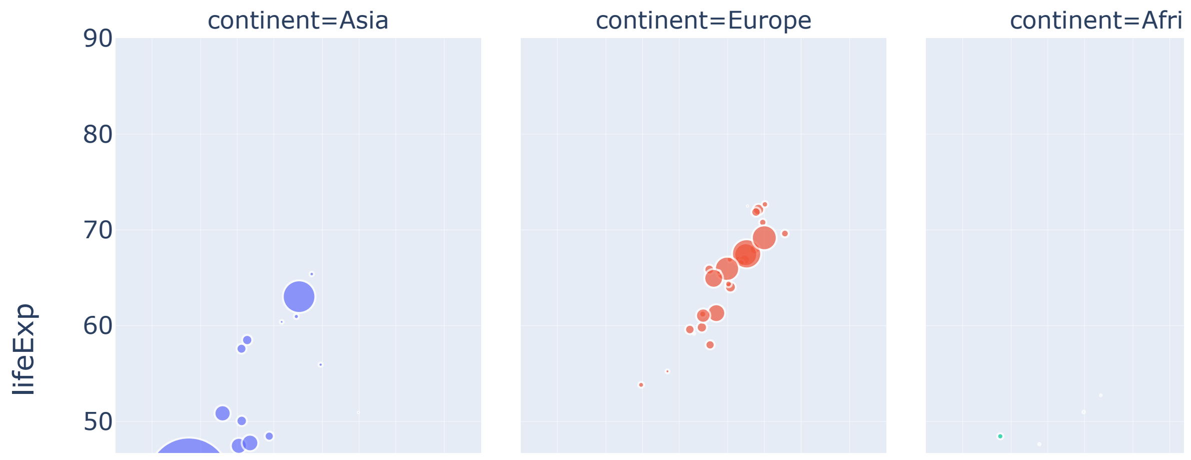
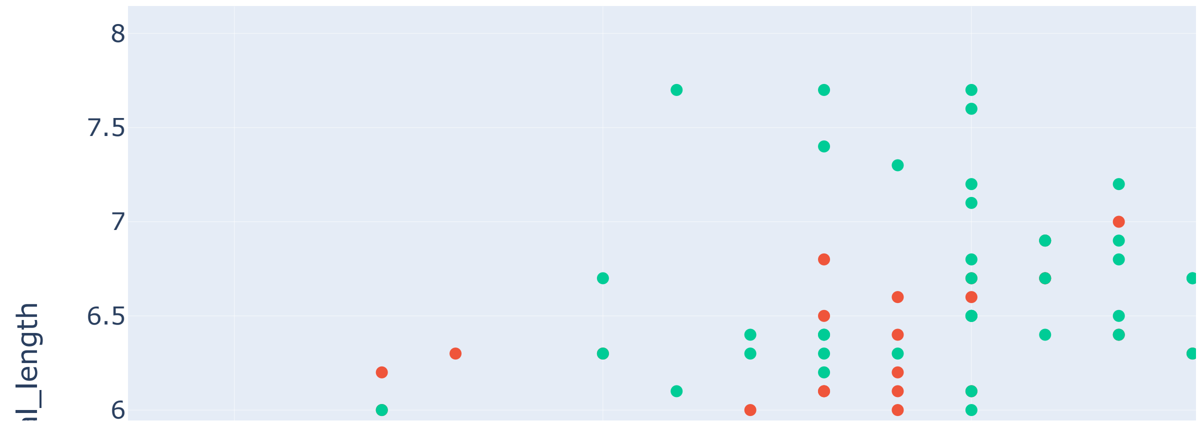
In [1]:
```
pip install plotly
```

```
Requirement already satisfied: plotly in c:\users\dell\anaconda3\lib\site-packages (5.7.
0)Note: you may need to restart the kernel to use updated packages.
Requirement already satisfied: six in c:\users\dell\anaconda3\lib\site-packages (from pl
otly) (1.16.0)
Requirement already satisfied: tenacity>=6.2.0 in c:\users\dell\anaconda3\lib\site-packa
ges (from plotly) (8.0.1)
```

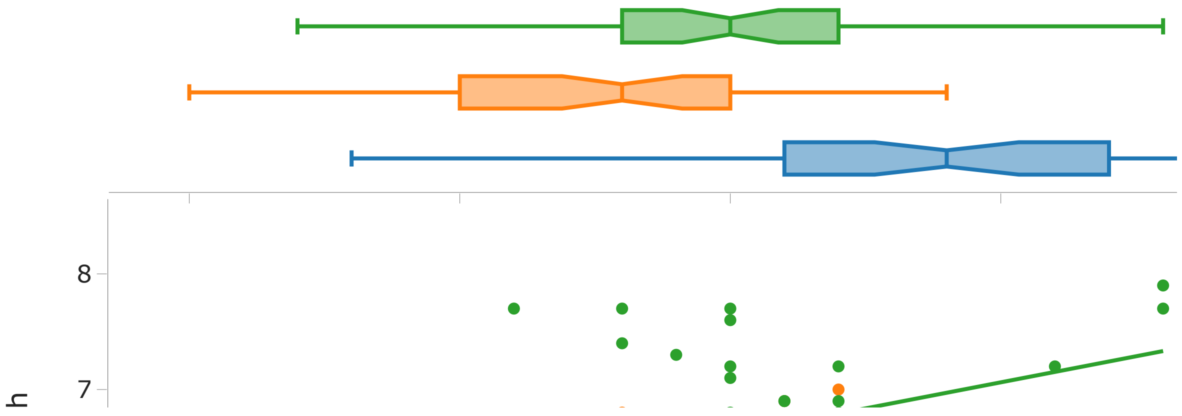
In [2]:
```
import plotly.express as px
df = px.data.gapminder()
px.scatter(df, x="gdpPercap", y="lifeExp", animation_frame="year", animation_group="cou
           size="pop", color="continent", hover_name="country",facet_col="continent",
           log_x=True, size_max=45, range_x=[100,100000], range_y=[25,90])
```



In [3]:
```
import plotly.express as px
df = px.data.iris()
fig = px.scatter(df, x="sepal_width", y="sepal_length", color="species")
fig.show()
```



```
In [4]:
import plotly.express as px
df = px.data.iris()
fig = px.scatter(df, x="sepal_width", y="sepal_length", color="species", marginal_y="vi
           marginal_x="box", trendline="ols", template="simple_white")
fig.show()
```



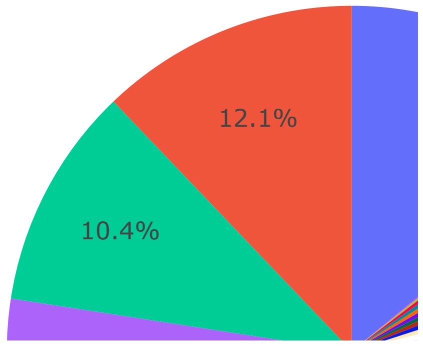
In [5]:

```
import plotly.express as px

df = px.data.gapminder()

fig = px.bar(df, x="continent", y="pop", color="continent",
  animation_frame="year", animation_group="country", range_y=[0,4000000000])
fig.show()
```

In [6]:

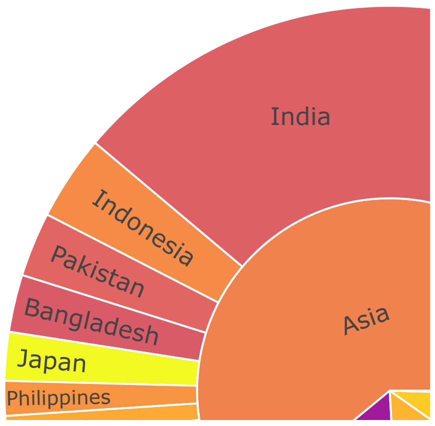
```
import plotly.express as px
df = px.data.gapminder().query("year == 2007").query("continent == 'Europe'")
df.loc[df['pop'] < 2.e6, 'country'] = 'Other countries' # Represent only large countrie
fig = px.pie(df, values='pop', names='country', title='Population of European continent
fig.show()
```

Population of European continent



In [7]:

```
import plotly.express as px
df = px.data.gapminder().query("year == 2007")
fig = px.sunburst(df, path=['continent','country'], values='pop',
                  color='lifeExp',hover_data=['iso_alpha'])
fig.show()
```



Libraries in python

Their usefulness lies in the fact that new codes are not required to be written everyb time the same process is required

to run.libraries in python play an important role in areas of data science,machine learning,data manipulation

application etc.

In [ ]:
```
# array
```

In [8]:
```
import numpy as np
a=np.array([1,2,3,4])
a
```

Out[8]:
```
array([1, 2, 3, 4])
```

In [4]:
```
import numpy as np
a = np.array([[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]])
a
```

Out[4]:
```
array([[ 1,  2,  3,  4],
       [ 5,  6,  7,  8],
       [ 9, 10, 11, 12]])
```

In [7]:
```
import numpy as np
a=np.zeros(2)
```

```
a=np.array([0., 0.])
a
```

Out[7]:
```
array([0., 0.])
```

In [12]:
```
import numpy as np
arr = np.array([2, 1, 5, 3, 7, 4, 6, 8])
arr
```

Out[12]:
```
array([2, 1, 5, 3, 7, 4, 6, 8])
```

In [16]:
```
import numpy as np
a=np.sort(arr)
a=np.array([1, 2, 3, 4, 5, 6, 7, 8])
a
```

Out[16]:
```
array([1, 2, 3, 4, 5, 6, 7, 8])
```

In [18]:
```
import numpy as np
a = np.array([1, 2, 3, 4])
b = np.array([5, 6, 7, 8])
a,b
```

Out[18]:
```
(array([1, 2, 3, 4]), array([5, 6, 7, 8]))
```

In [37]:
```
# resolve this then
import numpy as np
np.concatenate((a, b))
b=np.array([1, 2, 3, 4, 5, 6, 7, 8])
b
```

Out[37]:
```
array([1, 2, 3, 4, 5, 6, 7, 8])
```

In [31]:
```
import numpy as np
x = np.array([[1, 2], [3, 4]])
y = np.array([[5, 6]])
x,y
```

Out[31]:
```
(array([[1, 2],
        [3, 4]]),
 array([[5, 6]]))
```

In [36]:
```
import numpy as np
np.concatenate((x, y), axis=0)
np.array([[1, 2],
       [3, 4],
       [5, 6]])
```

Out[36]:
```
array([[1, 2],
       [3, 4],
       [5, 6]])
```

```
In [ ]:
```