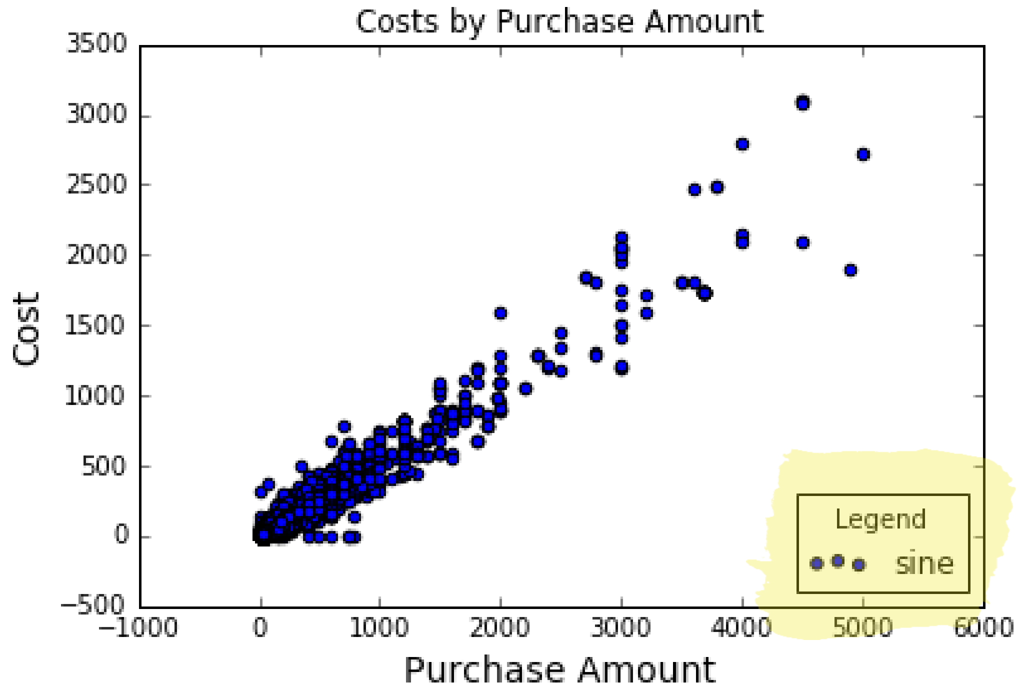
Adding further features to plots

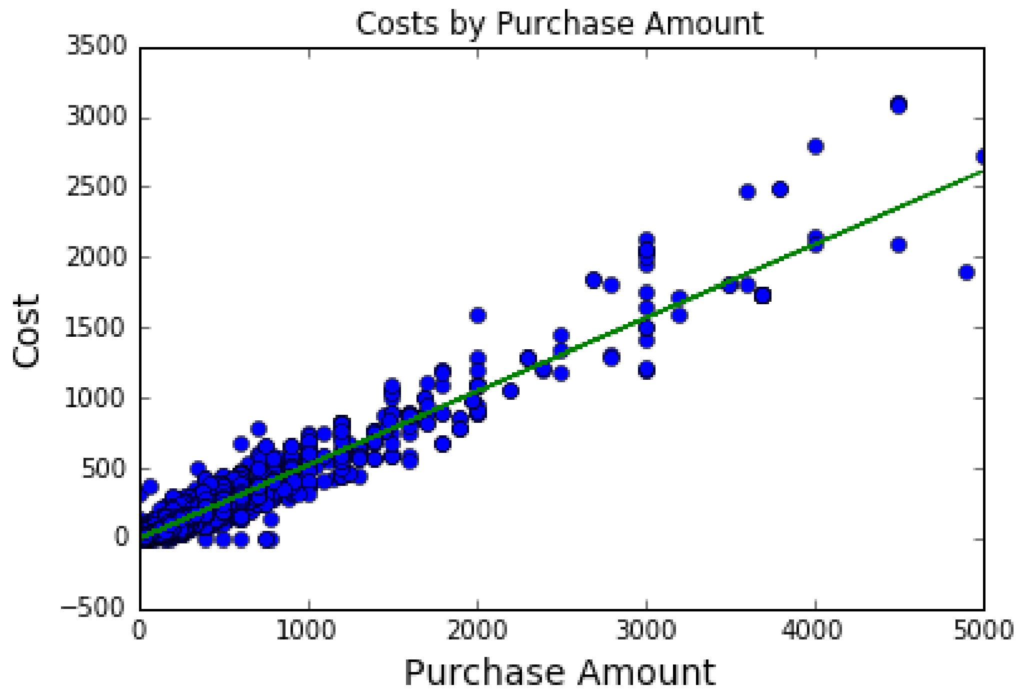
Further improve the aesthetic features of a plot: Add a legend to make your plot self-explanatory



```
plt.scatter( x=myData["PurchAmount"], y=myData["Cost"],
                        label='sine')
plt.legend(loc="lower right", fontsize=12, title="Legend")
                        plt.show()
```

Indicates the position of the legend in the plot

Further improve the aesthetic features of a plot: Lines can be added for extra information

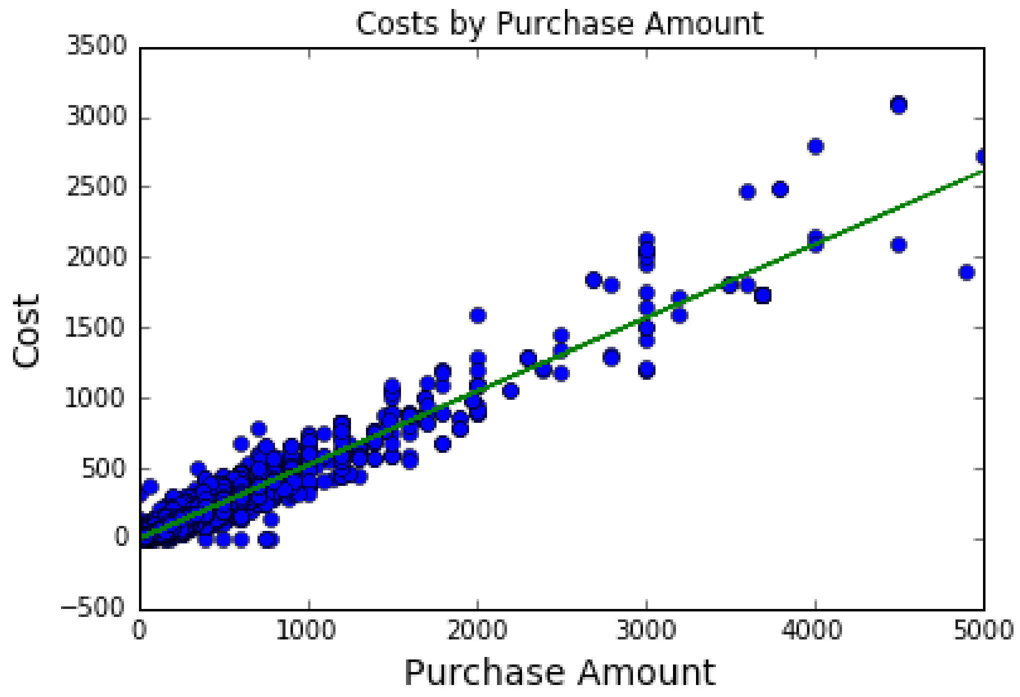


Fitting the linear regression line yields two values: ❶
1) Slope
2) Intercept

```
plt.plot(myData["PurchAmount"], myData["Cost"], "o")

fit = np.polyfit(x=myData["PurchAmount"], y=myData["Cost"], deg=1)
plt.plot(myData["PurchAmount"], fit[0] * myData["PurchAmount"] +
fit[1],"-", color="green")
plt.show()
```

Draw the regression line: ❷
y= m*x + b

Further improve the aesthetic features of a plot: Lines can be added for extra information



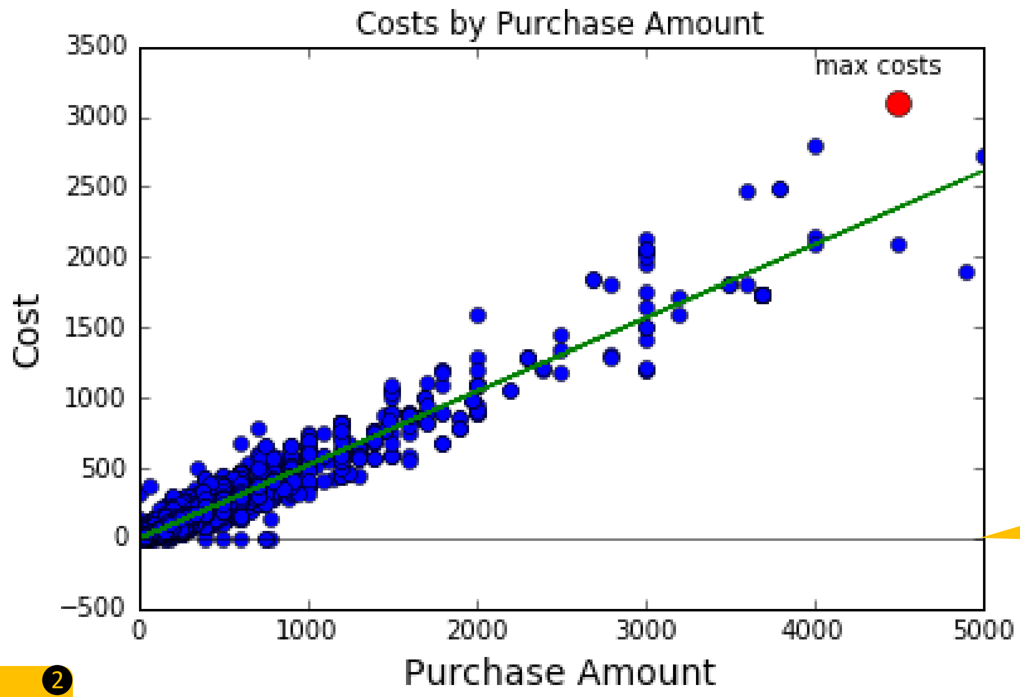
1 Fitting the linear regression line yields two values:
1) Slope
2) Intercept

```
plt.plot(myData["PurchAmount"], myData["Cost"], "o")

fit = np.polyfit(x=myData["PurchAmount"], y=myData["Cost"], deg=1)
 plt.plot(myData["PurchAmount"], fit[0] * myData["PurchAmount"] +
fit[1],"-", color="green")
plt.show()
```

3 Specify the line type

Draw the regression line:
y= m*x + b

Additional graphical elements can be added in the same fashion



❶ `axhline()` to add horizontal lines

❷ Also try `axvline()`

```
plt.axhline(y=0, xmin=min(myData["PurchAmount"]),
xmax=max(myData["PurchAmount"]), linewidth=0.5, color = 'k')
plt.text(4000,3300, "max costs")
plt.plot(4500, 3100, "o", color="red", markersize=10)
```

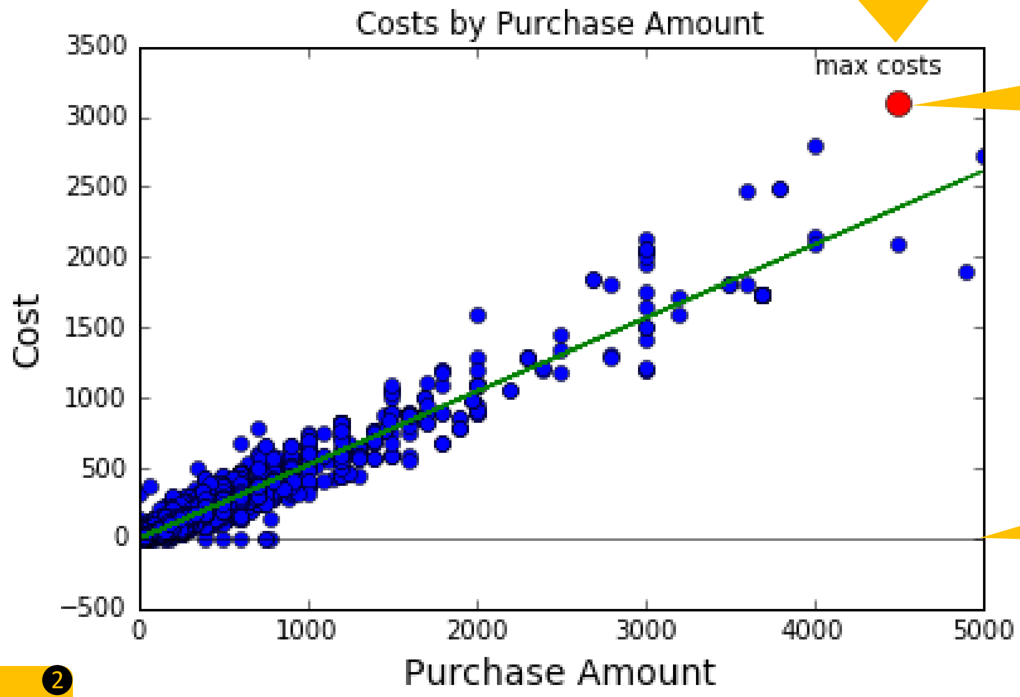
Additional graphical elements can be added in the same fashion

3 `text()` to add text.

4 `plt.plot()` to add additional points in different colors

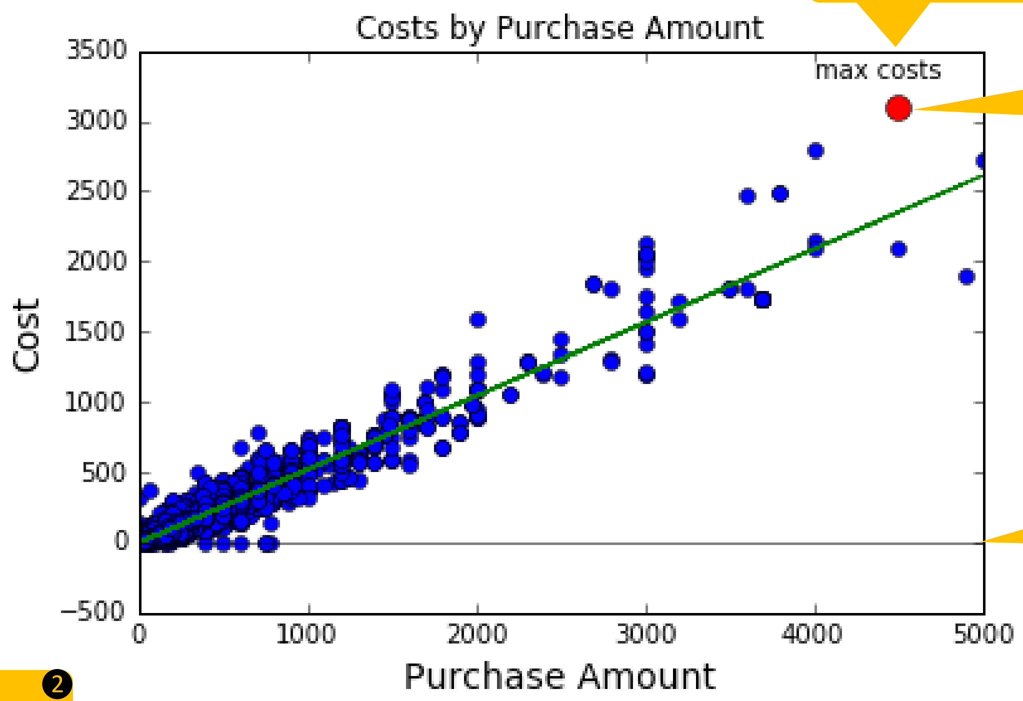
1 `axhline()` to add horizontal lines

2 Also try `axvline()`



```
plt.axhline(y=0, xmin=min(myData["PurchAmount"]),
xmax=max(myData["PurchAmount"]), linewidth=0.5, color = 'k')
plt.text(4000,3300, "max costs")
plt.plot(4500, 3100, "o", color="red", markersize=10)
```

Additional graphical elements can be added in the same fashion



3 `text()` to add text.

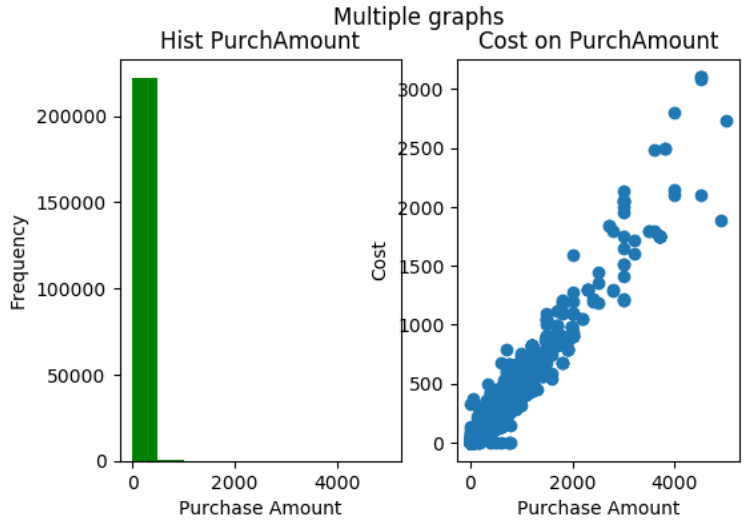
4 `plt.plot()` to add additional points in different colors

1 `axhline()` to add horizontal lines

2 Also try `axvline()`

```
plt.axhline(y=0, xmin=min(myData["PurchAmount"]),
xmax=max(myData["PurchAmount"]), linewidth=0.5, color = 'k')
plt.text(4000,3300, "max costs")
plt.plot(4500, 3100, "o", color="red", markersize=10)
```

You might want to plot multiple graphs in one image



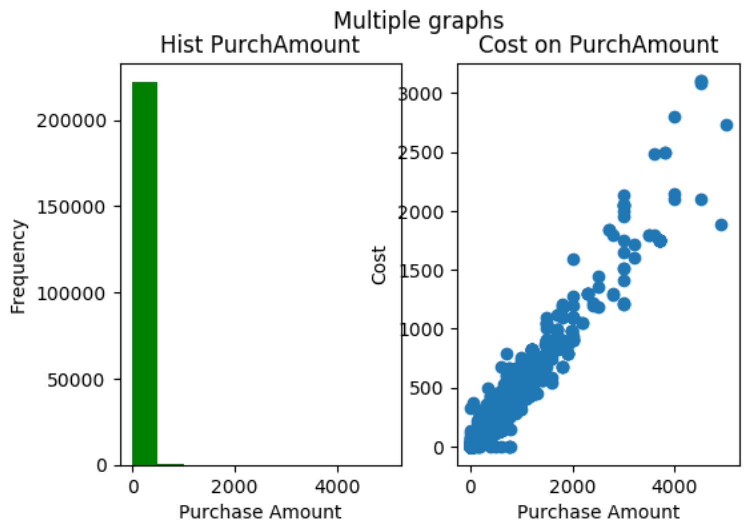
Rows ❶ Columns ❷ Plot 1 ❸

```
plt.subplot(1,2,1)
plt.hist(myData["PurchAmount"],
   color="green")
plt.title("Hist PurchAmount")
plt.xlabel("Purchase Amount")
plt.ylabel("Frequency")

plt.subplot(1,2,2)
plt.plot(myData["PurchAmount"],
   myData["Cost"], "o")
plt.title("Cost on PurchAmount")
…
plt.suptitle("Multiple graphs")
plt.show()
```

Plot 2 ❹

Common title ❺

You might want to plot multiple graphs in one image



Rows 1

Columns 2

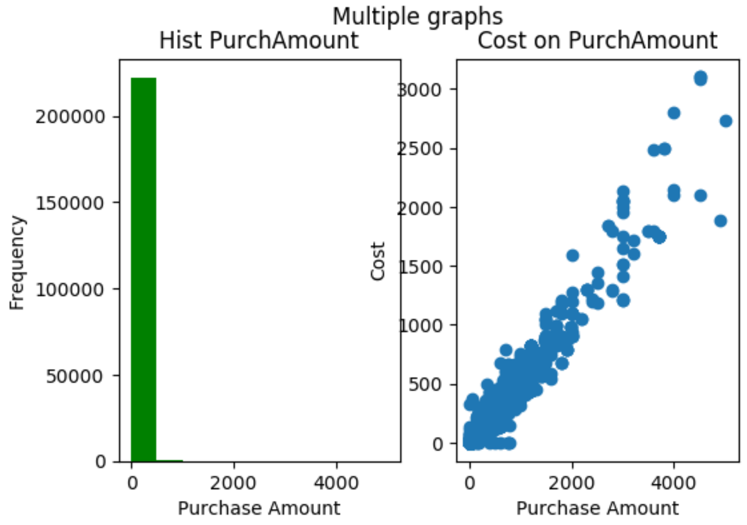
Plot 1 3

```
plt.subplot(1,2,1)
plt.hist(myData["PurchAmount"],
   color="green")
plt.title("Hist PurchAmount")
plt.xlabel("Purchase Amount")
plt.ylabel("Frequency")

plt.subplot(1,2,2)
plt.plot(myData["PurchAmount"],
   myData["Cost"], "o")
plt.title("Cost on PurchAmount")
…
plt.suptitle("Multiple graphs")
plt.show()
```

Plot 2 4

Common title 5

You might want to plot multiple graphs in one image



Rows 1

Columns 2

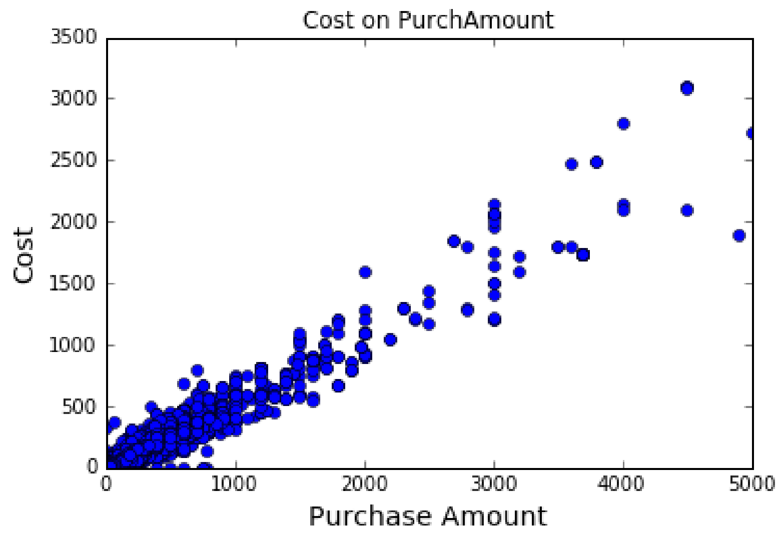
Plot 1 3

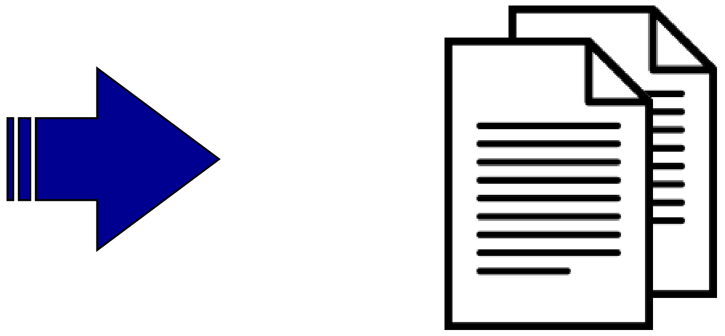
```
plt.subplot(1,2,1)
plt.hist(myData["PurchAmount"],
    color="green")
plt.title("Hist PurchAmount")
plt.xlabel("Purchase Amount")
plt.ylabel("Frequency")

plt.subplot(1,2,2)
plt.plot(myData["PurchAmount"],
    myData["Cost"], "o")
plt.title("Cost on PurchAmount")
…
plt.suptitle("Multiple graphs")
plt.show()
```

Plot 2 4

Common title 5

Step 6: Save your plot using the command line





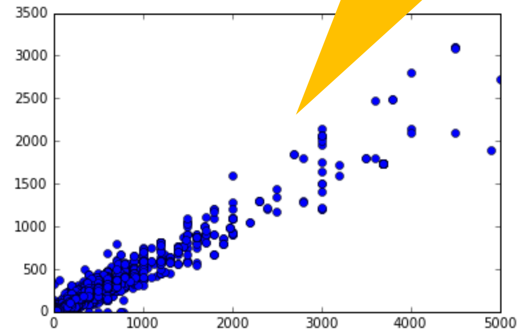
1 Caution: Do **not** show the plot with `plt.show()`: The picture will be stored blanc if you do

```
plt.plot(myData["PurchAmount"],myData["Cost"], "o")
plt.title("Cost on PurchAmount")
plt.savefig("Output.png")
```

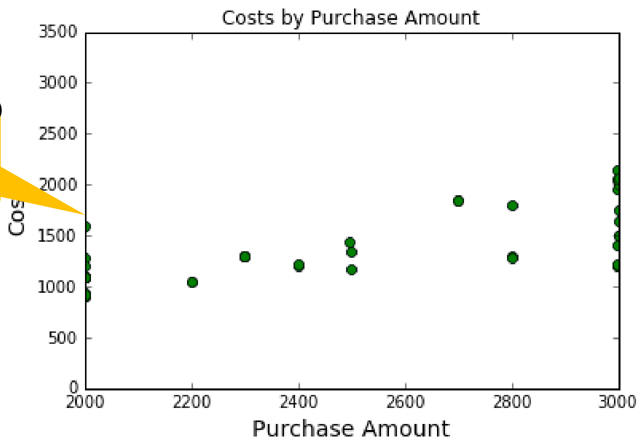
2 Save the figure

3 Use .jpg or .png as data formats

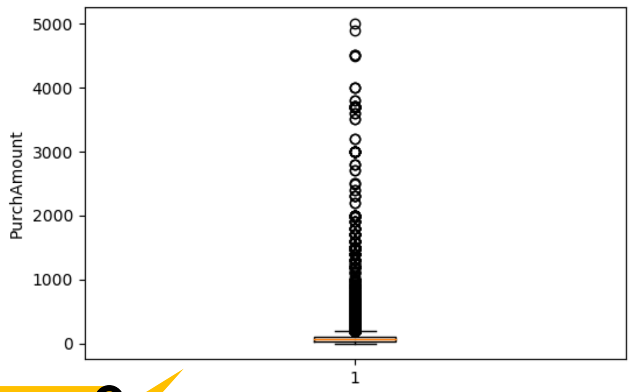
Be careful: Saving the plot with the wrong specifications can ruin your hard work!



❶ Axes labels and title are missing or not well specified.



❷ Incorrect x-limits showing the wrong extract.



❸ Width and/or height not appropriate.

- **Advice 1:** The point-and-click method helps to avoid this.
- **Advice 2:** Always save and comment your code, so that you can modify simple changes with little effort.

Adding further features to plots