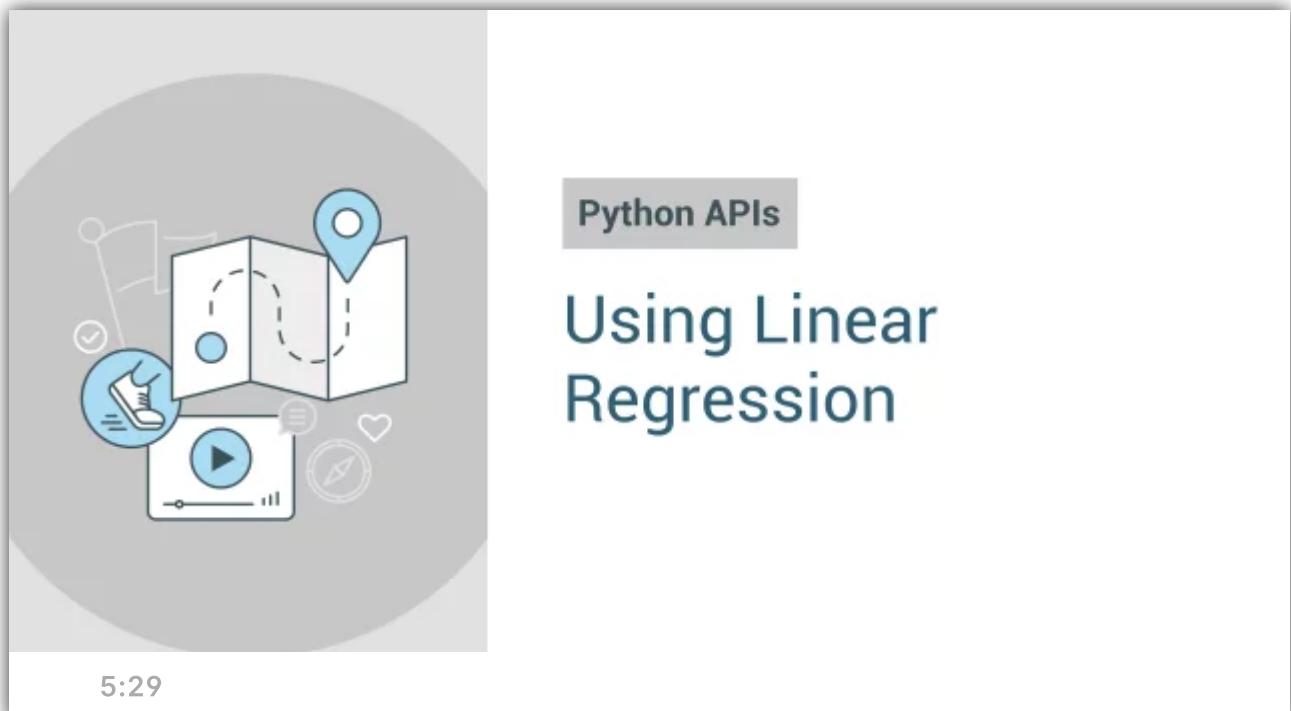


6.4.1

Use Linear Regression to Find the Relationship Between Variables

Jack loves your work on the scatter plots thus far, and the other tech companies were impressed, too. They ask you to put together something that can help the students explore how to determine correlations between weather data and latitude.

You and Jack decide to divide and conquer this task. He will write the piece on the scientific method and describe how to use linear regression on the scatter plots. You volunteer to use your scatter plot skills to create scatter plots for each weather parameter on the Northern and Southern Hemispheres. You'll need to also add a regression line equation and correlation coefficient to each scatter plot, so your first step is to brush up on linear regression. You know this is something Jack uses regularly, so you'll need to really understand it if you want to continue impressing the team.



Python APIs

Using Linear Regression

5:29

Linear regression is used to find a relationship between a dependent variable and one or more independent variables. The trick is to find something (a **dependent variable**) that depends on something else (the **independent variable**) and plot that relationship.

For example, let's say we wanted to understand how weather affects ice cream sales. We would model the linear regression between temperature (the independent variable) and ice cream sales (the dependent variable). Our hypothesis about the relationship would be that as temperatures rise, as they do in summer, more ice cream is sold. We will learn more hypothesis testing and building models in later modules.

For your project, you've already been working with independent and dependent variables. We have enough data to test relationships by creating scatter plots as we've done for each weather parameter vs. latitude. Plotting the data is the first step in determining if there might be an association between the two variables. For our scatter plots, the independent variable is the latitude, plotted on the x-axis, as its value is fixed. When we change the latitude, temperature changes, making it the dependent variable.

IMPORTANT

Independent variable: the variable changed by the analyst to observe how it affects the dependent variable

Dependent variable: the variable tested by the analyst to observe how it is affected by the independent variable

To determine if maximum temperature correlates to latitude, we can plot a linear regression line, a straight trendline predicting the average y-value, or dependent variable, for a given x-value, or independent variable. This line can be plotted using the equation $y = mx + b$, where "m" is the slope of the line and "b" is the y-intercept. For every x-value, or latitude we use in the equation, we will get a predicted temperature value.

To determine how strong the relationship is between the fitted line and the data, we find the correlation coefficient, or r-value. A correlation coefficient close to 1 shows a strong positive correlation, whereas close to -1 shows a strong negative correlation. A correlation coefficient close to zero is no correlation.

Let's practice using linear regression on fake weather data.

Practice Using Linear Regression

In a new cell of our `random_numbers` Jupyter Notebook file, we'll import the linear regression function from the SciPy statistics module.

```
# Import linear regression from the SciPy stats module.  
from scipy.stats import linregress
```

Next, we'll generate random latitudes as we did earlier. However, this time, latitudes will be in the Northern Hemisphere and therefore positive. In addition, we'll generate an equal number of random temperatures. Add the following list to a new cell and run the cell.

```
# Create an equal number of latitudes and temperatures.  
lats = [42.5, 43.9, 8.1, 36.8, 79.9, 69.1, 25.7, 15.3, 12.7, 64.5]  
temps = [80.5, 75.3, 90.9, 90.0, 40.4, 62.3, 85.4, 79.6, 72.5, 72.0]
```

Next, use the `linregress` function to calculate the slope, y-intercept, correlation coefficient (r-value), p-value, and standard deviation, and then we'll print out the equation for the line.

```
# Perform linear regression.  
(slope, intercept, r_value, p_value, std_err) = linregress(lats, temps)  
# Get the equation of the line.  
line_eq = "y = " + str(round(slope,2)) + "x + " + str(round(intercept,2))  
print(line_eq)  
print(f"The p-value is: {p_value:.3f}")
```

In the code to perform linear regression, the `linregress` function takes only two arguments, the x- and y-axes data (`lats` and `temps`) in the form of arrays. And it returns the following:

- Slope of the regression line as `slope`
- y-intercept as `intercept`
- Correlation coefficient as `r_value`
- p-value as `p_value`
- Standard error as `std_err`

IMPORTANT

The `slope`, `intercept`, `r_value`, `p_value`, and `std_err` are always returned when we run the `linregress` function. If you don't want to calculate one of these values but do not add it inside the parentheses, you'll get a `ValueError: too many values to unpack`.

To prevent this error, add a comma and underscore for each value you don't want to calculate.

For instance, if you don't want to print out the p-value and the standard error, write your function as `(slope, intercept, r_value, _, _) = linregress(x, y)`.

When we run the cell, we get the equation of a line for the data and the calculated probability (*p*-value).

```
y = -0.45x + 92.94
The p-value is: 0.011
```

NOTE

In statistics, the *p*-value is used to determine significance of results. In most cases, data scientists like to use a significance level of 0.05, which means:

A linear regression with a *p*-value > 0.05 is not statistically significant.

A linear regression with a *p*-value < 0.05 is statistically significant.

P-values can also be used to justify rejecting a null hypothesis. We will discuss *p*-values and hypothesis testing in more detail later in the course.

Now we can calculate the ideal temperatures (*y*-values) using the slope and intercept from the equation of the regression line. To do this, perform list comprehension on the latitudes by multiplying each latitude by the slope and adding the intercept.

Add the following code in a new cell and run the cell.

```
# Calculate the regression line "y values" from the slope and intercept.
regress_values = [(lat * slope + intercept) for lat in lats]
```

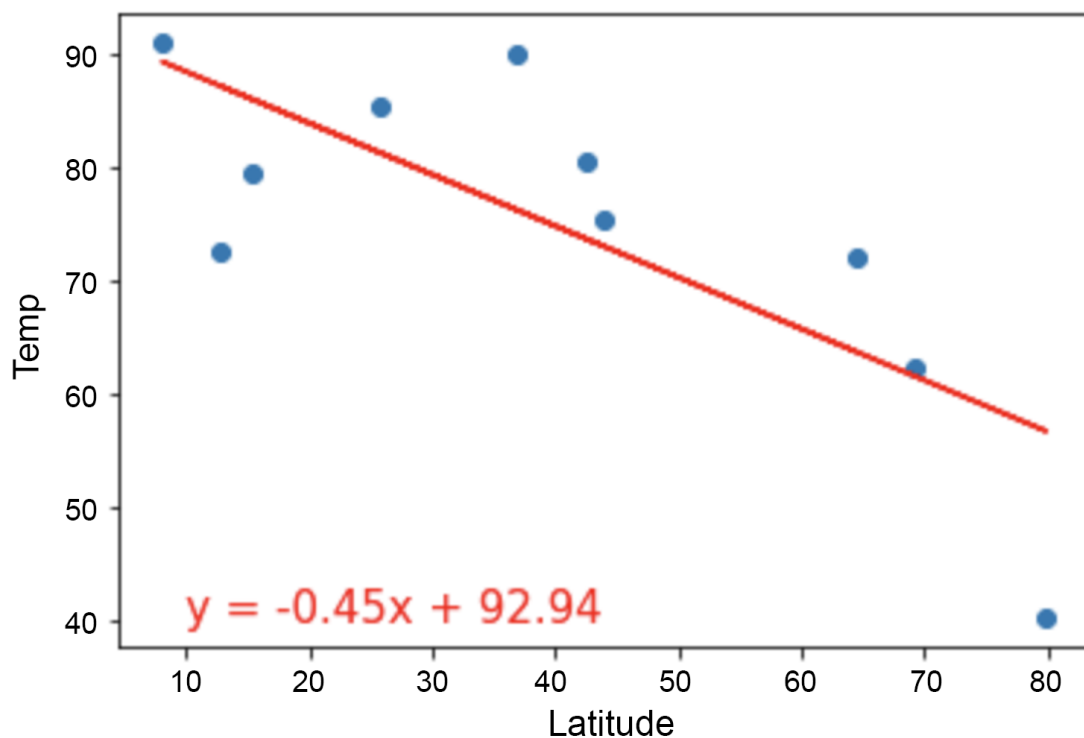
Add the following code to a new cell and run the cell to plot our data.

```
# Import Matplotlib.
import matplotlib.pyplot as plt
# Create a scatter plot of the x and y values.
plt.scatter(lats,temps)
# Plot the regression line with the x-values and the y coordinates based
plt.plot(lats,regress_values,"r")
# Annotate the text for the line equation and add its coordinates.
plt.annotate(line_eq, (10,40), fontsize=15, color="red")
plt.xlabel('Latitude')
plt.ylabel('Temp')
plt.show()
```

Let's review what this code does:

- We plot the latitudes and temperatures on a scatter plot.
- We create a line plot of our regression line with the ideal temperatures.
- We annotate the line plot by adding the equation of our regression line, where the x-axis is 10 and the y-axis is 40, and specify the font and color.
- We create x- and y-axes labels.

Now let's run the cell and plot our data.



Now that you are familiar with how to perform linear regression, let's use our data from the scatter plots.

NOTE

For more information, see the [documentation on the linear regression function](https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.linregress.html#scipy.stats.linregress) (<https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.linregress.html#scipy.stats.linregress>).

© 2020 - 2022 Trilogy Education Services, a 2U, Inc. brand. All Rights Reserved.