Interactive Visualization with Plotly - Plotly Express - 2

One should look for what is and not what he thinks should be. (Albert Einstein)

Module completion checklist

Objective	Complete
Describe bivariate plots and multivariate plots in Plotly	
Save plots in Plotly	

Introducing Iris flower dataset

- We are going to explore a new data set called Iris from `plotly express` package
- The `Iris flower` dataset is a multivariate data set introduced by the British statistician and biologist Ronald Fisher in his 1936 paper The use of multiple measurements in taxonomic problems.
- The data set consists of 50 samples from each of three species of Iris (**Iris Setosa**, **Iris virginica**, and **Iris versicolor**).
- Four features were measured from each sample: the length and the width of the sepals and petals (in centimeters)

Introducing Iris flower dataset (cont'd)

- The dataset contains a set of **150 records** under **5 attributes**
  - Petal Length
  - Petal Width
  - Sepal Length
  - Sepal width and
  - Class(Species)

Load the data

- Follow the steps below to read data from `plotly express`:

```
# Load the iris dataset from `plotly express`
iris_dataset = px.data.iris()

# Top 5 entries of the dataset
iris_dataset.head()
```

	sepal_length	sepal_width	petal_length	petal_width	species	species_id
0	5.1	3.5	1.4	0.2	setosa	1
1	4.9	3.0	1.4	0.2	setosa	1
2	4.7	3.2	1.3	0.2	setosa	1
3	4.6	3.1	1.5	0.2	setosa	1
4	5.0	3.6	1.4	0.2	setosa	1

Bivariate plots

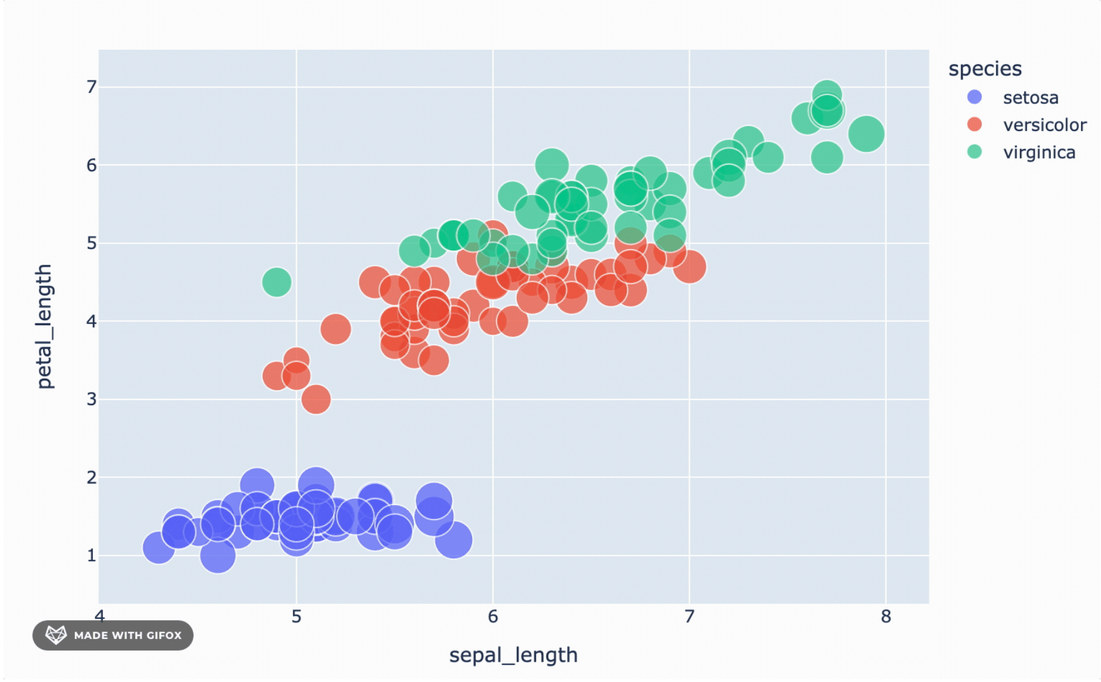
- A bivariate plot is a plot that allows us to identify the relationship between **two variables**
- `plotly express` has various bivariate plots available:
  - Scatter plots
  - Line plots
  - Funnel plots
  - Area plots
- We are going to discuss a subset of these
- For each of these plots, we will be using in-built datasets from the `plotly express` package

Scatter plot

- One of the most used ways to discern a relationship between two variables is to create a scatter plot
- We can create scatter plot using one variable for the x-axis and one variable for the y-axis from the dataset
- With **interactive plots**, we can add extra layers by changing the size and color of certain items based on other variables
- We can easily zoom in and toggle an **item** to make observations without creating multiple graphs

Scatter plot (cont'd)

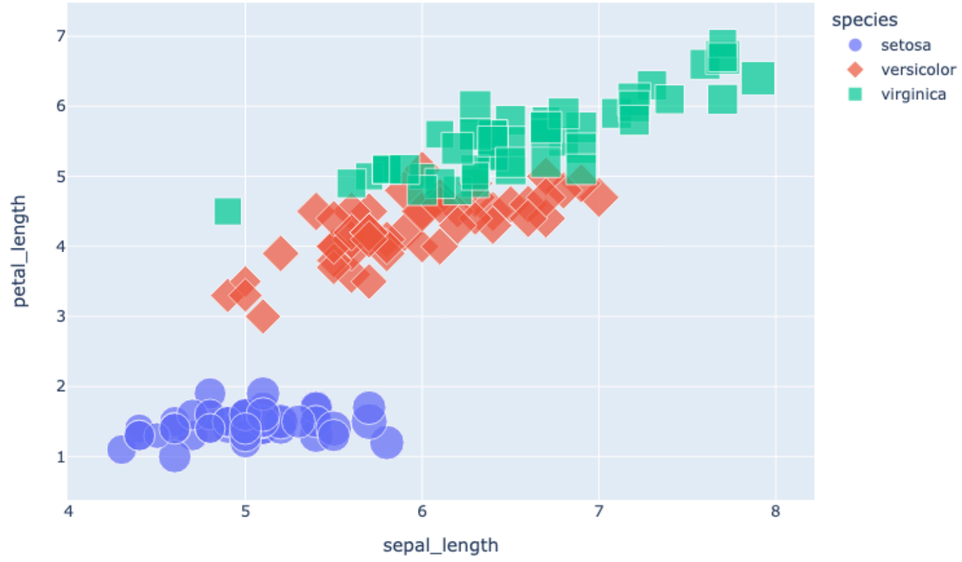
```
# Construct a scatter plot by simply giving the plotting function scatter
fig = px.scatter(iris_dataset,
           x='sepal_length',
           y='petal_length',
           color='species',
           size='sepal_width')
fig.show()
```



Scatter plot (cont'd)

- If you have a second item you wish to toggle, try using the `symbol` argument

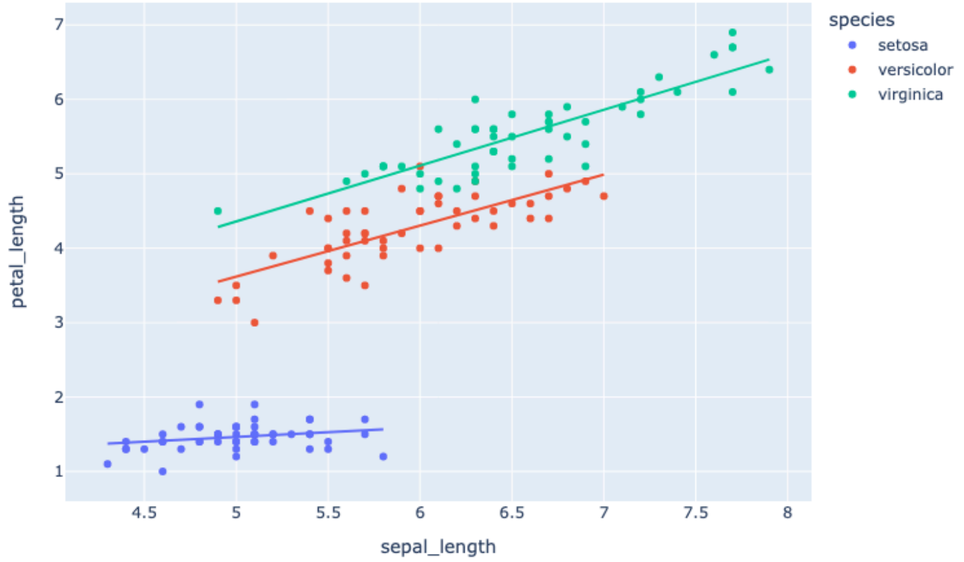
```
fig = px.scatter(iris_dataset,
           x='sepal_length',
           y='petal_length',
           color='species',
           size='sepal_width',
           symbol='species')
fig.show()
```



Scatter plot with a linear model

- Generally a scatter plot is used to see if there is a relationship between two variables, but it is also used to see if there is specifically a linear relationship
- `plotly` Express lets us create a linear regression for each of our groupings and see the model summary for each of those as well

```
fig = px.scatter(iris_dataset,
                 x='sepal_length',
                 y='petal_length',
                 color='species',
                 trendline="ols")
fig.show()
```



Scatter plot with a linear model (cont'd)

- We can also access the model summary from our graph to see how well the model fits our data

```
# Create and save model summary
results = px.get_trendline_results(fig)

# Access the model parameters
results.query("species=='setosa'").px_fit_results.iloc[0].summary()
```

- We'll see the result in next slide

Scatter plot with a linear model (cont'd)

OLS Regression Results

Dep. Variable:	y	**R-squared:**	0.070
Model:	OLS	**Adj. R-squared:**	0.050
Method:	Least Squares	**F-statistic:**	3.592
Date:	Wed, 31 Aug 2022	**Prob (F-statistic):**	0.0641
Time:	11:58:56	**Log-Likelihood:**	18.938
No. Observations:	50	**AIC:**	-33.88
Df Residuals:	48	**BIC:**	-30.05
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	0.8138	0.344	2.366	0.022	0.122	1.505
x1	0.1299	0.069	1.895	0.064	-0.008	0.268

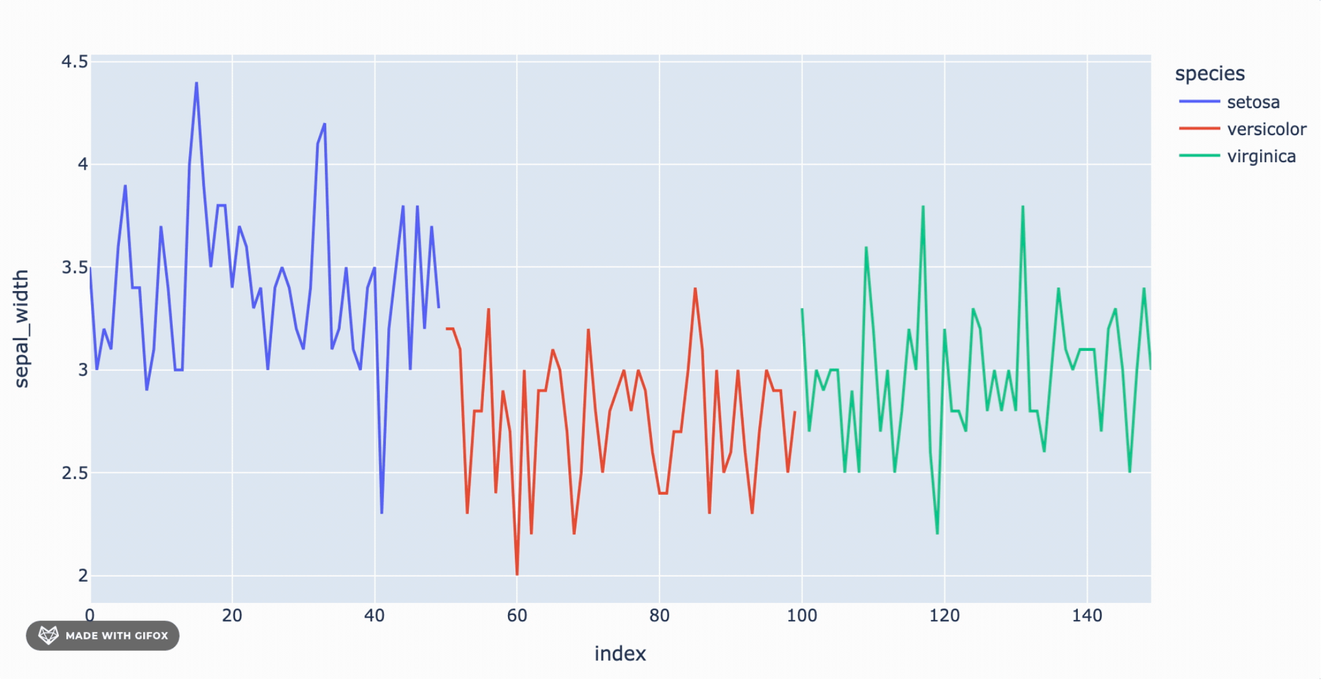
Omnibus:	2.950	**Durbin-Watson:**	1.434
Prob(Omnibus):	0.229	**Jarque-Bera (JB):**	2.272
Skew:	0.152	**Prob(JB):**	0.321
Kurtosis:	3.999	**Cond. No.**	75.0

Line plot

- If instead of looking at individual points, we actually want to see the trend over time or another variable, we may want to use a line plot instead

- You can see how `sepal_width` varies across observations for each `species`. Each color represents a species, and we can toggle species on and off in the legend to focus on one at a time

```
# Create a line graph by using the plotly function `line`
fig = px.line(iris_dataset,
              y='sepal_width',
              color='species')
fig.show()
```



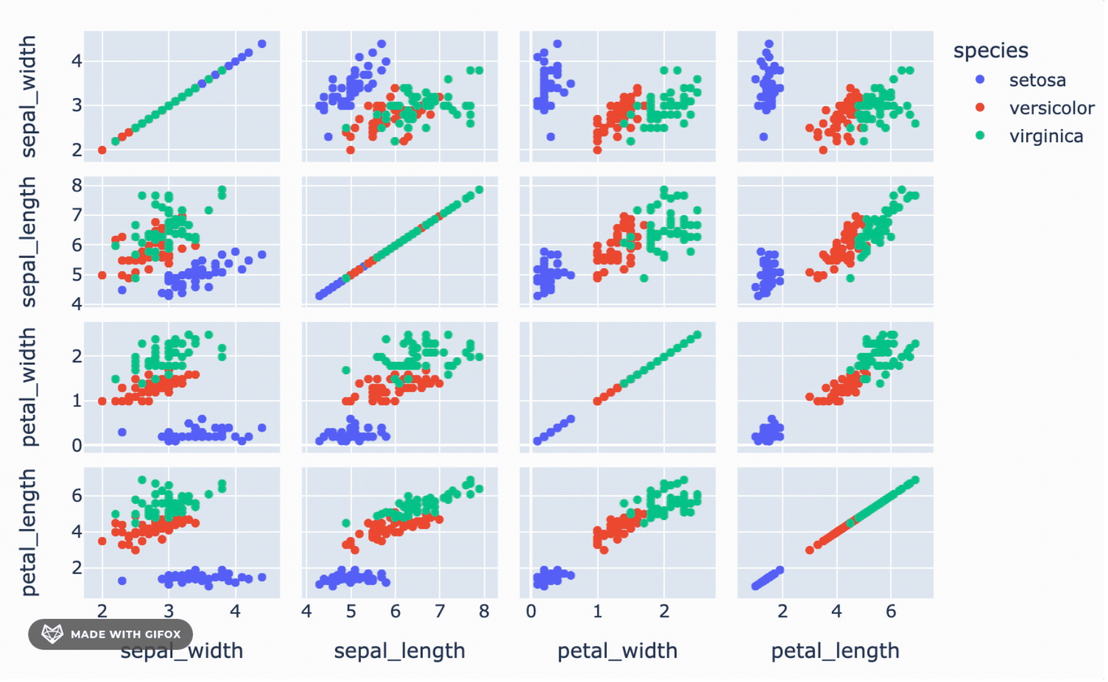
Multivariate plots

- Instead of just checking the relationship between two variables, maybe we want to look at the relationship between multiple pairs of variables. We can do that with plotly express as well
- Here are the types available:
  - Scatter Matrix plot
  - Parallel Coordinate plot
  - Parallel Category plot
- We'll be discussing `scatter matrix plot` in next slides

Scatter matrix plot

- One of the ways we can look at multiple variable pairs is the scatter matrix plot
- This is exactly what it sounds like, a **matrix of scatter plots** where each plot is a different pair of variables

```
# Construct a scatter plot by using plotly function
`scatter_matrix`
fig = px.scatter_matrix(iris_dataset,
          dimensions=["sepal_width", "sepal_length",
                      "petal_width", "petal_length"],
          color="species")
fig.show()
```



Module completion checklist

Objective	Complete
Describe bivariate plots and multivariate plots in Plotly	✔
Save plots in Plotly	

Saving plots in plotly to disk

- To save plots as HTML in `plotly express`, we simply need the variable we saved the plot to

```
# Save the plot as HTML
fig.write_html('scattermatrix.html')
```

- If you do save it without giving the full path, make sure you change your working directory so you know where the plot is saved to

Knowledge check

Module completion checklist

Objective	Complete
Describe bivariate plots and multivariate plots in Plotly	✔
Save plots in Plotly	✔

Congratulations on completing this module!

You are now ready to try Tasks 4-8 in the Exercise for this topic