pr_

October 20, 2019

1 Data Set Information:

It is a case of supervised learning with the use of Receiver Operating Characteristic (ROC) to select the minimal set of attributes preserving or increasing predictability of the data.

2 Attribute Information:

3 D = decision attribute (D) with values 0 (unhappy) and 1 (happy)

4 X1 = the availability of information about the city services

5 X2 = the cost of housing

6 X3 = the overall quality of public schools

7 X4 = your trust in the local police

8 X5 = the maintenance of streets and sidewalks

9 X6 = the availability of social community events

10 Importing required packeges

[19]:
```
import pandas as pd
import numpy as np
import seaborn as sns
from pandas.plotting import scatter_matrix
import matplotlib.pyplot as plt
from matplotlib import cm
```

11 reading data and displaying top 5 rows

[88]:
```
data = pd.read_csv("SomervilleHappinessSurvey.csv")
data.head()
labels = data["D"]
```

[253]:
```
data.head()
```

[253]:

	D	X1	X2	X3	X4	X5	X6
0	0	3	3	3	4	2	4
1	0	3	2	3	5	4	3
2	1	5	3	3	3	3	5
3	0	5	4	3	3	3	5
4	0	5	4	3	3	3	5

12 all data description

[3]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 143 entries, 0 to 142
Data columns (total 7 columns):
D     143 non-null int64
X1    143 non-null int64
X2    143 non-null int64
X3    143 non-null int64
X4    143 non-null int64
X5    143 non-null int64
X6    143 non-null int64
dtypes: int64(7)
memory usage: 7.9 KB
```

[76]:
```
data.dtypes
```

[76]:
```
D     int64
X1    int64
X2    int64
X3    int64
X4    int64
X5    int64
X6    int64
dtype: object
```

13 missing value

[72]:
```
data.isna().sum()
```

[72]:
```
D     0
X1    0
X2    0
X3    0
X4    0
X5    0
X6    0
dtype: int64
```

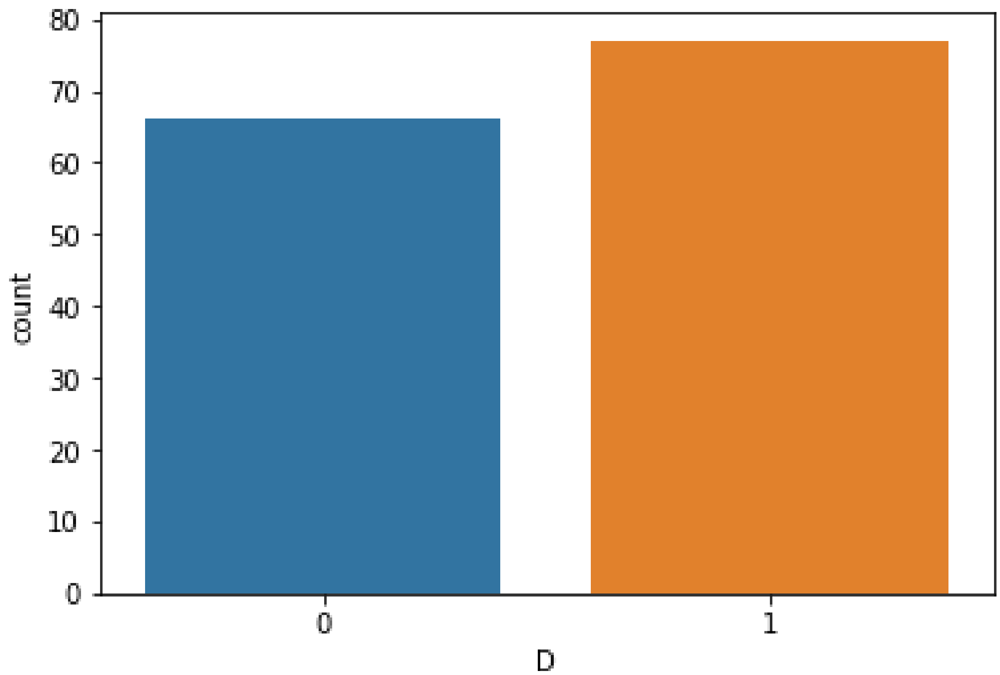
14 class of dataset

[47]:
```
cl = data.groupby("D")
```

[73]:
```
cl.size()
```

[73]:
```
D
0    66
1    77
dtype: int64
```

[14]:
```
sns.countplot(data['D'],label="Count")
plt.show() #0 -> unhappy(66) 1 -> happy(77)
```



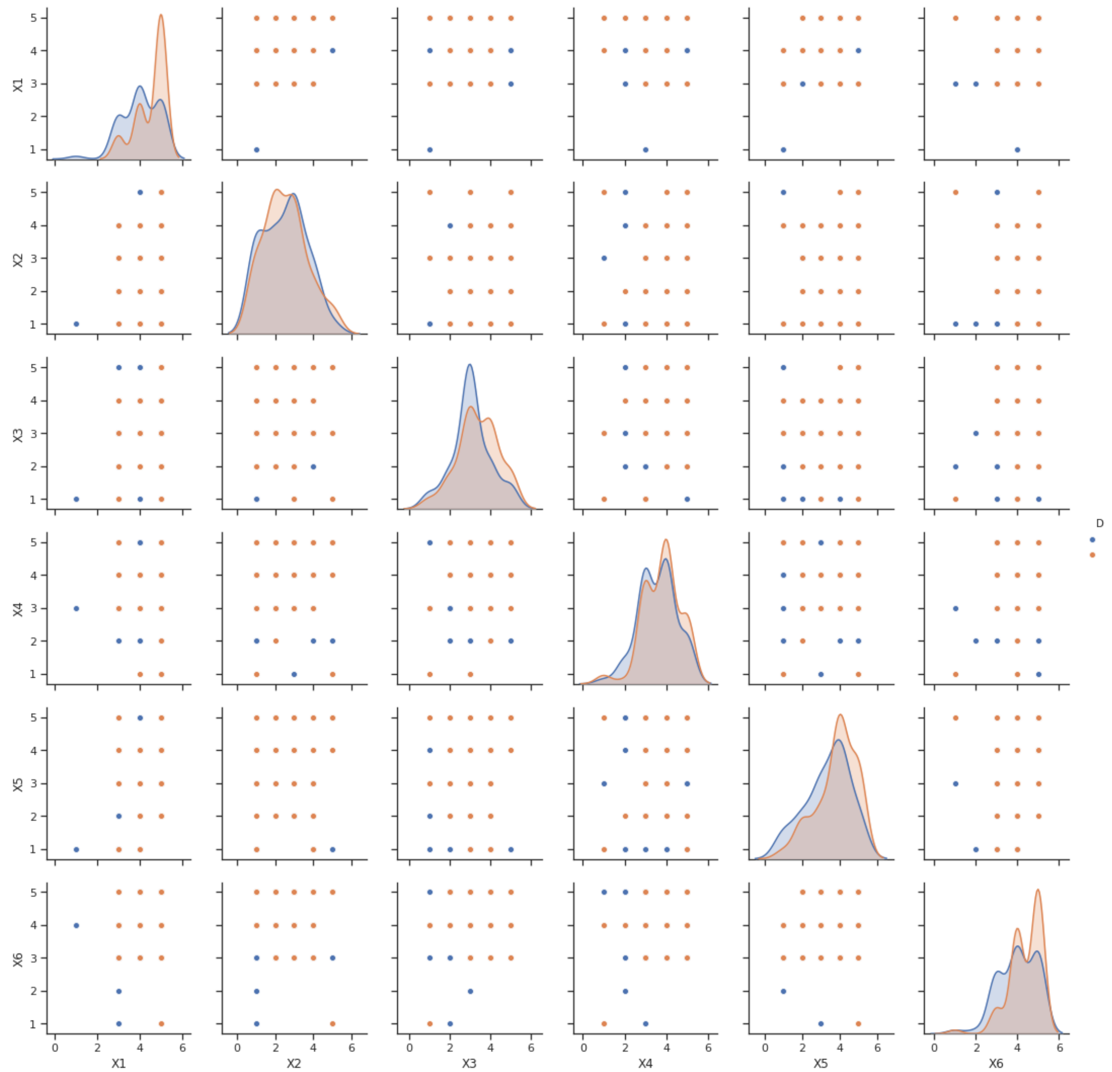
15 Scatter Matrix

[238]:
```
data.head(5)
```

[238]:

	D	X1	X2	X3	X4	X5	X6
0	0	3	3	3	4	2	4
1	0	3	2	3	5	4	3
2	1	5	3	3	3	3	5
3	0	5	4	3	3	3	5
4	0	5	4	3	3	3	5

[239]:
```
sns.set(style="ticks", color_codes=True)
sns.pairplot(data, hue='D', vars=["X1", "X2","X3","X4","X5","X6"])
plt.show()
```

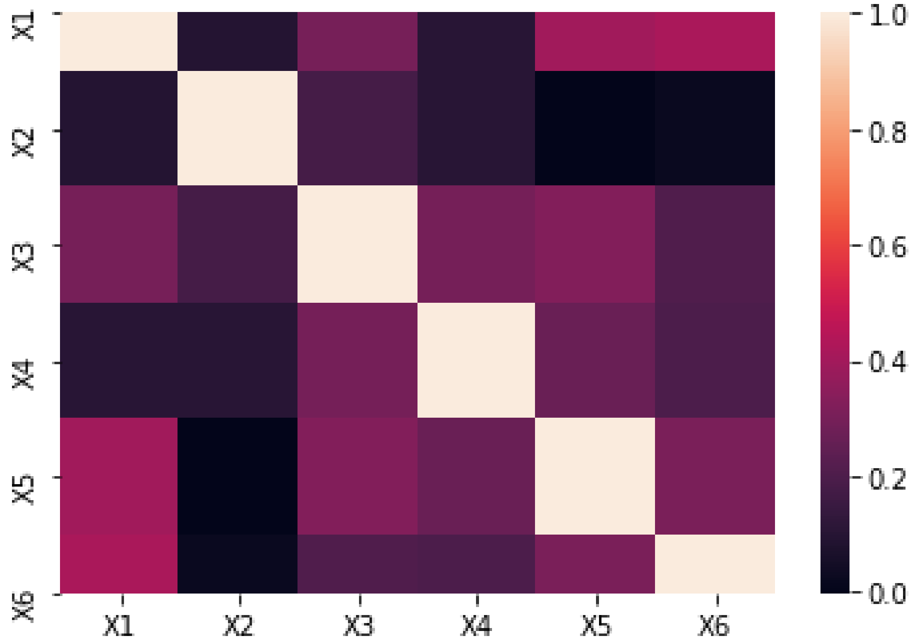


16 Correlation matrix

[34]:
```
data_f = data.drop("D" , axis = 1)
print(data_f.corr())
ax = sns.heatmap(data_f.corr())
```

```
          X1        X2        X3        X4        X5        X6
X1  1.000000  0.092676  0.301971  0.104378  0.399203  0.417521
X2  0.092676  1.000000  0.181081  0.107432 -0.002141  0.024546
X3  0.301971  0.181081  1.000000  0.298898  0.329874  0.207006
X4  0.104378  0.107432  0.298898  1.000000  0.269420  0.199151
X5  0.399203 -0.002141  0.329874  0.269420  1.000000  0.307402
```

```
X6  0.417521  0.024546  0.207006  0.199151  0.307402  1.000000
```



17 data Statistical Summary

18 outlier

[192]:
```
Q1 = data.quantile(0.25)
Q3 = data.quantile(0.75)
IQR = Q3 - Q1
outl = ((data < (Q1 - 1.5 * IQR)) | (data > (Q3 + 1.5 * IQR))).sum()
# outl = outl.values
```

[193]:
```
des = data_f.describe()
sw = data_f.skew()
kr = data_f.kurtosis()
des = des.T
des.insert(8,"skewness" , sw , True)
des.insert(9 , "kurtosis",kr , True)
des.insert(10 , "Outlier",outl , True)
print(des)
```

```
    count      mean       std  min  25%  50%  75%  max  skewness  kurtosis  \
X1  143.0  4.314685  0.799820  1.0  4.0  5.0  5.0  5.0 -0.966144  0.682811
```

```
X2  143.0  2.538462  1.118155  1.0  2.0  3.0  3.0  5.0  0.285491 -0.612389
X3  143.0  3.265734  0.992586  1.0  3.0  3.0  4.0  5.0 -0.118415 -0.073144
X4  143.0  3.699301  0.888383  1.0  3.0  4.0  4.0  5.0 -0.468723  0.422877
X5  143.0  3.615385  1.131639  1.0  3.0  4.0  4.0  5.0 -0.675393 -0.288766
X6  143.0  4.216783  0.848693  1.0  4.0  4.0  5.0  5.0 -1.062909  1.392131

    Outlier
X1        1
X2        7
X3        7
X4        3
X5        8
X6        3
```

19 happy and unhappy class

[78]:
```
happy = data[data["D"] == 1]
unhappy = data[data["D"] == 0]
```

[138]:
```
print("Happy class sample data \n" ,happy.head())
print("unhappy class sample \n",unhappy.head())
```

```
Happy class sample data
     D  X1  X2  X3  X4  X5  X6
2   1   5   3   3   3   3   5
5   1   5   5   3   5   5   5
7   1   5   4   4   4   4   5
12  1   5   2   4   5   5   5
15  1   3   2   4   3   4   4
unhappy class sample
    D  X1  X2  X3  X4  X5  X6
0  0   3   3   3   4   2   4
1  0   3   2   3   5   4   3
3  0   5   4   3   3   3   5
4  0   5   4   3   3   3   5
6  0   3   1   2   2   1   3
```

[80]:
```
happy_f = happy.drop("D" , axis = 1 )
unhappy_f = unhappy.drop("D" , axis = 1)
```

20 class happy Statistical Summary

[198]:
```
Q1 = happy_f.quantile(0.25)
Q3 = happy_f.quantile(0.75)
IQR = Q3 - Q1
outl = ((happy_f < (Q1 - 1.5 * IQR)) | (happy_f > (Q3 + 1.5 * IQR))).sum()
# outl = outl.values
```

[199]:
```
des_h = happy_f.describe()
des_h = des_h.T
swh = happy_f.skew()
krh = happy_f.kurtosis()
des_h.insert(8,"skewness" , swh , True)
des_h.insert(9 , "kurtosis",krh , True)
des_h.insert(10 , "Outlier",outl , True)
print(des_h)
```

```
   count      mean       std  min  25%  50%  75%  max  skewness  kurtosis  \
X1  77.0  4.545455  0.679502  3.0  4.0  5.0  5.0  5.0 -1.202039  0.180847
X2  77.0  2.558442  1.117958  1.0  2.0  2.0  3.0  5.0  0.429804 -0.364991
X3  77.0  3.415584  1.004603  1.0  3.0  3.0  4.0  5.0 -0.281859 -0.167029
X4  77.0  3.792208  0.878660  1.0  3.0  4.0  4.0  5.0 -0.651823  1.069124
X5  77.0  3.831169  1.056342  1.0  3.0  4.0  5.0  5.0 -0.821443  0.034411
X6  77.0  4.389610  0.763576  1.0  4.0  5.0  5.0  5.0 -1.527673  3.760978

   Outlier
X1       0
X2       5
X3       3
X4       2
X5       0
X6       1
```

21 Class Unhappy

[200]:
```
Q1 = unhappy_f.quantile(0.25)
Q3 = unhappy_f.quantile(0.75)
IQR = Q3 - Q1
outl = ((unhappy_f < (Q1 - 1.5 * IQR)) | (unhappy_f > (Q3 + 1.5 * IQR))).sum()
# outl = outl.values
```

[201]:
```
des_uh = unhappy_f.describe()
des_uh = des_uh.T
swuh = unhappy_f.skew()
kruh = unhappy_f.kurtosis()
```

```
des_uh.insert(8,"skewness" , swuh , True)
des_uh.insert(9 , "kurtosis",kruh , True)
des_uh.insert(10 , "Outlier",outl , True)
print(des_uh)
```

```
    count      mean       std  min   25%  50%   75%  max  skewness  kurtosis  \
X1   66.0  4.045455  0.849105  1.0  3.25  4.0  5.00  5.0 -0.710011  0.861880
X2   66.0  2.515152  1.126498  1.0  2.00  3.0  3.00  5.0  0.127636 -0.877138
X3   66.0  3.090909  0.956392  1.0  3.00  3.0  3.75  5.0  0.031476  0.414186
X4   66.0  3.590909  0.894036  1.0  3.00  4.0  4.00  5.0 -0.282383  0.030042
X5   66.0  3.363636  1.171933  1.0  3.00  4.0  4.00  5.0 -0.518101 -0.509589
X6   66.0  4.015152  0.902857  1.0  3.00  4.0  5.00  5.0 -0.677083  0.381676

    Outlier
X1        0
X2        2
X3       10
X4        1
X5        6
X6        0
```

22 pre-proccessing

[108]:
```
from sklearn.preprocessing import StandardScaler

standardized_data = StandardScaler().fit_transform(data_f)
```

[111]:
```
standardized_data.std()
```

[111]: 0.9999999999999999

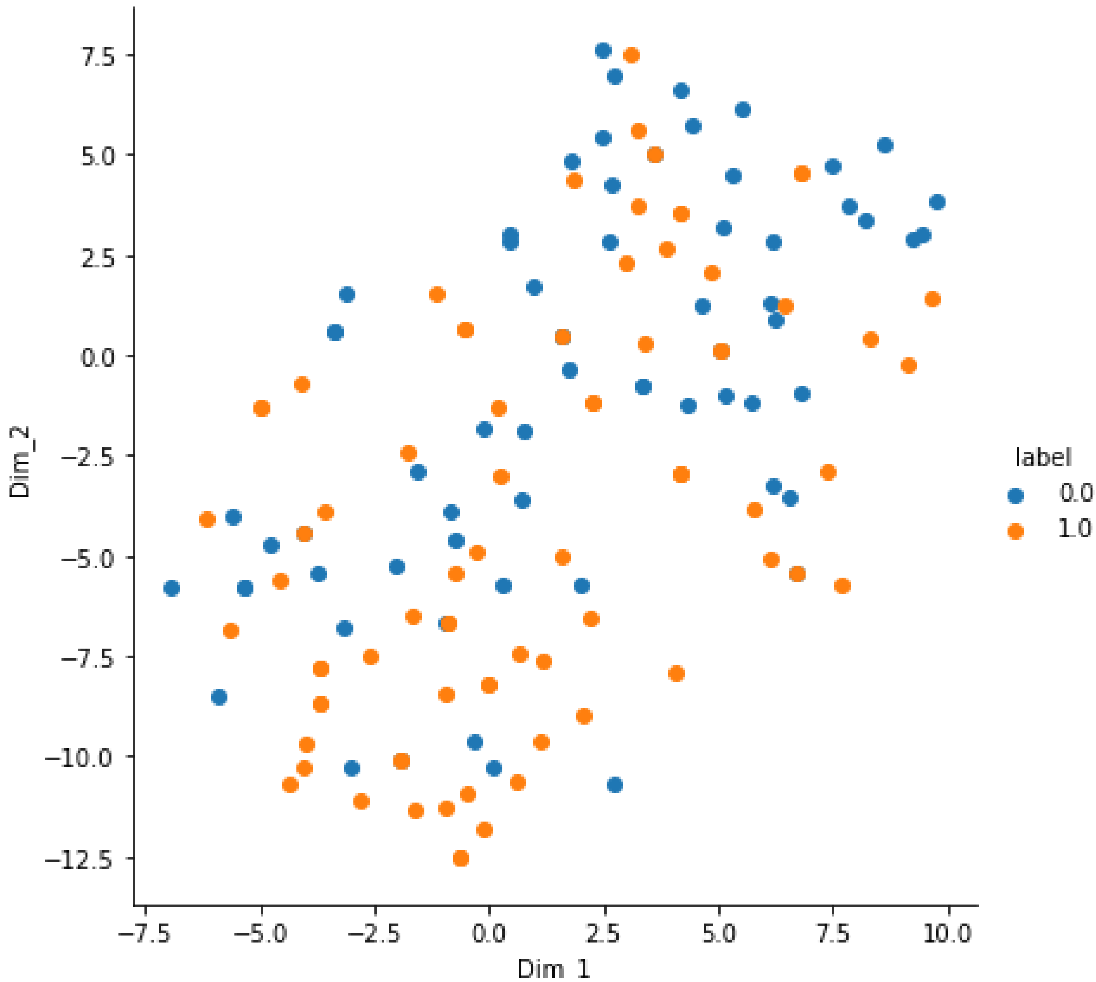
[123]:
```
int(standardized_data.mean())
```

[123]: 0

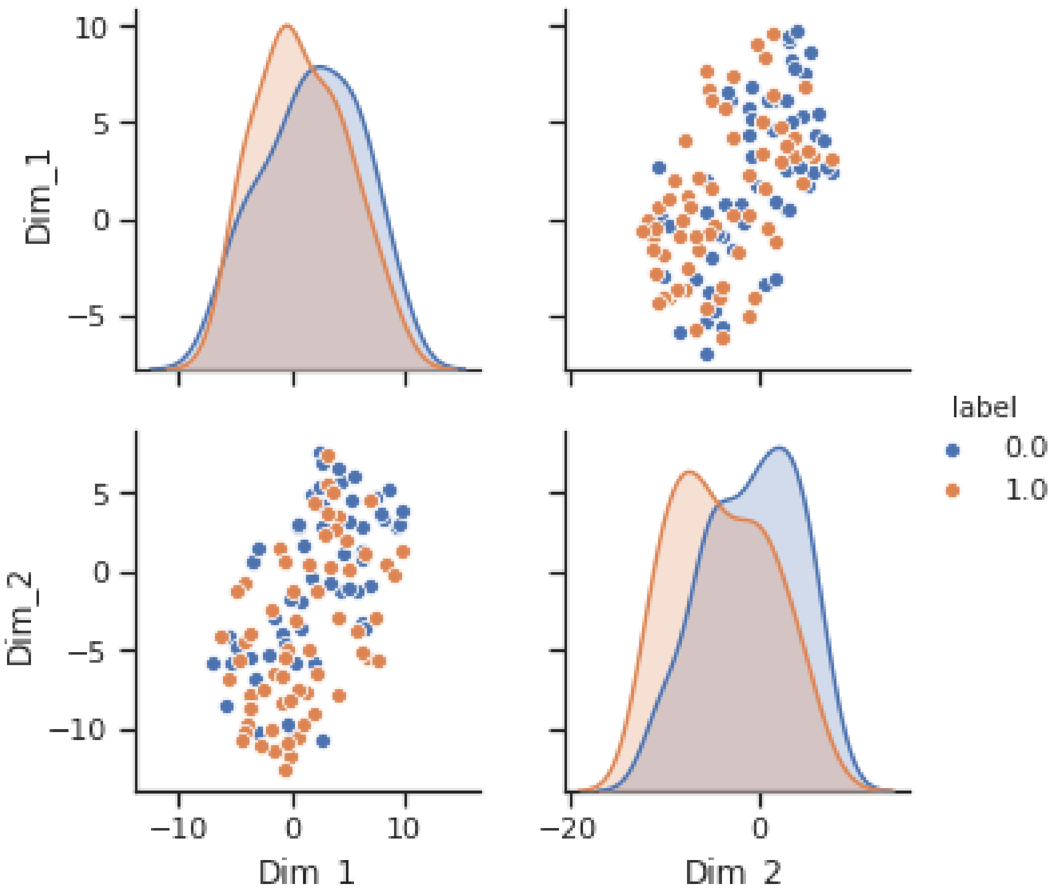
23 Dimension reduction t-SNE

[126]:
```
from sklearn.manifold import TSNE
from sklearn.preprocessing import StandardScaler
model = TSNE(n_components=2, random_state=0)
tsne_data = model.fit_transform(standardized_data)
tsne_data1 = np.vstack((tsne_data.T, labels)).T
tsne_df = pd.DataFrame(data=tsne_data1, columns=("Dim_1", "Dim_2", "label"))
```

```
# Ploting the result of tsne
sns.FacetGrid(tsne_df, hue="label", height=6).map(plt.scatter, 'Dim_1', 'Dim_2').
 ↪add_legend()
plt.show()
```



[252]:
```
sns.set(style="ticks", color_codes=True)
sns.pairplot(tsne_df, hue='label', vars=["Dim_1", "Dim_2"])
plt.show()
```



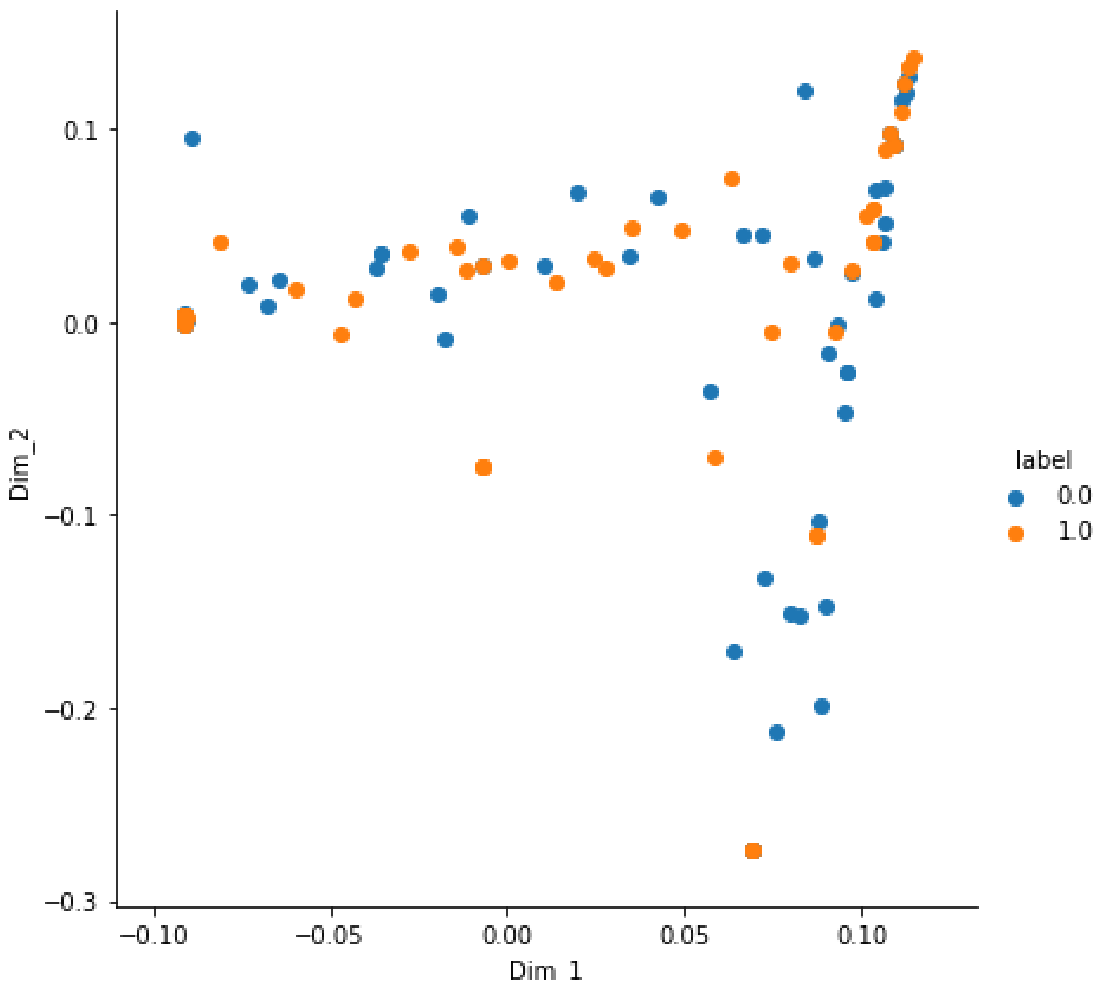
[153]:
```
cls1 = tsne_df[tsne_df["label"] == 1]
cls2 = tsne_df[tsne_df["label"] == 0]
print(cls1,cls2)
```

```
        Dim_1      Dim_2  label
2   -4.551148  -5.585553    1.0
5    1.187833  -7.597860    1.0
7   -0.034002  -8.209891    1.0
12  -0.938716 -11.304203    1.0
15   6.445683   1.238485    1.0
..        ...        ...    ...
134  2.981453   2.287377    1.0
135  7.706266  -5.697924    1.0
137  5.773182  -3.832636    1.0
139 -3.577958  -3.905294    1.0
140 -0.915704  -6.686235    1.0

[77 rows x 3 columns]          Dim_1     Dim_2  label
0    4.420581  5.751287    0.0
```

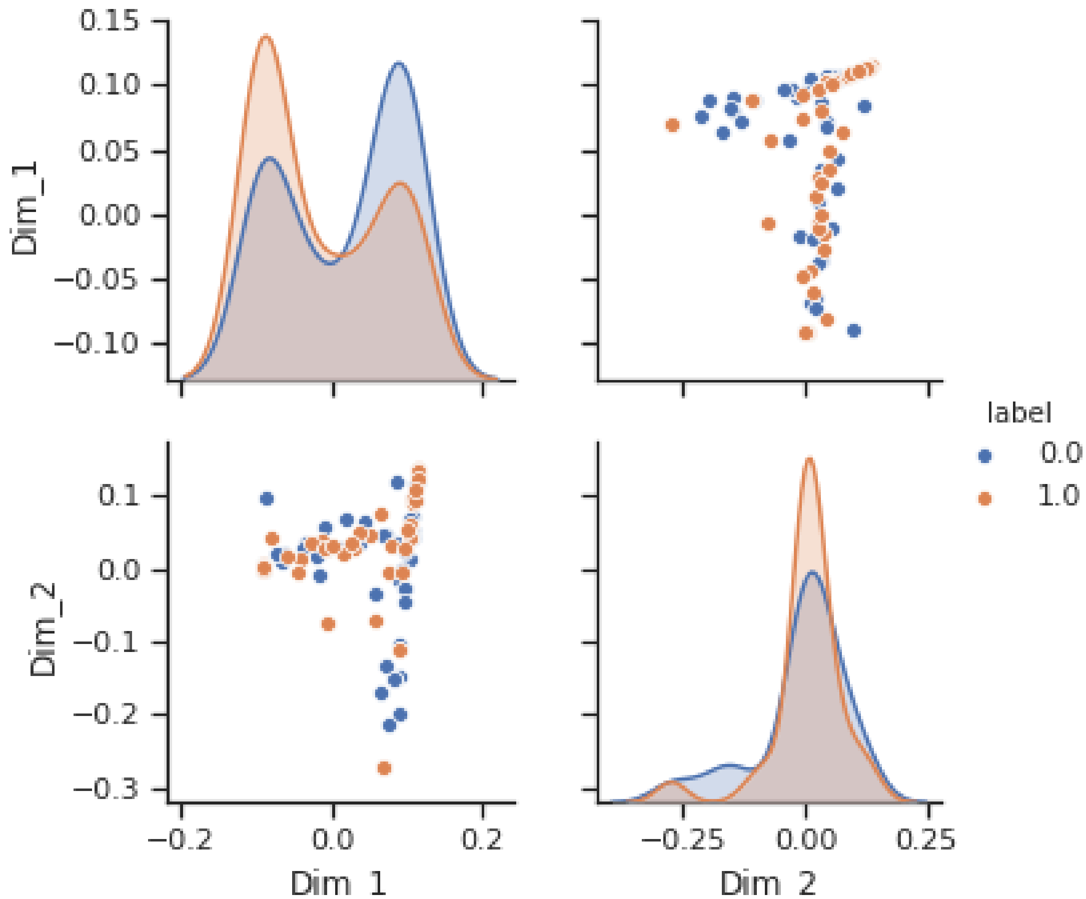
```
1     0.455209  2.833410    0.0
3    -5.333675 -5.805637    0.0
4    -5.333699 -5.805988    0.0
6     9.248189  2.890118    0.0
..         ...       ...    ...
131 -0.301861 -9.629655    0.0
136  2.744741  6.995521    0.0
138 -6.930221 -5.790246    0.0
141  0.737557 -3.610036    0.0
142  0.314275 -5.752754    0.0

[66 rows x 3 columns]
```

24 LLE

[127]:
```
from sklearn import manifold, datasets

#print("Computing LLE embedding")
LLE_data, err = manifold.locally_linear_embedding(standardized_data,␣
↪n_neighbors= 4,n_components=2)
#print("Done. Reconstruction error: %g" % err)

LLE_data1 = np.vstack((LLE_data.T, labels)).T
LLE_df = pd.DataFrame(data=LLE_data1, columns=("Dim_1", "Dim_2", "label"))

# Ploting the result of tsne
sns.FacetGrid(LLE_df, hue="label", height=6).map(plt.scatter, 'Dim_1', 'Dim_2').
↪add_legend()
plt.show()
```

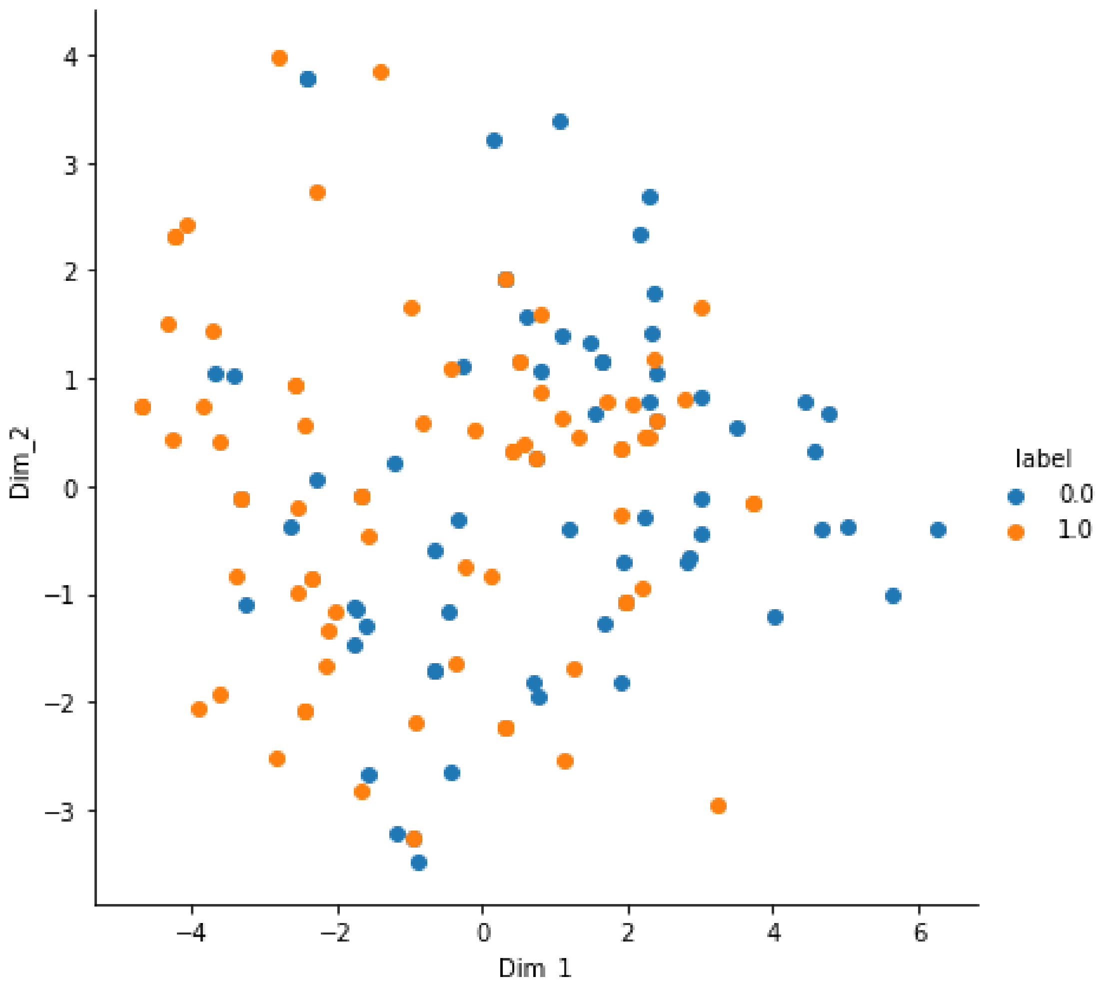


[251]:
```
sns.set(style="ticks", color_codes=True)
sns.pairplot(LLE_df, hue='label', vars=["Dim_1", "Dim_2"])
plt.show()
```

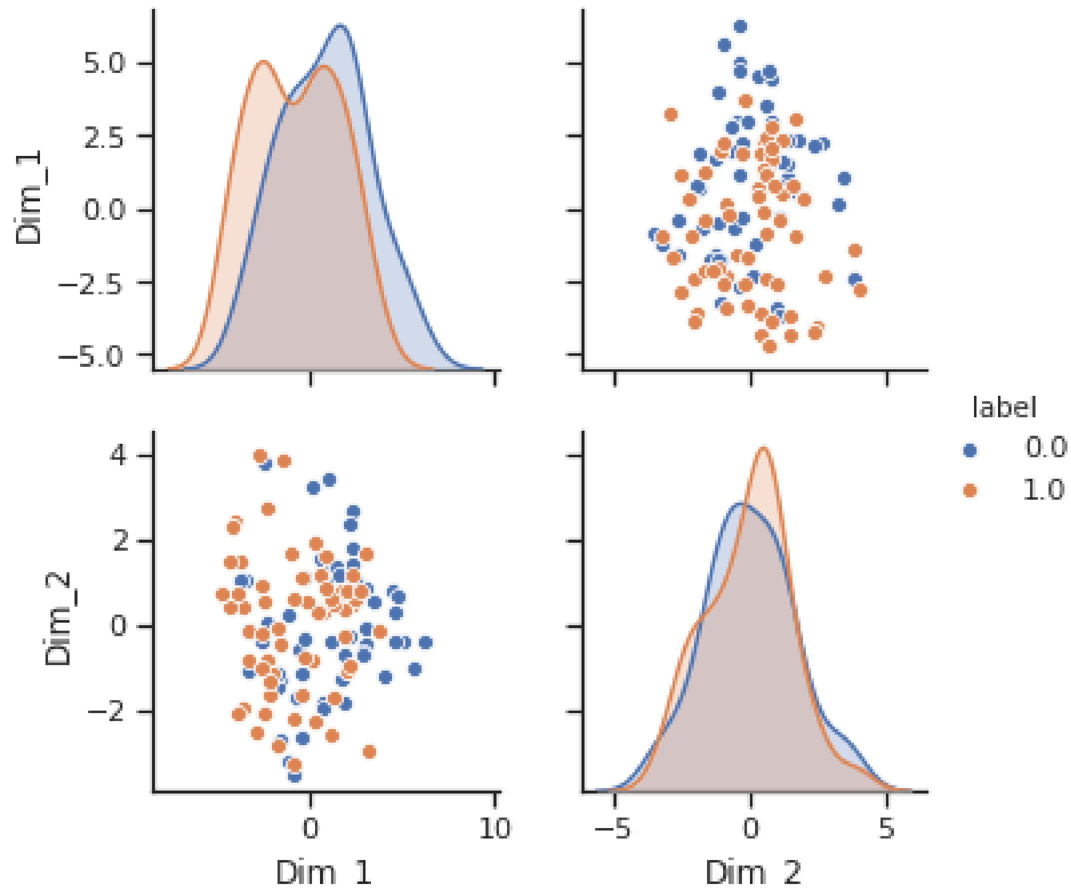


25 ISOmap

[149]:
```
iso_data = manifold.Isomap(n_neighbors=6, n_components=2).
 ↪fit_transform(standardized_data)



iso_data1 = np.vstack((iso_data.T, labels)).T
iso_df = pd.DataFrame(data=iso_data1, columns=("Dim_1", "Dim_2", "label"))
# Ploting the result of tsne
sns.FacetGrid(iso_df, hue="label", height=6).map(plt.scatter, 'Dim_1', 'Dim_2').
 ↪add_legend()
plt.show()
```



[250]:
```
sns.set(style="ticks", color_codes=True)
sns.pairplot(iso_df, hue='label', vars=["Dim_1", "Dim_2"])
plt.show()
```



26 PCA

[146]:
```
from sklearn.decomposition import PCA
from sklearn.preprocessing import normalize

pca = PCA(n_components = 2)

X_normalized = normalize(standardized_data)

# Converting the numpy array into a pandas DataFrame
X_normalized = pd.DataFrame(X_normalized)
X_principal = pca.fit_transform(X_normalized)
X_principal = pd.DataFrame(X_principal)
X_principal.columns = ['D1', 'D2']
```

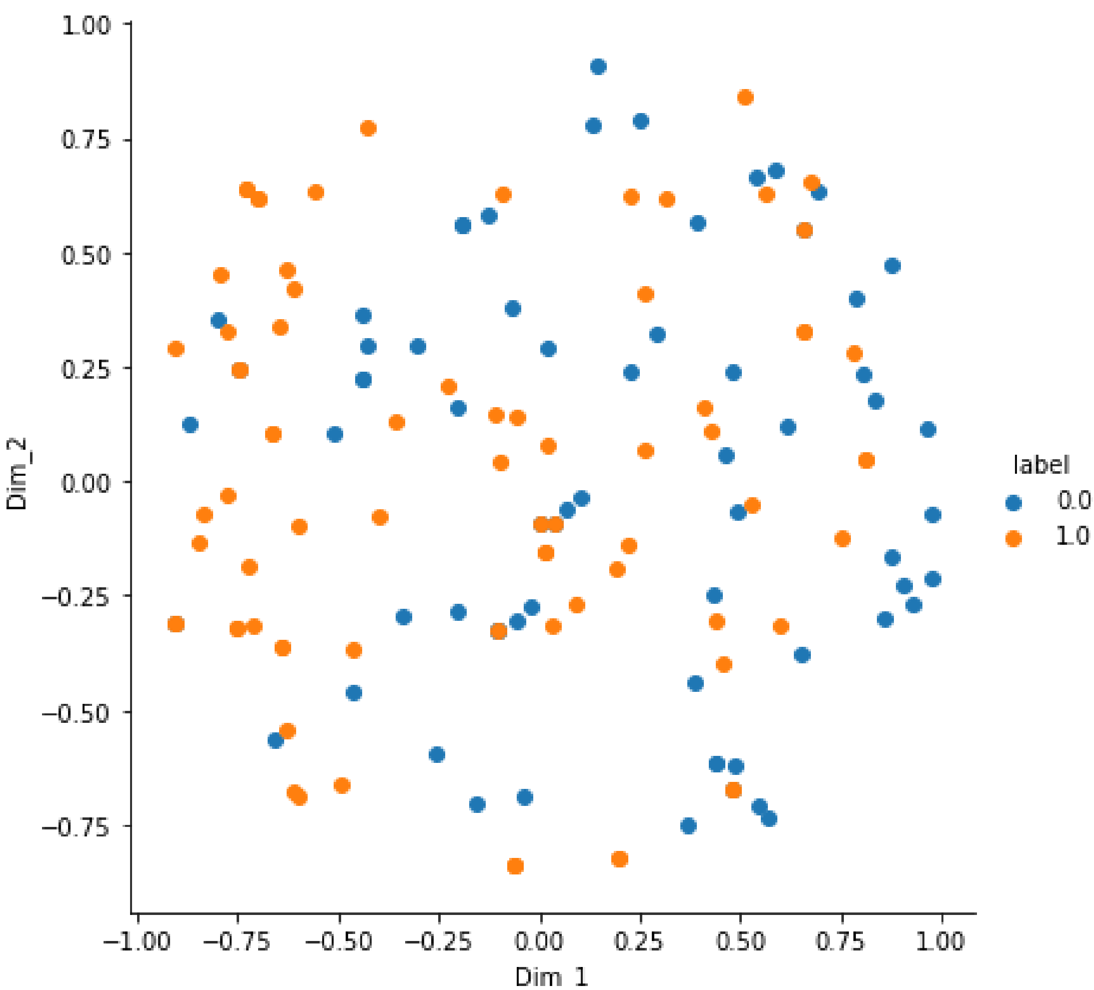
```
print(X_principal.head())
```

```
         D1        D2
0  0.789671  0.398983
1  0.494312 -0.063420
2 -0.226964  0.207125
3 -0.193307  0.562926
4 -0.193307  0.562926
```

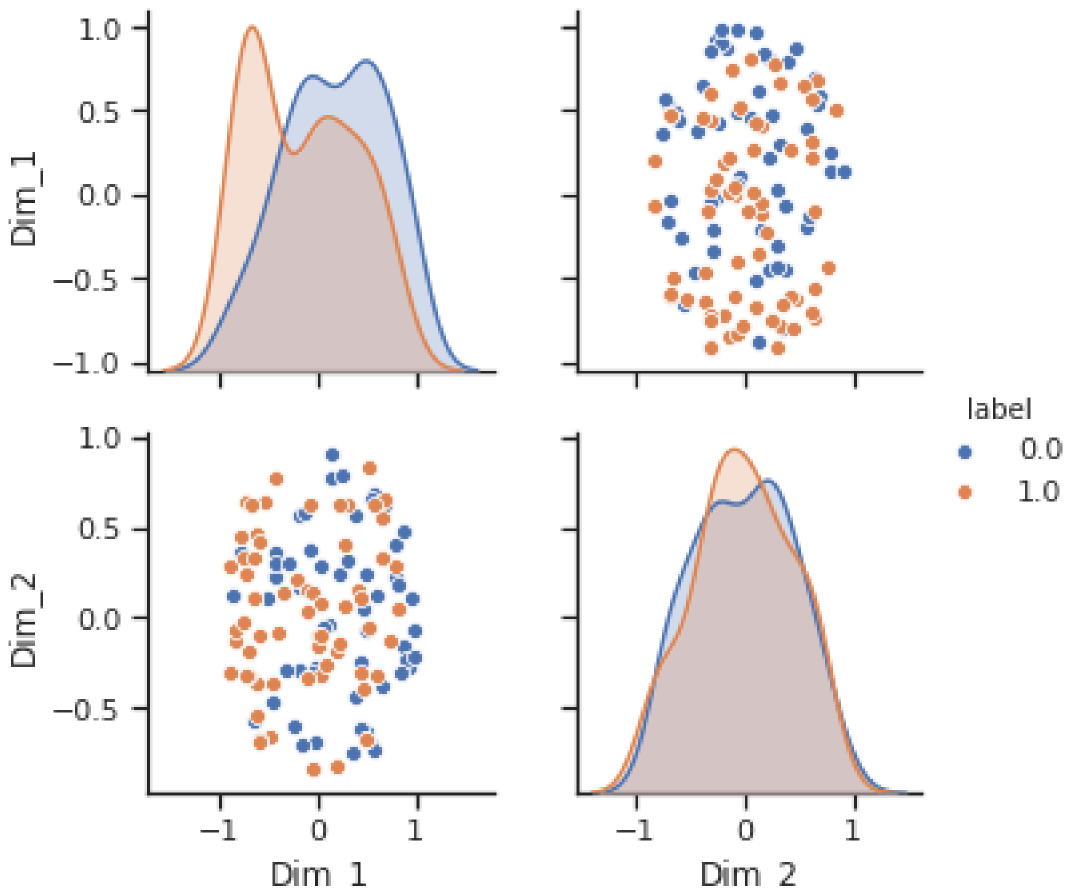
[144]:
```
type(X_principal)
```

[144]: pandas.core.frame.DataFrame

[143]:
```
pca_data1 = np.vstack((X_principal.T, labels)).T
pca_df = pd.DataFrame(data=pca_data1, columns=("Dim_1", "Dim_2", "label"))
sns.FacetGrid(pca_df, hue="label", height=6).map(plt.scatter, 'Dim_1', 'Dim_2').
 ↪add_legend()
plt.show()
```



[249]:
```
sns.set(style="ticks", color_codes=True)
sns.pairplot(pca_df, hue='label', vars=["Dim_1", "Dim_2"])
plt.show()
```



27 fastICA

[164]:
```
from sklearn.decomposition import FastICA
transformer = FastICA(n_components=2,random_state=0)
X_transformed = transformer.fit_transform(standardized_data)
```

[165]:
```
ica_df = pd.DataFrame(X_transformed)
```

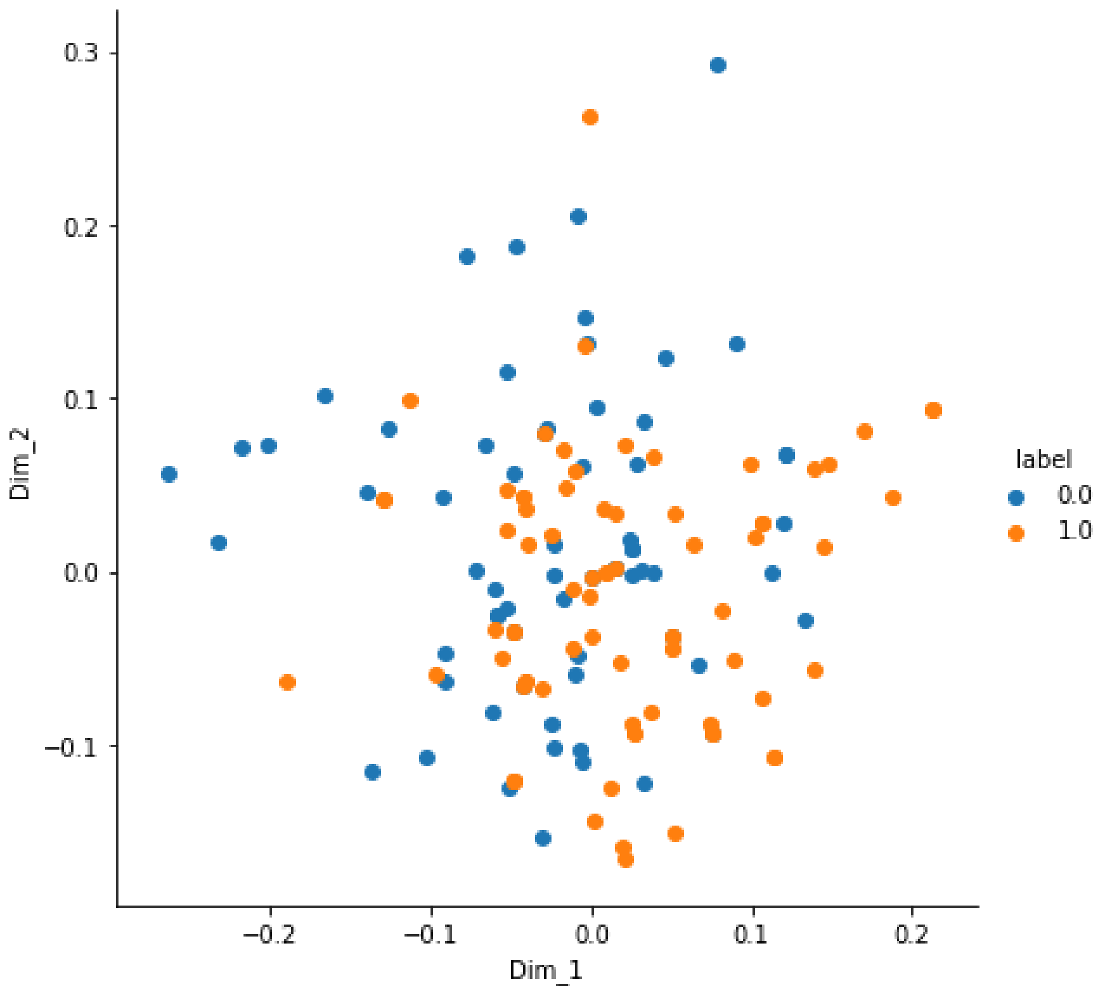
[166]:
```
ica_df
```

[166]:

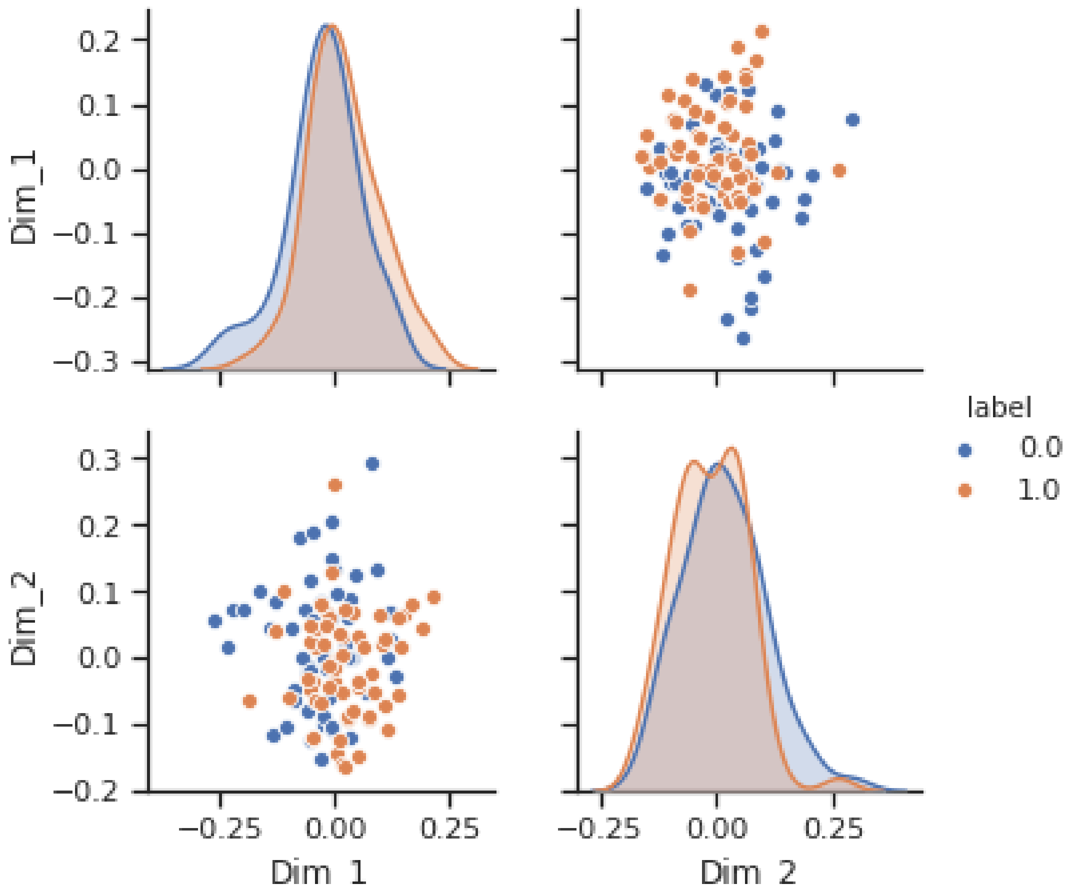
```
            0         1
0   -0.052969  0.115773
1   -0.027758  0.082460
2    0.000310 -0.037377
3    0.024800  0.012647
4    0.024800  0.012647
..        ...       ...
138 -0.061765 -0.080362
139 -0.011543 -0.044287
140  0.049748 -0.037507
141  0.025858 -0.001720
142  0.067039 -0.053034

[143 rows x 2 columns]
```

[168]:

```
ica_data1 = np.vstack((X_transformed.T, labels)).T
ica_df = pd.DataFrame(data=ica_data1, columns=("Dim_1", "Dim_2","label"))
sns.FacetGrid(ica_df, hue="label", height=6).map(plt.scatter, 'Dim_1', 'Dim_2').
 ↪add_legend()
plt.show()
```



[248]:
```
sns.set(style="ticks", color_codes=True)
sns.pairplot(ica_df, hue='label', vars=["Dim_1", "Dim_2"])
plt.show()
```

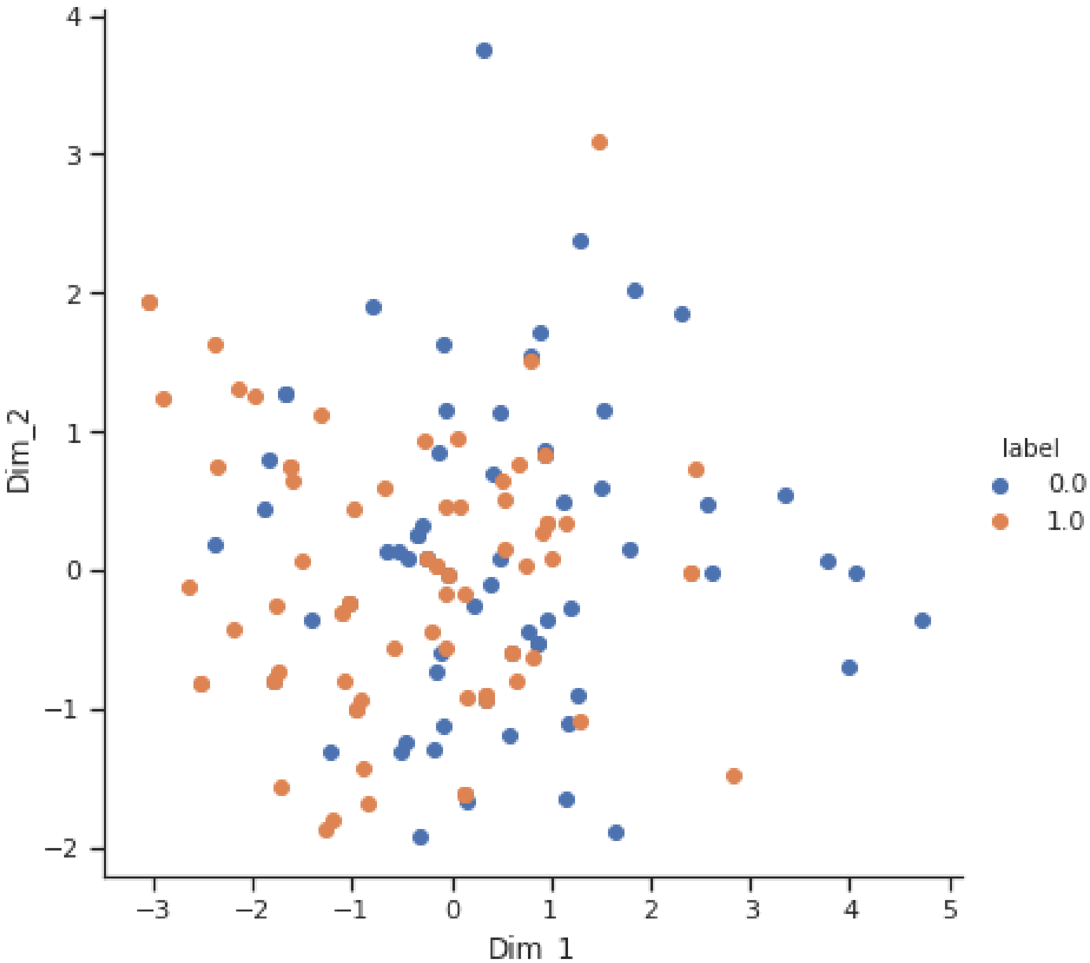


28 kernelPCA

[244]:
```python
from sklearn.decomposition import KernelPCA
transformer = KernelPCA(n_components=2, kernel='linear')
X_transformed_k = transformer.fit_transform(standardized_data)
X_transformed_k.shape
```

[244]: (143, 2)

[243]:
```python
kpca = np.vstack((X_transformed_k.T, labels)).T
kpca_df = pd.DataFrame(data=kpca, columns=("Dim_1", "Dim_2","label"))
sns.FacetGrid(kpca_df, hue="label", height=6).map(plt.scatter, 'Dim_1', 'Dim_2').
 ↪add_legend()
plt.show()
```



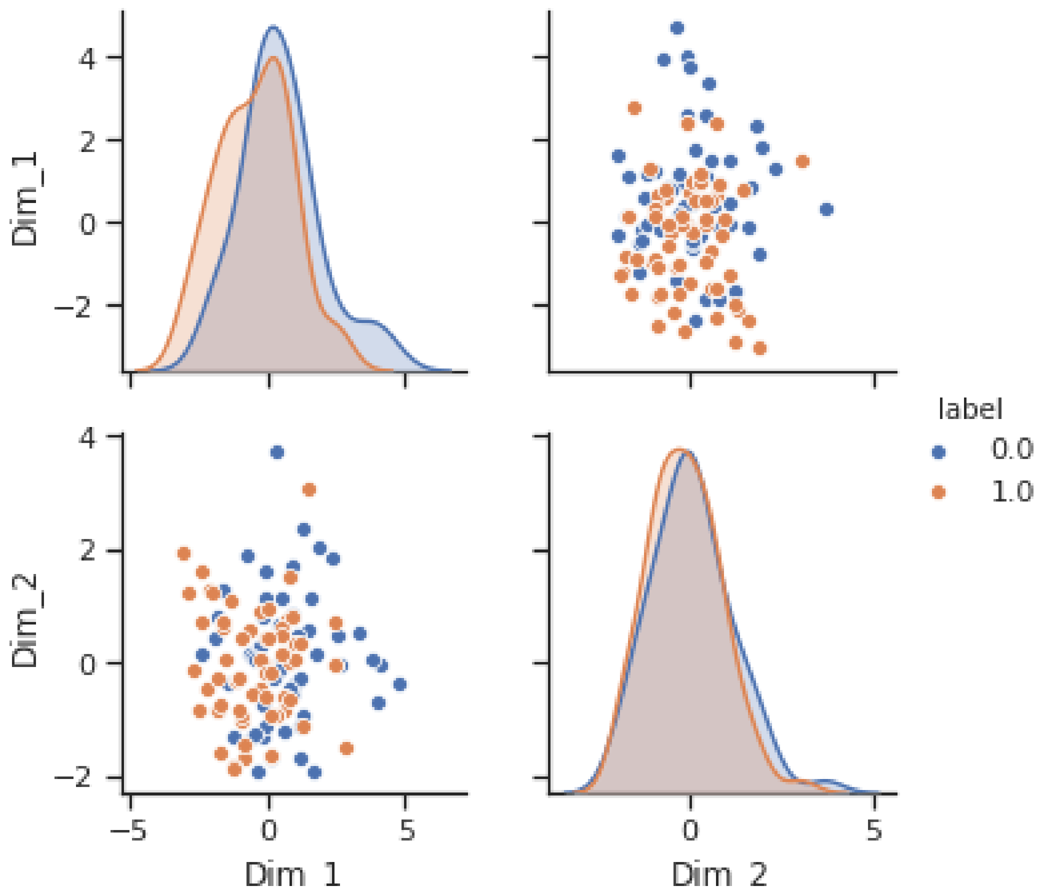
[246]:
```
kpca_df
```

[246]:

	Dim_1	Dim_2	label
0	1.530048	1.156667	0.0
1	0.922871	0.862817	0.0
2	-0.212857	-0.439066	1.0
3	-0.344936	0.245913	0.0
4	-0.344936	0.245913	0.0
..	...	...	...
138	0.587595	-1.188046	0.0
139	-0.052813	-0.566792	1.0
140	-1.041273	-0.247338	1.0
141	-0.442477	0.080820	0.0
142	-1.417043	-0.362645	0.0

[143 rows x 3 columns]

[247]:
```
sns.set(style="ticks", color_codes=True)
sns.pairplot(kpca_df, hue='label', vars=["Dim_1", "Dim_2"])
plt.show()
```



[ ]: