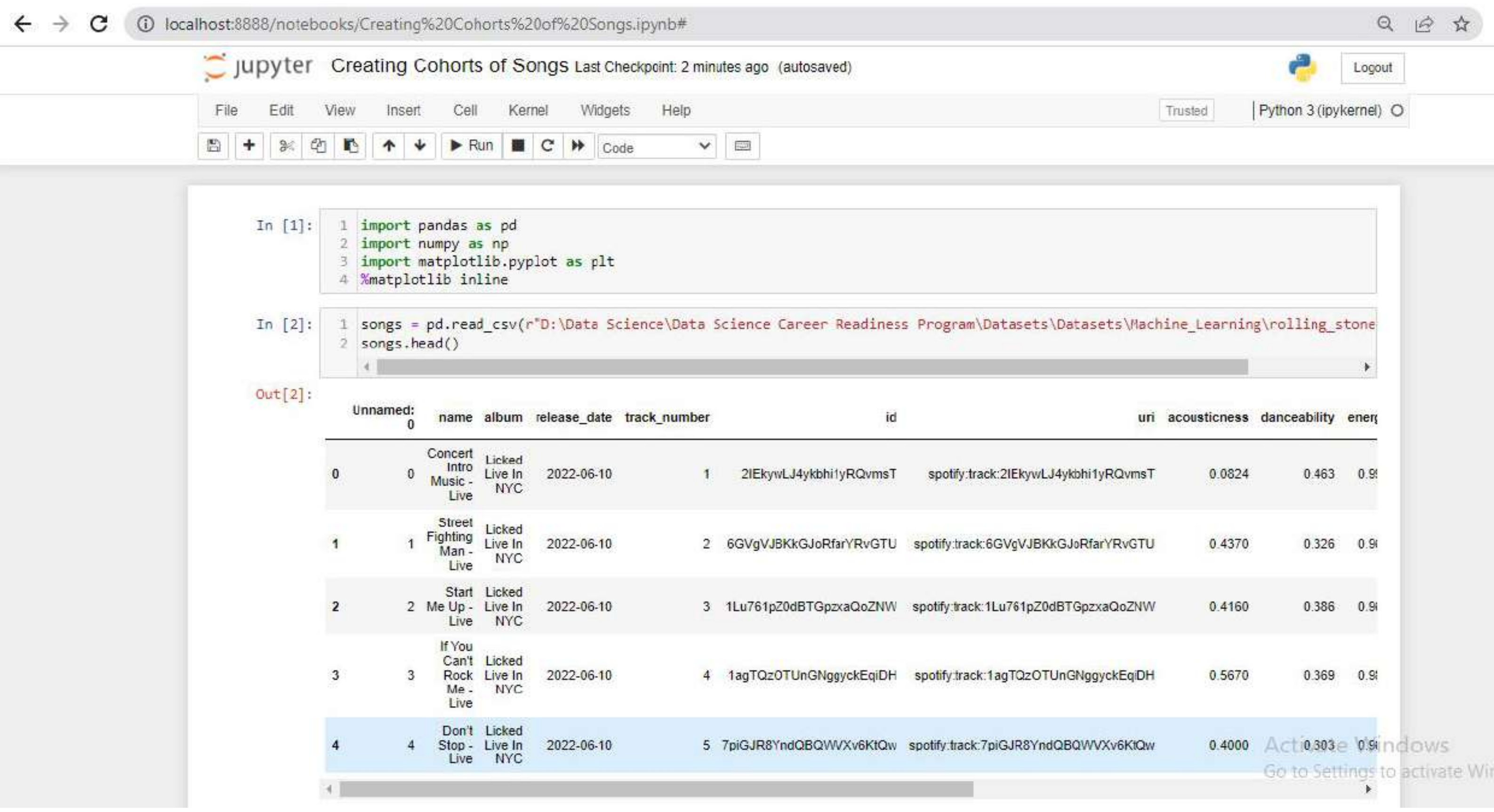
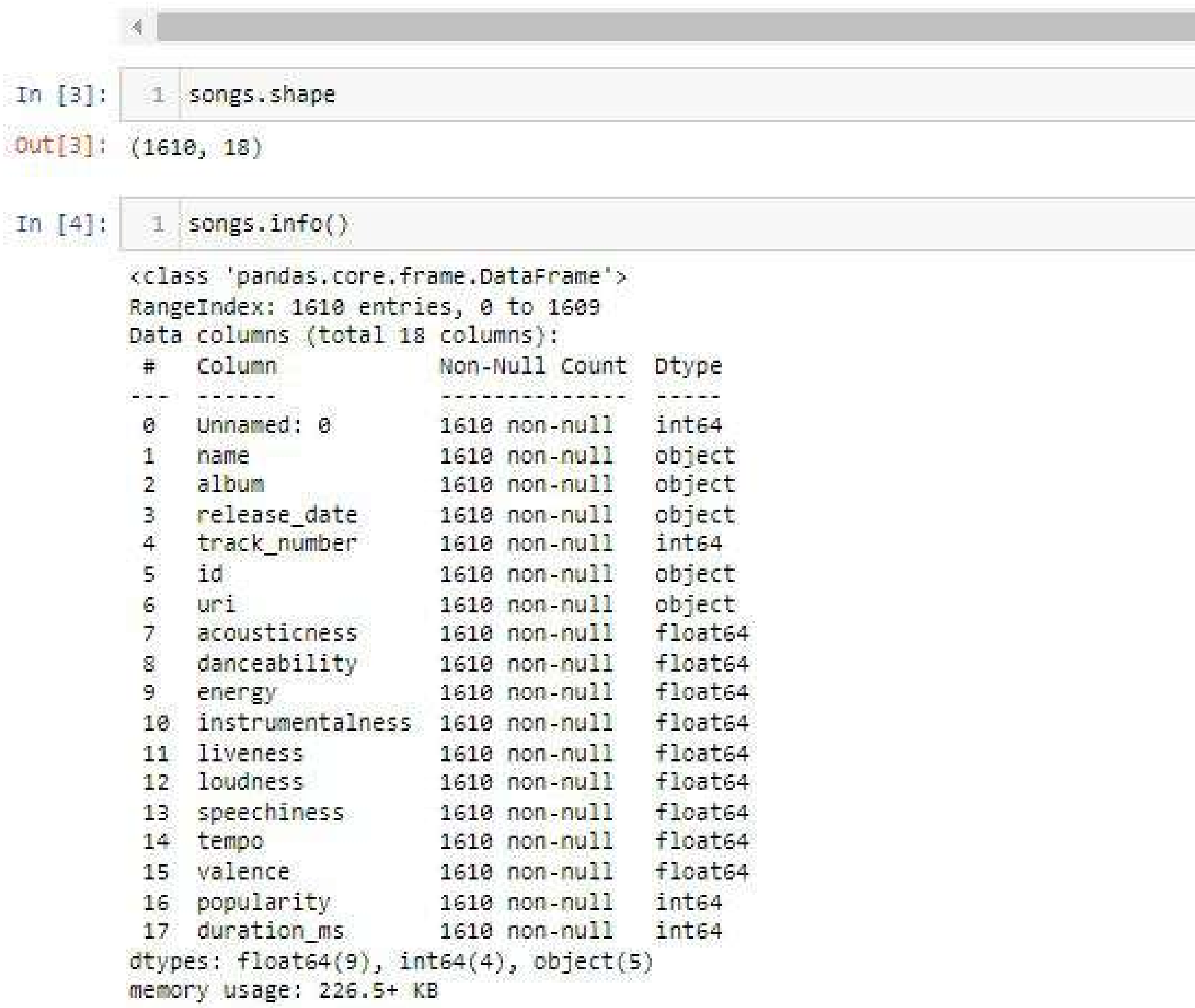

localhost:8888/notebooks/Creating%20Cohorts%20of%20Songs.ipynb#
jupyter Creating Cohorts of Songs Last Checkpoint: 2 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

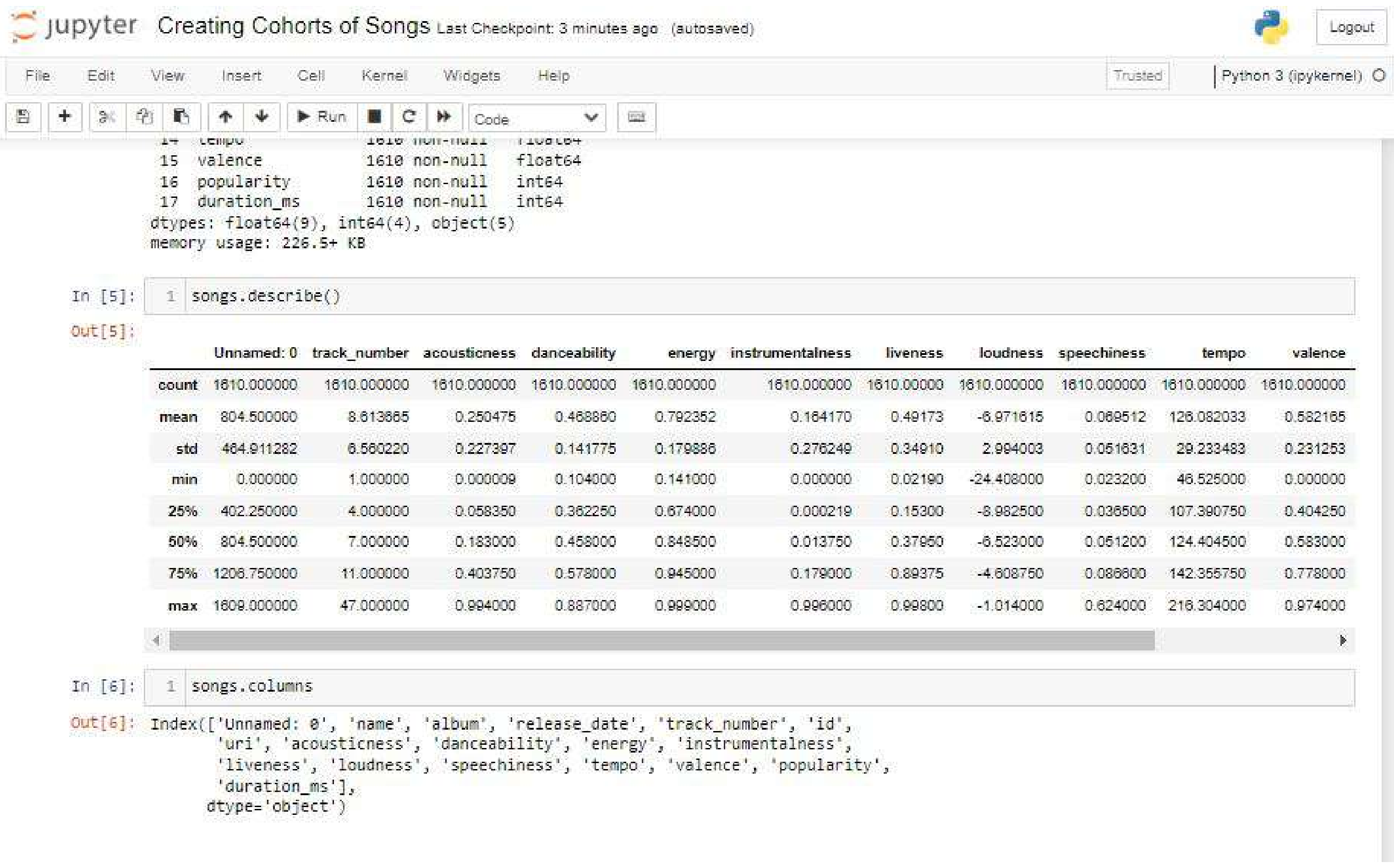
In [2]:
```
songs = pd.read_csv(r"D:\Data Science\Data Science Career Readiness Program\Datasets\Datasets\Machine_Learning\rolling_stone
songs.head()
```

Out[2]:

	Unnamed: 0	name	album	release_date	track_number	id	uri	acousticness	danceability	ener
0	0	Concert Intro Music - Live	Licked Live In NYC	2022-06-10	1	2IEkywLJ4ykbhi1yRQvmsT	spotify:track:2IEkywLJ4ykbhi1yRQvmsT	0.0824	0.463	0.9
1	1	Street Fighting Man - Live	Licked Live In NYC	2022-06-10	2	6GVgVJBKkGJoRfarYRvGTU	spotify:track:6GVgVJBKkGJoRfarYRvGTU	0.4370	0.326	0.9
2	2	Start Me Up - Live	Licked Live In NYC	2022-06-10	3	1Lu761pZ0dBTGpzxaQoZNW	spotify:track:1Lu761pZ0dBTGpzxaQoZNW	0.4160	0.386	0.9
3	3	If You Can't Rock Me - Live	Licked Live In NYC	2022-06-10	4	1agTQzOTUnGNggyckEqiDH	spotify:track:1agTQzOTUnGNggyckEqiDH	0.5670	0.369	0.9
4	4	Don't Stop - Live	Licked Live In NYC	2022-06-10	5	7piGJR8YndQBQWVXv6KtQw	spotify:track:7piGJR8YndQBQWVXv6KtQw	0.4000	0.303	0.9



```
In [3]:  songs.shape

Out[3]: (1610, 18)

In [4]:  songs.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1610 entries, 0 to 1609
Data columns (total 18 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Unnamed: 0        1610 non-null   int64  
 1   name              1610 non-null   object 
 2   album             1610 non-null   object 
 3   release_date      1610 non-null   object 
 4   track_number      1610 non-null   int64  
 5   id                1610 non-null   object 
 6   uri               1610 non-null   object 
 7   acousticness      1610 non-null   float64
 8   danceability      1610 non-null   float64
 9   energy            1610 non-null   float64
 10  instrumentalness  1610 non-null   float64
 11  liveness          1610 non-null   float64
 12  loudness          1610 non-null   float64
 13  speechiness       1610 non-null   float64
 14  tempo             1610 non-null   float64
 15  valence           1610 non-null   float64
 16  popularity        1610 non-null   int64  
 17  duration_ms       1610 non-null   int64  
dtypes: float64(9), int64(4), object(5)
memory usage: 226.5+ KB
```



```
 14  tempo             1610 non-null   float64
 15  valence           1610 non-null   float64
 16  popularity        1610 non-null   int64
 17  duration_ms       1610 non-null   int64
dtypes: float64(9), int64(4), object(5)
memory usage: 226.5+ KB
```

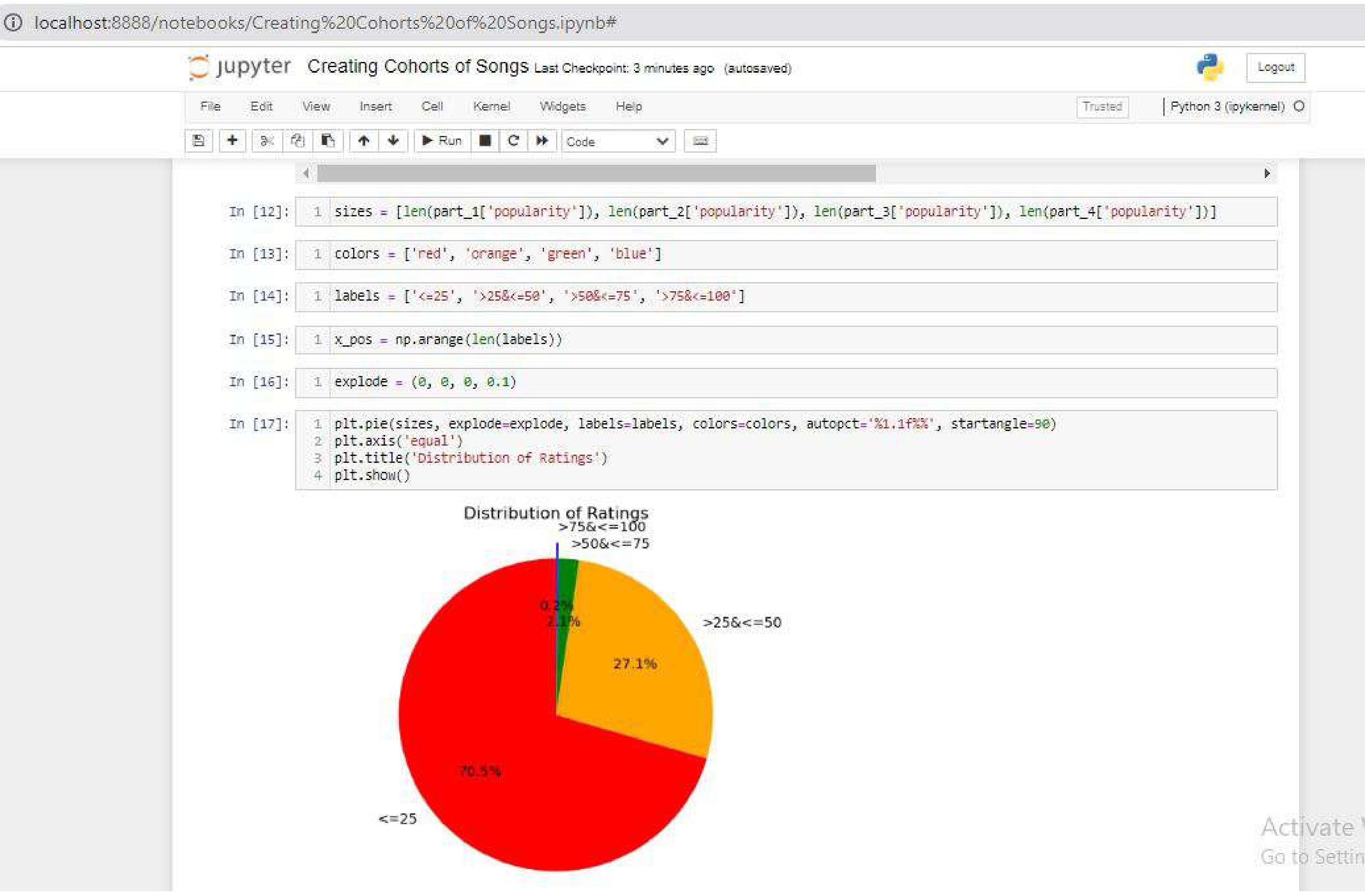
In [5]:
```
songs.describe()
```

Out[5]:

	Unnamed: 0	track_number	acousticness	danceability	energy	instrumentalness	liveness	loudness	speechiness	tempo	valence
count	1610.000000	1610.000000	1610.000000	1610.000000	1610.000000	1610.000000	1610.00000	1610.000000	1610.000000	1610.000000	1610.000000
mean	804.500000	8.613665	0.250475	0.468860	0.792352	0.164170	0.49173	-6.971615	0.069512	126.082033	0.582165
std	464.911282	6.560220	0.227397	0.141775	0.179886	0.276249	0.34910	2.994003	0.051631	29.233483	0.231253
min	0.000000	1.000000	0.000009	0.104000	0.141000	0.000000	0.02190	-24.408000	0.023200	46.525000	0.000000
25%	402.250000	4.000000	0.058350	0.362250	0.674000	0.000219	0.15300	-8.982500	0.036500	107.390750	0.404250
50%	804.500000	7.000000	0.183000	0.458000	0.848500	0.013750	0.37950	-6.523000	0.051200	124.404500	0.583000
75%	1206.750000	11.000000	0.403750	0.578000	0.945000	0.179000	0.89375	-4.608750	0.086600	142.355750	0.778000
max	1609.000000	47.000000	0.994000	0.887000	0.999000	0.996000	0.99800	-1.014000	0.624000	216.304000	0.974000

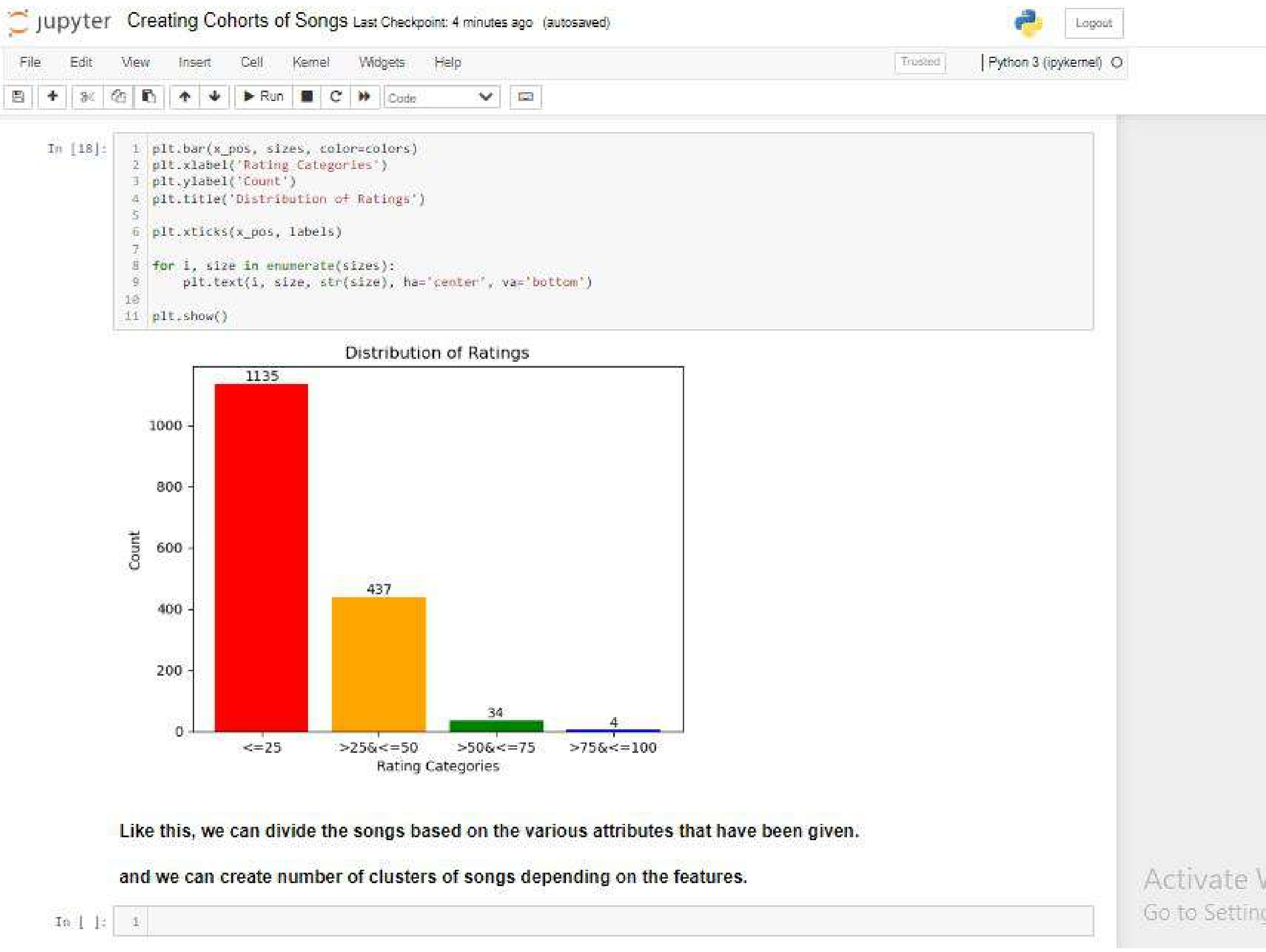
In [6]:
```
songs.columns
```

Out[6]:
```
Index(['Unnamed: 0', 'name', 'album', 'release_date', 'track_number', 'id',
       'uri', 'acousticness', 'danceability', 'energy', 'instrumentalness',
       'liveness', 'loudness', 'speechiness', 'tempo', 'valence', 'popularity',
       'duration_ms'],
      dtype='object')
```

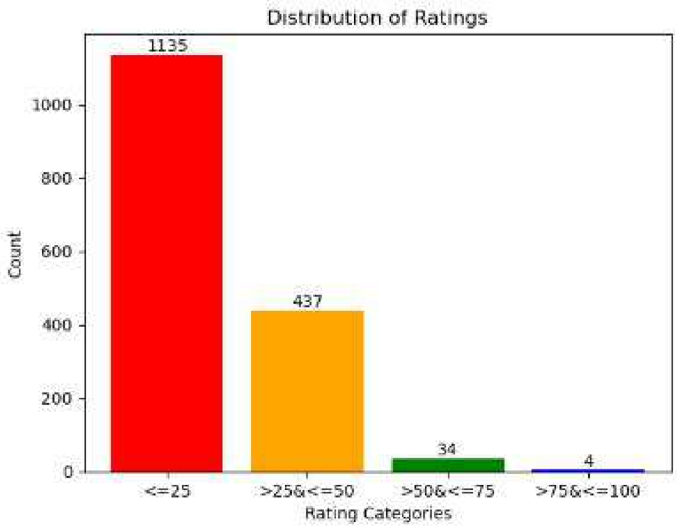

localhost:8888/notebooks/Creating%20Cohorts%20of%20Songs.ipynb#
jupyter Creating Cohorts of Songs Last Checkpoint: 3 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code

```
In [12]: sizes = [len(part_1['popularity']), len(part_2['popularity']), len(part_3['popularity']), len(part_4['popularity'])]

In [13]: colors = ['red', 'orange', 'green', 'blue']

In [14]: labels = ['<=25', '>25&<=50', '>50&<=75', '>75&<=100']

In [15]: x_pos = np.arange(len(labels))

In [16]: explode = (0, 0, 0, 0.1)

In [17]: plt.pie(sizes, explode=explode, labels=labels, colors=colors, autopct='%1.1f%%', startangle=90)
         plt.axis('equal')
         plt.title('Distribution of Ratings')
         plt.show()
```

Distribution of Ratings
>75&<=100
>50&<=75
0.2%
2.1%
>25&<=50
27.1%
70.5%
<=25
Activate
Go to Settin


jupyter
Creating Cohorts of Songs Last Checkpoint: 4 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code

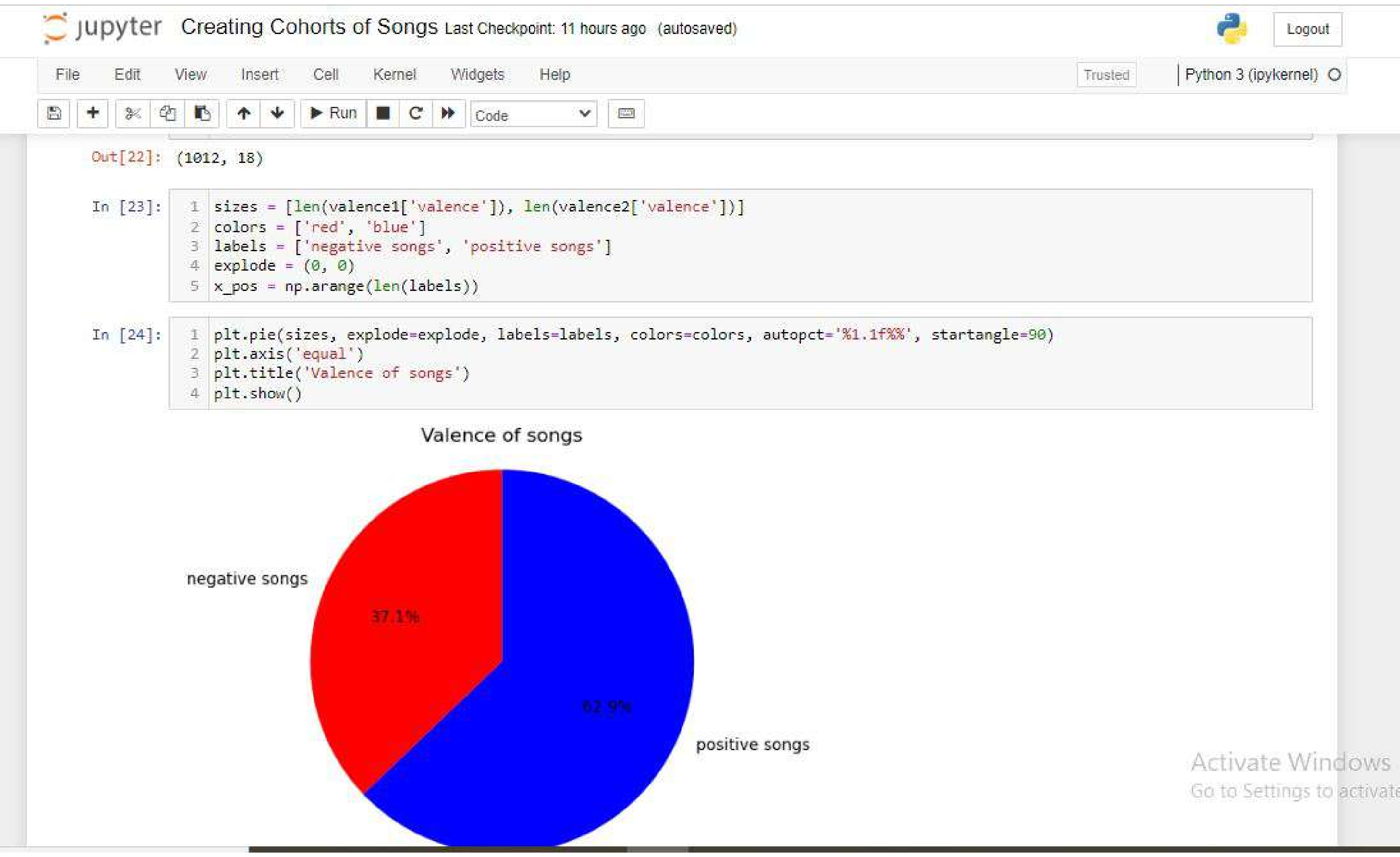
In [18]:
```python
plt.bar(x_pos, sizes, color=colors)
plt.xlabel('Rating Categories')
plt.ylabel('Count')
plt.title('Distribution of Ratings')

plt.xticks(x_pos, labels)

for i, size in enumerate(sizes):
    plt.text(i, size, str(size), ha='center', va='bottom')

plt.show()
```



Like this, we can divide the songs based on the various attributes that have been given.

and we can create number of clusters of songs depending on the features.

In [ ]:



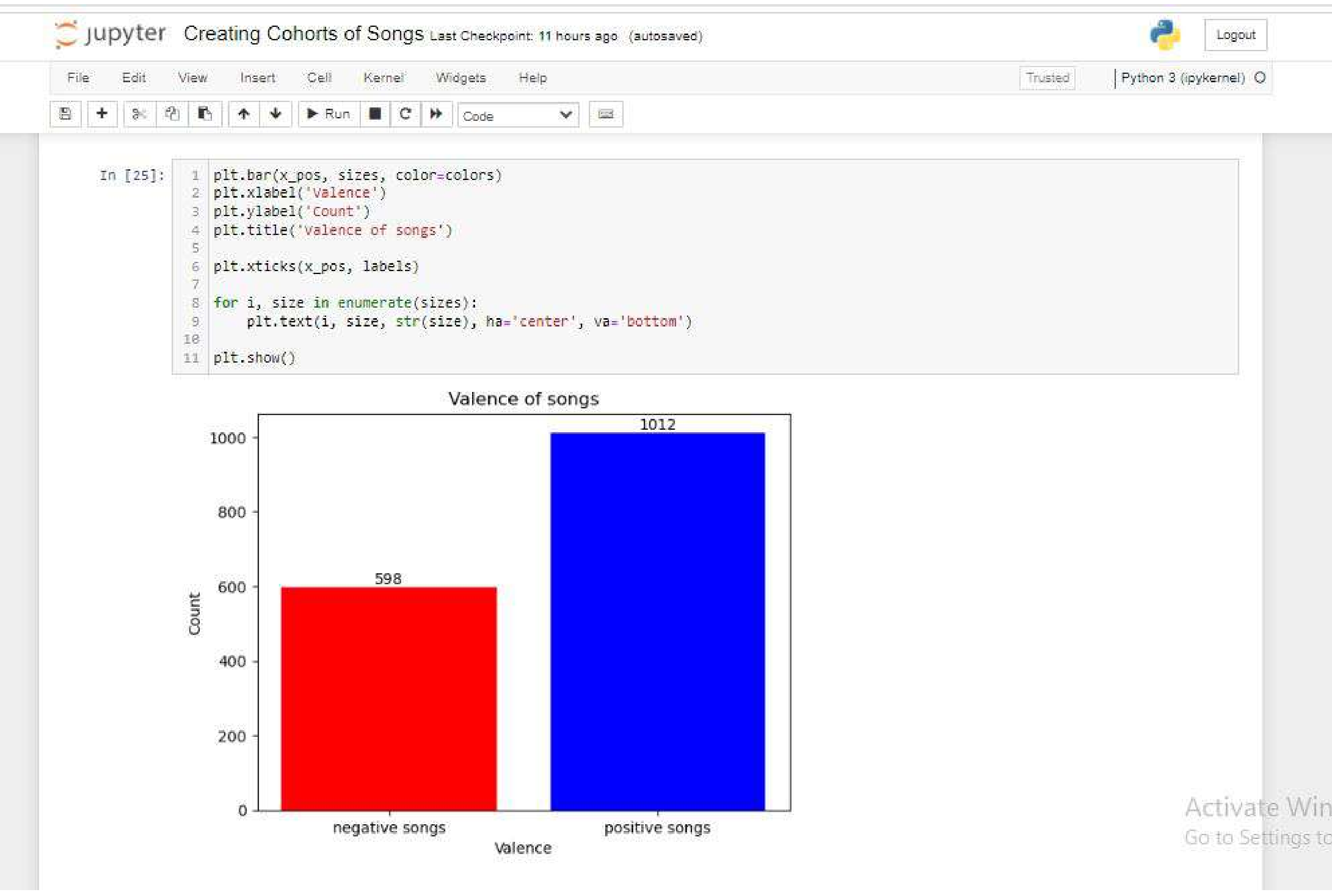
jupyter
Creating Cohorts of Songs Last Checkpoint: 11 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code

```
Out[22]: (1012, 18)
```

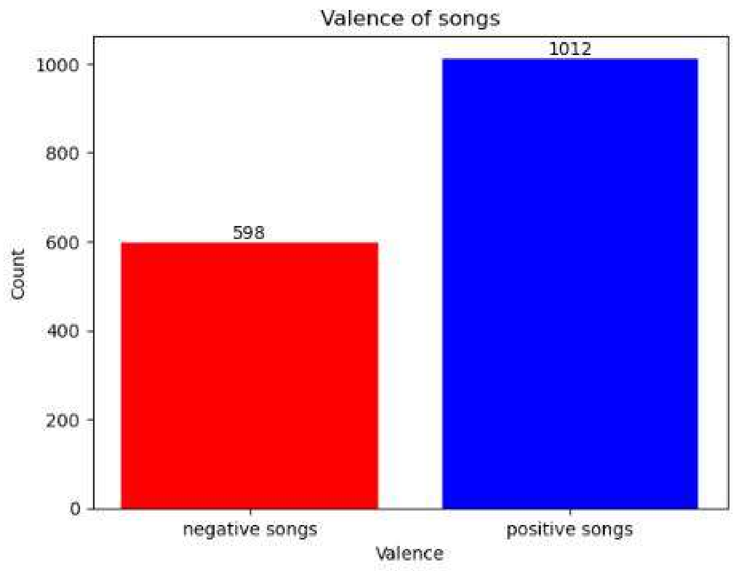
```
In [23]:
sizes = [len(valence1['valence']), len(valence2['valence'])]
colors = ['red', 'blue']
labels = ['negative songs', 'positive songs']
explode = (0, 0)
x_pos = np.arange(len(labels))
```

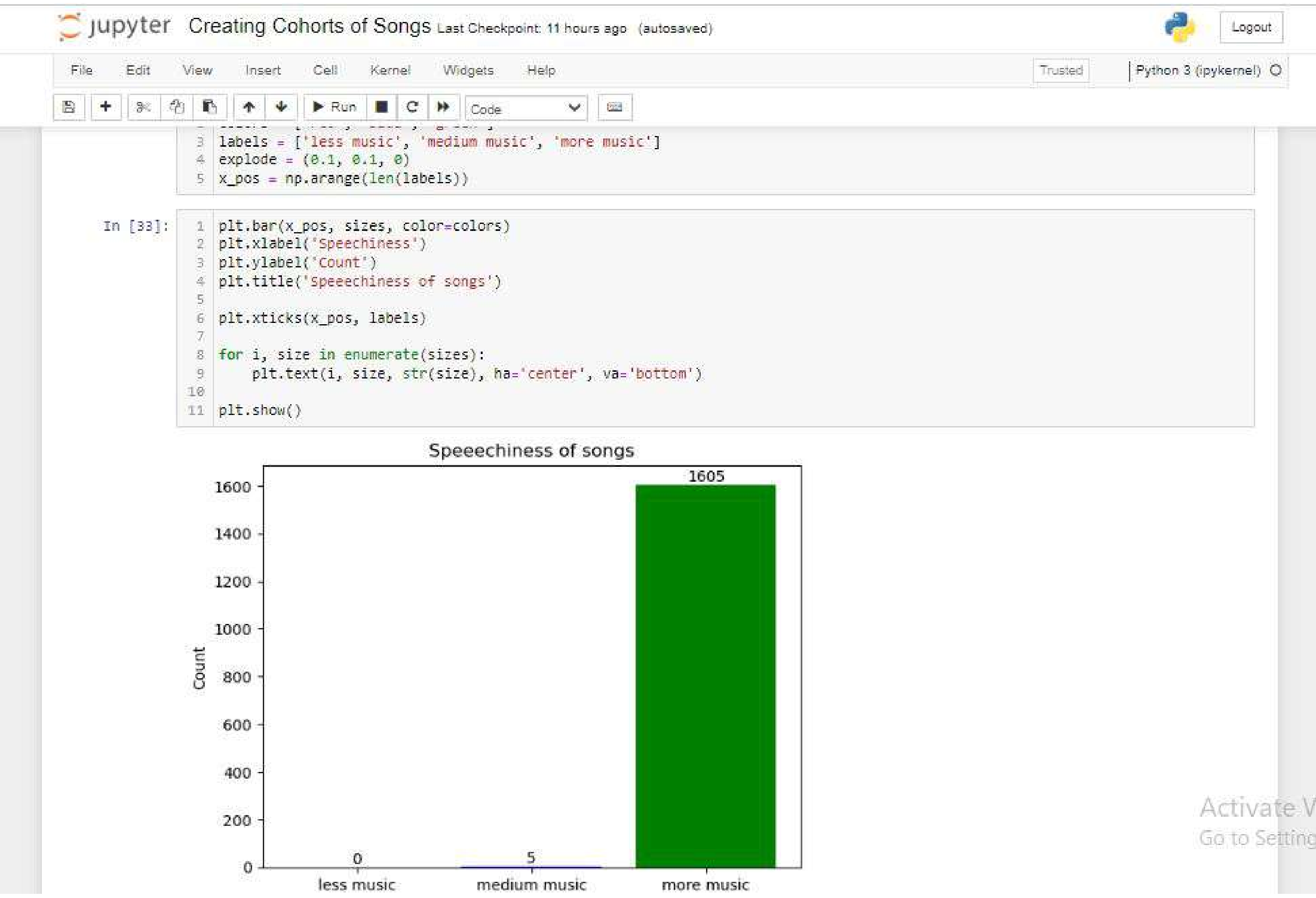
```
In [24]:
plt.pie(sizes, explode=explode, labels=labels, colors=colors, autopct='%1.1f%%', startangle=90)
plt.axis('equal')
plt.title('Valence of songs')
plt.show()
```

Valence of songs
negative songs
37.1%
62.9%
positive songs


jupyter
Creating Cohorts of Songs Last Checkpoint: 11 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code

```
In [25]:
plt.bar(x_pos, sizes, color=colors)
plt.xlabel('Valence')
plt.ylabel('Count')
plt.title('Valence of songs')

plt.xticks(x_pos, labels)

for i, size in enumerate(sizes):
    plt.text(i, size, str(size), ha='center', va='bottom')

plt.show()
```

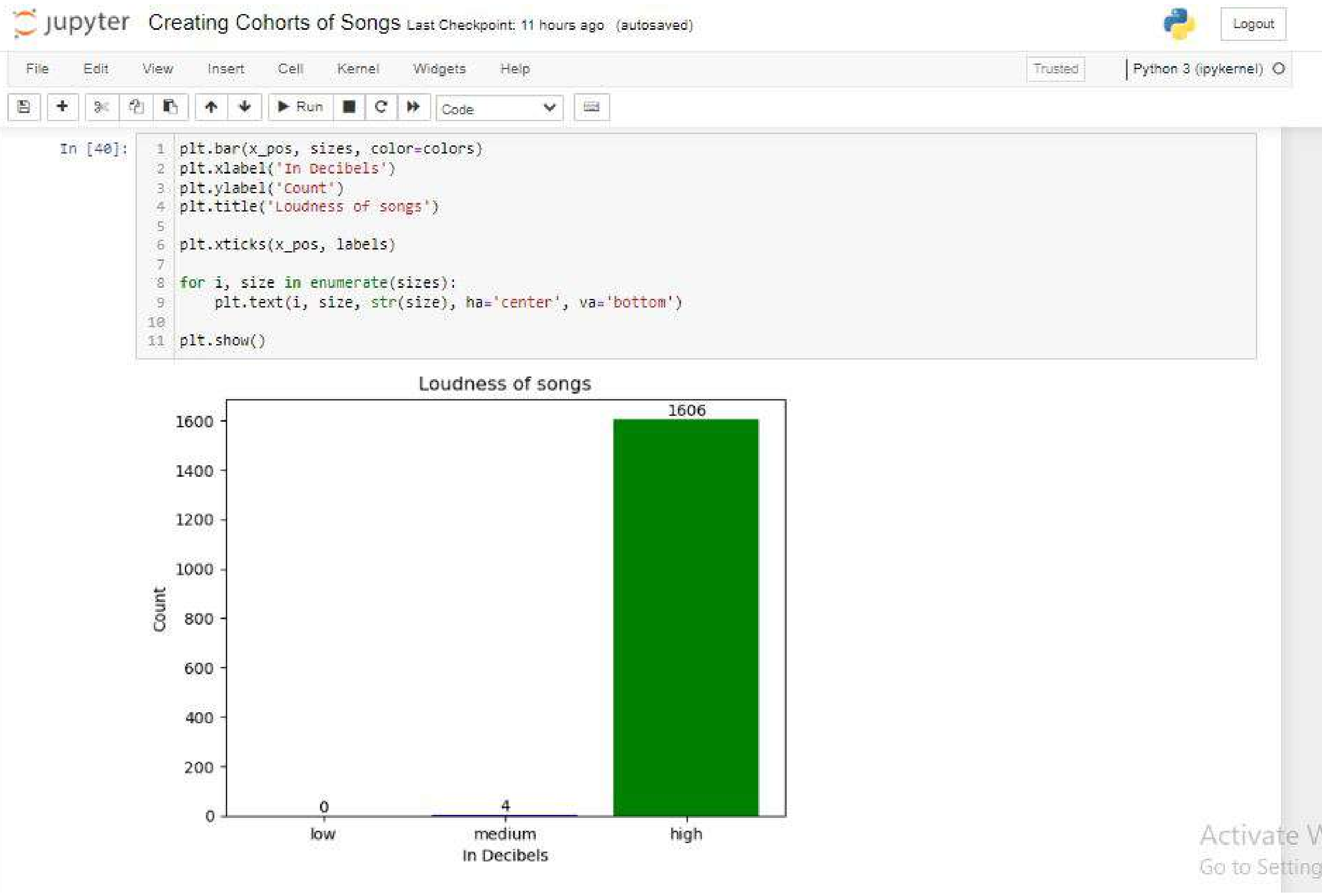




```
labels = ['less music', 'medium music', 'more music']
explode = (0.1, 0.1, 0)
x_pos = np.arange(len(labels))
```

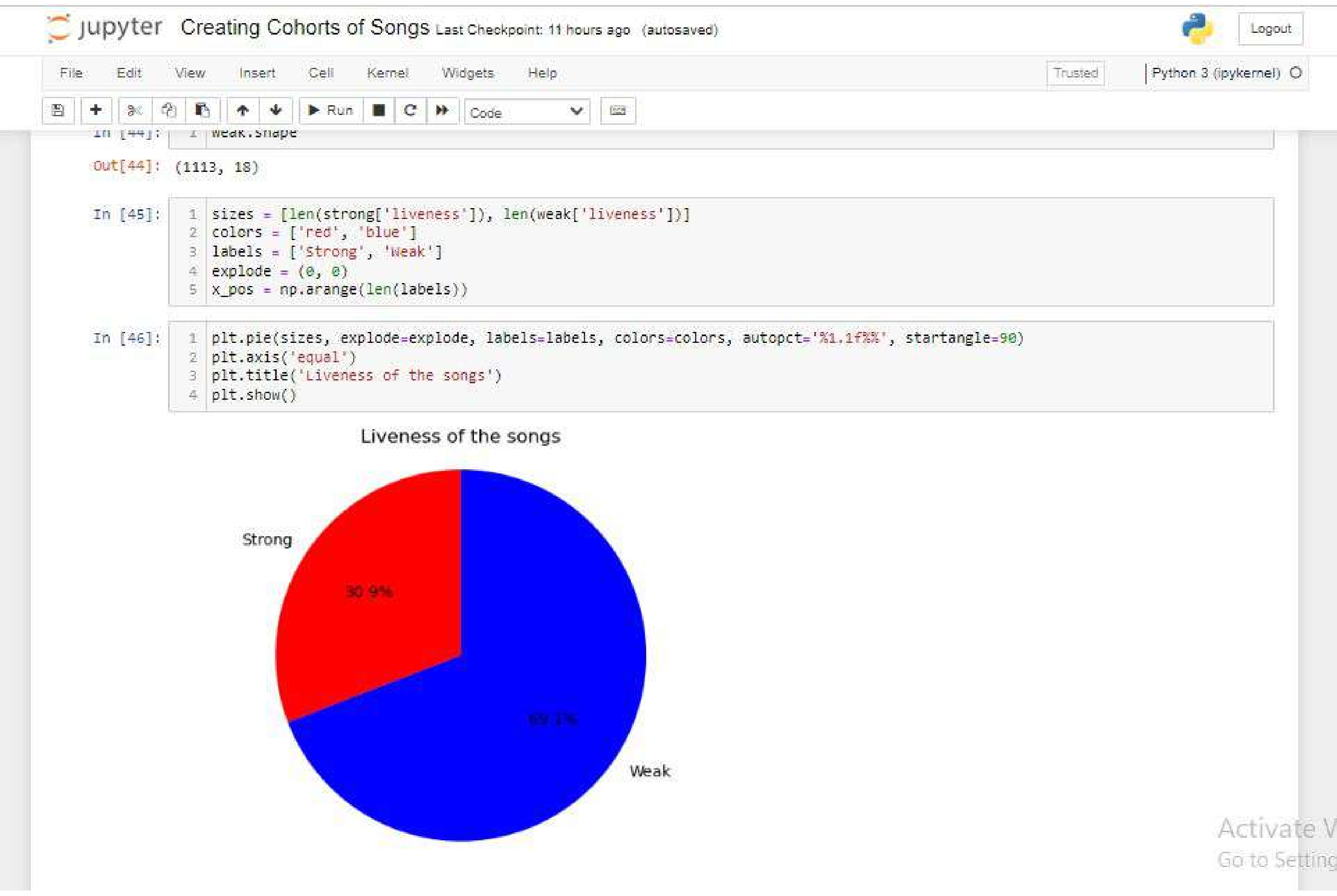
In [33]:

```
plt.bar(x_pos, sizes, color=colors)
plt.xlabel('Speechiness')
plt.ylabel('Count')
plt.title('Speeechiness of songs')

plt.xticks(x_pos, labels)

for i, size in enumerate(sizes):
    plt.text(i, size, str(size), ha='center', va='bottom')

plt.show()
```



jupyter
Creating Cohorts of Songs Last Checkpoint: 11 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code
In [40]:

```
plt.bar(x_pos, sizes, color=colors)
plt.xlabel('In Decibels')
plt.ylabel('Count')
plt.title('Loudness of songs')

plt.xticks(x_pos, labels)

for i, size in enumerate(sizes):
    plt.text(i, size, str(size), ha='center', va='bottom')

plt.show()
```

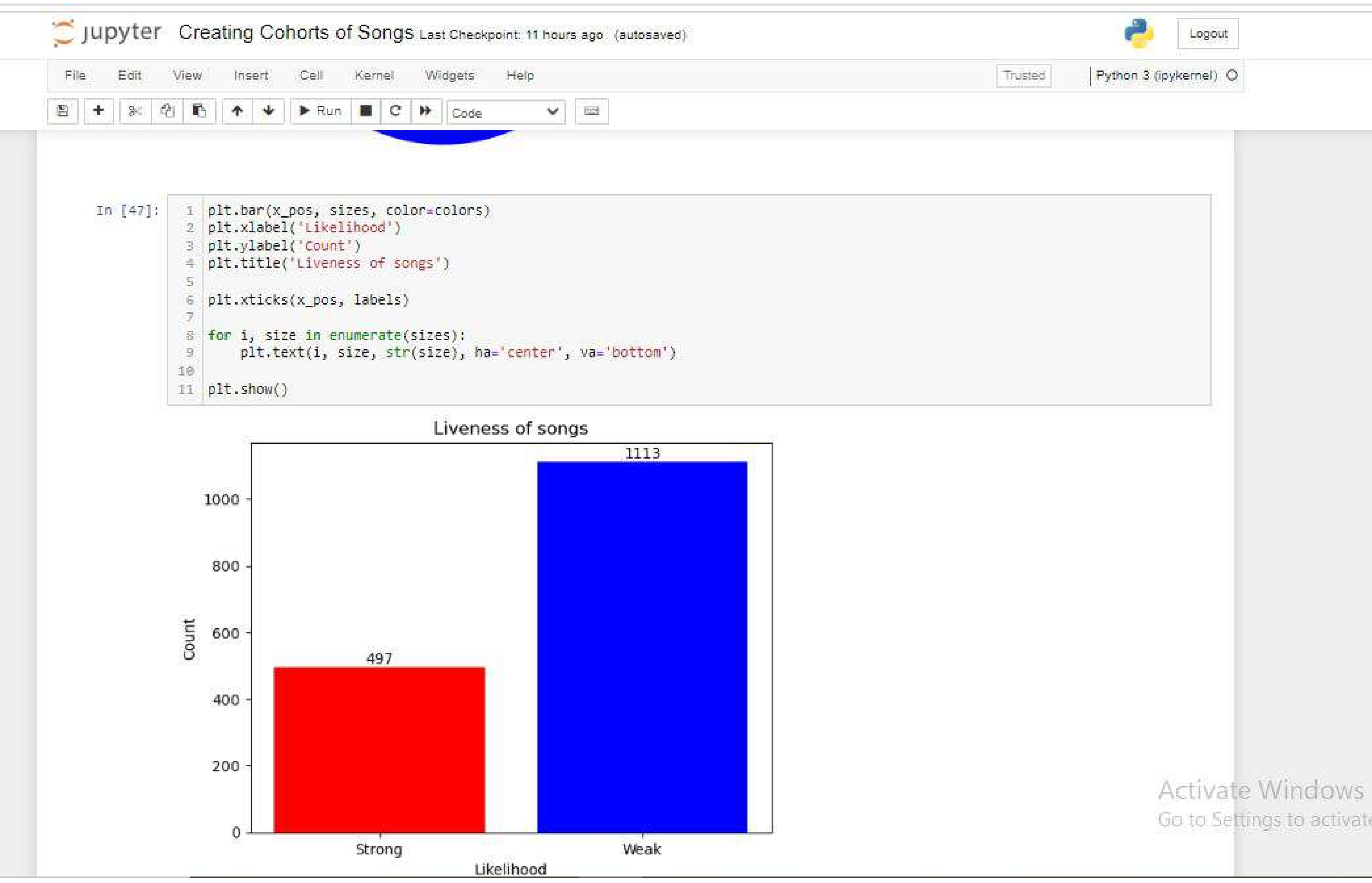


```
In [44]: weak.shape
```

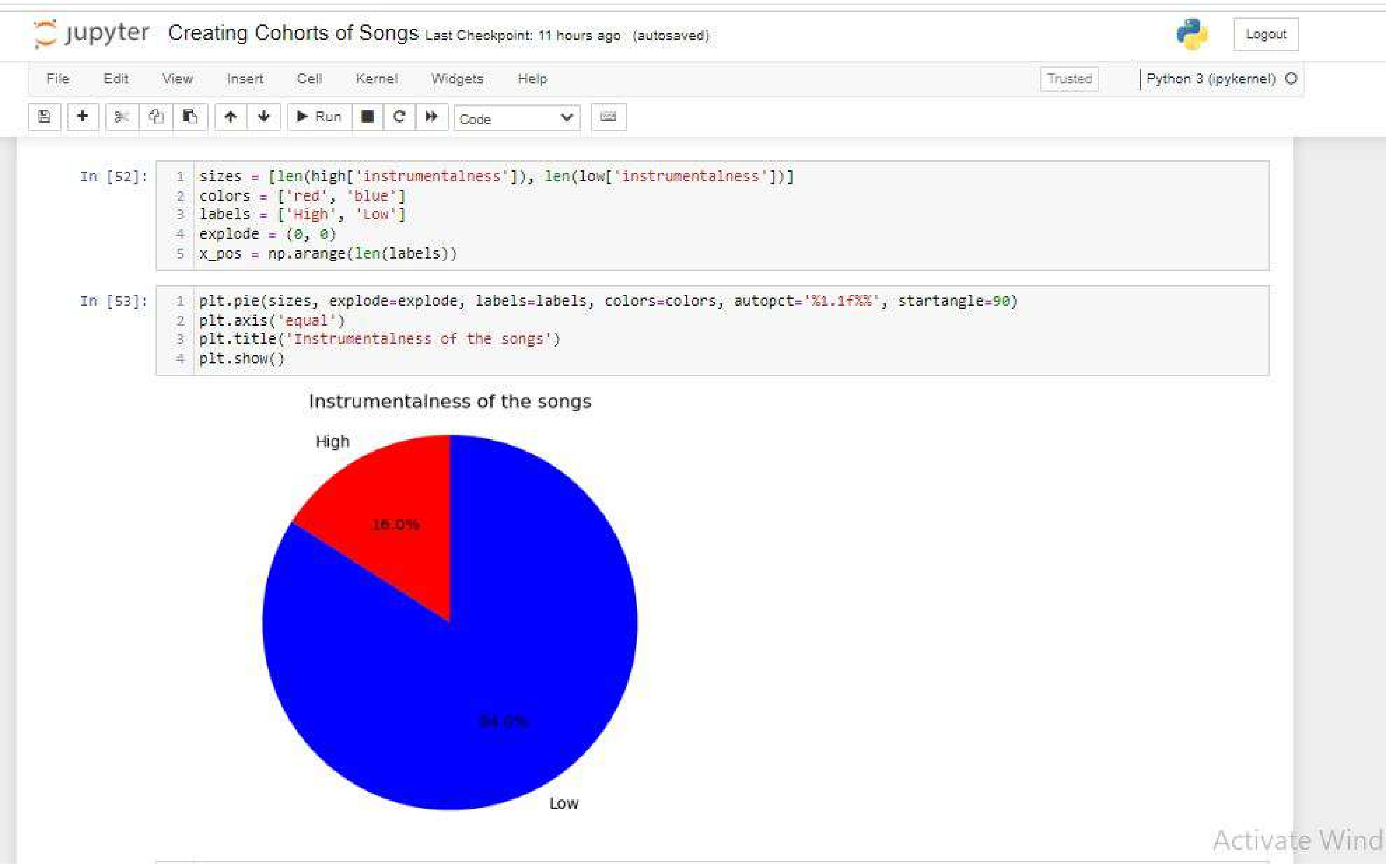
```
Out[44]: (1113, 18)
```

```
In [45]:
sizes = [len(strong['liveness']), len(weak['liveness'])]
colors = ['red', 'blue']
labels = ['Strong', 'Weak']
explode = (0, 0)
x_pos = np.arange(len(labels))
```

```
In [46]:
plt.pie(sizes, explode=explode, labels=labels, colors=colors, autopct='%1.1f%%', startangle=90)
plt.axis('equal')
plt.title('Liveness of the songs')
plt.show()
```

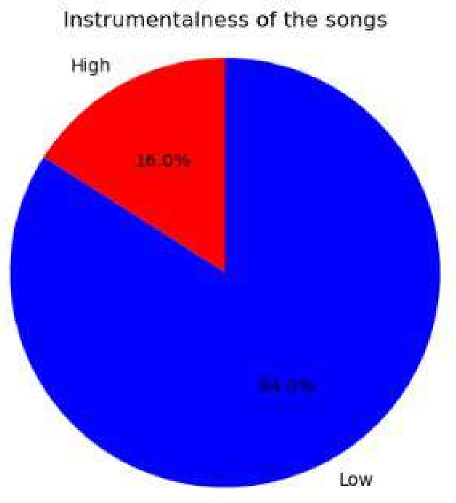


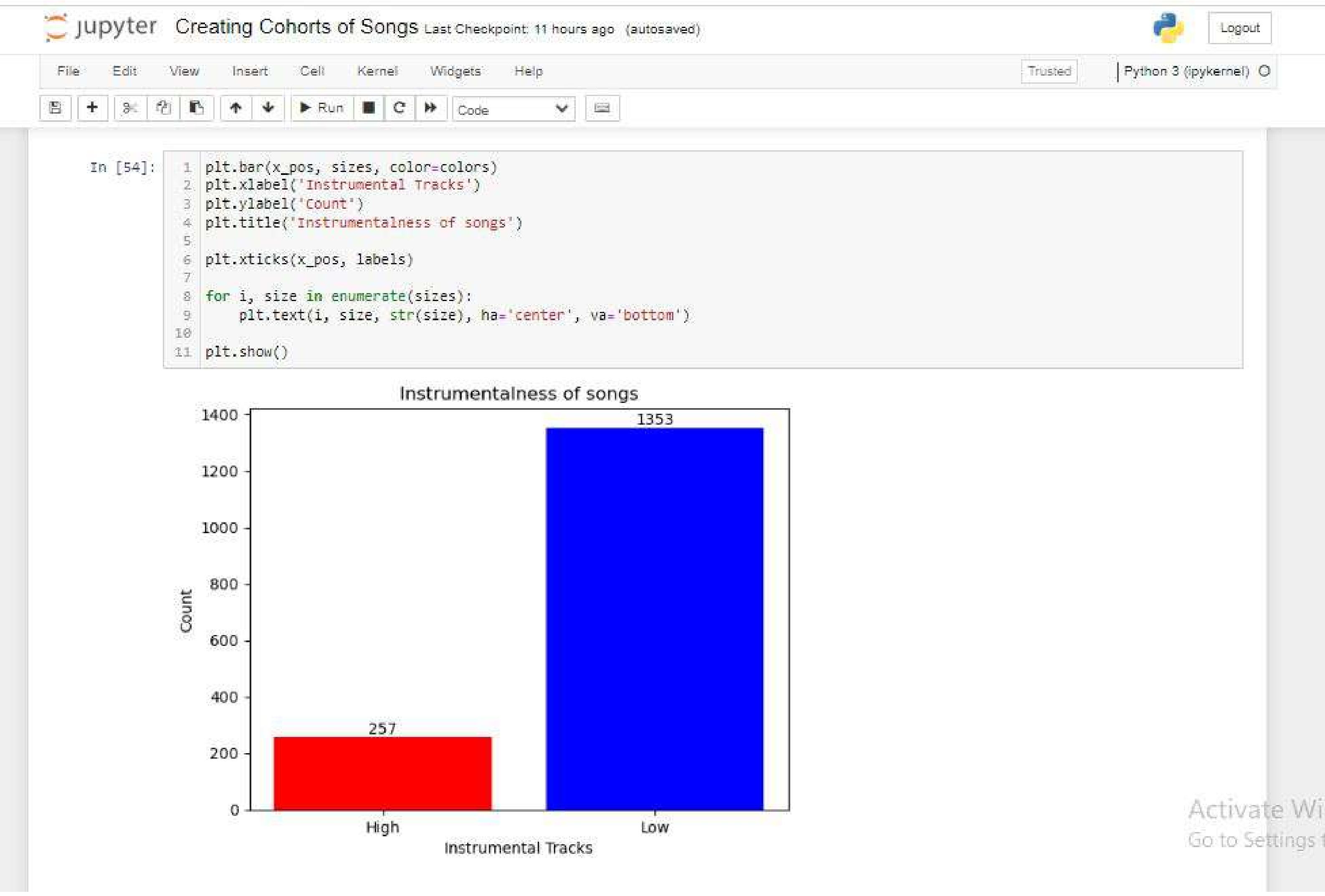
```
plt.bar(x_pos, sizes, color=colors)
plt.xlabel('Likelihood')
plt.ylabel('Count')
plt.title('Liveness of songs')

plt.xticks(x_pos, labels)

for i, size in enumerate(sizes):
    plt.text(i, size, str(size), ha='center', va='bottom')

plt.show()
```


jupyter
Creating Cohorts of Songs Last Checkpoint: 11 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code
Activate Wind

```
In [52]: sizes = [len(high['instrumentalness']), len(low['instrumentalness'])]
         colors = ['red', 'blue']
         labels = ['High', 'Low']
         explode = (0, 0)
         x_pos = np.arange(len(labels))
```

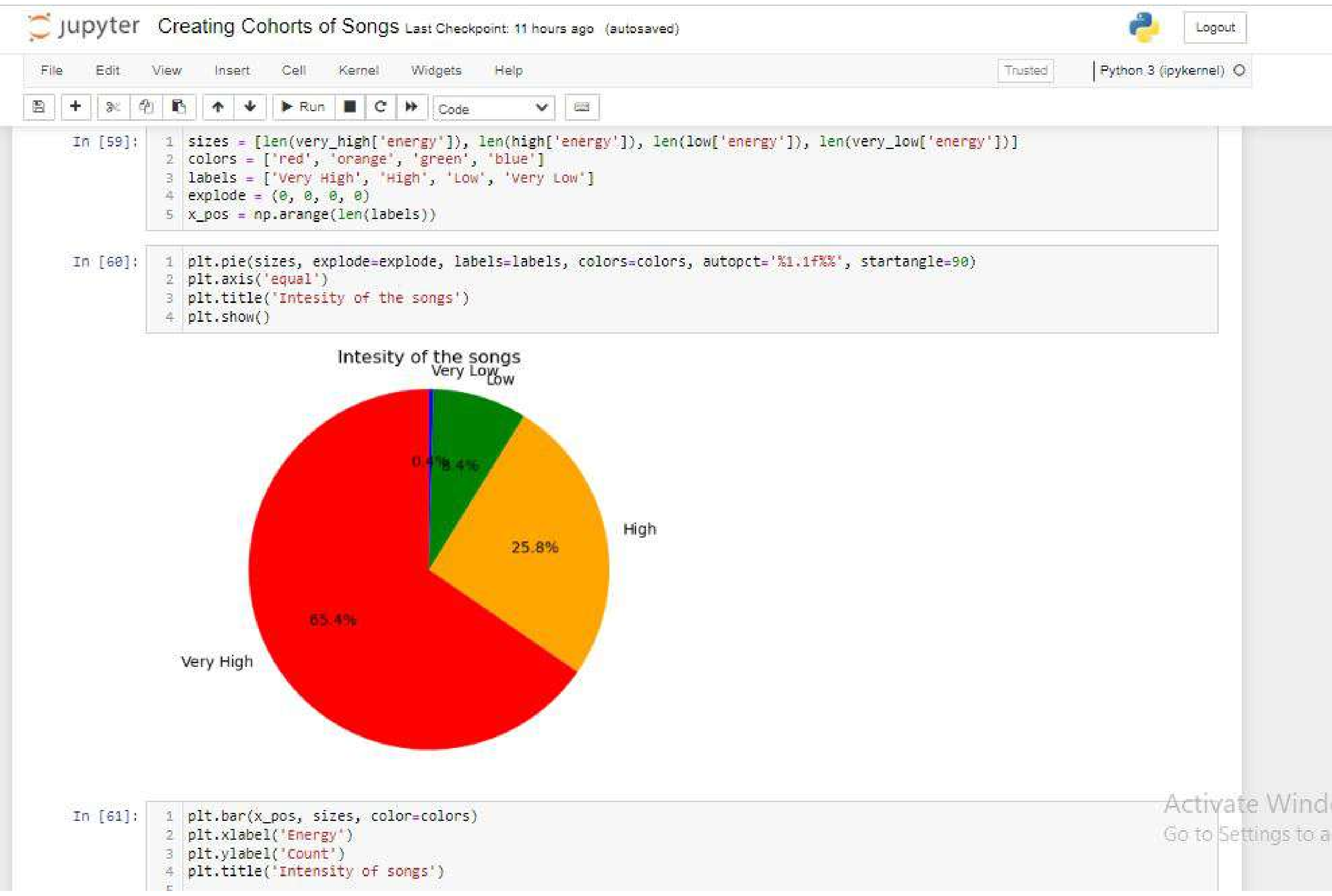
```
In [53]: plt.pie(sizes, explode=explode, labels=labels, colors=colors, autopct='%1.1f%%', startangle=90)
         plt.axis('equal')
         plt.title('Instrumentalness of the songs')
         plt.show()
```





In [54]:

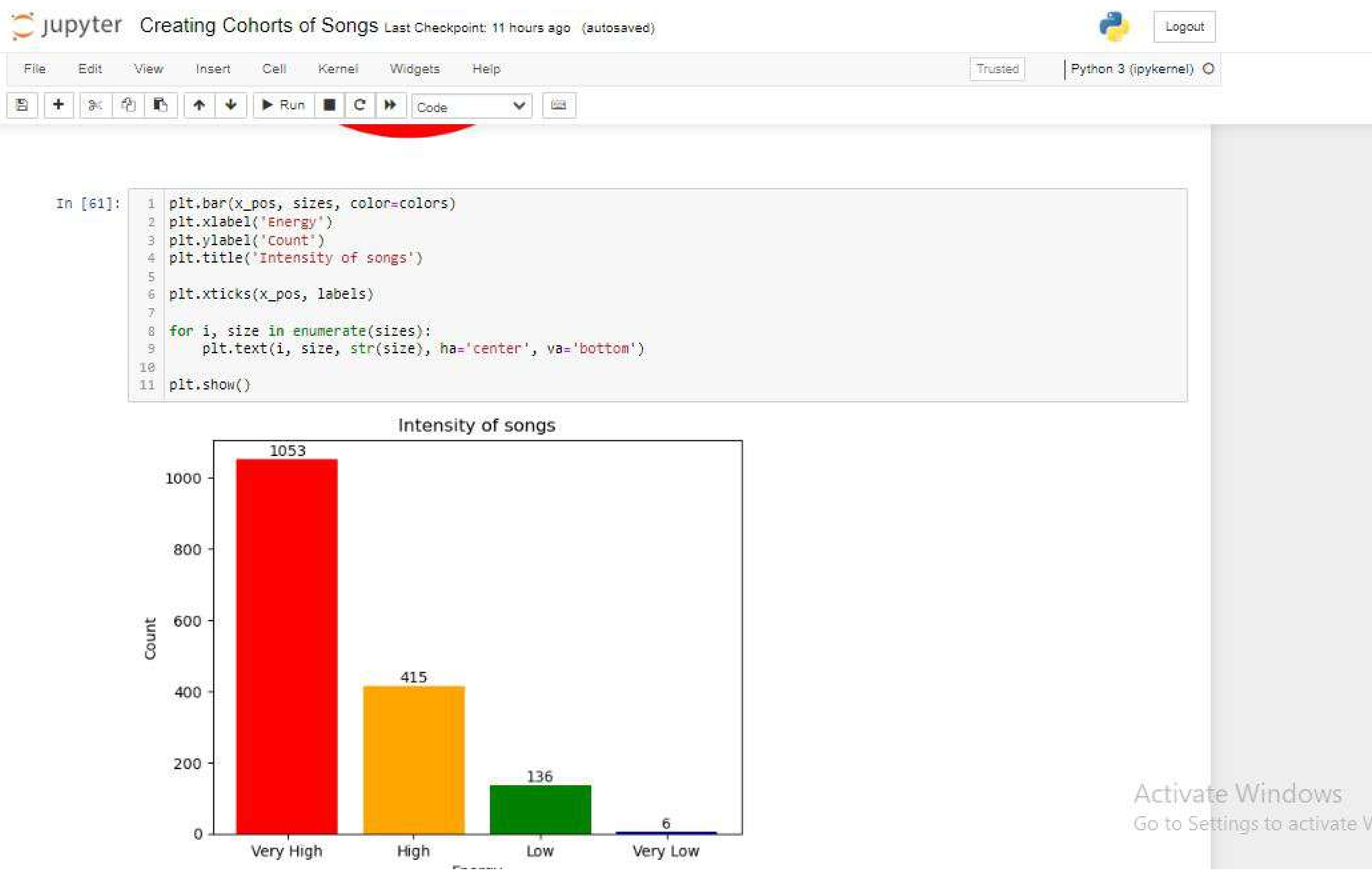
```python
plt.bar(x_pos, sizes, color=colors)
plt.xlabel('Instrumental Tracks')
plt.ylabel('Count')
plt.title('Instrumentalness of songs')

plt.xticks(x_pos, labels)

for i, size in enumerate(sizes):
    plt.text(i, size, str(size), ha='center', va='bottom')

plt.show()
```



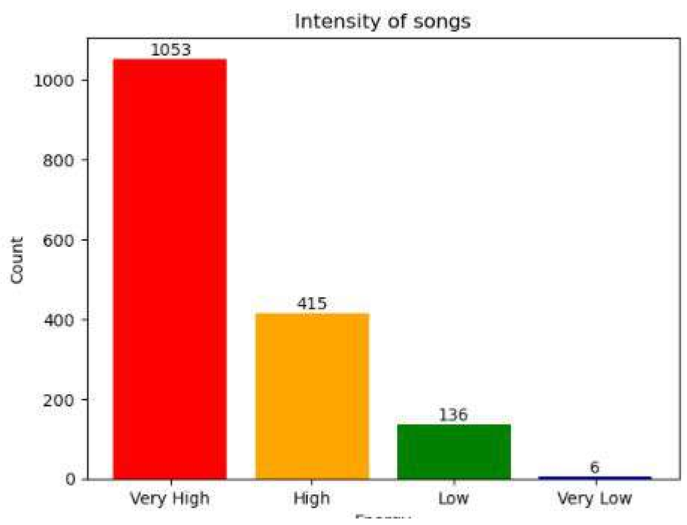
In [59]:
```python
sizes = [len(very_high['energy']), len(high['energy']), len(low['energy']), len(very_low['energy'])]
colors = ['red', 'orange', 'green', 'blue']
labels = ['Very High', 'High', 'Low', 'Very Low']
explode = (0, 0, 0, 0)
x_pos = np.arange(len(labels))
```

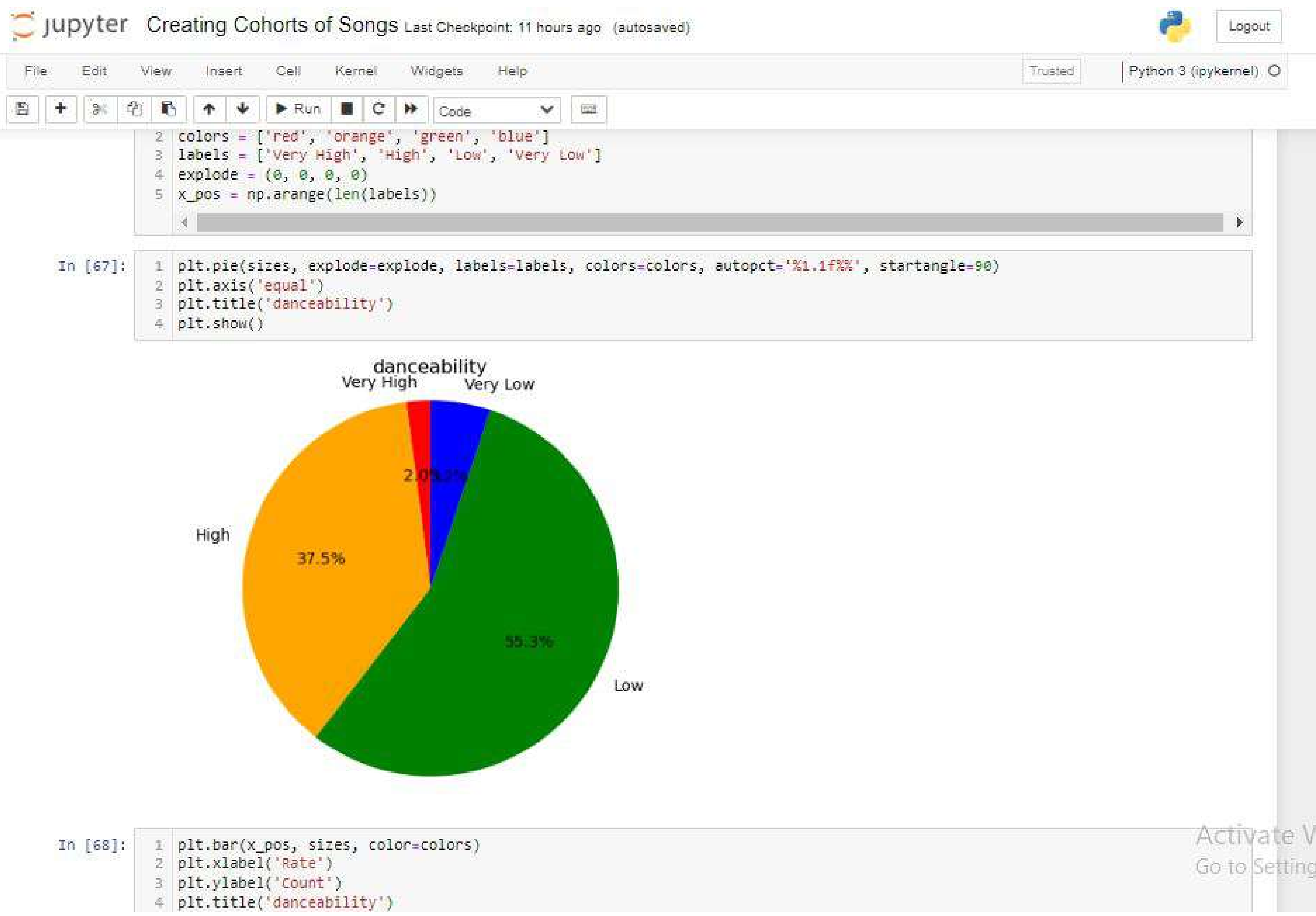
In [60]:
```python
plt.pie(sizes, explode=explode, labels=labels, colors=colors, autopct='%1.1f%%', startangle=90)
plt.axis('equal')
plt.title('Intesity of the songs')
plt.show()
```

In [61]:
```python
plt.bar(x_pos, sizes, color=colors)
plt.xlabel('Energy')
plt.ylabel('Count')
plt.title('Intensity of songs')
```



In [61]:
```python
plt.bar(x_pos, sizes, color=colors)
plt.xlabel('Energy')
plt.ylabel('Count')
plt.title('Intensity of songs')

plt.xticks(x_pos, labels)

for i, size in enumerate(sizes):
    plt.text(i, size, str(size), ha='center', va='bottom')

plt.show()
```




jupyter
Creating Cohorts of Songs Last Checkpoint: 11 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code

```
colors = ['red', 'orange', 'green', 'blue']
labels = ['Very High', 'High', 'Low', 'Very Low']
explode = (0, 0, 0, 0)
x_pos = np.arange(len(labels))
```

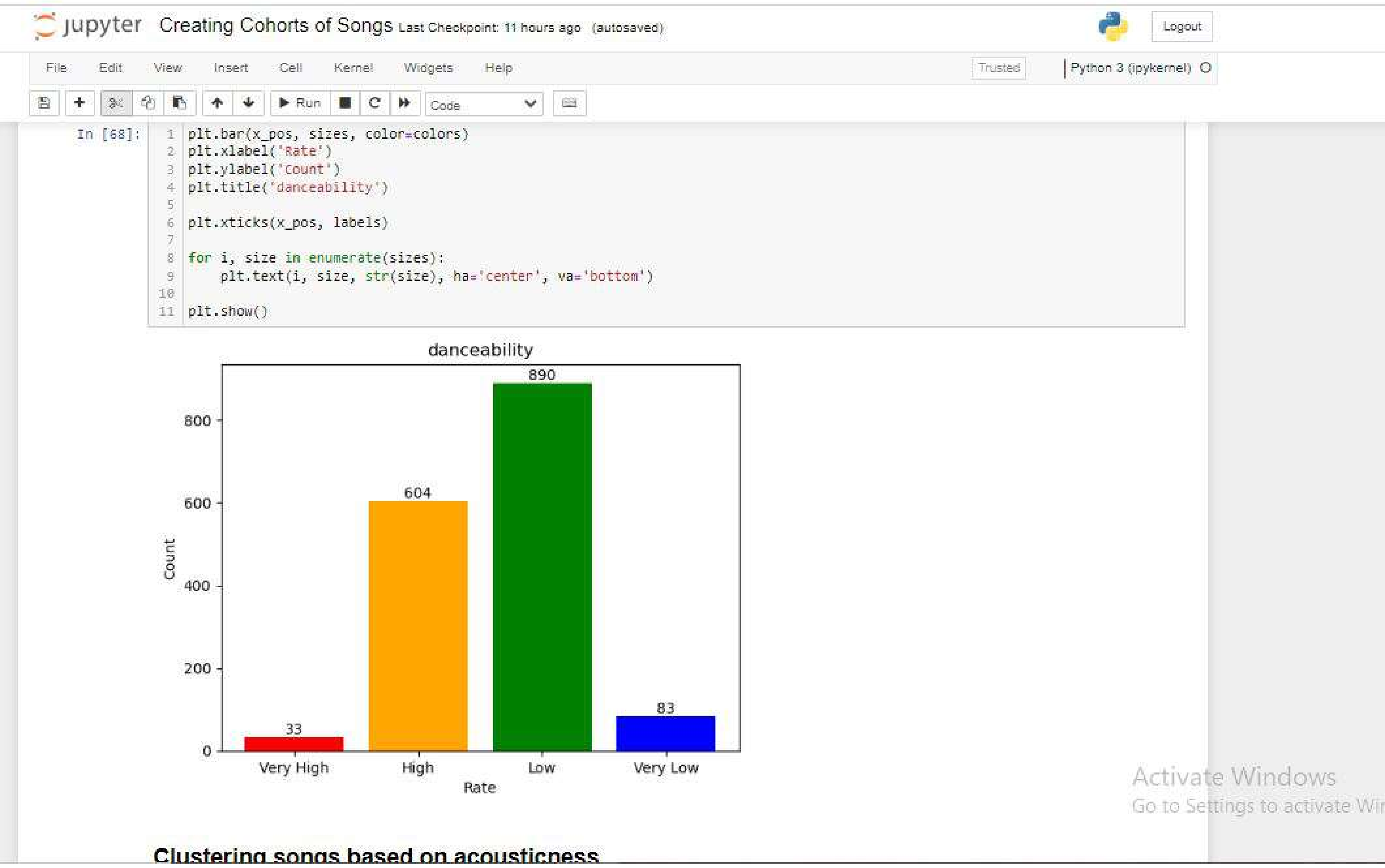
In [67]:

```
plt.pie(sizes, explode=explode, labels=labels, colors=colors, autopct='%1.1f%%', startangle=90)
plt.axis('equal')
plt.title('danceability')
plt.show()
```

danceability
Very High
Very Low
High
Low
2.0%
5.2%
37.5%
55.3%

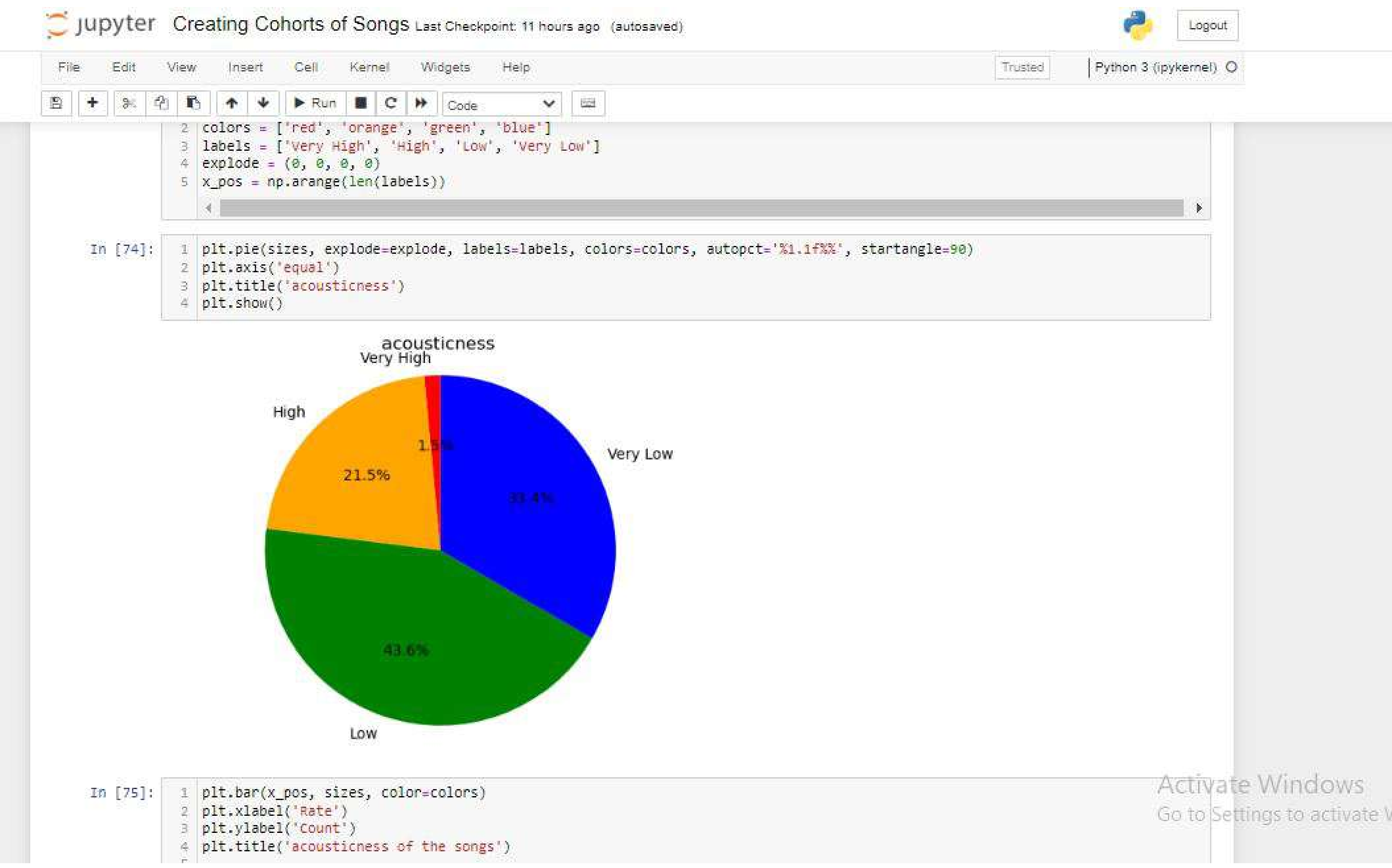
In [68]:

```
plt.bar(x_pos, sizes, color=colors)
plt.xlabel('Rate')
plt.ylabel('Count')
plt.title('danceability')
```



In [68]:
```
plt.bar(x_pos, sizes, color=colors)
plt.xlabel('Rate')
plt.ylabel('Count')
plt.title('danceability')

plt.xticks(x_pos, labels)

for i, size in enumerate(sizes):
    plt.text(i, size, str(size), ha='center', va='bottom')

plt.show()
```

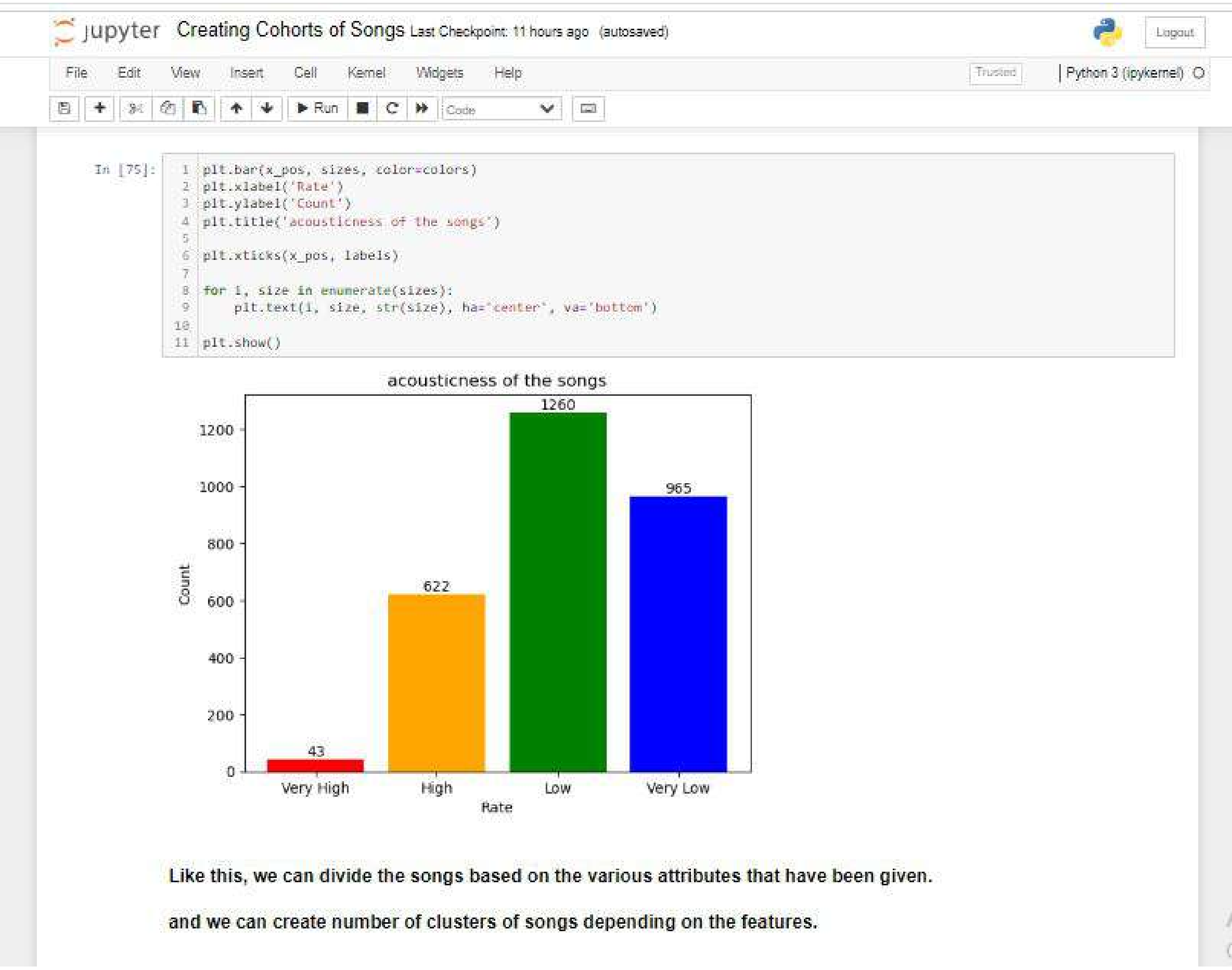
Clustering songs based on acousticness



```python
colors = ['red', 'orange', 'green', 'blue']
labels = ['Very High', 'High', 'Low', 'Very Low']
explode = (0, 0, 0, 0)
x_pos = np.arange(len(labels))
```

In [74]:
```python
plt.pie(sizes, explode=explode, labels=labels, colors=colors, autopct='%1.1f%%', startangle=90)
plt.axis('equal')
plt.title('acousticness')
plt.show()
```

In [75]:
```python
plt.bar(x_pos, sizes, color=colors)
plt.xlabel('Rate')
plt.ylabel('Count')
plt.title('acousticness of the songs')
```

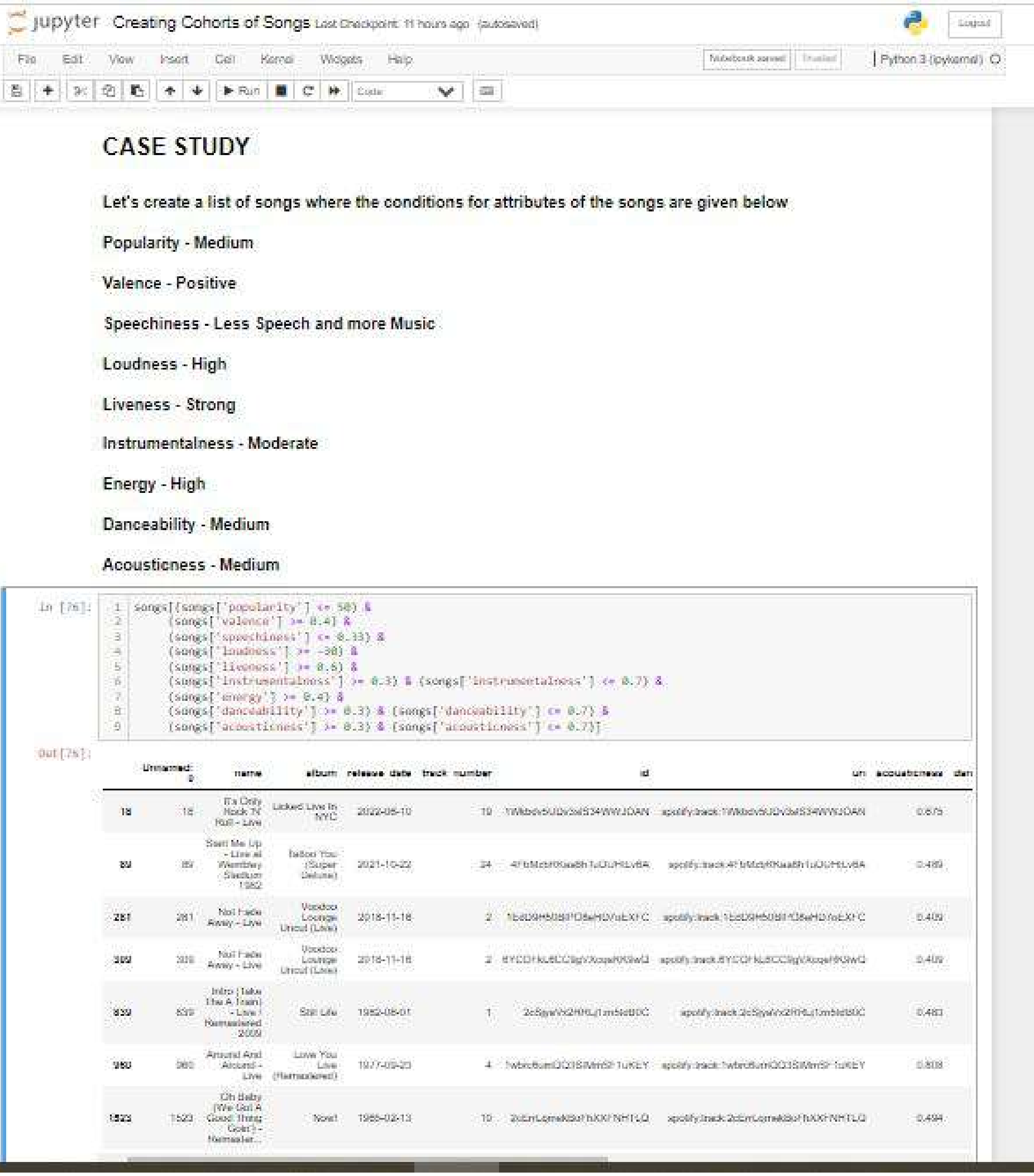


In [75]:

```
plt.bar(x_pos, sizes, color=colors)
plt.xlabel('Rate')
plt.ylabel('Count')
plt.title('acousticness of the songs')

plt.xticks(x_pos, labels)

for i, size in enumerate(sizes):
    plt.text(i, size, str(size), ha='center', va='bottom')

plt.show()
```

Like this, we can divide the songs based on the various attributes that have been given.

and we can create number of clusters of songs depending on the features.


jupyter Creating Cohorts of Songs Last Checkpoint: 11 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Notebook saved Trusted Python 3 (ipykernel)
Run Code

CASE STUDY

Let's create a list of songs where the conditions for attributes of the songs are given below

Popularity - Medium

Valence - Positive

Speechiness - Less Speech and more Music

Loudness - High

Liveness - Strong

Instrumentalness - Moderate

Energy - High

Danceability - Medium

Acousticness - Medium

In [76]:
```
songs[(songs['popularity'] <= 50) &
      (songs['valence'] >= 0.4) &
      (songs['speechiness'] <= 0.33) &
      (songs['loudness'] >= -30) &
      (songs['liveness'] >= 0.6) &
      (songs['instrumentalness'] >= 0.3) & (songs['instrumentalness'] <= 0.7) &
      (songs['energy'] >= 0.4) &
      (songs['danceability'] >= 0.3) & (songs['danceability'] <= 0.7) &
      (songs['acousticness'] >= 0.3) & (songs['acousticness'] <= 0.7)]
```

Out[76]:

	Unnamed: 0	name	album	release_date	track_number	id	uri	acousticness	dan
18	18	It's Only Rock 'N' Roll - Live	Licked Live In NYC	2022-06-10	19	1Wkbdv5UDv3al534WWJOAN	spotify:track:1Wkbdv5UDv3al534WWJOAN	0.675	
89	89	Start Me Up - Live at Wembley Stadium 1982	Tattoo You (Super Deluxe)	2021-10-22	24	4FbMzbfKaa8h1uOUHLvBA	spotify:track:4FbMzbfKaa8h1uOUHLvBA	0.489	
281	281	Not Fade Away - Live	Voodoo Lounge Uncut (Live)	2018-11-16	2	1EdD9H50BPO8eHDVoLXFC	spotify:track:1EdD9H50BPO8eHDVoLXFC	0.409	
309	309	Not Fade Away - Live	Voodoo Lounge Uncut (Live)	2018-11-16	2	8YCOFkL8CCBgVXopaRK9wQ	spotify:track:8YCOFkL8CCBgVXopaRK9wQ	0.409	
839	839	Intro (Take The A Train) - Live / Remastered 2009	Still Life	1982-06-01	1	2cSgjaVX2HKLj1mbIsB0C	spotify:track:2cSgjaVX2HKLj1mbIsB0C	0.483	
960	960	Around And Around - Live	Love You Live (Remastered)	1977-09-23	4	1wbn8umQO3SMmSF1uKEY	spotify:track:1wbn8umQO3SMmSF1uKEY	0.808	
1523	1523	Oh Baby (We Got A Good Thing Goin') - Remaster...	Now!	1965-02-13	10	2dzmLomeKBoFhXXFNH1LQ	spotify:track:2dzmLomeKBoFhXXFNH1LQ	0.494	