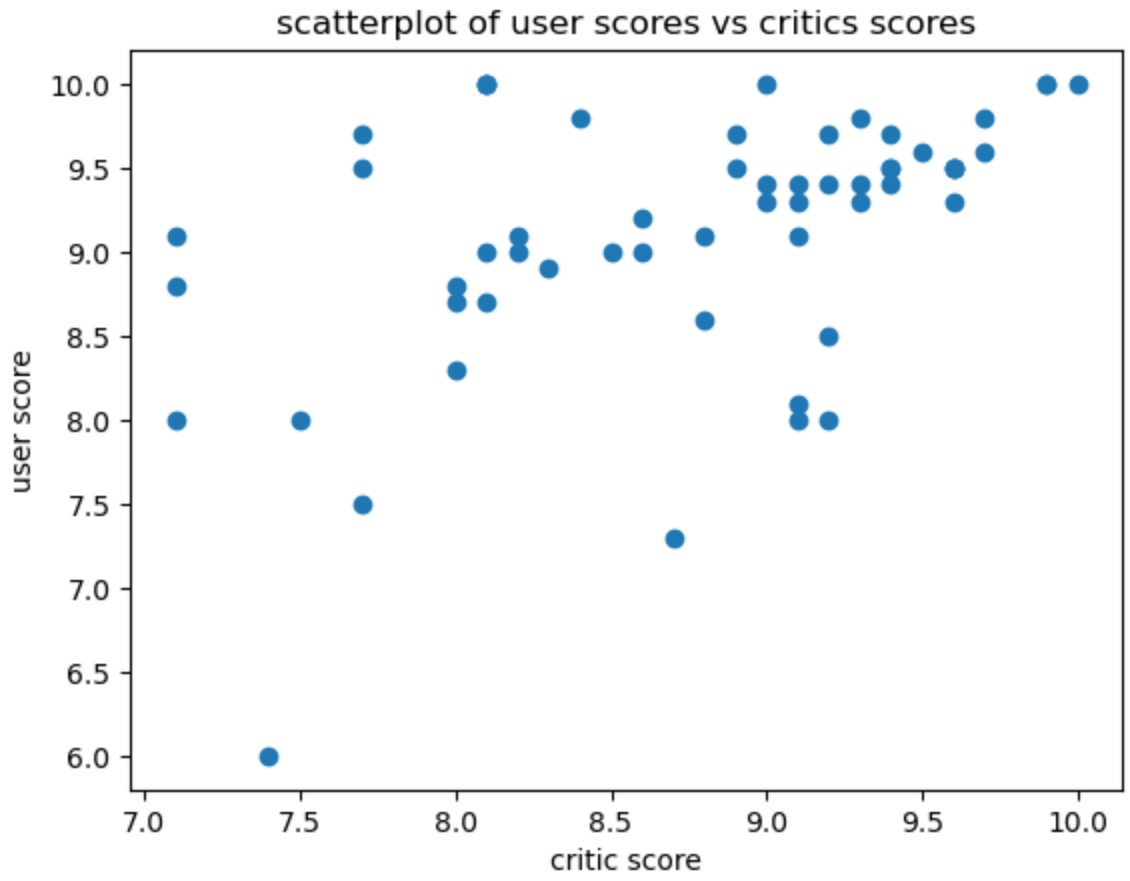
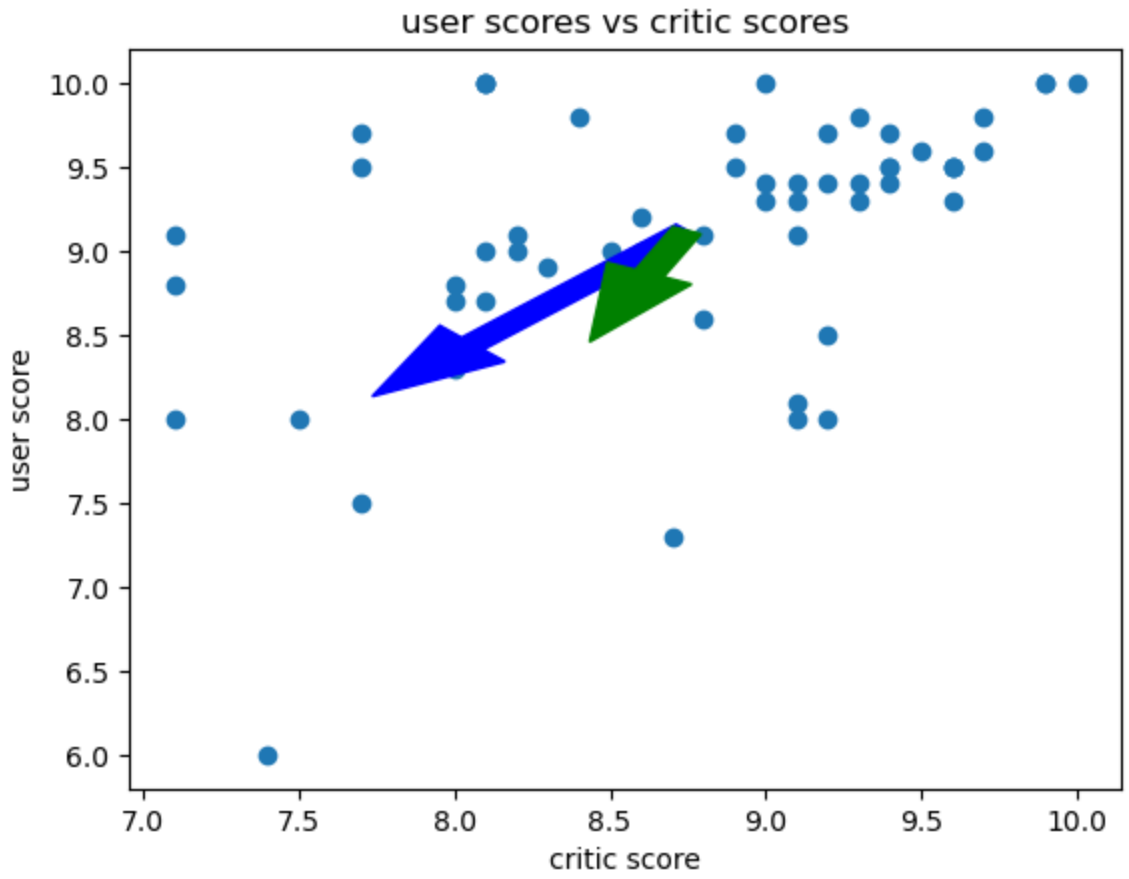
In [2]:
```
#a) Is the vector 2 | an eigenvector of A? Verify your answer with a calculation in Py

#load np
import numpy as np

#matrix A
A = np.array([[4, 0, 1],
              [-1, -6, -2],
              [5, 0, 0]])

# Define the vectors
v1 = np.array([1, 2, 3])
v2 = np.array([0, 1, 0])

#is v1 an eigenvector
eigenvalues, eigenvectors = np.linalg.eig(A)
print("eigenvalues of A:", eigenvalues)
print("eigenvectors of A:", eigenvectors)

#is v2  an eigenvector
v2_check = A.dot(v2)
if np.array_equal(v2_check, eigenvalues[0] * v2) or np.array_equal(v2_check, eigenvalu
    print("v2 is an eigenvector of A.")
else:
    print("v2 is not an eigenvector of A.")
```

```
eigenvalues of A: [-6.  5. -1.]
eigenvectors of A: [[ 0.          0.69431384 -0.18493168]
 [ 1.         -0.18935832 -0.33287702]
 [ 0.          0.69431384  0.9246584 ]]
v2 is an eigenvector of A.
```

In [3]:
```
#video_game_data.csv
#(a) Make a scatterplot of the user scores versus critics scores.



#load pd, plt, sklearn
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA

#video_game_data.csv
data = pd.read_csv('video_game_data.csv')

#plot
plt.scatter(data['critic_score'], data['user_score'])
plt.title('scatterplot of user scores vs critics scores')
plt.xlabel('critic score')
plt.ylabel('user score')
plt.show()
```



In [7]:
```
#(b) On your scatterplot from part (a), sketch the approximate directions of the first
# principal components. This can be done by hand or in Python.



#numeric columns for PCA
numeric_data = data[['critic_score', 'user_score', 'total_shipped']]

#standarized data
standardized_data = (numeric_data - numeric_data.mean()) / numeric_data.std()

#PCA algorithm
pca = PCA()
principal_components = pca.fit_transform(standardized_data)

#pca components
first_pc = pca.components_[0]
second_pc = pca.components_[1]

#plot
plt.scatter(data['critic_score'], data['user_score'])
plt.title('user scores vs critic scores')
plt.xlabel('critic score')
plt.ylabel('user score')

#draw arrows?
#try first/second_pc[0]
plt.arrow(data['critic_score'].mean(), data['user_score'].mean(), first_pc[0], first_p
plt.arrow(data['critic_score'].mean(), data['user_score'].mean(), second_pc[0], second

plt.show()
```


user scores vs critic scores
user score
critic score
10.0
9.5
9.0
8.5
8.0
7.5
7.0
6.5
6.0
7.0
7.5
8.0
8.5
9.0
9.5
10.0