Chapter 17 - Exercise 1: Admit to university?

In [1]:
```
# from google.colab import drive
# drive.mount("/content/gdrive", force_remount=True)
# %cd'/content/gdrive/My Drive/LDS6_MachineLearning/practice/Chapter17_LLE/'
```

Xem xét việc có được vào trường đại học hay không dựa trên bộ dữ liệu sinh viên 400 mẫu có tên là binary.csv

Yêu cầu: Hãy đọc dữ liệu từ tập tin này, áp dụng Logistic Regression để thực hiện việc xác định có được vào trường đại học hay không dựa vào các thông tin như: gre, gpa, rank.

Áp dụng thuật toán LLE để trực quan hóa dữ liệu với 2 thành phần thay vì 3 thành phần. Xem xét việc scale dữ liệu???

Chạy lại thuật toán Logistic Regression cho dữ liệu đã giảm chiều và kiểm chứng lại kết quả? So sánh với kết quả ban đầu để quyết định có giảm chiều cho bài toán này hay không?

In [2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
import math
from pandas import DataFrame
from sklearn import preprocessing

from sklearn.model_selection import train_test_split
from numpy import loadtxt, where
from pylab import scatter, show, legend, xlabel, ylabel
```

In [3]:
```
data = pd.read_csv("binary.csv")
```

In [4]:
```
# data.info()
```

In [5]:
```
# data.describe()
```

In [6]:
```
# data.head()
```

In [7]:
```
X = data[['gre', 'gpa', 'rank']]
X.head(3)
```

Out[7]:

	gre	gpa	rank
0	380	3.61	3
1	660	3.67	3
2	800	4.00	1

In [8]:
```
X.corr()
```

Out[8]:

	gre	gpa	rank
gre	1.000000	0.384266	-0.123447
gpa	0.384266	1.000000	-0.057461
rank	-0.123447	-0.057461	1.000000

In [9]:
```
Y = data[['admit']]
Y.head(3)
```

Out[9]:

	admit
0	0
1	1
2	1

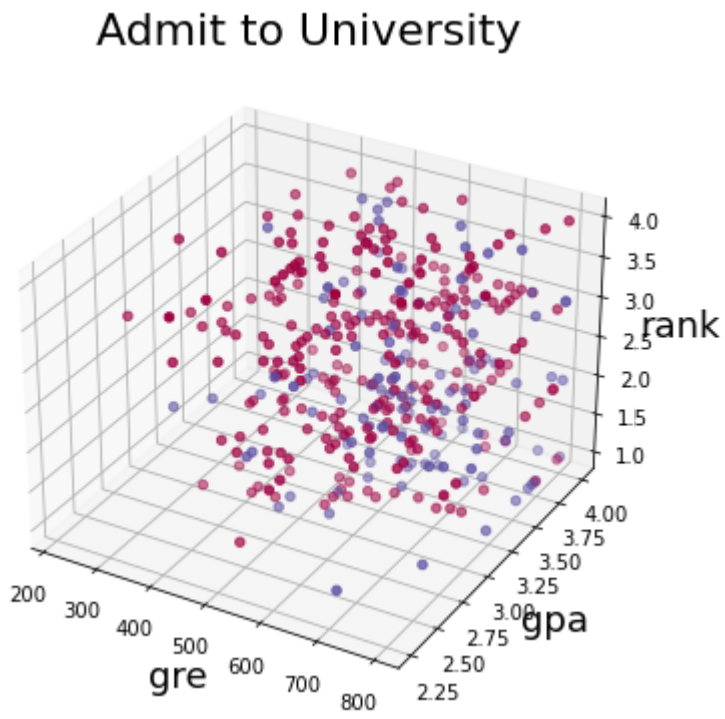
Trực quan hóa dữ liệu

In [10]:
```
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
```

In [11]:
```
types = np.reshape(Y.values, -1)
```

In [12]:
```
fig = plt.figure(figsize=(6,6))
ax = fig.add_subplot(1, 1, 1, projection='3d')
ax.scatter(X['gre'], X['gpa'], X['rank'], c=types, cmap=plt.cm.Spectral)
ax.set_xlabel("gre", fontsize=18)
ax.set_ylabel("gpa", fontsize=18)
ax.set_zlabel("rank", fontsize=18)
ax.set_title("Admit to University", fontsize=22)
```

Out[12]: Text(0.5, 0.92, 'Admit to University')



In [13]:
```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
# Fit on training set only.
scaler.fit(X)
# Apply transform to both the training set and the test set.
X_scaler = scaler.transform(X)
```

In [14]:
```
from sklearn.manifold import LocallyLinearEmbedding
lle = LocallyLinearEmbedding(n_components=2, n_neighbors=10)
```

In [15]:
```
X_reduced = lle.fit_transform(X_scaler)
```

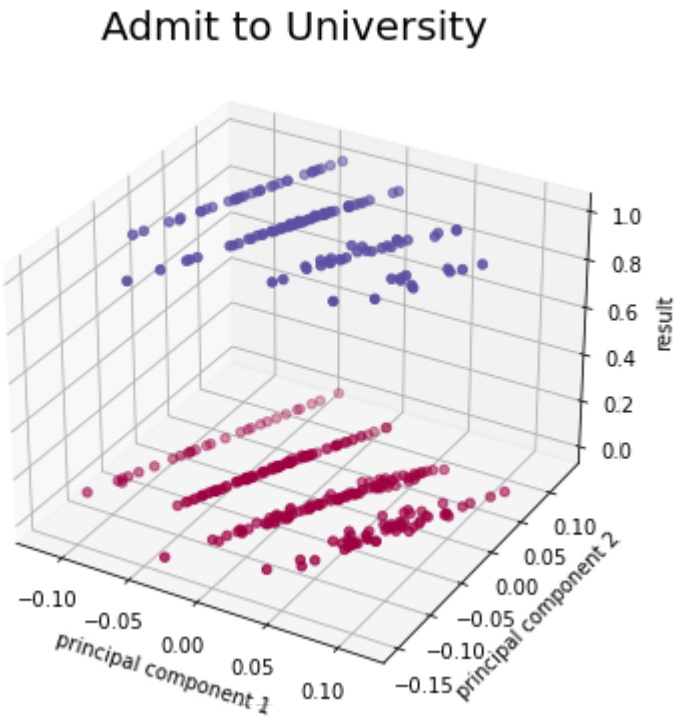
In [16]:
```
X_reduced[:3]
```

Out[16]:
```
array([[ 0.00561058, -0.0432736 ],
       [ 0.04323459,  0.0235638 ],
       [-0.03852613,  0.13080992]])
```

In [17]:
```
lle.eigen_solver
```

Out[17]:
```
'auto'
```

In [18]:
```
from mpl_toolkits.mplot3d import Axes3D
fig = plt.figure(figsize=(6,6))
ax = fig.add_subplot(111, projection='3d')
ax.scatter(X_reduced[:, 0], X_reduced[:, 1], types,
           c=types, cmap=plt.cm.Spectral)
ax.set_xlabel('principal component 1')
ax.set_ylabel('principal component 2')
ax.set_zlabel('result')
ax.set_title('Admit to University', fontsize =20)
plt.show()
```



In [19]:
```
lle_r = pd.DataFrame(data = X_reduced,
                             columns = ['c1',
                                        'c2'])
```

In [20]:
```
lle_r = pd.concat([lle_r, Y], axis = 1)
lle_r.head(3)
```

Out[20]:

	c1	c2	admit
0	0.005611	-0.043274	0
1	0.043235	0.023564	1
2	-0.038526	0.130810	1

Explaining dataset with 2 main components (LLE)

In [21]:
```
lle_r = lle_r.join(X)
```

In [22]:
```
lle_r.head(3)
```

Out[22]:

	c1	c2	admit	gre	gpa	rank
0	0.005611	-0.043274	0	380	3.61	3
1	0.043235	0.023564	1	660	3.67	3
2	-0.038526	0.130810	1	800	4.00	1

In [23]:
```
lle
```

Out[23]:
```
LocallyLinearEmbedding(eigen_solver='auto', hessian_tol=0.0001, max_iter=100,
                       method='standard', modified_tol=1e-12, n_components=2,
                       n_jobs=None, n_neighbors=10, neighbors_algorithm='auto',
                       random_state=None, reg=0.001, tol=1e-06)
```

In [24]:
```
# Áp dụng Logistic Regression với X_reduced và Y
# Đo lại: acc của train, test => kết luận ...
# Dựa trên kết luận => Có/không giảm chiều cho dữ liệu này???
```