Friday, 18 September 2020

Submit all experiments in one google colab link.

1 Pairplot (10 mins)

1.1 Objectives

1. Be able to study how to use the `pairplot` function of the seaborn library
2. Be able to see the discrimination power of features of the data

1.2 Description

The `pairplot` function of the seaborn library is a very useful tools to visualize the data. In this experiment, you have to explore the new thing by yourself. Please read the document relating to the pairplot function and try to plot the scatter plot of all pairs of variables.

If you want to select the columns of the data that can be used to classify the Species of each flower, which variable(s) you will choose? (Answer in a text cell after your output chart.)

1.3 Procedure

1. Read the documents relating to `pairplot` function of `seaborn` library. The link is https://seaborn.pydata.org/generated/seaborn.pairplot.html.
2. Study the `iris` data set (https://www.kaggle.com/uciml/iris).
3. Plot the pairplot of iris data as shown in the Sample Output.
4. Choose the attributes that can classify three classes from each other.

1.4 Sample Output

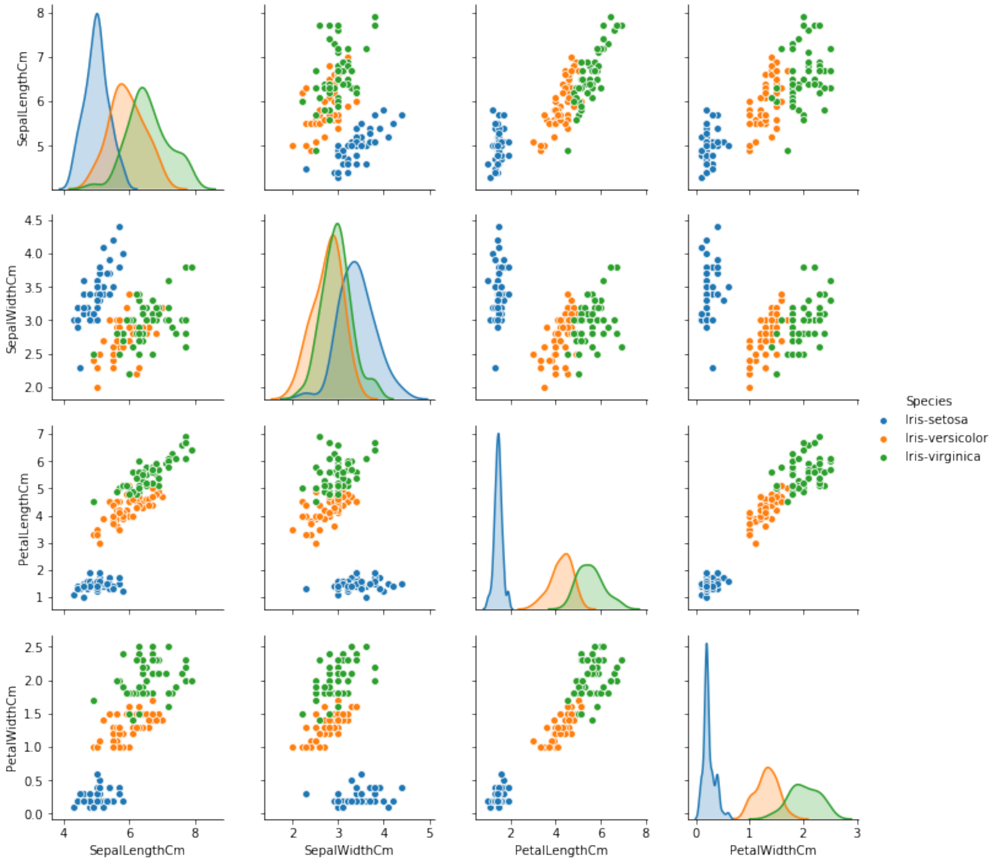

Figure 1: *Output chart*

2 Colored PM2.5(20 mins)

2.1 Objectives

1. Can modify the barplot function of seaborn
2. Can combine pandas and seaborn
3. Know and can use function of python

2.2 Description

This time you have to create your own function. The format of function in python is quite simple. You can define a function like below:

```
def max_of_two(a,b):
  max = a
  if b > max:
    return b

print(max_of_two(4,5))
```

To be able to control the color of barplot you have to provide a list of colors of the bar. In this case if you use the standard function of barplot of the pm2.5 dataframe (the code that creates this dataframe is shown below), you will get the chart shown in the example.

```
pm_df=pd.read_csv('pm25.csv')
pm_df.columns=['dt','pm']
pm_df=pm_df[pm_df['pm'] != '-']
pm_df['pm']=pd.to_numeric(pm_df['pm'])
pm_df['dt']=pd.to_datetime(pm_df['dt'])
pm_df['hr']=pm_df['dt'].dt.hour

sb.barplot(data=pm_df,x='hr',y='pm')
```

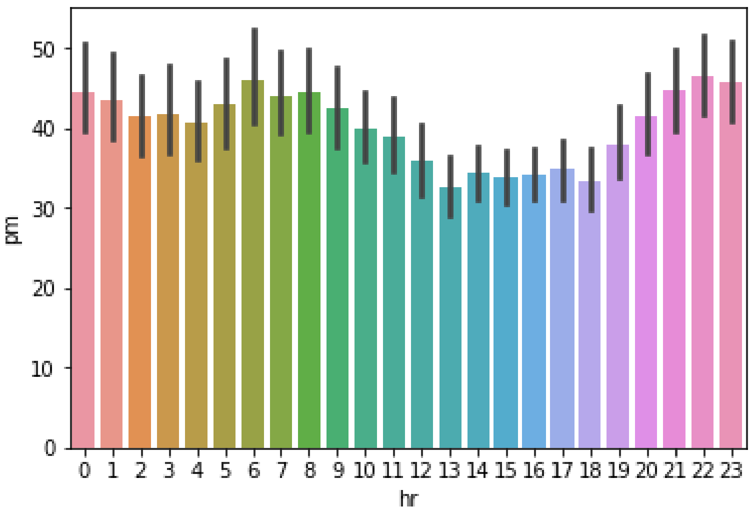

Figure 2: *Standard chart*

The method that you can change the color is to provide a list of colors to the parameter `palette`. For example, if you create a list of colors like

```
colors = ['grey']*6+['black']*18
sb.barplot(data=pm_df,x='hr',y='pm',palette=colors)
```

You will get the chart below.

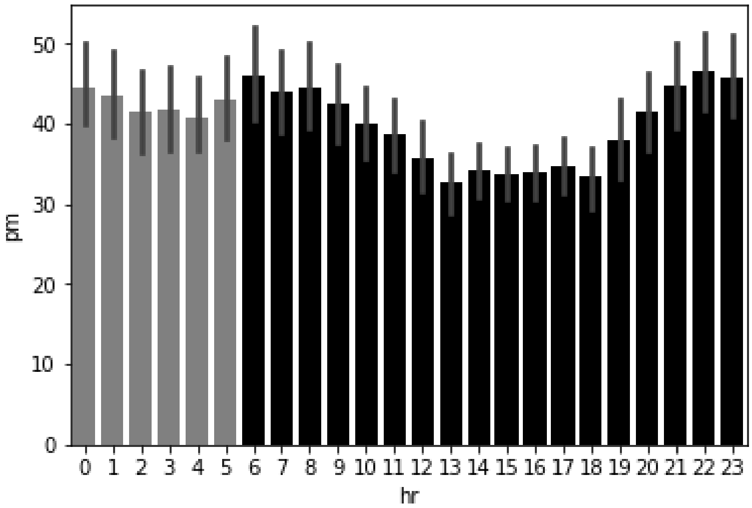

Figure 3: *Standard chart*

Please create your own list of colors that can show the level of pm2.5 using the criteria below.

```
if level >= 50:
  color = 'red'
elif level >= 37:
  color = 'yellow'
else:
  color = 'green'
```

You can use the code to create the chart as shown below. But you have to create your own `get_color()` function by yourself.

```
def get_color(x):
  # declare your own function


mean_by_hr=pm_df.groupby('hr').mean()
color_bar = [get_color(x) for x in mean_by_hr['pm']]
sb.barplot(data=pm_df,x='hr',y='pm',palette=color_bar)
```

2.3 Sample output

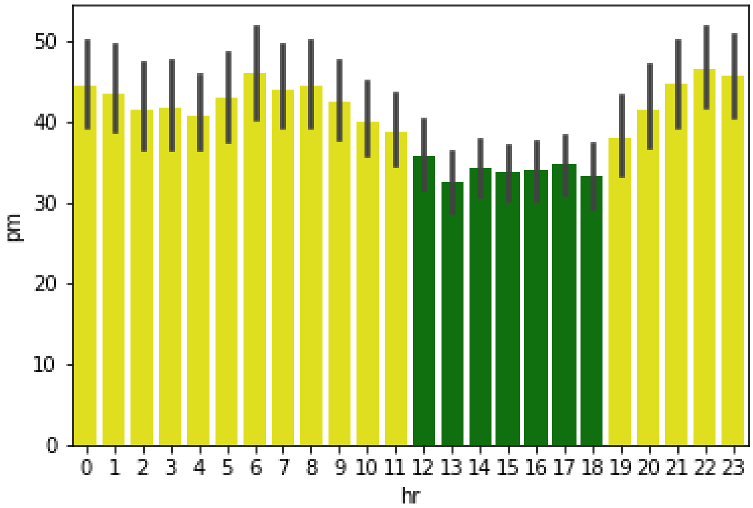

Figure 4: *Final output*

3 Other Idea (30 mins)

3.1 Description

This experiment is your turn. You have to create an interesting question and try to answer that question using visualization tools from pandas, seaborn, or plotly. You can choose the `Covid19`, `pm25.csv`, `titanic`, or any other data from online resources.