Exploring Mushrooms

When asked to think of a mushroom, you might imagine the following:

It's a beige, convex mushroom top with a uniform, thick stem. Maybe you thought of it cut up on a slice of delicious pizza or braised with sauce over rice. Regardless, you most likely did not consider this:

Mushrooms exist in a variety of different colors, shapes, sizes, textures, etc. In this project, you will analyze an extensive mushroom dataset from UCI using bar charts and acquaint yourself with the diverse array of mushrooms that exist worldwide.

Looking Over the Data

1. Take a look at the code block below where we've loaded **mushroom_data.csv**. It contains 23 columns of data describing thousands of mushrooms. In the output below the code block, data about five different mushrooms is shown.

   Read through this table to get a sense of the type(s) of variables in the data and the structure of the table. It may also be helpful to read through the information on Kaggle.

   Before you move on to plotting any of this data, answer the following questions:

   - What type(s) of variables does **mushroom_data.csv** contain?
   - How many of the variables can we visualize effectively with a bar graph?

   View **Exploring Mushrooms_Solution.ipynb** to see our answers.

In [2]:
```
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns

# load in the data
df = pd.read_csv("mushroom_data.csv")
print(df.head())
```

```
      Class Cap Shape Cap Surface Cap Color  Bruises     Odor  \
0  Poisonous    Convex      Smooth     Brown     True  Pungent
1     Edible    Convex      Smooth    Yellow     True   Almond
2     Edible      Bell      Smooth     White     True    Anise
3  Poisonous    Convex       Scaly     White     True  Pungent
4     Edible    Convex      Smooth      Gray    False      NaN

  Gill Attachment Gill Spacing Gill Size Gill Color  ...  \
0            Free        Close    Narrow      Black  ...
1            Free        Close     Broad      Black  ...
2            Free        Close     Broad      Brown  ...
3            Free        Close    Narrow      Brown  ...
4            Free      Crowded     Broad      Black  ...

  Stalk Surface Below Ring Stalk Color Above Ring Stalk Color Below Ring  \
0                   Smooth                  White                  White
1                   Smooth                  White                  White
2                   Smooth                  White                  White
3                   Smooth                  White                  White
4                   Smooth                  White                  White

  Veil Type Veil Color Ring Number   Ring Type Spore Print Color Population  \
0   Partial      White         One     Pendant             Black  Scattered
1   Partial      White         One     Pendant             Brown   Numerous
2   Partial      White         One     Pendant             Brown   Numerous
3   Partial      White         One     Pendant             Black  Scattered
4   Partial      White         One  Evanescent             Brown   Abundant

   Habitat
0    Urban
1  Grasses
2  Meadows
3    Urban
4  Grasses

[5 rows x 23 columns]
```

2. There are 23 variables in this dataset (one for each column). One of them (the `Bruises` variable) has `True` or `False` responses. This will create problems when we try to plot this column later on.

   One way we can fix this issue is by converting each `True` and `False` value to a string. Iterate through the elements in the `Bruises` variable and convert each value to a string using the `str()` method.

   Now, all our variables are of `object` types, which means we can graph them. Graphing each one individually would be tedious; luckily, you will use loops.

   If you look at the code block below, you will see an attribute called `columns`. This attribute returns the name of each variable in **mushroom_data.csv**.

   - Create a loop that traverses each `column` in the `columns` list.
   - Print each `column` in `columns` while iterating through the loop. This is to check that your `for` loop is working correctly.

In [3]:
```
# list of all column headers

for index in range(0, len(df['Bruises'])):
```

```
    df['Bruises'][index] = str(df['Bruises'][index])
columns = df.columns.tolist()

for column in columns:
    print(column)
```

```
Class
Cap Shape
Cap Surface
Cap Color
Bruises
Odor
Gill Attachment
Gill Spacing
Gill Size
Gill Color
Stalk Shape
Stalk Root
Stalk Surface Above Ring
Stalk Surface Below Ring
Stalk Color Above Ring
Stalk Color Below Ring
Veil Type
Veil Color
Ring Number
Ring Type
Spore Print Color
Population
Habitat
```

```
C:\Users\xre22\AppData\Local\Temp\ipykernel_13584\3484195794.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#retu
rning-a-view-versus-a-copy
  df['Bruises'][index] = str(df['Bruises'][index])
C:\Users\xre22\AppData\Local\Temp\ipykernel_13584\3484195794.py:4: FutureWarning: Setting an item of incompatibl
e dtype is deprecated and will raise in a future error of pandas. Value 'True' has dtype incompatible with bool,
please explicitly cast to a compatible dtype first.
  df['Bruises'][index] = str(df['Bruises'][index])
```

3. In the output above, you should see 23 column names pop up starting with `class` and ending with `habitat`.

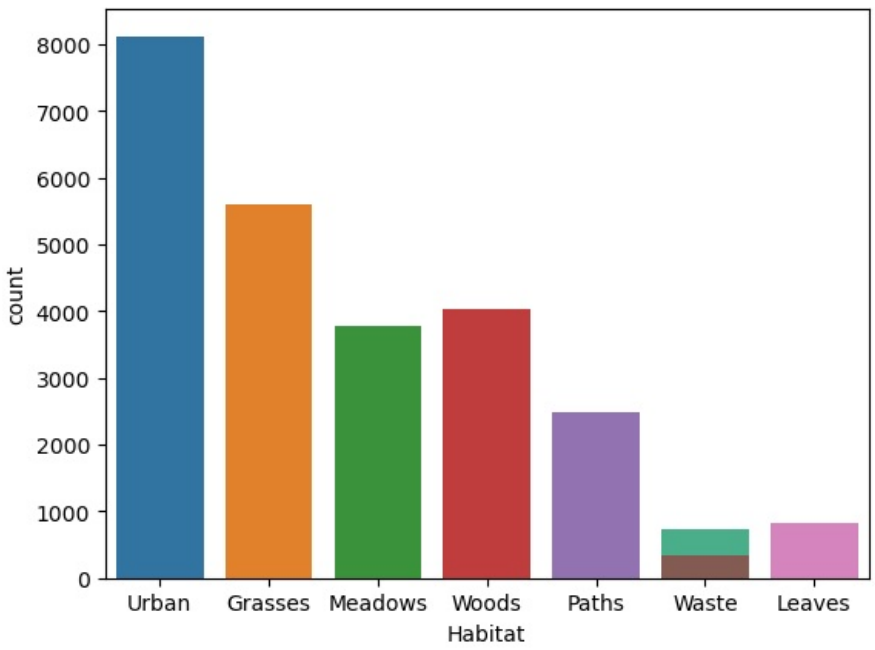
   Great! Your `for` loop is working, so feel free to comment out your print statement.

   You can now plot your data using the `.countplot()` method from the seaborn library. Follow these steps:

   - Call `.countplot()` in the `for` loop
   - Use `column` and the `df` pandas DataFrame to graph the value counts of each variable in **mushroom_data.csv**.

   Please wait until the next task to use `plt.show()`.

In [4]:
```
for i in columns:
    sns.countplot(x=i, data=df)
```

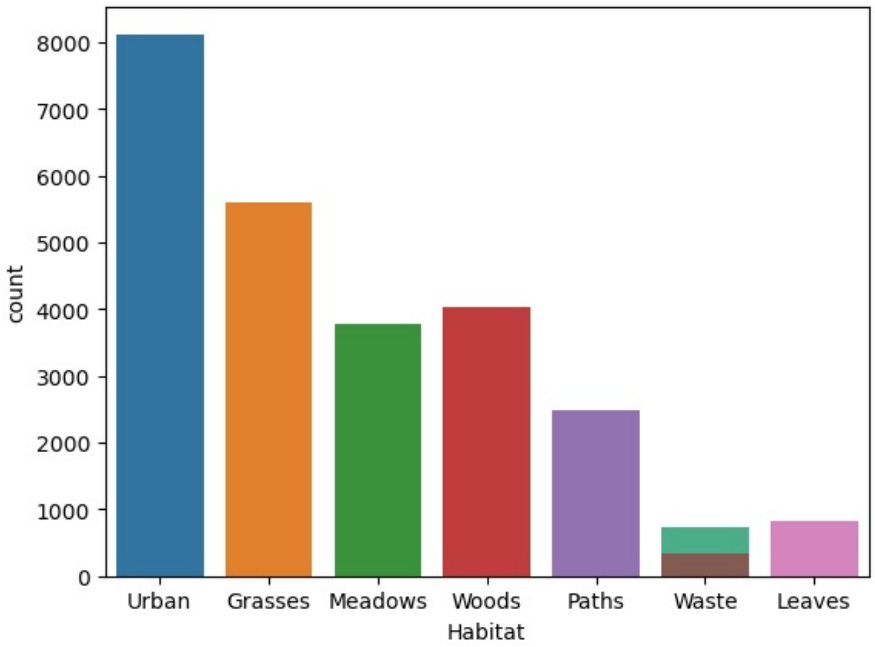


4. At the end of your `for` loop, add the following lines of code to show your plots:

```
plt.show()
plt.clf()
```

The `.show()` Matplotlib method should look familiar from the previous lesson, but .clf() might be unfamiliar. This method is also from the Matplotlib library. It clears any previous figure formatting. This will keep any graphs you are plotting from bunching up on each other. Instead, your plots will be neatly stacked on top of each other with spacing ideal for viewing.

In [5]:
```
for column in columns:
    sns.countplot(x=column, data=df)
```



5. After using `plt.show()` and `plt.clf()`, 23 plots should appear in the output below the previous code block. Scroll through each of the graphs, and see what sort of trends you immediately see.

   - What variables have an obvious mode?
   - Do any of them have a notably diverse array of values?
   - What habitat are you most likely to find mushrooms in?

   What questions did you have before seeing the graphs? What questions are popping up now that you see them?

   In the next few steps, you will clean up the graphs and make them more readable and useful for finding patterns.

Cleaning the Bar Graphs

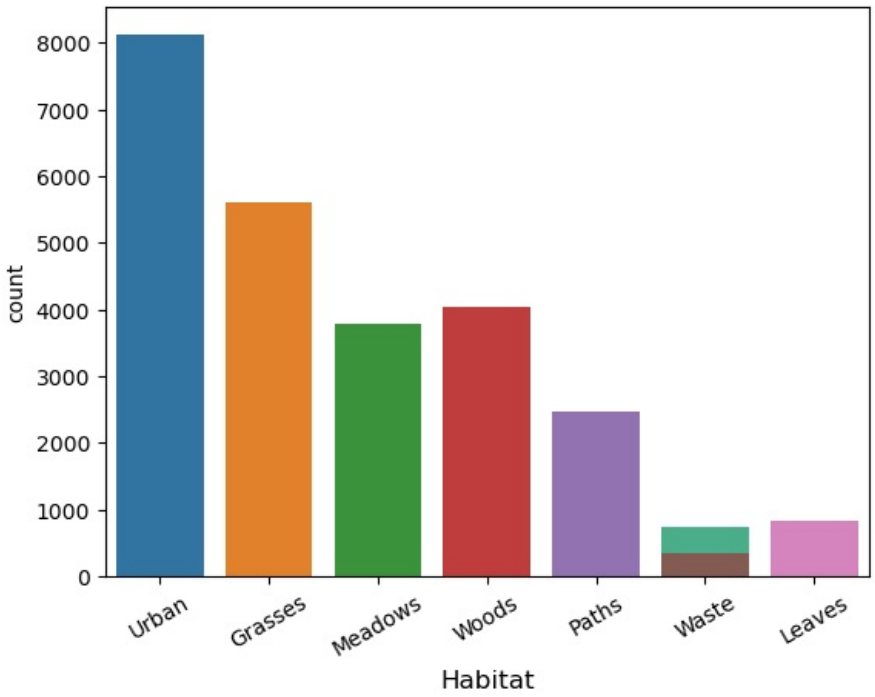
6. As you scroll through the graphs, you may notice some imperfections. For example, some of the x-axis labels overlap each other. The font size for the labels along the x-axis is also pretty small, making them tough to read.

   Let's fix these up with two lines of code.

   Following your `.countplot()` method, add the following two lines of code in your `for` loop:

   ```
   # rotates the value labels slightly so they don't overlap, also slightly increases font size
   plt.xticks(rotation=30, fontsize=10)
   # increases the variable label font size slightly to increase readability
   plt.xlabel(column, fontsize=12)
   ```

In [6]:
```
for column in columns:
    sns.countplot(x=column, data=df)
    plt.xticks(rotation=30, fontsize=10)
    plt.xlabel(column, fontsize=12)
```

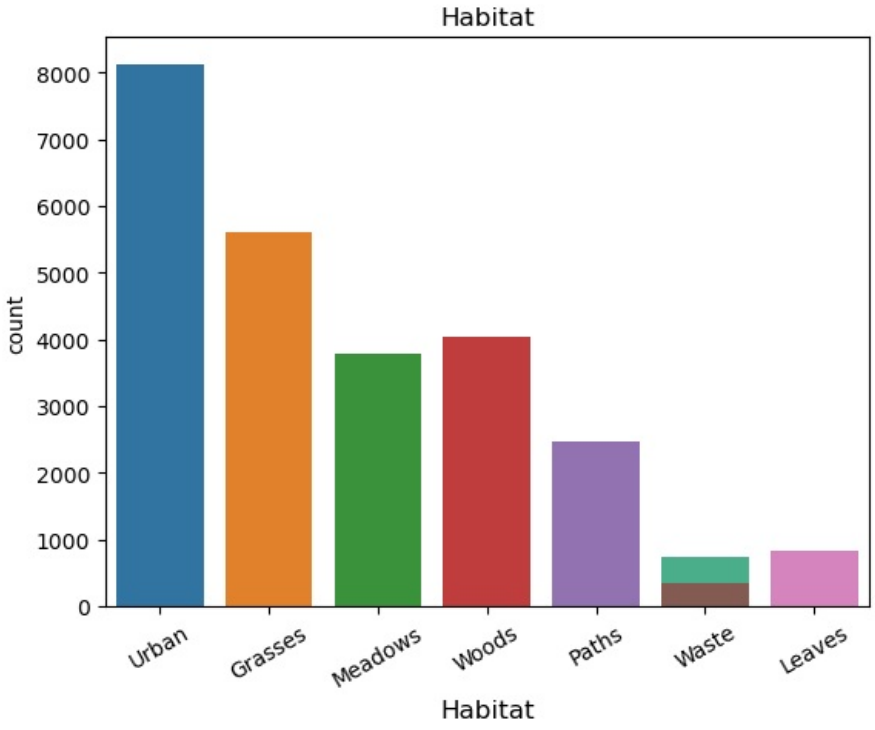


7. One more thing you can do to increase readability is to add an informative title. Using `.title()` from the Matplotlib library, give your graph the following title:

   `{Variable Name} Value Counts`

   Use `column` to capture each column name. Be sure to call this method after `.countplot()` inside of your `for` loop.

In [7]:
```
for column in columns:
    sns.countplot(x=column, data=df)
    plt.xticks(rotation=30, fontsize=10)
    plt.xlabel(column, fontsize=12)
    plt.title(column)
```



Ordering the Bars for Analysis

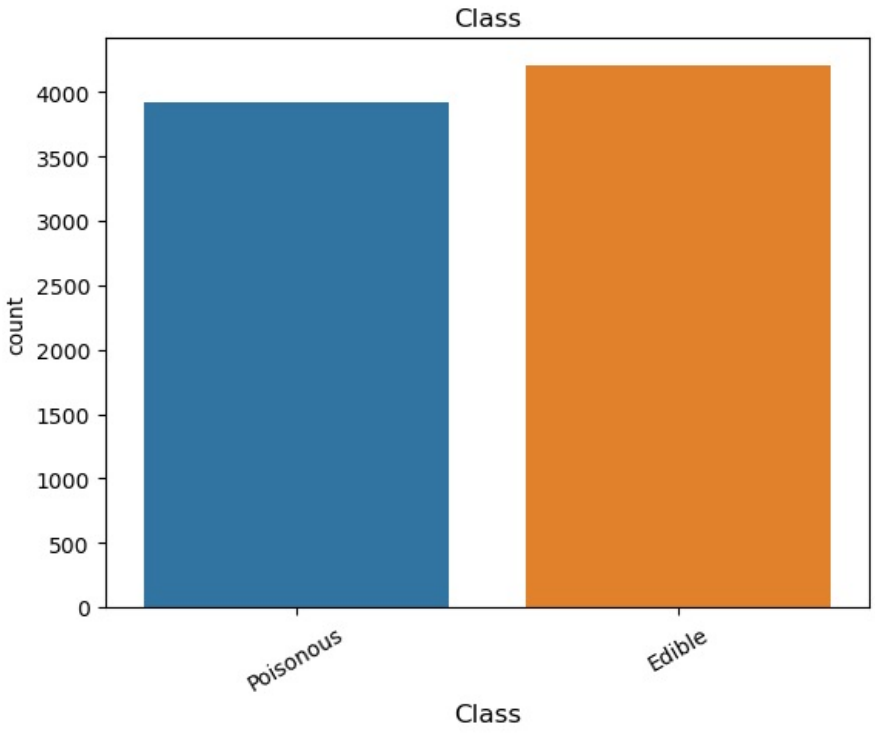
8. The graphs are readable, but you can take it another step further.

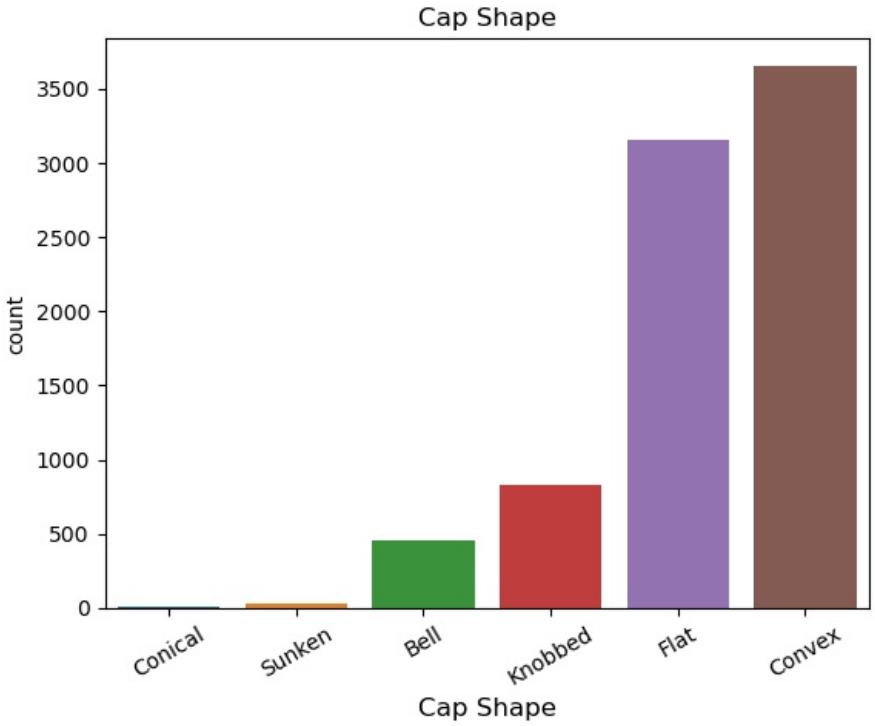
   - Add the `order` parameter to your `.countplot()` method.
   - Set the parameter to that the value counts in each column are in descending order.

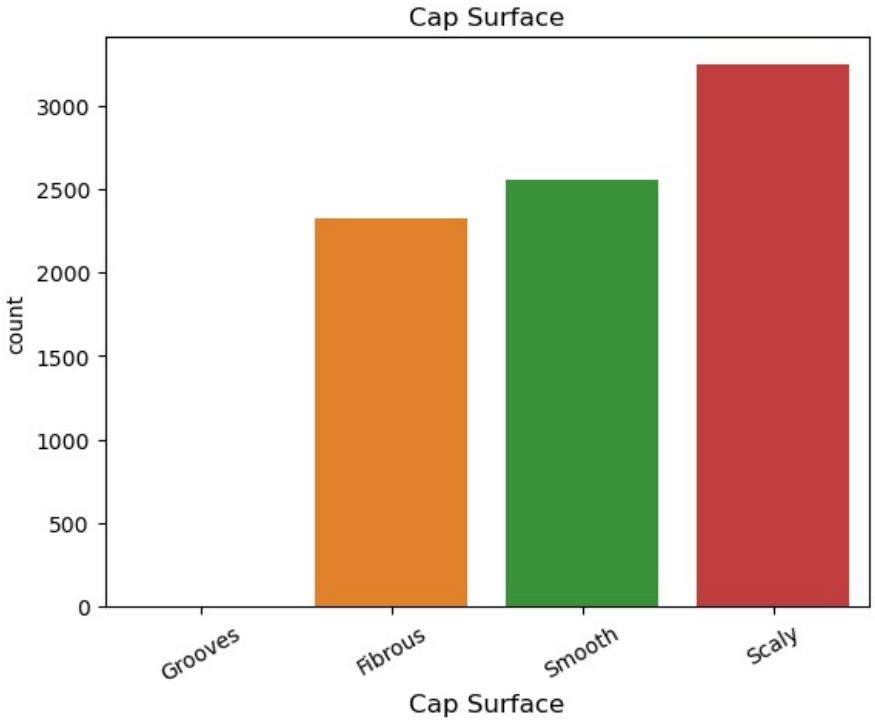
   You will need to use the `.value_counts()` pandas method and the `.index` pandas object.

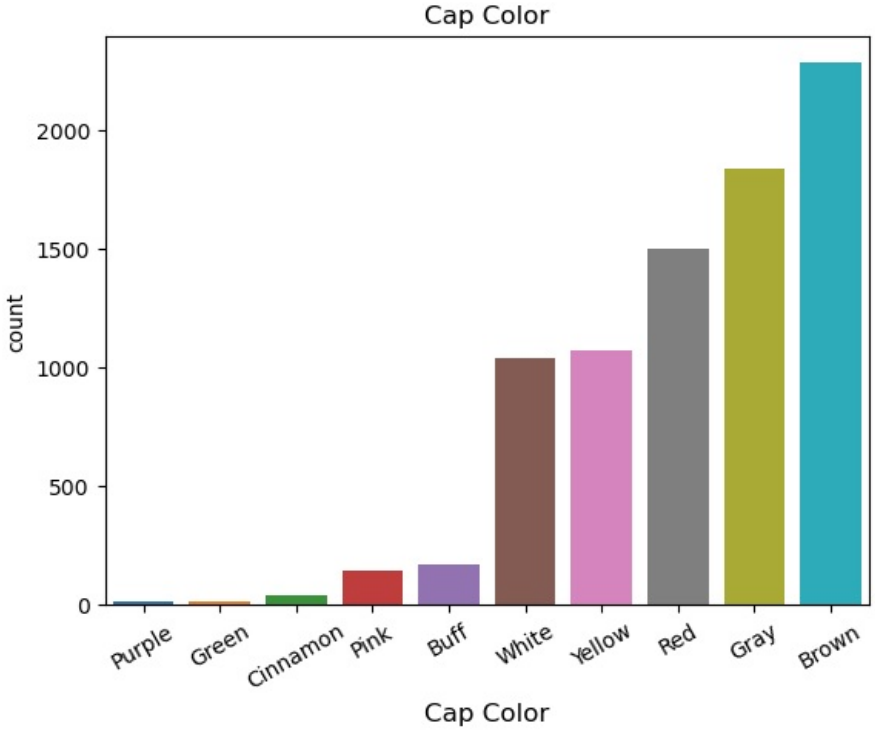
In [8]:

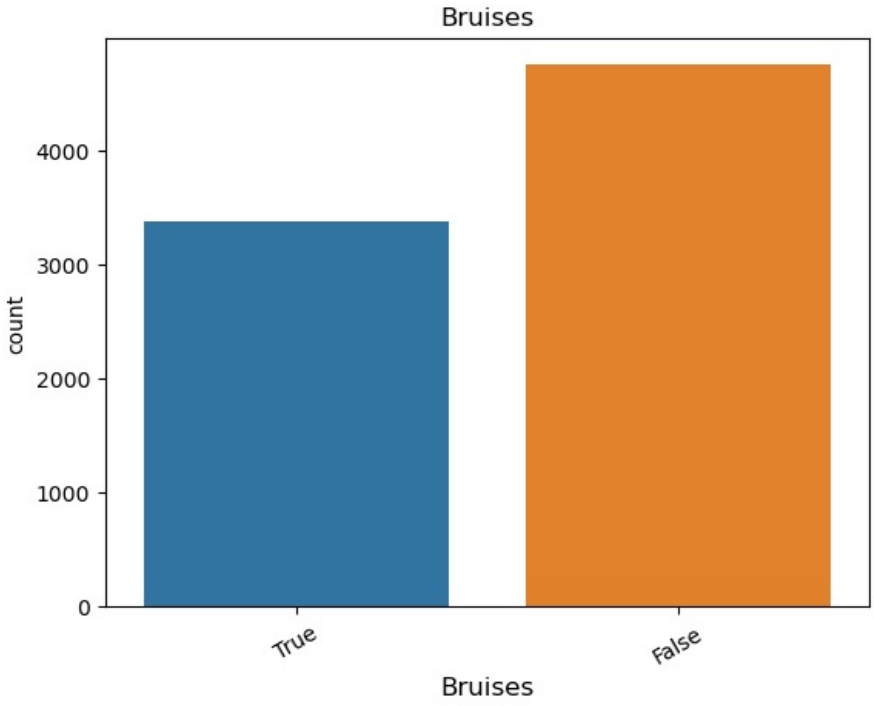
```
for column in columns:
    sns.countplot(x=column, data=df, order=df[column].value_counts(ascending=True).index)
    plt.xticks(rotation=30, fontsize=10)
    plt.xlabel(column, fontsize=12)
    plt.title(column)
    plt.show()
    plt.clf()
```

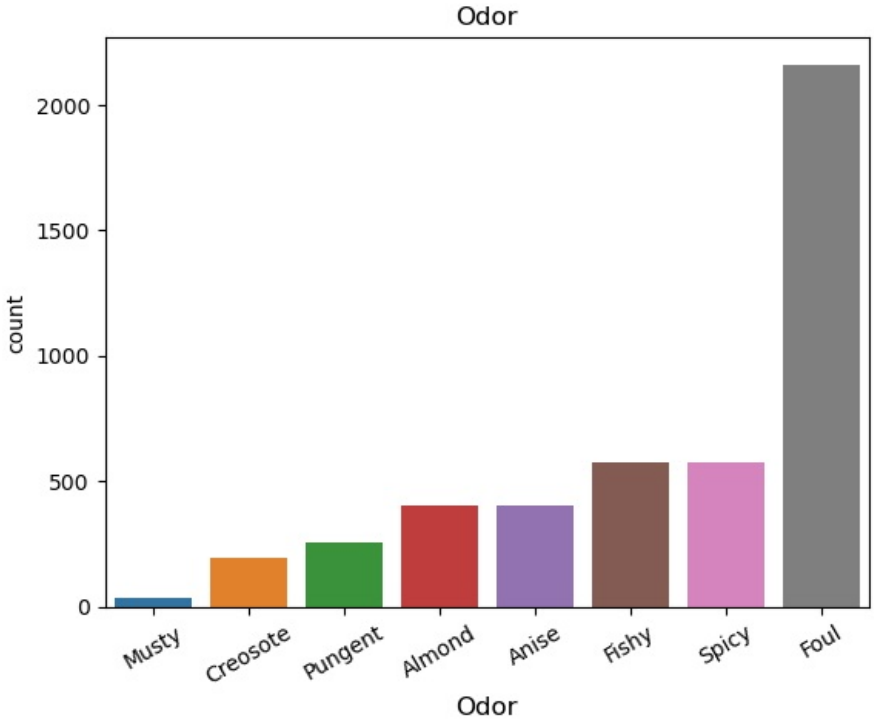


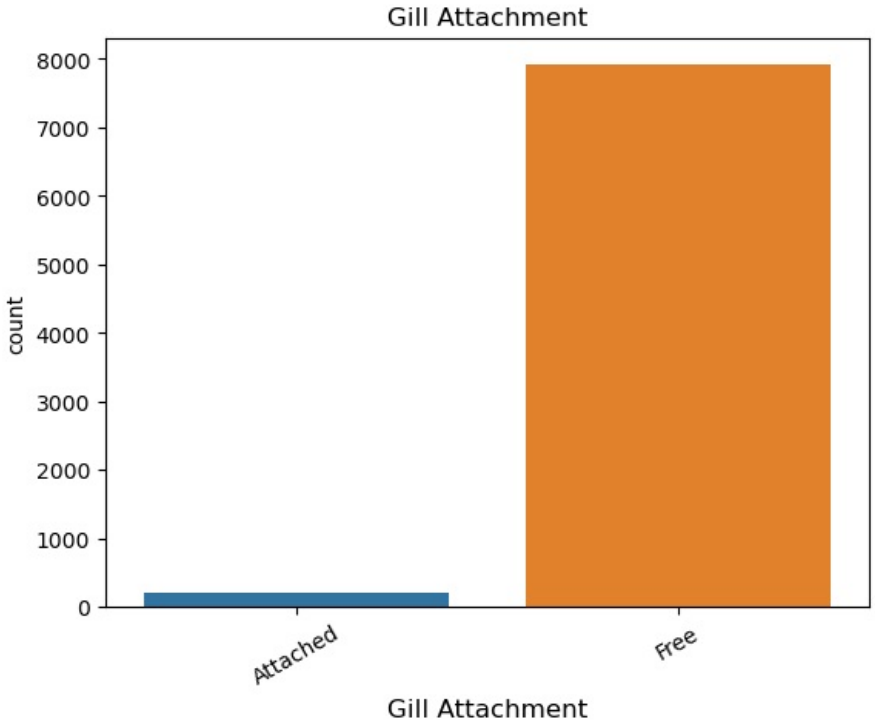


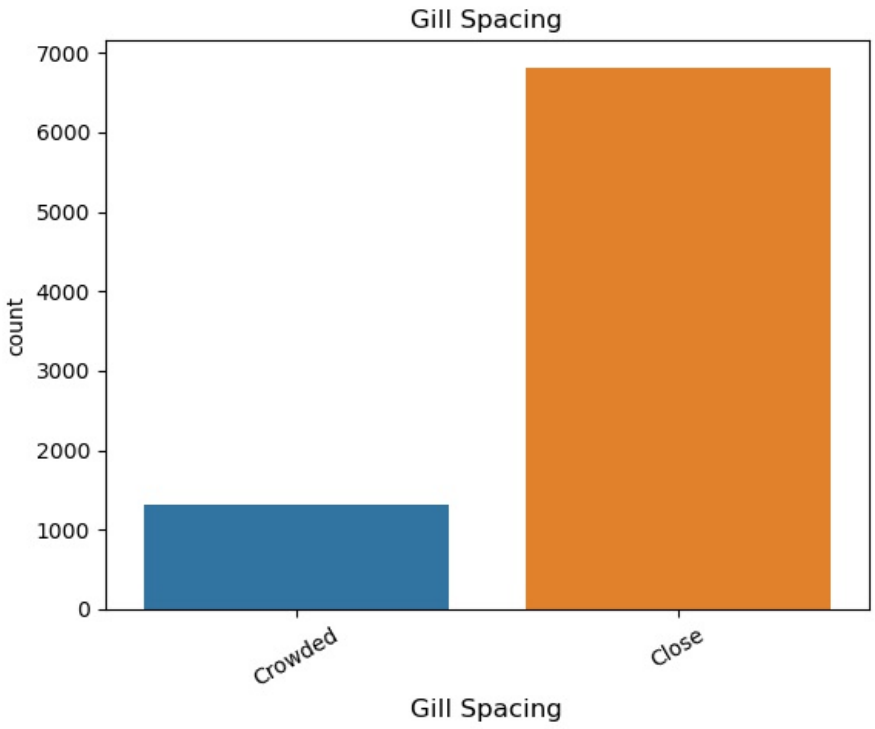

Cap Surface
count
3000
2500
2000
1500
1000
500
0
Grooves
Fibrous
Smooth
Scaly
Cap Surface

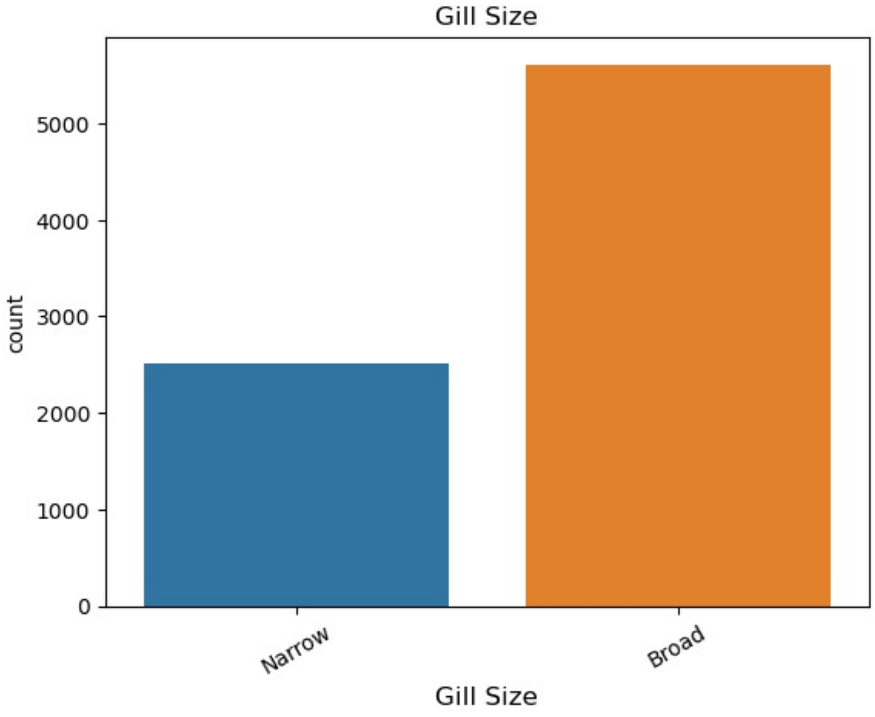

Cap Color
count
2000
1500
1000
500
0
Purple
Green
Cinnamon
Pink
Buff
White
Yellow
Red
Gray
Brown
Cap Color

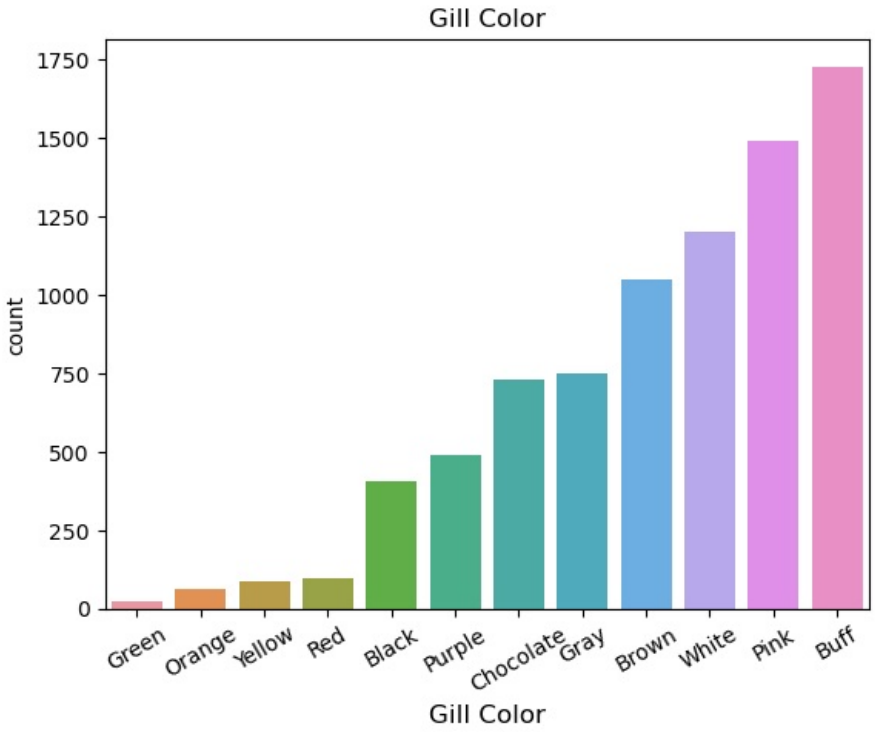

Bruises
4000
3000
2000
1000
0
count
True
False
Bruises

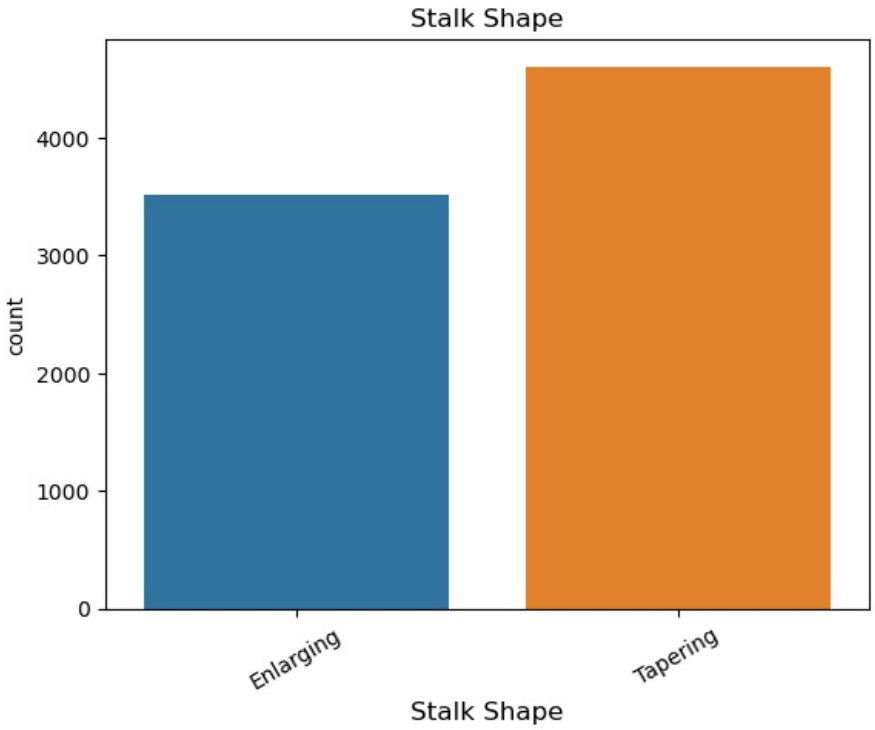

Odor
2000
1500
1000
500
0
count
Musty
Creosote
Pungent
Almond
Anise
Fishy
Spicy
Foul
Odor

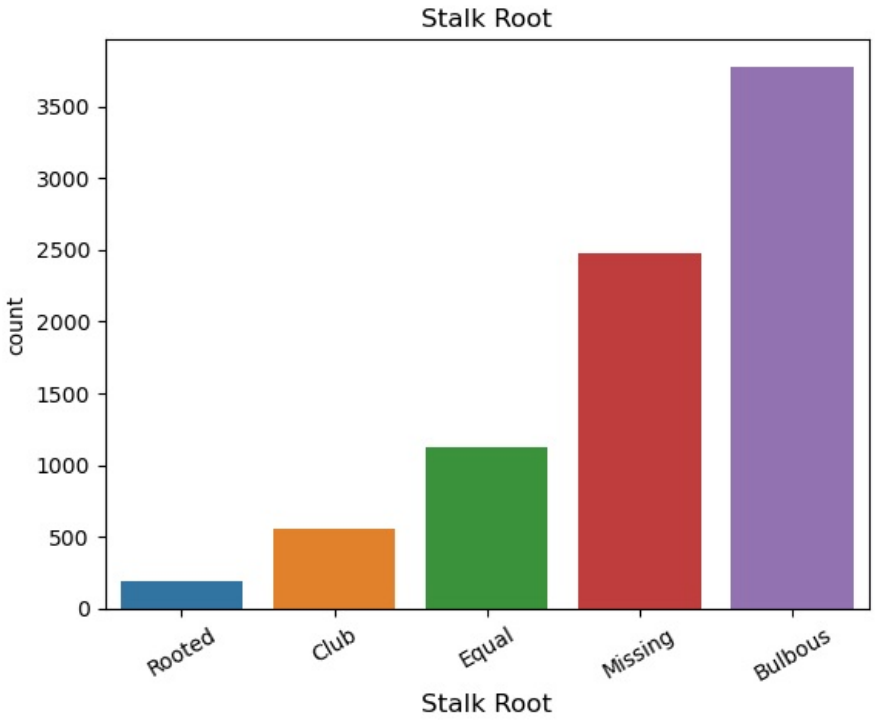

Gill Attachment
8000
7000
6000
5000
4000
3000
2000
1000
0
count
Attached
Free
Gill Attachment

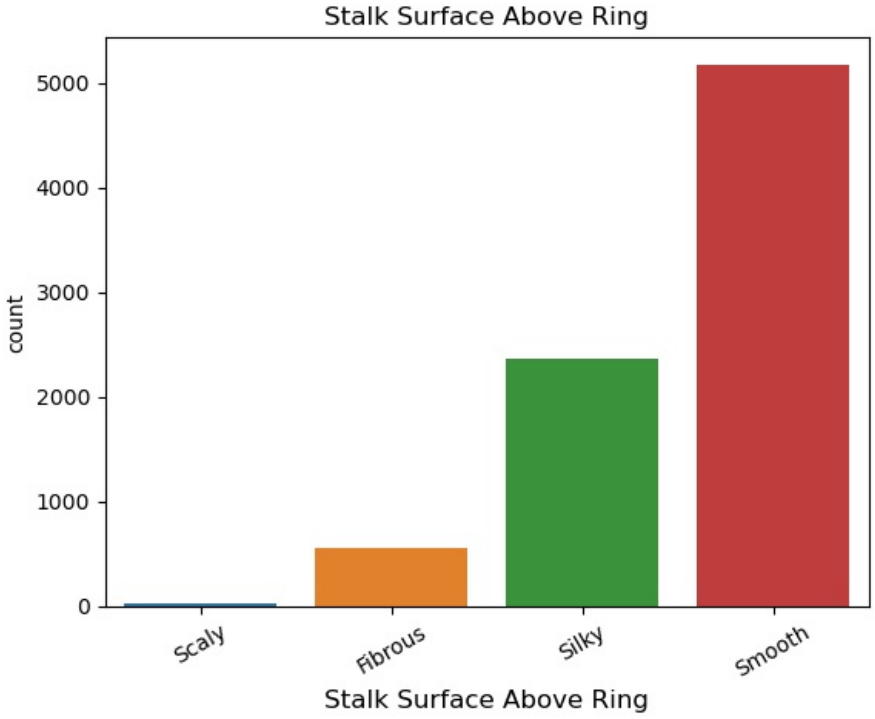

Gill Spacing
7000
6000
5000
4000
3000
2000
1000
0
count
Crowded
Close
Gill Spacing

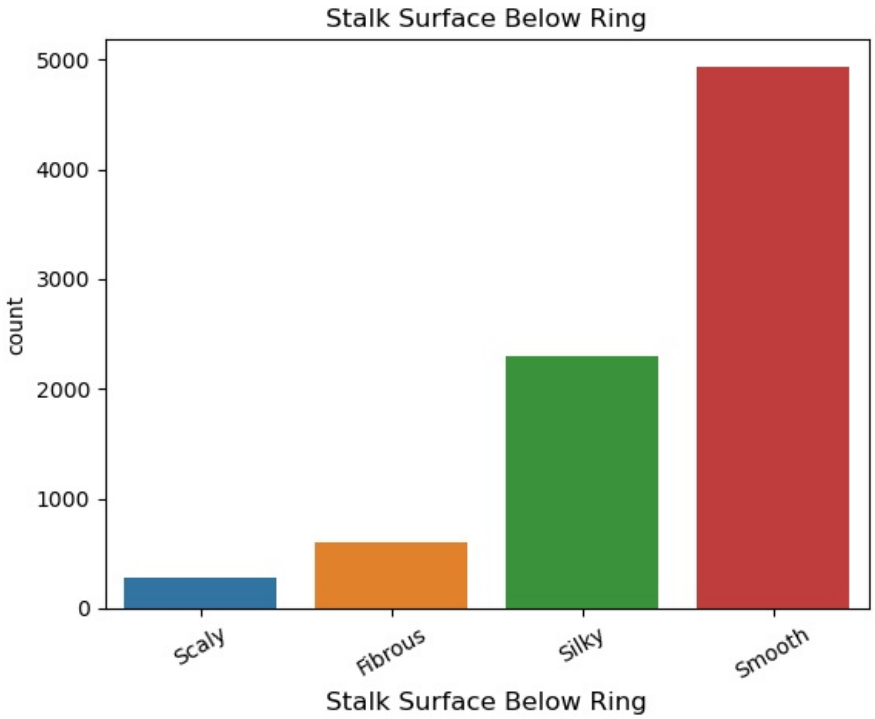

Gill Size
count
5000
4000
3000
2000
1000
0
Narrow
Broad
Gill Size

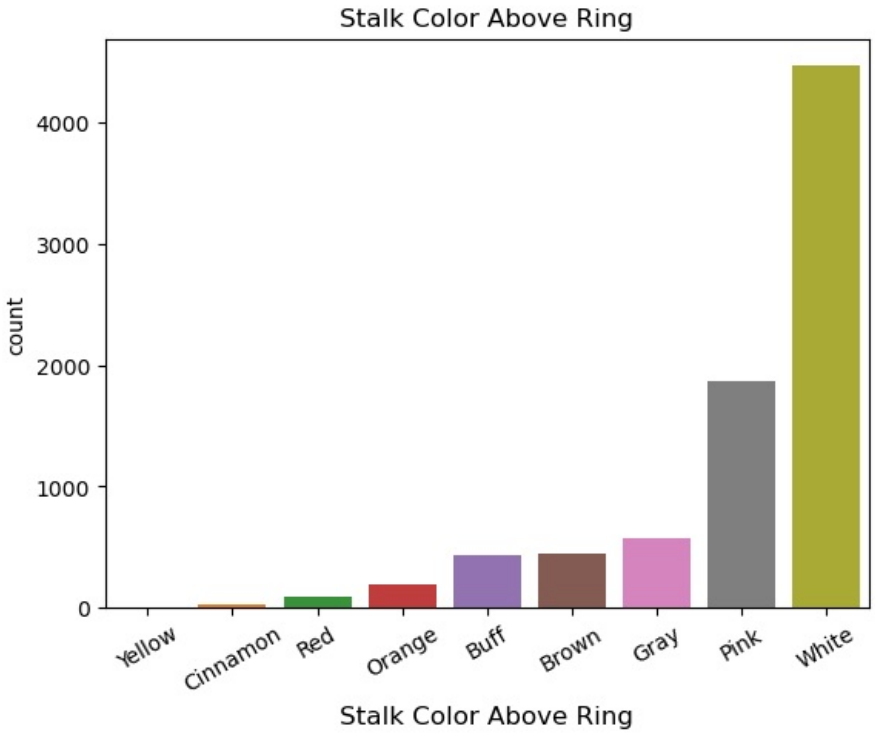

Gill Color
count
1750
1500
1250
1000
750
500
250
0
Green
Orange
Yellow
Red
Black
Purple
Chocolate
Gray
Brown
White
Pink
Buff
Gill Color

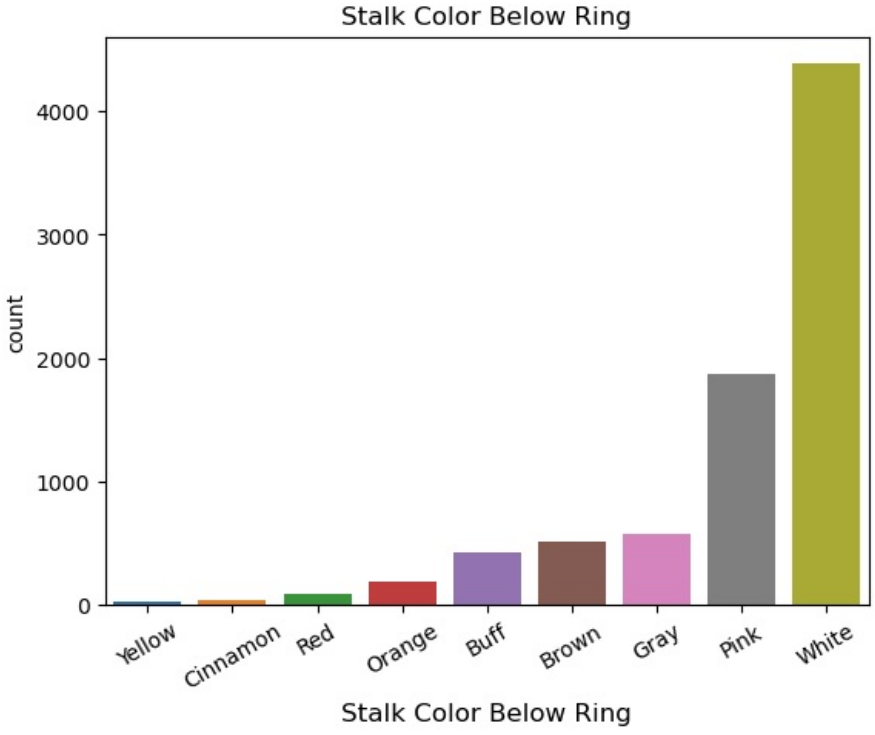

Stalk Shape
count
4000
3000
2000
1000
0
Enlarging
Tapering
Stalk Shape

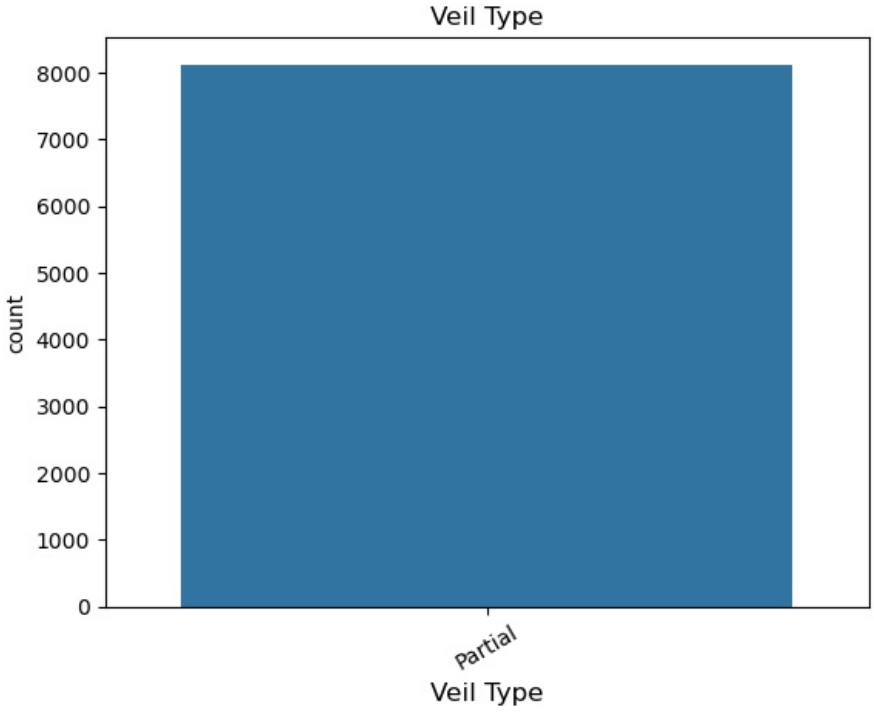

Stalk Root
count
3500
3000
2500
2000
1500
1000
500
0
Rooted
Club
Equal
Missing
Bulbous
Stalk Root

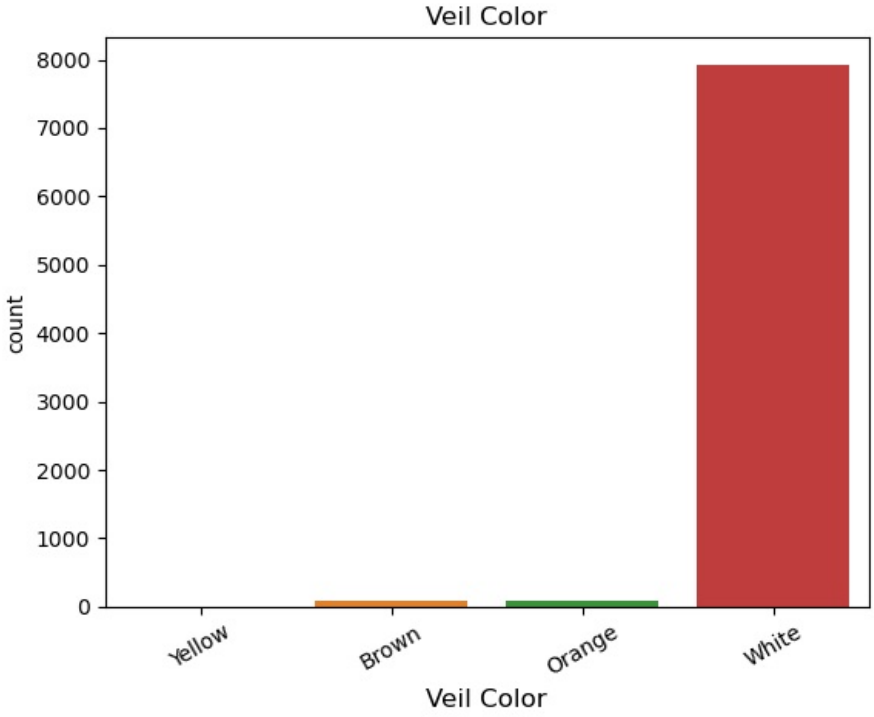

Stalk Surface Above Ring
count
5000
4000
3000
2000
1000
0
Scaly
Fibrous
Silky
Smooth
Stalk Surface Above Ring

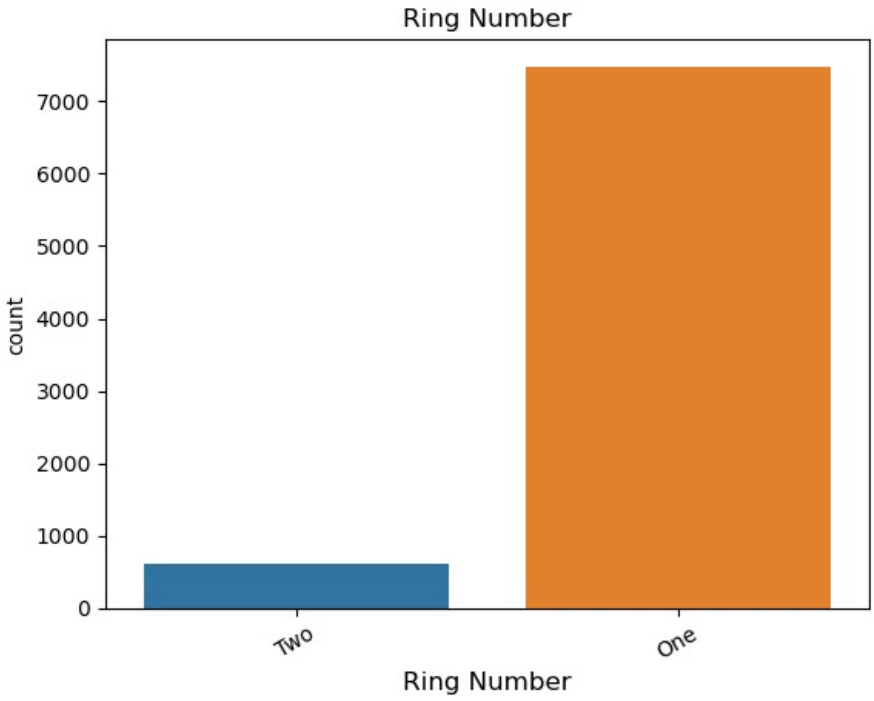

Stalk Surface Below Ring
count
5000
4000
3000
2000
1000
0
Scaly
Fibrous
Silky
Smooth
Stalk Surface Below Ring

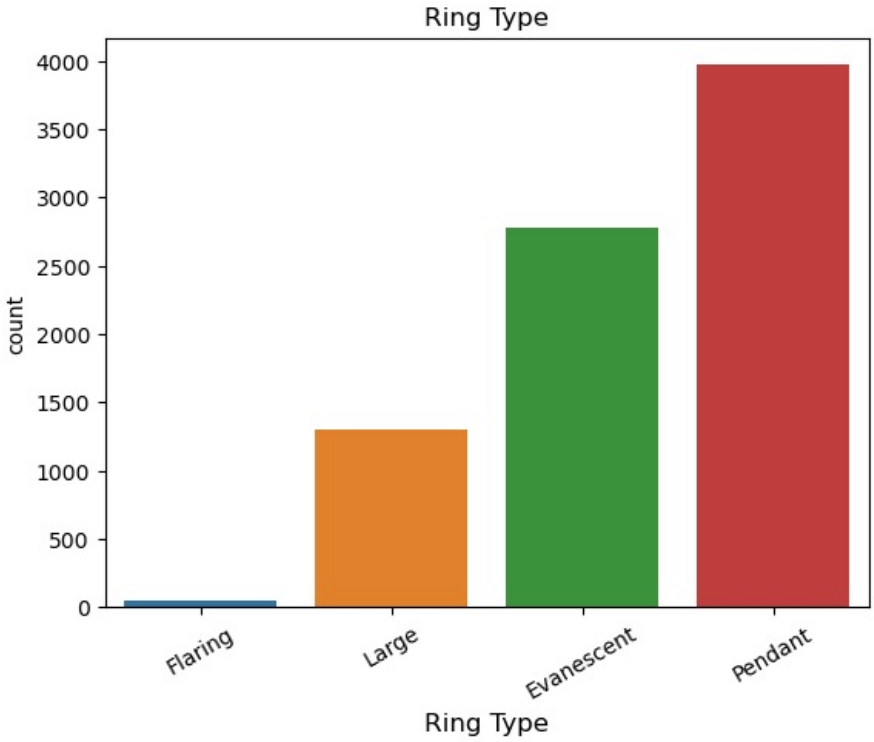

Stalk Color Above Ring
count
4000
3000
2000
1000
0
Yellow
Cinnamon
Red
Orange
Buff
Brown
Gray
Pink
White
Stalk Color Above Ring

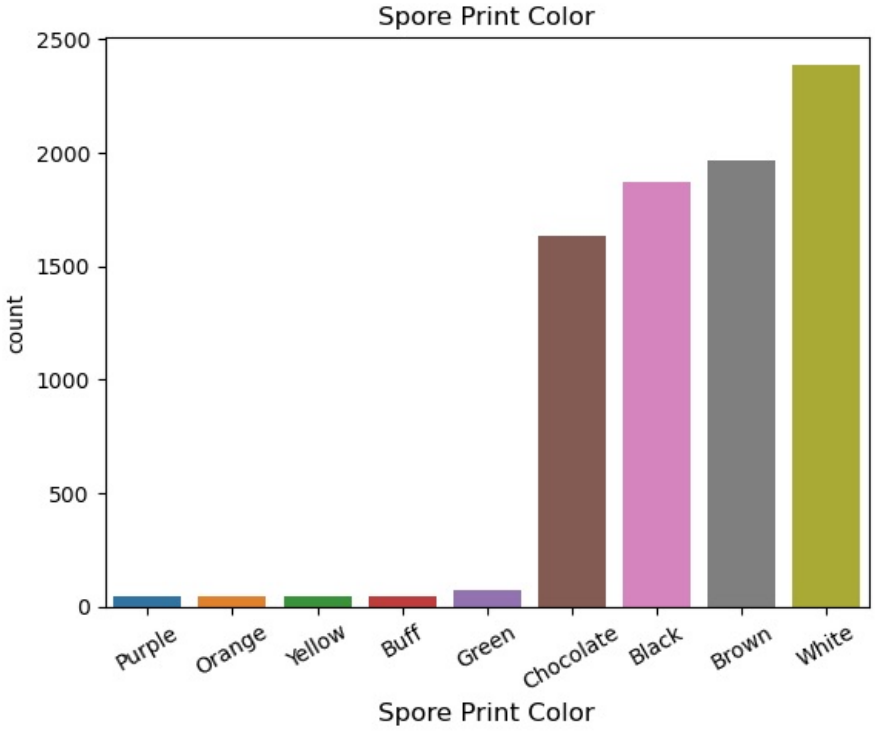

Stalk Color Below Ring
count
4000
3000
2000
1000
0
Yellow
Cinnamon
Red
Orange
Buff
Brown
Gray
Pink
White
Stalk Color Below Ring

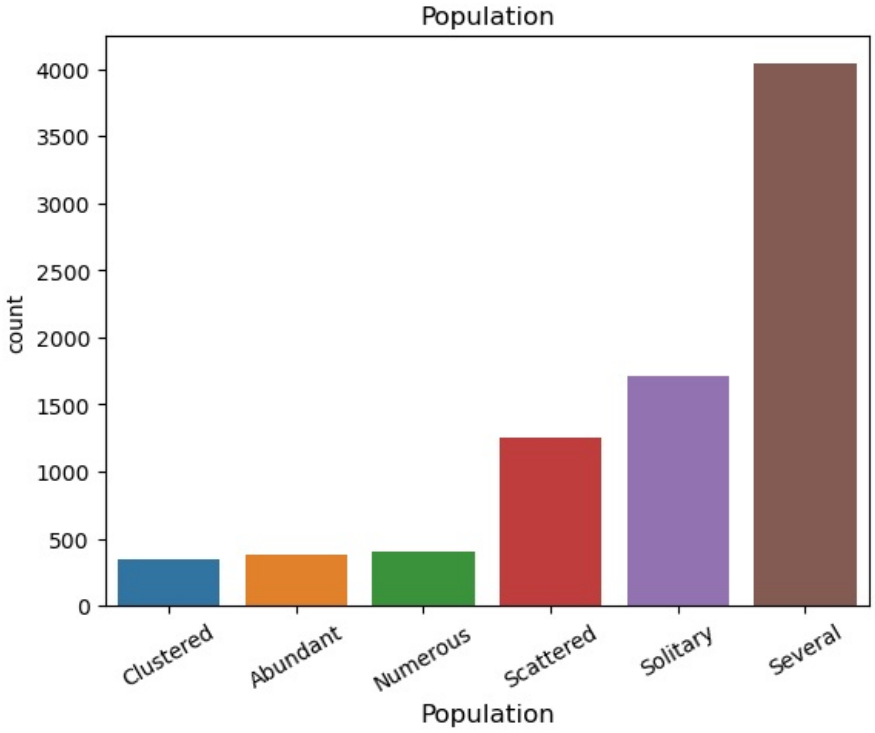

Veil Type
8000
7000
6000
5000
4000
3000
2000
1000
0
count
Partial
Veil Type

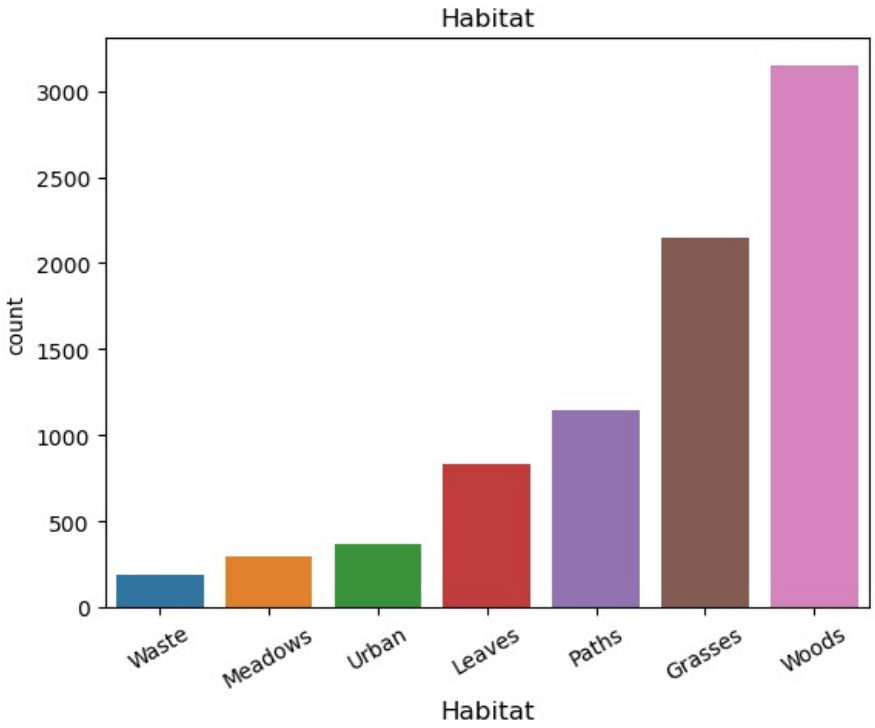

Veil Color
8000
7000
6000
5000
4000
3000
2000
1000
0
count
Yellow
Brown
Orange
White
Veil Color


Ring Number
count
7000
6000
5000
4000
3000
2000
1000
0
Two
One
Ring Number


Ring Type
count
4000
3500
3000
2500
2000
1500
1000
500
0
Flaring
Large
Evanescent
Pendant
Ring Type


Spore Print Color
2500
2000
1500
1000
500
0
count
Purple
Orange
Yellow
Buff
Green
Chocolate
Black
Brown
White
Spore Print Color


Population
4000
3500
3000
2500
2000
1500
1000
500
0
count
Clustered
Abundant
Numerous
Scattered
Solitary
Several
Population




```
<Figure size 640x480 with 0 Axes>
```

9. Great job! In relatively few lines of code, you have created 23 informative plots. Now that they are titled, labeled clearly, and ordered, you can really dive in on your analysis.

   Think about how someone could best use these visualizations. It is easy to tell which features of mushrooms are most common and rare, and we get insights into the variety of mushrooms in the fungi kingdom.

   Spend some time looking over the graphs. Write down exciting insights you find. Here are some examples to get you started:

   - It is a roughly equal split between mushrooms that are edible vs. poisonous.
   - The majority of mushrooms in this dataset have a scaly surface.
   - There are a non-insignificant amount of mushrooms that give off an almond scent?!!!
   - Most top surfaces of mushrooms in this dataset are scaly rather than smooth.

   Some of your analysis may also require research into mushroom features for any of the x-labels. We hope you enjoy continuing to explore the world of these fun guys!

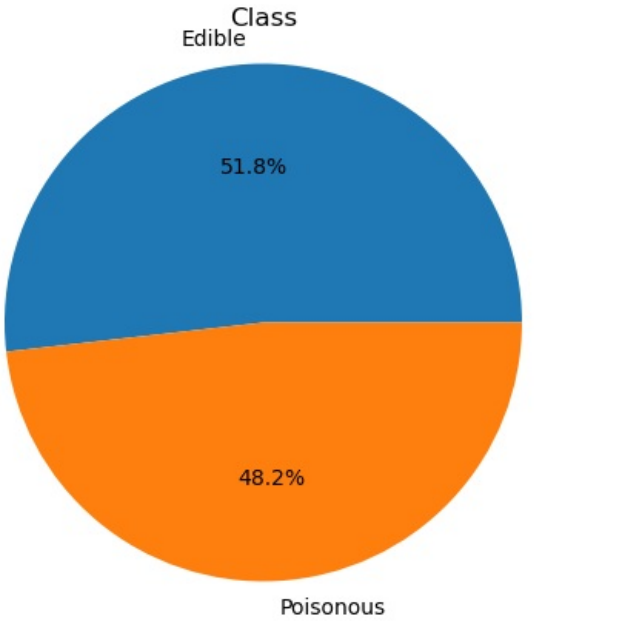
Extensions

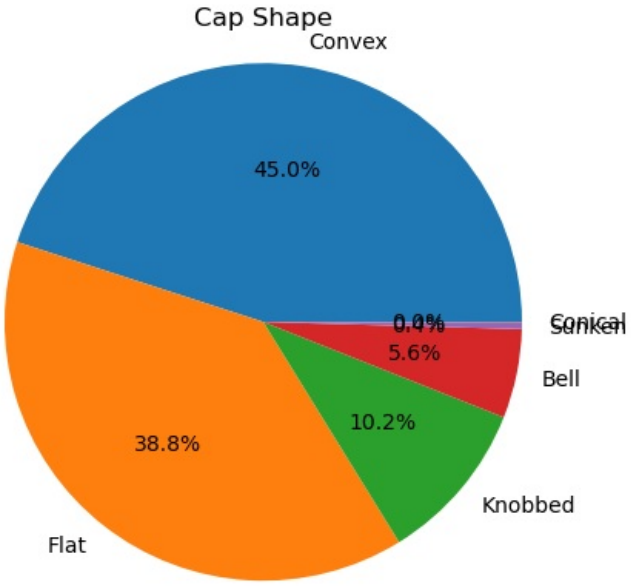
10. Feel free to play around with the graphs and customize them any way you want to help in your analysis! Here are some ideas to get yourself started:

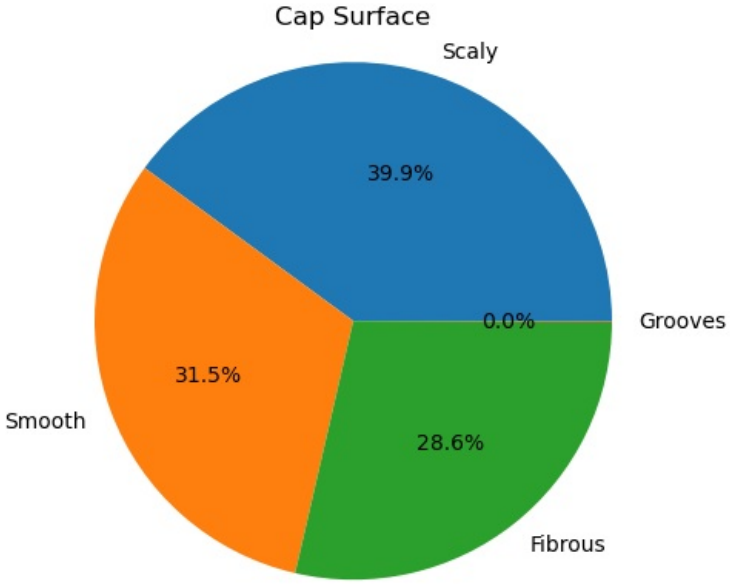
    - Turn any bar graph with less than six bars into a pie chart (hint: use a conditional statement!).
    - Create your bar charts using a list comprehension instead of a `for` loop.
    - Change the color theme of your graphs using the seaborn color or palette parameters.
    - Remove any graphs you find uninformative.
    - Change around the current title or label formatting.

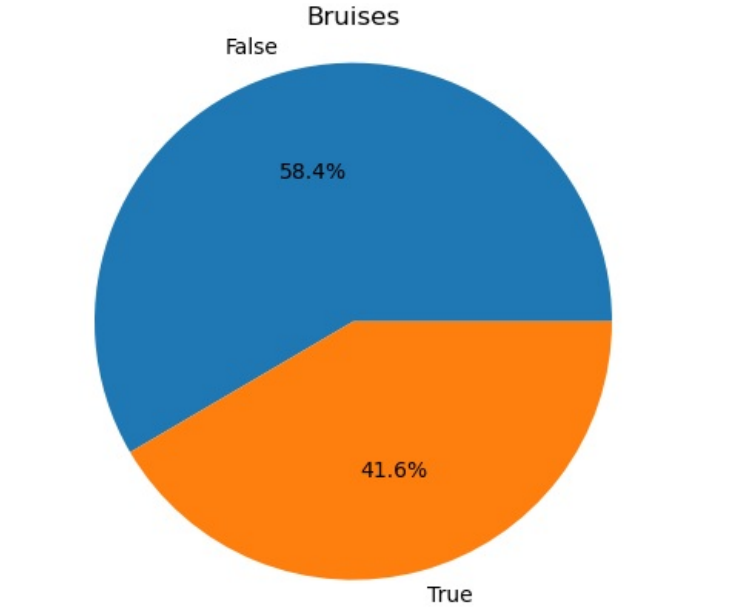
Happy Coding! :)

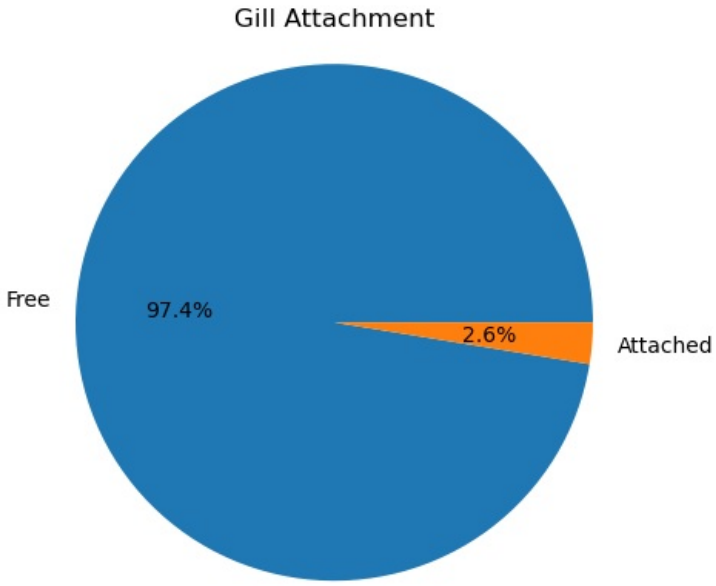
In [9]:
```
for column in columns:
    class_counts = df[column].value_counts()
    if len(class_counts) <= 6:
        plt.pie(class_counts, labels=class_counts.index, autopct='%0.1f%%')
        plt.title(column)
        plt.axis('equal')
        plt.show()
        plt.clf()
```

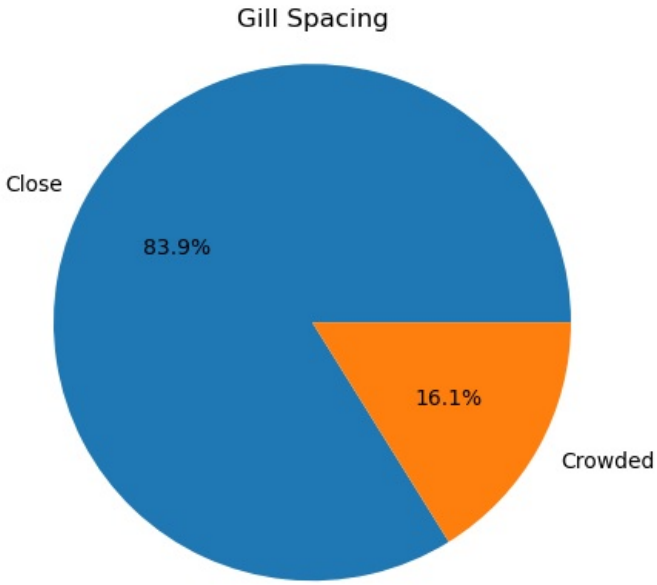


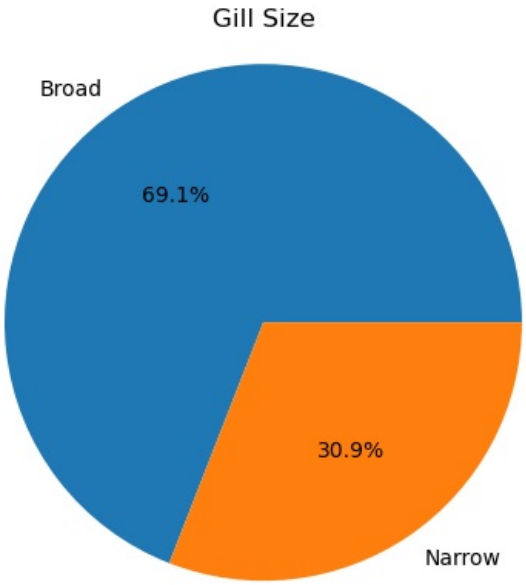


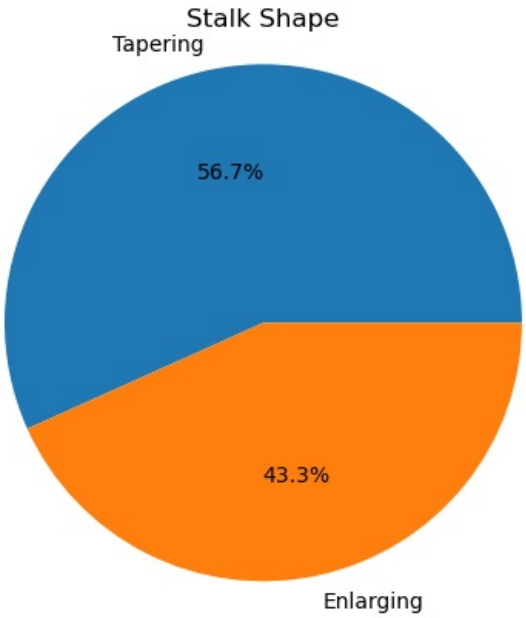

Cap Surface
Scaly
39.9%
0.0%
Grooves
31.5%
Smooth
28.6%
Fibrous

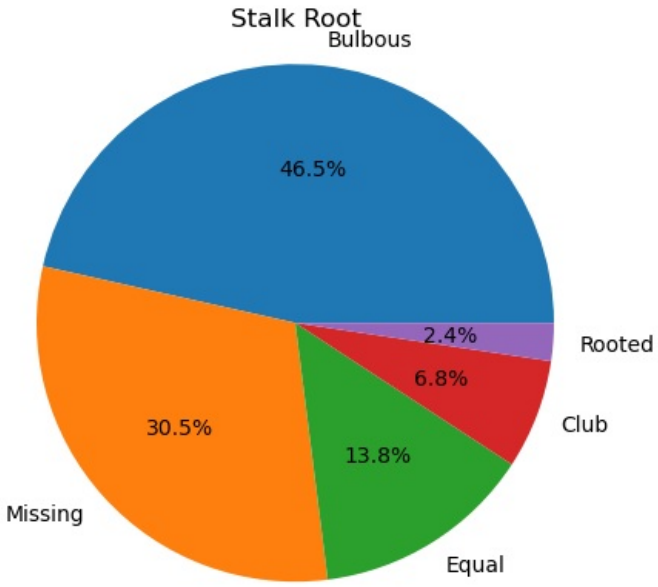

Bruises
False
58.4%
41.6%
True

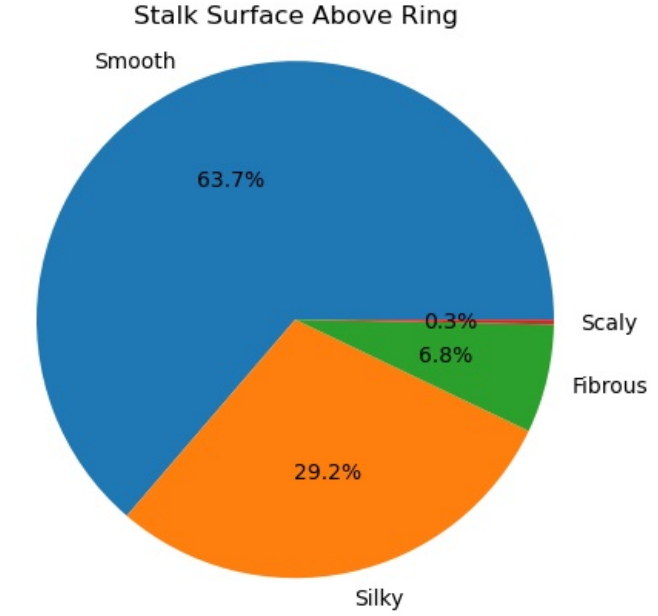

Gill Attachment
Free
97.4%
2.6%
Attached

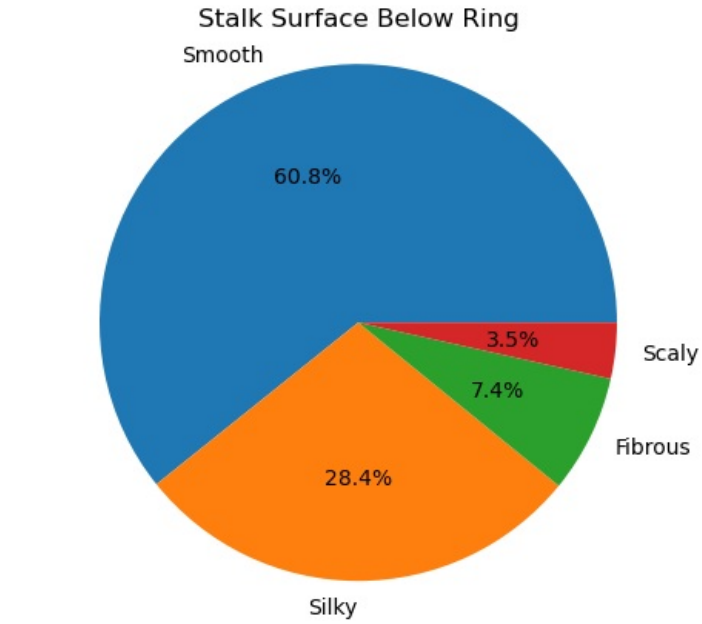

Gill Spacing
Close
83.9%
16.1%
Crowded

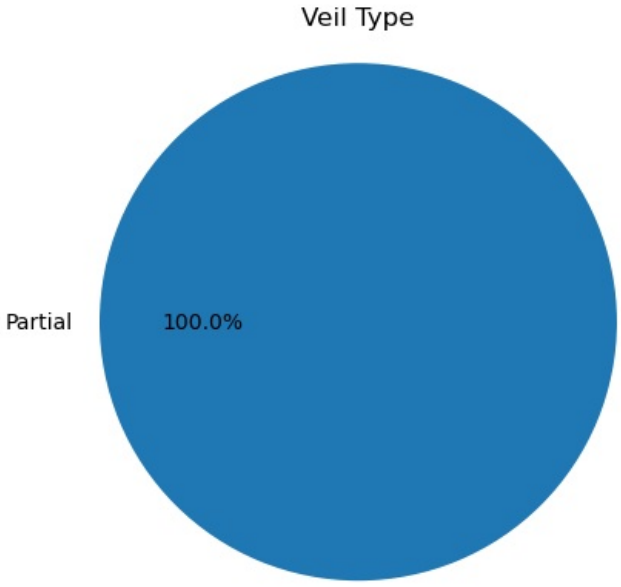

Gill Size
Broad
69.1%
30.9%
Narrow

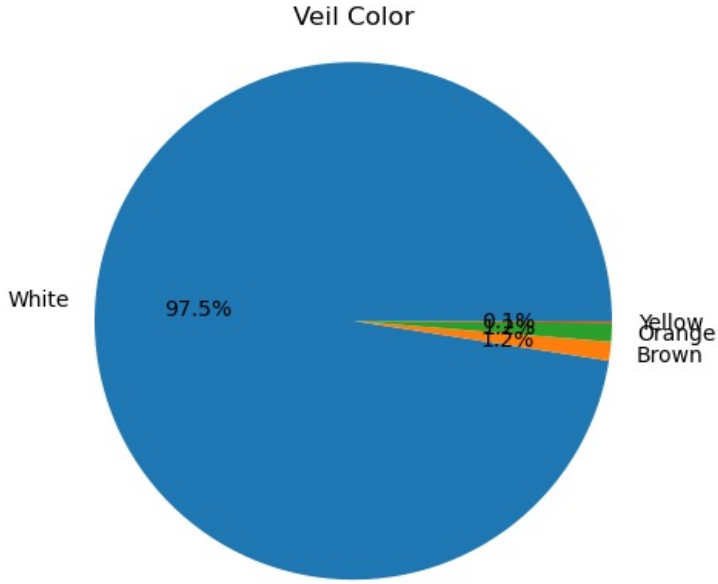

Stalk Shape
Tapering
56.7%
43.3%
Enlarging

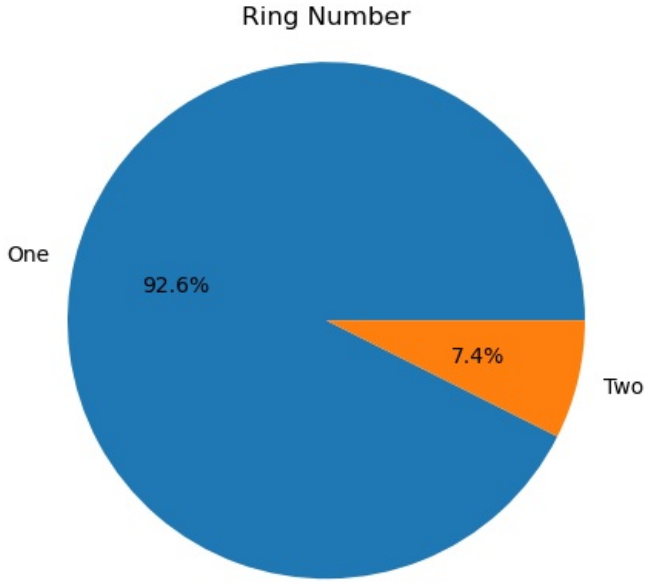

Stalk Root
Bulbous
46.5%
2.4%
Rooted
6.8%
Club
30.5%
13.8%
Missing
Equal

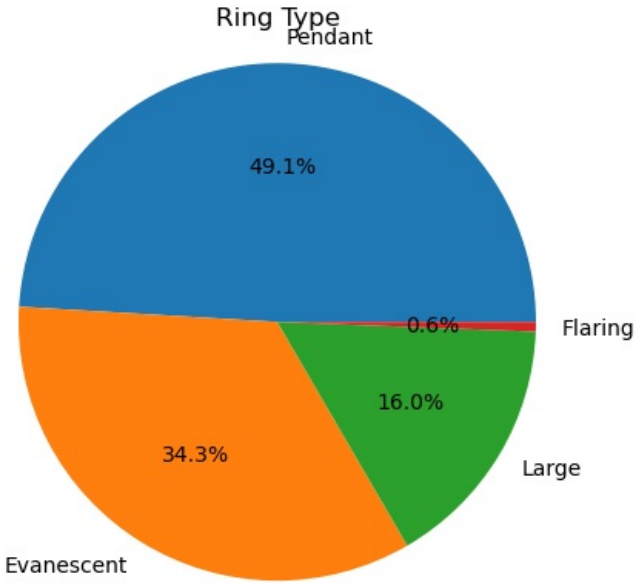

Stalk Surface Above Ring
Smooth
63.7%
0.3%
Scaly
6.8%
Fibrous
29.2%
Silky

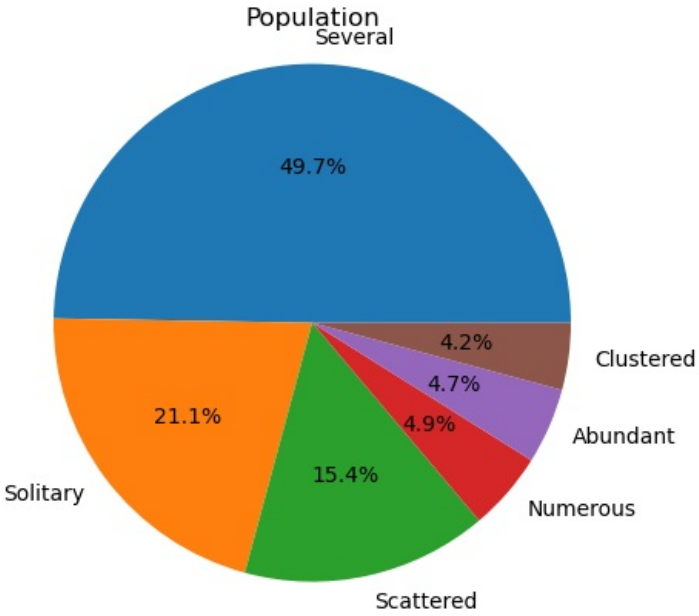

Stalk Surface Below Ring
Smooth
60.8%
3.5%
Scaly
7.4%
Fibrous
28.4%
Silky


Veil Type
Partial
100.0%


Veil Color
White
97.5%
0.1%
Yellow
Orange
1.2%
Brown


Ring Number
One
92.6%
7.4%
Two






```
<Figure size 640x480 with 0 Axes>
```