1 Data Preprocessing

1.0.1 1.数据质量问题

1.0.2 较差的数据质量可能会对数据挖掘产生不利影响。常见的数据质量问题包括噪声、异常值、缺失值和重复数据。

1.0.3 数据：breast+cancer+wisconsin+original/breast-cancer-wisconsin.data

In [59]:
```
import pandas as pd
data = pd.read_csv('breast+cancer+wisconsin+original/breast-cancer-wisconsin.data', h
data.columns = ['Sample code', 'Clump Thickness', 'Uniformity of Cell Size', 'Uniform
                'Marginal Adhesion', 'Single Epithelial Cell Size', 'Bare Nuclei', 'B
                'Normal Nucleoli', 'Mitoses','Class']
# 打印数据集实例个数和属性个数
print("数据集实例个数:", data.shape[0])
print("属性个数:", data.shape[1])
```

```
数据集实例个数: 699
属性个数: 11
```

1.0.3.1 1.1缺失值

1.0.3.2 缺失值在数据集中编码为“?”，将缺失值转换为NaN，并计算每列数据中缺失值的数量。

In [12]:
```
import numpy as np
# 将缺失值编码为 NaN
data1=data.replace('?', np.NaN)

# 计算每列中缺失值的数量
missing_values_count = data1.isna().sum()
print("每列中缺失值的数量:")
print(missing_values_count)
```

```
每列中缺失值的数量:
Sample code                  0
Clump Thickness              0
Uniformity of Cell Size      0
Uniformity of Cell Shape     0
Marginal Adhesion            0
Single Epithelial Cell Size  0
Bare Nuclei                 16
Bland Chromatin              0
Normal Nucleoli              0
Mitoses                      0
Class                        0
dtype: int64
```

In [10]:
```
# 将缺失值替换为该列的中值
data2=data1.fillna(data1.median())
```

In [13]:
```
# 丢弃包含缺失值的数据点，并打印丢弃后数据集的数据量
data3=data1.dropna()
print("数据集实例个数:", data3.shape[0])
```

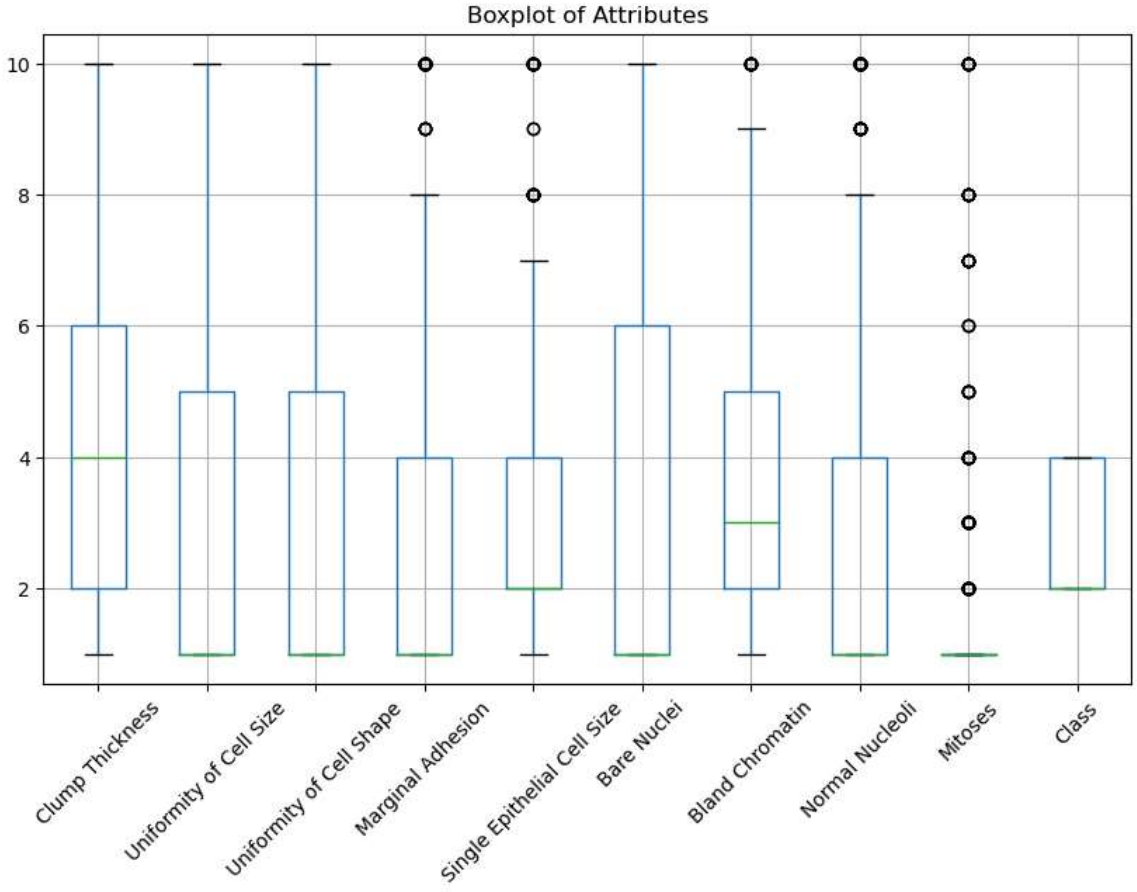
数据集实例个数：683

1.0.3.3 1.2异常值

1.0.3.4 通过绘制boxplot来识别数据中包含异常值的列

1.0.3.5 由于“Bare Nuclei”列中的值存储为字符串对象，应该先将该列转换为数值

In [16]:
```
import matplotlib.pyplot as plt
# 将Bare Nuclei属性的值转换为数值，利用boxplot绘制箱线图
data2['Bare Nuclei'] = pd.to_numeric(data['Bare Nuclei'], errors='coerce')
data4=data2.drop(columns=['Sample code'])
# 绘制 boxplot
data4.boxplot(figsize=(10, 6))
plt.title('Boxplot of Attributes')
plt.xticks(rotation=45)
plt.show()
```



1.0.3.6 为了丢弃异常值，我们可以计算每个属性的Z分数，并删除那些包含Z分数异常高或异常低的属性的实例（例如，如果Z>3或Z<=-3）。

In [33]:
```
# 数据标准化，计算Z分数
Z_scores = (data4 - data4.mean()) / data4.std()
```

In [40]:
```
# 按照“Z > 3 or Z <= -3”这个原则删除异常值，打印原始数据量和删除异常值后的数据量
selected_rows = Z_scores[(Z_scores > -3).sum(axis=1) == 9]  # 大于 -3 的元素数量为 9
selected_rows2 = Z_scores[(Z_scores <= 3).sum(axis=1) == 9]  # 小于等于 3 的元素数量
intersection_index=selected_rows.index.intersection(selected_rows2.index)
# 删除具有异常 Z 分数的实例
data5 = data4.loc[intersection_index]
```

1.0.3.7 1.3重复数据

In [43]:
```
# 检查数据中的重复样本，打印重复样本个数
duplicate_count = data4.duplicated().sum()
print("重复样本个数:", duplicate_count)
```

```
重复样本个数: 236
```

In [45]:
```
# 删除重复行，打印删除前后数据集样本量
data6=data4.drop_duplicates()
print("样本量:", data4.shape[0],data6.shape[0])
```

```
样本量: 699 463
```

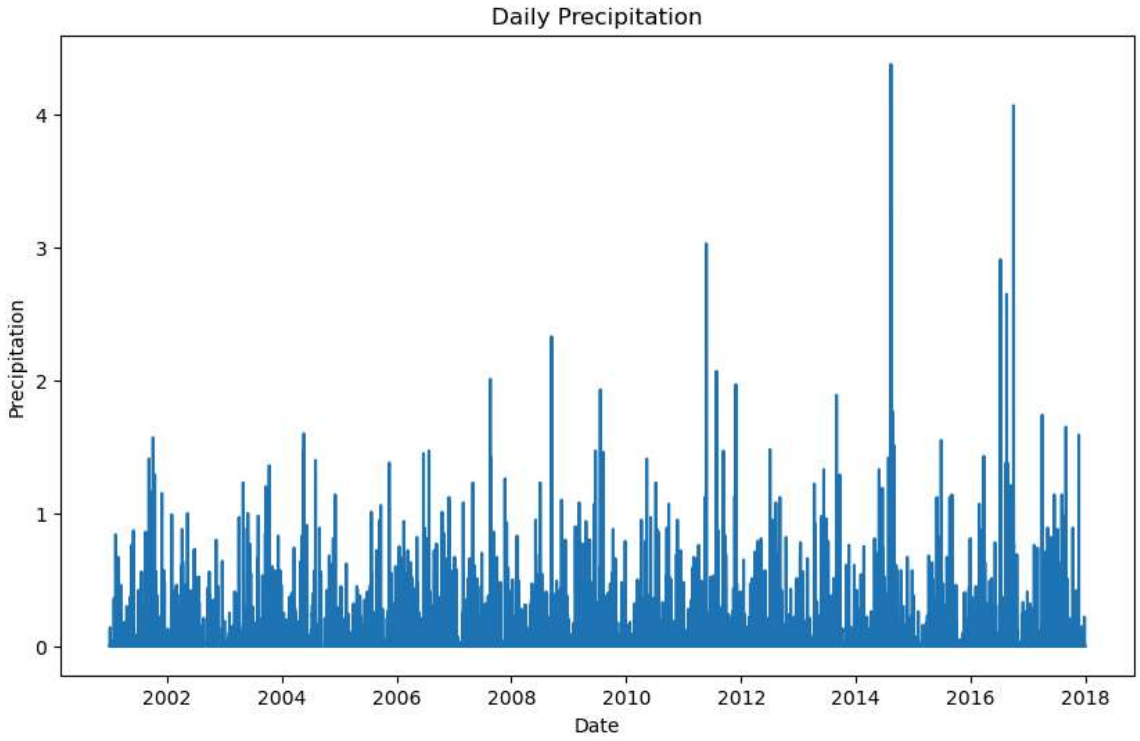
1.0.4 2.数据聚合

1.0.5 目的：(1)减小要处理的数据的大小;(2)改变分析的粒度(从细粒度到粗粒度);(3)提高数据的稳定性

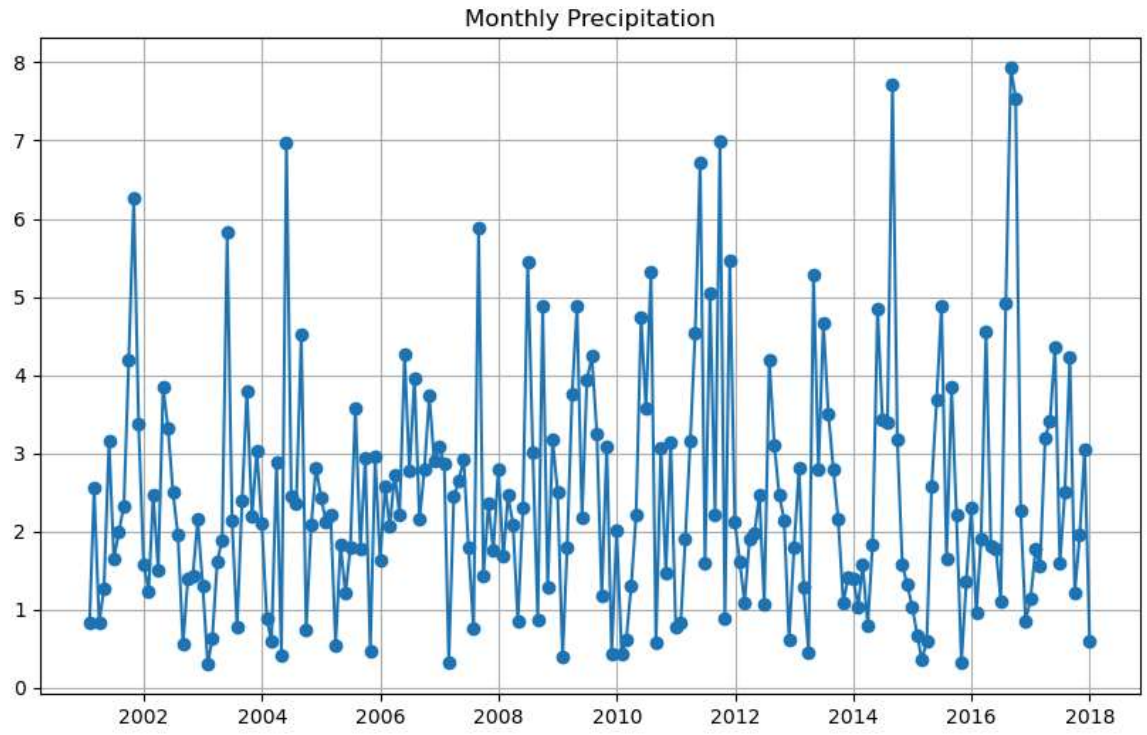
1.0.6 数据：DTW_prec.csv

In [54]:
```
data = pd.read_csv('DTW_prec.csv', header=None)
data.columns = ['Date', 'Precipitation']
data = data.iloc[1:]
print("数据集实例个数:", data.shape[0])
print("属性个数:", data.shape[1])
data['Precipitation'] = pd.to_numeric(data['Precipitation'])
data['Date'] = pd.to_datetime(data['Date'])
data_variance = data['Precipitation'].var()
print("数据的方差:", data_variance)
# 绘制其每日时间序列的折线图，并打印数据的方差
plt.figure(figsize=(10, 6))
plt.plot(data['Date'], data['Precipitation'])
plt.title('Daily Precipitation')
plt.xlabel('Date')
plt.ylabel('Precipitation')
plt.show()
```

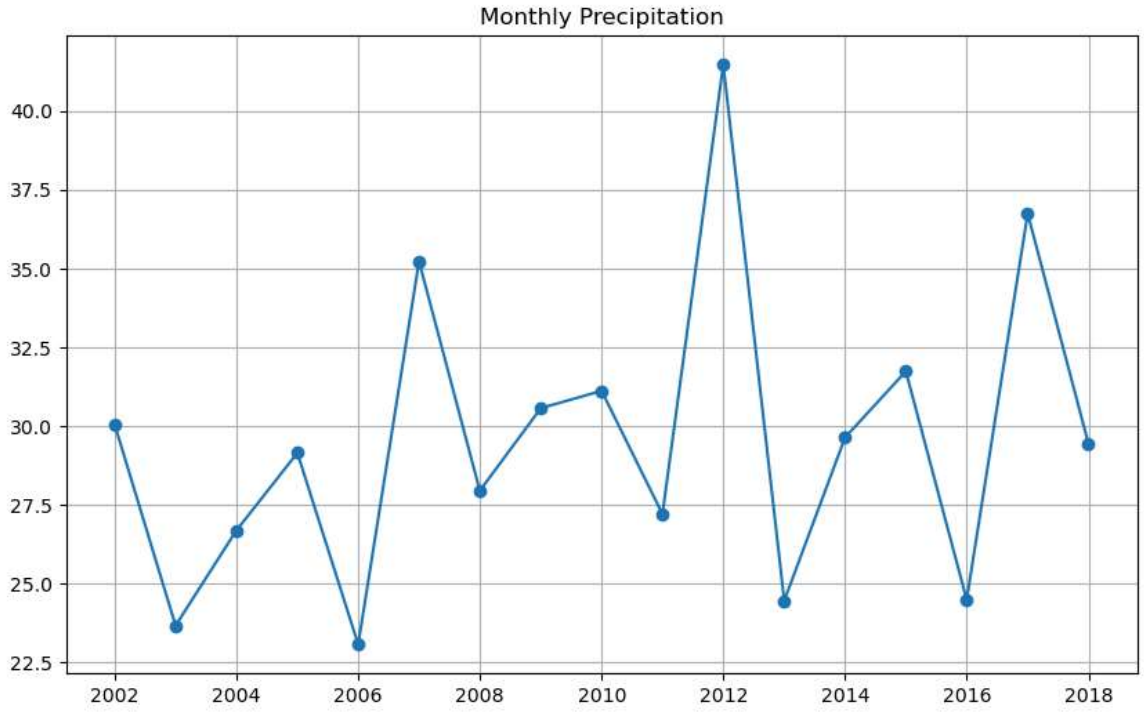
数据集实例个数: 6191
属性个数: 2
数据的方差: 0.05304985960911706



In [57]:
```
# 每日降水量的时间序列过于混乱，不同时间步长之间的变化很大。将其按月分组和聚合，获得月
# 绘制其每月时间序列的折线图，并打印数据的方差
# data.set_index('Date', inplace=True)

# 按月分组并聚合数据
monthly_data = data.groupby(pd.Grouper(freq='M')).sum()

# 绘制每月时间序列的折线图
plt.figure(figsize=(10, 6))
plt.plot(monthly_data.index, monthly_data['Precipitation'], marker='o', linestyle='-'
plt.title('Monthly Precipitation')
plt.grid(True)
plt.show()
```



In [58]:
```
# 绘制其每年时间序列的折线图，并打印数据的方差
# 按年分组并聚合数据
monthly_data = data.groupby(pd.Grouper(freq='Y')).sum()

# 绘制每年时间序列的折线图
plt.figure(figsize=(10, 6))
plt.plot(monthly_data.index, monthly_data['Precipitation'], marker='o', linestyle='-'
plt.title('Monthly Precipitation')
plt.grid(True)
plt.show()
```



1.0.7 3.采样(可分为替换采样和不替换采样)

1.0.8 不替换采样，其中每个选定实例都从数据集中删除；替换采样，每个选定实例不删除，从而允许在样本中多次选择

1.0.9 数据：breast+cancer+wisconsin+original/breast-cancer-wisconsin.data

In [60]:
```
# 从原始数据中随机选择（不替换）大小为3的样本
sample=data.sample(n=3, replace=False)
print(sample)
```

```
     Sample code  Clump Thickness  Uniformity of Cell Size  \
407      1234554                1                        1
215      1222936                8                        7
457      1259008                8                        8

     Uniformity of Cell Shape  Marginal Adhesion  Single Epithelial Cell Size  \
407                         1                  1                            2
215                         8                  7                            5
457                         9                  6                            6

    Bare Nuclei  Bland Chromatin  Normal Nucleoli  Mitoses  Class
407           1                2                1        1      2
215           5                5               10        2      4
457           3               10               10        1      4
```

In [61]:
```
# 随机选择1%的数据（不替换）并显示所选样本
sample=data.sample(frac=0.01, replace=False)
print(sample)
```

```
     Sample code  Clump Thickness  Uniformity of Cell Size  \
137      1182410                3                        1
463      1280258                4                        1
201      1216694               10                        8
589      1272166                5                        1
486      1070522                3                        1
114      1173235                3                        3
469      1181685                1                        1

     Uniformity of Cell Shape  Marginal Adhesion  Single Epithelial Cell Size  \
137                         1                  1                            2
463                         1                  1                            2
201                         8                  4                           10
589                         1                  1                            2
486                         1                  1                            1
114                         2                  1                            2
469                         2                  1                            2

    Bare Nuclei  Bland Chromatin  Normal Nucleoli  Mitoses  Class
137           1                1                1        1      2
463           1                1                2        1      2
201          10                8                1        1      4
589           1                1                1        1      2
486           1                2                1        1      2
114           3                3                1        1      2
469           1                2                1        1      2
```

In [62]:
```
# （替换）采样1%的数据
sample=data.sample(frac=0.01, replace=True)
print(sample)
```

```
     Sample code  Clump Thickness  Uniformity of Cell Size  \
62       1116116                9                       10
66       1117152                4                        1
594      1315506                4                        8
655      1326892                3                        1
630      1225382                6                        2
455      1246562               10                        2
52       1110102               10                        3

     Uniformity of Cell Shape  Marginal Adhesion  Single Epithelial Cell Size  \
62                         10                  1                           10
66                          1                  1                            2
594                         6                  3                            4
655                         1                  1                            2
630                         3                  1                            2
455                         2                  1                            2
52                          6                  2                            3

    Bare Nuclei  Bland Chromatin  Normal Nucleoli  Mitoses  Class
62            8                3                3        1      4
66            1                3                1        1      2
594          10                7                1        1      4
655           1                2                1        1      2
630           1                1                1        1      2
455           6                1                1        2      4
52            5                4               10        2      4
```

1.0.10 4.离散化（将连续属性转换为分类属性）

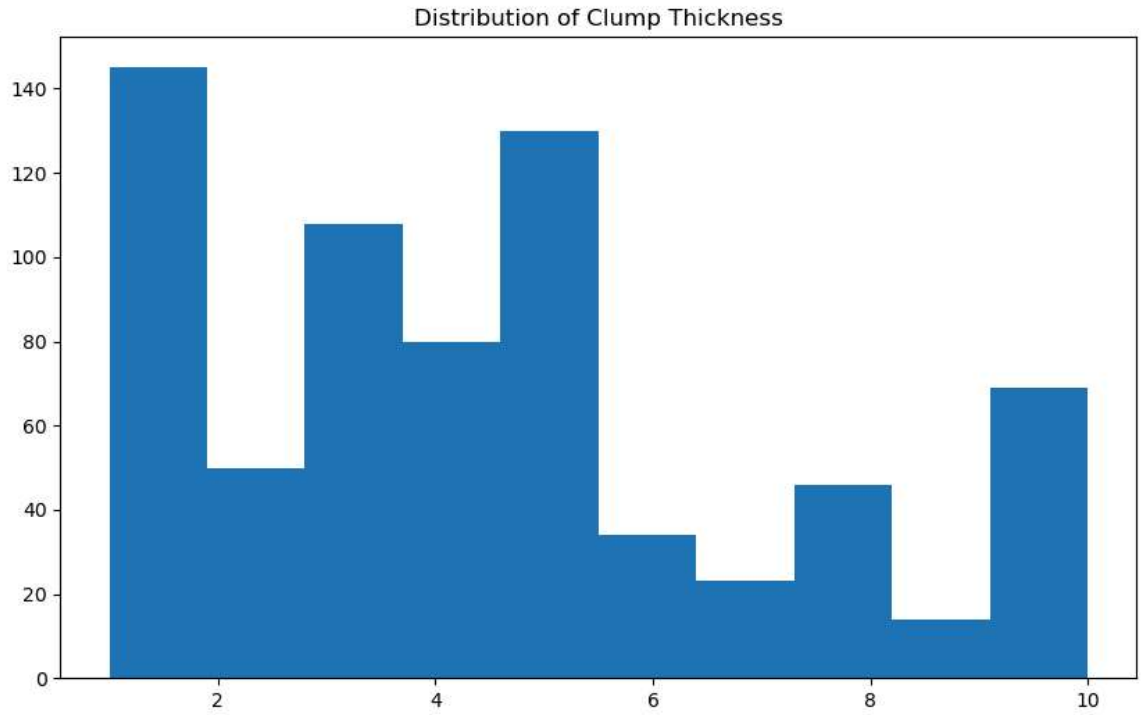
1.0.11 数据：breast+cancer+wisconsin+original/breast-cancer-wisconsin.data

1.0.12 对乳腺癌症数据集中的“Clump Thickness”属性使用无监督离散方法（equal width和equal depth）

In [65]:
```
# 使用Counter对数据集Clump Thickness属性的取值进行计数
from collections import Counter
clump_thickness_counts = Counter(data['Clump Thickness'])
clump_thickness_counts
```

Out[65]:
```
Counter({5: 130,
         3: 108,
         6: 34,
         4: 80,
         8: 46,
         1: 145,
         2: 50,
         7: 23,
         10: 69,
         9: 14})
```

In [64]:
```
# 绘制直方图，显示属性值的分布
plt.figure(figsize=(10, 6))
plt.hist(data['Clump Thickness'])
plt.title('Distribution of Clump Thickness')
plt.show()
```



In [73]:
```
# equal width：应用cut()将属性离散为4个间隔宽度相似的bin
# 使用value_counts()确定每个bin中的实例数

bins = pd.cut(data['Clump Thickness'], bins=4) #加入include_lowest=True可以得到0.99
bin_counts = bins.value_counts()
print(bin_counts)
```

```
Clump Thickness
(0.99, 3.25]    303
(3.25, 5.5]     210
(7.75, 10.0]    129
(5.5, 7.75]      57
Name: count, dtype: int64
```

In [74]:
```
# equal frequency：qcut()函数可用于将值划分为4个bin，以便每个bin具有几乎相同数量的实
bins = pd.qcut(data['Clump Thickness'], q=4)

bin_counts = bins.value_counts()
bin_counts
```

Out[74]:
```
Clump Thickness
(0.999, 2.0]    195
(2.0, 4.0]      188
(4.0, 6.0]      164
(6.0, 10.0]     152
Name: count, dtype: int64
```

1.0.13 5.主成分分析（PCA）

1.0.14 PCA是一种通过将数据从其原始高维空间投影到低维空间来减少数据中属性数量的经典方法

1.0.15 PCA创建的新属性具有以下特点：(1)它们是原始属性的线性组合;(2)它们彼此正交（垂直）;(3)它们捕获数据中的最大变化量

1.0.16 数据：pics文件夹下包含16个RGB图像文件，每个文件的大小为111x111像素

In [75]:
```
# 读取图像数据，将RGB图像转换为111x111x3=36963个特征值，最终得到一个16x36963的矩阵
from PIL import Image
import numpy as np
import os

folder_path = 'pics/'

image_matrix = np.zeros((16, 111 * 111 * 3))

for idx, filename in enumerate(os.listdir(folder_path)):
    image = Image.open(os.path.join(folder_path, filename))
    image = image.convert('RGB')
    image = image.resize((111, 111))
    image_array = np.array(image)
    feature_vector = image_array.flatten()
    image_matrix[idx] = feature_vector

image_matrix.shape
```

Out[75]: (16, 36963)

In [80]:
```
# 使用PCA，将数据矩阵投影到其前两个主成分
# 无需编写PCA代码，直接导入sklearn.decomposition中的PCA类
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
projected_data = pca.fit_transform(image_matrix)
projected_data.shape
```

Out[80]: (16, 2)

In [81]:
```
# 绘制散点图来显示投影值
colors = ['red', 'blue', 'green', 'orange']

plt.figure(figsize=(8, 6))
for i in range(len(projected_data)):
    plt.scatter(projected_data[i, 0], projected_data[i, 1], color=colors[i % 4])
plt.title('Scatter Plot of Projected Data')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.show()
```