 -0.80144925] 27274 0.52359713]]  82 0.02085386]  into 2D data a  taset into 2D for vi  transform(X) a = principalCompone 0f2, df[['target']]]  nent 2 target 819397 0 177001 0 144949 0 818299 0 826755 0  f the model transformed formed to 2D: che data transformed formed to 2D: he data transformed formed to 2D: che data transformed	<pre>med from 4D to 2D \n{pca2.components_}\n") to 2D: \n{pca2.explaine a2.score(X)}")</pre>	
Out[14]:	[ 0.5804131 -0.02449161 0.142: [ 0.56485654 -0.06694199 0.6342]  Eigenvalues: [ 2.93808505 0.9201649 0.1477418]  PCA for transformation  #Apply PCA to transform iris date pca2 = PCA (n_components=2) principalComponents2 = pca2.fit_principalDf2 = pd.DataFrame (data finalDf2 = pd.concat([principalIf finalDf2.head(5)]  principal component 1 principal compo  -2.684126 0.3  principal component 1 principal compo  -2.714142 -0.7  2 -2.888991 -0.7  3 -2.745343 -0.3  4 -2.728717 0.3  #check the statistical values of print(f"components_ in the data print(f"explained_variance_ in the print(f"score in the data transform [ [ 0.36138659 -0.08452251 0.8566 [  0.65658877 0.73016143 -0.1733]  explained_variance_ in the data transformed to -2.699796510675664  #Visualize the data into 2D fig = plt.figure(figsize = (12,8 ax = fig.add_subplot(1,1,1))	12637 -0.80144925] 27274 0.52359713]]  82 0.02085386]  into 2D data a  taset into 2D for vi  transform(X) a = principalCompone 0f2, df[['target']]]  nent 2 target 819397 0 177001 0 144949 0 818299 0 826755 0  f the model transform transformed to 2D: the data transformed formed to 2D: \n(pc) med to 2D: 67061 0.3582892 ] 37266 -0.07548102]] transformed to 2D: 2D:  2D:	<pre>med from 4D to 2D \n{pca2.components_}\n") to 2D: \n{pca2.explaine a2.score(X)}")</pre>	
Out[14]:	[ 0.5804131	into 2D data a  into 2D data a  taset into 2D for vi  transform(X)  a = principalCompone  of2, df[['target']]]  nent 2 target  319397	<pre>med from 4D to 2D \n{pca2.components_}\n") to 2D: \n{pca2.explaine a2.score(X)}")</pre>	
Out[14]:	Eigenvalues: [2.93808505 0.9201649 0.1477416  PCA for transformation  #Apply PCA to transform iris data pca2 = PCA(n_components=2) principalComponents2 = pca2.fit_principalDf2 = pd.DataFrame(data finalDf2.head(5)  principal component1 principal components2 = pca2.fit_principalDf2 = pd.concat([principalIf finalDf2.head(5)]  principal component1 principal component2 = pca2.fit_principalDf2.head(5)  principal component1 principal component2 = pca2.fit_principalDf2.head(5)  #check the statistical values of print(f"components_ in the data print(f"explained_variance_ in the print(f"explained_variance_ in the data transform [0.36138659 -0.08452251 0.8566 [0.65658877 0.73016143 -0.1733]  explained_variance_ in the data transform [1.036138659 -0.08452251 0.8566 [0.65658877 0.73016143 -0.1733]  explained_variance_ in the data transformed to -2.699796510675664  #Visualize the data into 2D fig = plt.figure(figsize = (12,8 ax = fig.add_subplot(1,1,1) ax.set_xlabel('Principal Components_xset_ylabel('Principal Components_xset_ylabel('Principal Components_xset_ylabel('Principal Components_xset_ylabel('Principal Components_xset_ylabel('Principal Components_xset_ylabel('Principal Components_ylabel('Principal Components_ylabel('Princ	into 2D data a  into 2D data a  taset into 2D for vi  transform(X)  a = principalCompone.  of2, df[['target']]]  nent 2 target  319397	med from 4D to 2D  \n{pca2.components_}\n")  to 2D : \n{pca2.explained a2.score(X)}")  -virginica']	
Out[14]:	[ 0.5804131 -0.02449161 0.142: [ 0.56485654 -0.06694199 0.6342  Eigenvalues: [2.93808505 0.9201649 0.1477418  PCA for transformation  #Apply PCA to transform iris date pca2 = PCA (n_components2) principalComponents2 = pca2.fit_principalDf2 = pd.DataFrame (data finalDf2 = pd.concat([principalIf finalDf2.head(5))  principal component 1 principal component 1 principal component 2 = c.884126 0.3  principal component 1 principal component 3 = c.745343 -0.3  4 -2.728717 0.3  #check the statistical values of print(f"components_ in the data print(f"explained_variance_ in the print(f"score in the data transform [[ 0.36138659 -0.08452251 0.8566 [ 0.65658877 0.73016143 -0.1733]  explained_variance_ in the data f[ 0.36138659 -0.08452251 0.8566 [ 0.65658877 0.73016143 -0.1733]  explained_variance_ in the data f[ 0.269796510675664 ]  #Visualize the data into 2D fig = plt.figure(figsize = (12, 8 as = fig.add_subplot(1, 1, 1) ax.set_xlabel('Principal Component ax.set_ylabel('Principal Component ax	into 2D data a  into 2D data a  taset into 2D for vi  transform(X)  a = principalCompone.  of 2, df[['target']]]  nent 2 target  319397	med from 4D to 2D  \n{pca2.components_}\n")  to 2D : \n{pca2.explained a2.score(X)}")  -virginica']	
Out[14]:	[ 0.5804131 -0.02449161 0.142: [ 0.56485654 -0.06694199 0.6342: [ 2.93808505 0.9201649 0.1477418]  PCA for transformation  #Apply PCA to transform iris dat pca2 = PCA(n_components=2) principalComponents2 = pca2.fit_principalDf2 = pd.DataFrame(datafinalDf2 = pd.concat([principalIfinalDf2.head(5))  # principal component1 principal components	into 2D data a  into 2D data a  taset into 2D for vi  transform(X)  a = principalCompone.  of 2, df[['target']]]  nent 2 target  319397	med from 4D to 2D  \n{pca2.components_}\n")  to 2D: \n{pca2.explaine a2.score(X)}")  -virginica']  al component 1'] ipal component 2']	
Out[14]:	[ 0.5804131 -0.02449161 0.142: [ 0.56485654 -0.06694199 0.6342  Eigenvalues: [2.93808505 0.9201649 0.1477418  PCA for transformation  #Apply PCA to transform iris date pca2 = PCA (n_components2) principalComponents2 = pca2.fit_principalDf2 = pd.DataFrame (data finalDf2 = pd.concat([principalIf finalDf2.head(5))  principal component 1 principal component 1 principal component 2 = c.884126 0.3  principal component 1 principal component 3 = c.745343 -0.3  4 -2.728717 0.3  #check the statistical values of print(f"components_ in the data print(f"explained_variance_ in the print(f"score in the data transform [[ 0.36138659 -0.08452251 0.8566 [ 0.65658877 0.73016143 -0.1733]  explained_variance_ in the data f[ 0.36138659 -0.08452251 0.8566 [ 0.65658877 0.73016143 -0.1733]  explained_variance_ in the data f[ 0.269796510675664 ]  #Visualize the data into 2D fig = plt.figure(figsize = (12, 8 as = fig.add_subplot(1, 1, 1) ax.set_xlabel('Principal Component ax.set_ylabel('Principal Component ax	into 2D data a  into 2D data a  taset into 2D for vi  transform(X)  a = principalCompone  of 2, df[['target']]]  nent 2 target  319397	med from 4D to 2D  \n{pca2.components_}\n")  to 2D: \n{pca2.explaine a2.score(X)}")  -virginica']  al component 1'] ipal component 2']	
Out[14]:	[ 0.5804131 -0.02449161 0.142: [ 0.56485654 -0.06694199 0.6342: [ 2.93808505 0.9201649 0.1477418]  PCA for transformation  #Apply PCA to transform iris dat pca2 = PCA(n_components=2) principalComponents2 = pca2.fit_principalDf2 = pd.DataFrame(datafinalDf2 = pd.concat([principalIfinalDf2.head(5))  # principal component1 principal components	into 2D data a  into 2D data a  taset into 2D for vi  transform(X)  a = principalCompone  of 2, df[['target']]]  nent 2 target  319397	med from 4D to 2D  \n{pca2.components_}\n")  to 2D: \n{pca2.explaine a2.score(X)}")  -virginica']  al component 1'] ipal component 2']	
Out[14]:	Eigenvalues: [2.93808505 0.9201649 0.1477416  PCA for transformation  #Apply PCA to transform iris data pca2 = PCA (n_components2 = pca2.fit_principalComponents2 = pca2.fit_principalDf2 = pd.concat([principalIfinalDf2.head(5))  principalcomponent 1 principalcomponents2 = pca2.fit_principalComponent 2 = pca2.fit_principalDf2.head(5)  principalcomponent 1 principalcompo  -2.684126 0.3  -2.745343 -0.3  -2.745343 -0.3  4 -2.728717 0.3  #check the statistical values of print(f"components_ in the data print(f"explained_variance_ in the print(f"score in the data transfors [0.36138659 -0.0845251 0.8566 [0.6565887 0.73016143 -0.173]  explained_variance_ in the data f(4.22824171 0.24267075]  score in the data transformed to -2.699796510675664  #Visualize the data into 2D fig = plt.figure(figsize = (12.6 ax = fig.add_subplot(1,1,1)) ax.set_xlabel('Principal Components ax.set_ylabel('Principal Components ylabel)  ###################################	into 2D data a  into 2D data a  taset into 2D for vi  transform(X)  a = principalCompone  of 2, df[['target']]]  nent 2 target  319397	med from 4D to 2D  \n{pca2.components_}\n")  to 2D: \n{pca2.explaine a2.score(X)}")  -virginica']  al component 1'] ipal component 2']	
Out[14]:	Eigenvalues: [2.93808505 0.9201649 0.1477416  PCA for transformation  #Apply PCA to transform iris data pca2 = PCA (n_components2 = pca2.fit_principalComponents2 = pca2.fit_principalDf2 = pd.concat([principalIfinalDf2.head(5))  principalcomponent 1 principalcomponents2 = pca2.fit_principalComponent 2 = pca2.fit_principalDf2.head(5)  principalcomponent 1 principalcompo  -2.684126 0.3  -2.745343 -0.3  -2.745343 -0.3  4 -2.728717 0.3  #check the statistical values of print(f"components_ in the data print(f"explained_variance_ in the print(f"score in the data transfors [0.36138659 -0.0845251 0.8566 [0.6565887 0.73016143 -0.173]  explained_variance_ in the data f(4.22824171 0.24267075]  score in the data transformed to -2.699796510675664  #Visualize the data into 2D fig = plt.figure(figsize = (12.6 ax = fig.add_subplot(1,1,1)) ax.set_xlabel('Principal Components ax.set_ylabel('Principal Components ylabel)  ###################################	into 2D data a  into 2D data a  taset into 2D for vi  transform(X)  a = principalCompone  of 2, df[['target']]]  nent 2 target  319397	med from 4D to 2D  \n{pca2.components_}\n")  to 2D: \n{pca2.explaine a2.score(X)}")  -virginica']  al component 1'] ipal component 2']	
Out[14]:	Eigenvalues: [2.93808505 0.9201649 0.1477416  PCA for transformation  #Apply PCA to transform iris data pca2 = PCA (n_components2 = pca2.fit_principalComponents2 = pca2.fit_principalDf2 = pd.concat([principalIfinalDf2.head(5))  principalcomponent 1 principalcomponents2 = pca2.fit_principalComponent 2 = pca2.fit_principalDf2.head(5)  principalcomponent 1 principalcompo  -2.684126 0.3  -2.745343 -0.3  -2.745343 -0.3  4 -2.728717 0.3  #check the statistical values of print(f"components_ in the data print(f"explained_variance_ in the print(f"score in the data transfors [0.36138659 -0.0845251 0.8566 [0.6565887 0.73016143 -0.173]  explained_variance_ in the data f(4.22824171 0.24267075]  score in the data transformed to -2.699796510675664  #Visualize the data into 2D fig = plt.figure(figsize = (12.6 ax = fig.add_subplot(1,1,1)) ax.set_xlabel('Principal Components ax.set_ylabel('Principal Components ylabel)  ###################################	into 2D data a  into 2D data a  taset into 2D for vi  transform(X)  a = principalCompone  of 2, df[['target']]]  nent 2 target  319397	med from 4D to 2D  \n{pca2.components_}\n")  to 2D: \n{pca2.explaine a2.score(X)}")  -virginica']  al component 1'] ipal component 2']	
Out[14]:		into 2D data a  into 2D data a  taset into 2D for vi  transform(X)  a = principalCompone  of 2, df[['target']]]  nent 2 target  319397	med from 4D to 2D  \n{pca2.components_}\n")  to 2D: \n{pca2.explaine a2.score(X)}")  -virginica']  al component 1'] ipal component 2']	
Out[14]:	0.5804131   -0.02449161   0.142:   (0.56485654 -0.06694199   0.634:   Eigenvalues:   (2.93808505 0.9201649   0.1477414	into 2D data a  into 2D data a  taset into 2D for vi  transform(X)  a = principalCompone  of 2, df[['target']]]  nent 2 target  319397	med from 4D to 2D  \n{pca2.components_}\n")  to 2D: \n{pca2.explaine a2.score(X)}")  -virginica']  al component 1'] ipal component 2']	
Out[14]:	0.5804131 -0.02449161	into 2D data a  into 2D data a  taset into 2D for vi  transform(X)  a = principalCompone  of 2, df[['target']]]  nent 2 target  319397	med from 4D to 2D  \n{pca2.components_}\n")  to 2D: \n{pca2.explaine a2.score(X)}")  -virginica']  al component 1'] ipal component 2']	
Out[14]:	0.5804131 -0.02449161	into 2D data a  into 2D data a  taset into 2D for vi  transform(X)  a = principalCompone  of 2, df[['target']]]  nent 2 target  319397	med from 4D to 2D \n{pca2.components_}\n")  to 2D : \n{pca2.explained a2.score(X)}")  ent PCA	
Out[14]:	C.5804131 -0.02449161 0.142:   (0.56485654 -0.06694199 0.634:   Eigenvalues:   (2.93808505 0.9201649 0.147741:   PCA for transform iris data pea2 = PCA (n_components=2) principalComponents2 = pea2.fit principalComponents2 = pea2.fit principalComponents2 = pea2.fit principalComponents2 = pea2.fit principalComponent	12637 -0.80144925    27274	med from 4D to 2D \n{pca2.components_}\n")  to 2D : \n{pca2.explaine a2.score(X)}")  al component 1'] ipal component 2']  ent PCA	ed_variance_}\n")  • 0 • 1 • 2 • • • • • • • • • • • • • • • • • •
Out[14]:  In [15]:	Eigenvalues: [2.93808505 0.9201649 0.1477416  PCA for transformation  #Apply PCA to transform iris data pca2 = PCA (n_components=2) principalComponents2 = pca2.fit principalDf2 = pd.DataFrame(data finalDf2 = pd.concat([principalI finalDf2.head(5)]  principal component 1 principal components = components = component =	into 2D data a  into 2D data a  taset into 2D for vi  transform(X)  a = principalCompone  fig., df[['target']]]  nent 2 target  319397	med from 4D to 2D \(\text{\te\	ed_variance_}\n")  • 0 • 1 • 2 • • • • • • • • • • • • • • • • • •
Out[14]:  In [15]:	0.5804131 -0.02449161	12637 -0.80144925    27274	med from 4D to 2D \(\text{\te\	ed_variance_)\n")  • 0 • 1 • 2 • • • • • • • • • • • • • • • • • •
In [15]:  In [16]:	0.5804131 -0.02449161 0.142:   (0.56485654 -0.06694199 0.6342:   Eigenvalues:   (2.93808505 0.9201649 0.1477414   PCA for transform iris date poa2 = PCA(n_components=2)   principalComponents2 = pca2.fit_principalDf2 = pd.DataFrame(date finalDf2 = pd.Concat([principalIf inalDf2.head(5))   principal component 1 principal components2 = pca2.fit_principalDf2.head(5)   principal component 1 principal components2 = pca2.fit_principalCf2.head(5)   principal component 1 principal components2 = pca2.fit_principalCf3	12637 -0.80144925    27274	med from 4D to 2D \(\text{\tex	ed_variance_)\n")  • 0 • 1 • 2 • • • • • • • • • • • • • • • • • •
In [15]:  In [16]:	0.5804131	into 2D data a  into 2D data a  taset into 2D for vi  transform(X) a = principalCompone of 2, df[['target']]]  nent 2 target 319397	med from 4D to 2D \( \text{\tint{\text{\tikt{\tex{\tex	ed_variance_)\n")  • 0 • 1 • 2 • • • • • • • • • • • • • • • • • •
In [15]:  In [16]:	0.5804131 -0.02449161 0.142:	into 2D data a  into 2D data a  into 2D data a  into 2D for vi  transform(X) a = principalCompone of 2, df[['target']]]  nent 2 target  319397	med from 4D to 2D \( \text{N\foatset} \) \( \	ed_variance_)\n")  • 0 • 1 • 2 • • • • • • • • • • • • • • • • • •
In [15]:  In [16]:	0.5804131 -0.02449161 0.142:	12637 -0.80144925    27274	suallization  nts2, columns = ['princip, axis = 1)  med from 4D to 2D \n\{pca2.components_\n''\}  to 2D : \n\{pca2.explains a2.score(X)}'')  al component 1'] ipal component 2']  pont PCA  ent PCA  possible to the princip are as a second and a second an	ed_variance_}\n")  al component 1', 'principal compone
<pre>In [17]:</pre> Out[17]:	0.5804131 -0.02449161 0.142:	12637 -0.80144925    27274	med from 4D to 2D \( \text{\tex{\tex	ed_variance_}\n")  al component 1', 'principal compone
<pre>In [17]:</pre> Out[17]:	C.5804131 -0.02449161 0.142:   C.56485654 -0.06694199 0.634;   Eigenvalues:   (2.93808505 0.9201649 0.147741)    PCA for transformation   FApply FCA to transform iris dat pca2 = PCA(n. components=2) principalComponents2 = pca2.fit principalIf = pd. DataFrame(data finalDf2 = pd. concat([principalIf inalDf2.head(5))   Principal component1 principal components2 = pca2.fit principal component1 principal component3	into 2D data a  into 2D data a  into 2D data a  into 2D data a  into 2D for vi  transform(X)  a = principalCompone if2, df[['target']]]  nent 2 target  39397	med from 4D to 2D \( \text{\tex{\tex	ed_variance_}\n")  al component 1', 'principal compone
<pre>In [17]:</pre> Out[17]:	0.5804131 -0.02449161 0.142:     0.56485654 -0.06694199 0.634:     Eigenvalues:   (2.93808505 0.9201649 0.1477418     (2.93808505 0.9201649 0.1477418     PCA for transformation     #Apply FCA to transform iris dat pca2 = FCA(n components=2)     principalComponents2 = pca2.fit principalComponents2 = pca2.fit principalComponent1     principal component1   principal component2     principal component1   principal component3     1	12637 -0.80144925    12637 -0.80144925    127274	med from 4D to 2D \( \text{\tex{\tex	ed_variance_}\n")  al component 1', 'principal compone
<pre>In [17]:</pre> Out[17]:	0.5804131 -0.02449161 0.142:	into 2D data a  into 2D data a  into 2D data a  taset into 2D for vi  transform(X)  a = principalCompone  of 2, df[['target']]]  ment 2 target  319397	med from 4D to 2D \( \text{\tex{\tex	ed_variance_}\n")  al component 1', 'principal compone
<pre>In [17]:</pre> Out[17]:	0.5804131 -0.02449161 0.142:     0.56485654 -0.06694199 0.634:     Eigenvalues:     (2.93808505 0.9201649 0.147741)     PCA for transformation     FApply PCA to transform iris dat pea2 = PCA(In_components=2)	12637 - 0.80144925    27274	med from 4D to 2D \n(pca2.components_\)\n")  med from 4D to 2D \n(pca2.components_\)\n")  to 2D : \n(pca2.explaine a2.score(X))")  al component 1'] ipal component 2']  ent PCA  ponent 1  suallization  nts2, columns = ['princip a2.score(X)]")  med from 4D to 2D \n(pca2.components_\)\n")	ed_variance_}\n")  al component 1', 'principal compone

color 2

1.5

0.5

import numpy as np
import pandas as pd

In [2]:

from pylab import \*
import pylab as pl
%matplotlib inline
import seaborn as sns

import matplotlib.pyplot as plt
from sklearn import datasets
import plotly.express as px

from itertools import cycle

iris = datasets.load\_iris()

#import iris dataset from sklearn

 $\textbf{from} \text{ sklearn.} \\ \text{decomposition } \textbf{import} \text{ PCA}$ 

 $\textbf{from} \ \text{sklearn.preprocessing} \ \textbf{import} \ \text{StandardScaler}$