Introduction to Linear Regression - Introduction

Introduction

In this section, you're going to learn about one of the most basic machine learning models, linear regression! Many of the ideas you learn in this section will be foundational knowledge for more complex machine learning models.

Statistical Learning Theory

We'll start this section by exploring Statistical Learning Theory and how dependent and independent variables relate to it. Statistical Learning Theory provides an important framework for understanding machine learning.

Linear Regression

In this section, we'll introduce our first machine learning model - linear regression. It's really just a fancy way of saying "(straight) line of best fit", but it will introduce a number of concepts that will be important as you continue to learn about more sophisticated models.

Coefficient of Determination

We're then going to introduce the idea of "R squared" as the coefficient of determination to quantify how well a particular line fits a particular data set.

A Complete Regression

From there we look at calculating a complete linear regression, just using code. We'll cover some of the assumptions that must be held for a "least squares regression", introduce Ordinary Least Squares in Statsmodels and introduce some tools for diagnosing your linear regression such as Q-Q plots, the Jarque-Bera test for normal distribution of residuals and the Goldfield-Quandt test for heteroscedasticity. We then look at the interpretation of significance and p-value and finish up by doing a regression model of the Boston Housing data set.

Summary

Congratulations! You've made it through much of the introductory data and we've finally got enough context to take a look at our first machine learning model, while broadening our experience of both coding and math so we'll be able to introduce more sophisticated machine learning models as the course progresses.

Statistical Learning Theory

Introduction

In this lesson, you'll be introduced to Statistical Learning Theory and some key components in the framework of this theory. This is a particularly important theory as it encompasses the majority of statistical inference and functional analyses approaches. Statistical Learning Theory has applications in a wide variety of fields such as image and speech recognition, bioinformatics, sports, etc.

Objectives

You will be able to:

- Identify independent and dependent variables in a statistical model
- Describe loss and its importance in relation to model creation

Statistical Learning Theory

> Statistical Learning Theory is based on the idea of using data along with statistics to provide a framework for learning.

In Statistical Learning Theory, the main idea is to **construct a model** to draw certain conclusions from data, and next, to **use this model** to make predictions.

Types of Data in Statistical Learning

In the context of Statistical learning, there are two main types of data:

- **Dependent variables**: data that can be controlled directly (other names: outcome variables, target variables, response variables)
- **Independent variables**: data that cannot be controlled directly (other names: predictor variables, input variables, explanatory variables, features)

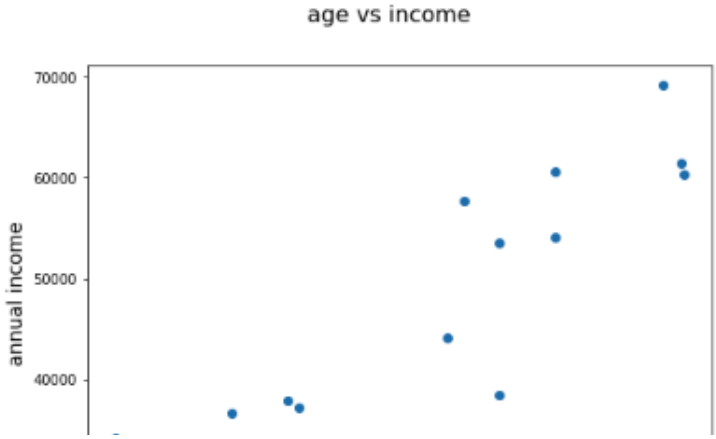
In models, the independent variable(s) are the variables that will affect (or will lead to a change in) the dependent variable(s).

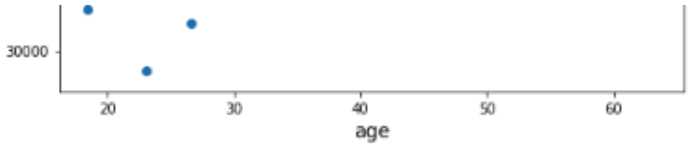
Two examples of common **independent variables** are age and time. There is nothing you can do to speed up or slow down time or increase or decrease age. They are independent of everything else.

An example of a **dependent variable** is how much you weigh at different ages. Here, the dependent variable (weight) depends on the independent variable (age). As someone's weight fluctuates over time, you can observe and record your weight as a dependent variable on your age.

Independent and dependent variables are normally shown on a graph under a standardized approach. This makes it easy for you to quickly see which variable is independent and which is dependent when looking at a graph or chart.

Conventionally, the independent variable goes on the x-axis, or the horizontal axis. Let's consider another example, one where we look at someone's income depending on their age. Below, you see a scatter plot where age is the independent variable, and income is the dependent variable. In this setting, **we want to study if age has some effect on annual income**.



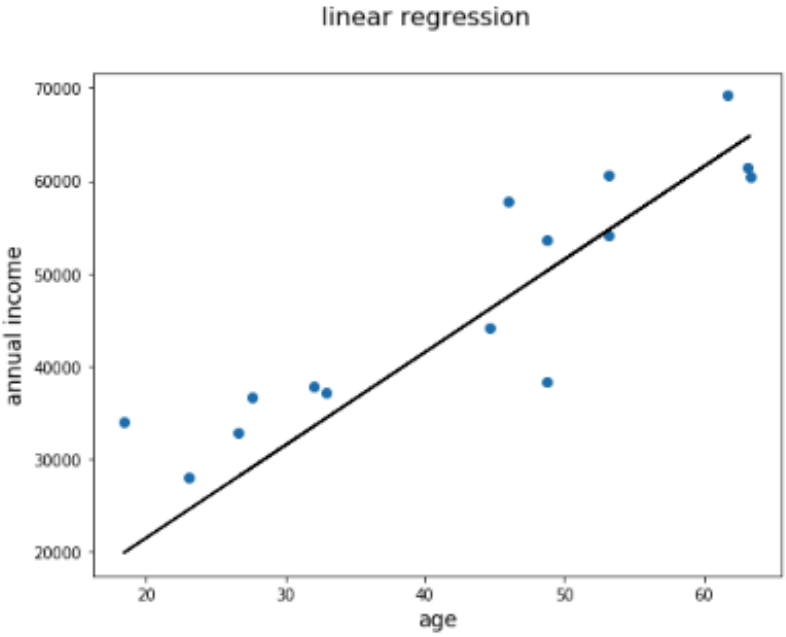


Statistical Model

> **A statistical model can be thought of as some kind of a transformation that helps us express dependent variables as a function of one or more independent variables**.

A statistical model defines a **relationship** between a dependent and an independent variable.

For the plot we see above, the relationship between age and income can be shown using a **straight line** connecting all the individual observations in the data. So this line here would be our **model** as shown in the image below.



We can define and **fit** such a straight line to our data following a straight line equation:

$$y = mx + c$$

You'll often come across greek letters talking about models like this. Another common way of writing a linear equation is (β is the Greek letter "beta"):

$$y = \beta_0 + \beta_1 x$$

β_0 has the same role as c in the first expression and denotes the *intercept with the y-axis*. β_1 has the same role as m in the first expression and denotes the *slope of the line*. More on this below.

Such a simple model would describe a person's height has **almost** a linear relationship with weight i.e. weight increases with height.

So this is our simple model for the relationship. Of course, we can use more sophisticated models like quadratic equations or polynomial equations for a **better fit**, and you may see this later on if you dig into more advanced modeling.

Looking at this line above, we can define it as **Income = 1500 + 1000 * Age**, based on slope (m or β_1) and intercept (c or β_0) values.

This would be our **linear model** (Linear refers to a model consisting of a straight line, or "linear regression"), which can help us work out a weight value for a given height. In summary,

> A model is expressed as a mathematical equation showing the relationship between dependent and independent variables.

Statistical Model Parameters

> **Model Parameters are the coefficients of the model equation for estimating the output**.

Statistical Learning is all about learning these parameters. A statistical learning approach would help us **learn** these parameters so we have a clear description of their relationship which we can replicate and analyze under different circumstances.

For the straight line above, we need to learn the **slope** and **intercept** for the line that best describes the relationship between the data elements in the dataset. We gave you the two values here, but in general, you'll have to **learn these values**. These values are denoted by **parameters**.

Once we have learned the m (or β_1) and c (or β_0) values, we can predict a value of y (income in our example) for a given value of x (age). In our next lab, you'll learn how to calculate these for a given dataset. Let's have a look at another example:

What Else Determines an Individual's Income?

If we suppose that income is a function of not only age, but also education level. A model that estimates the income could look like:

$$income = \beta_0 + \beta_1 * \text{age} + \beta_2 * \text{education level}$$

Here we have two independent variables i.e. age and education level, with the same dependent variable, income. β_0, β_1 and β_2 are model parameters.

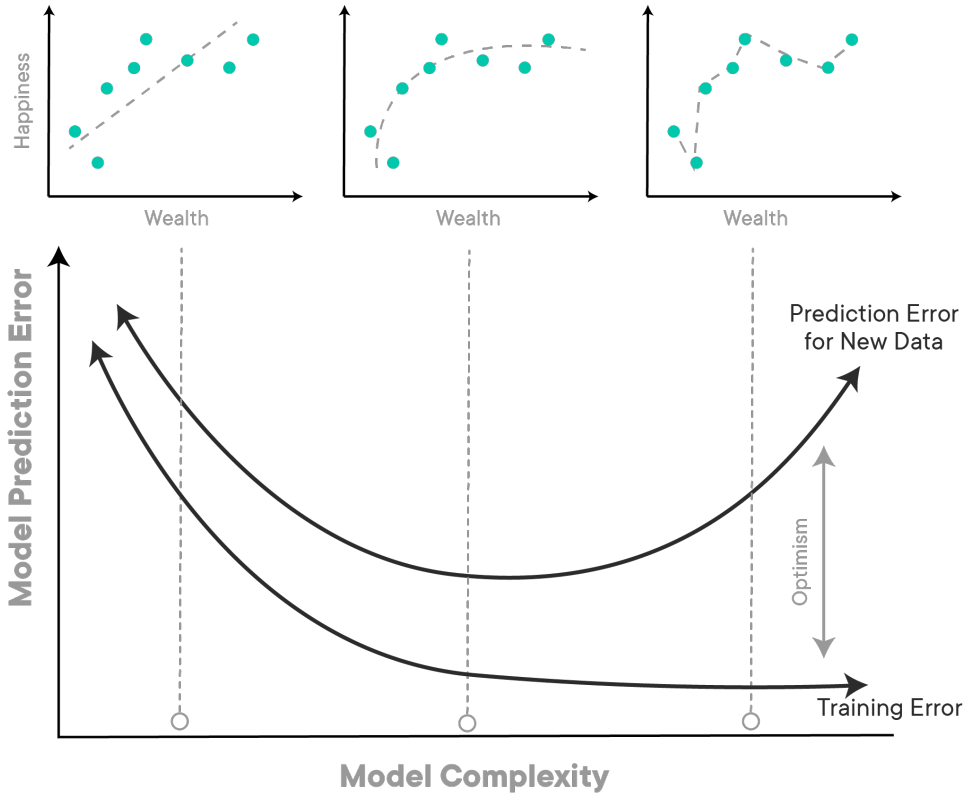
Model Generalization

As the data which is available to us for modeling is finite, the available data needs to be used very effectively to build and **validate** a model. Validation of the model usually makes the model more **generalizable** for unseen situations.

Training the model is like the infancy stage for humans. Examples are presented to the model and the model tweaks its parameters to better understand the data. Once the training is over, the model is unleashed upon new data and then uses what it has learned to explain that data. This is where problems can emerge. If we **over-train** the model on the training data i.e. make the model every detail of shown data, it will be able to identify all the relevant information in the training data, but will fail miserably when presented with the new data.

We then say that the **model is not capable of generalizing**, or that **model is over-fitting the training data**.

Here's a great example of the phenomenon: modeling happiness as a function of wealth.



In the top three diagrams, we have data and models (dashed curves). From left to right the models have been trained longer and longer on the training data. The training error curve in the bottom box shows that the training error gets better and better as we train longer (increasing model complexity). You may think that if we train longer we'll get better! Well, yes, but **only better at describing the training data**. The top right box shows a very complex model that hits all the data points. This model does great on the training data, but when presented with new data (examine the Prediction error curve in the bottom box) then it does worse!

In order to create good predictive models in machine learning that are capable of generalizing, one needs to know when to stop training the model so that it doesn't over-fit.

training the model so that it doesn't overfit.

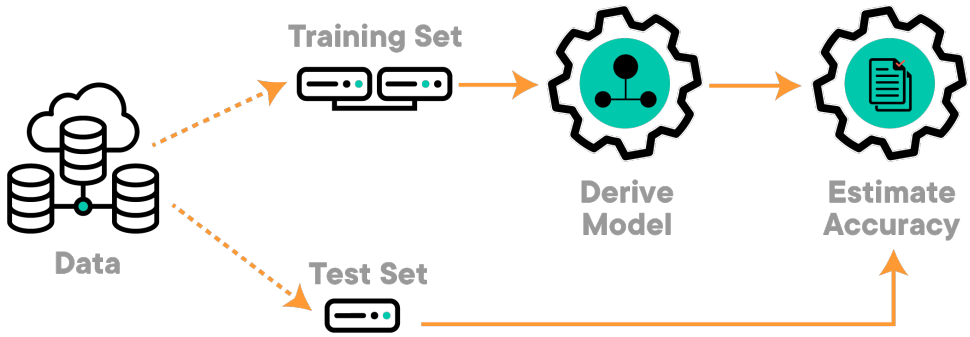
Model Validation

> **Model validation is a process of controlling overfitting and allows a higher degree of generalizability.**

Here is how we perform validation, in its simplest form:

- Split the data into two parts with a 70/30, 80/20 or a similar split
- Use the larger part for training so the model learns from it. This set of data is normally called the **Training Data**
- Use the smaller part for testing the model. This is data is not being used during the model learning process and used only for testing the performance of a learned model. This dataset is called as the **Testing Data**

This setup looks like as shown below:

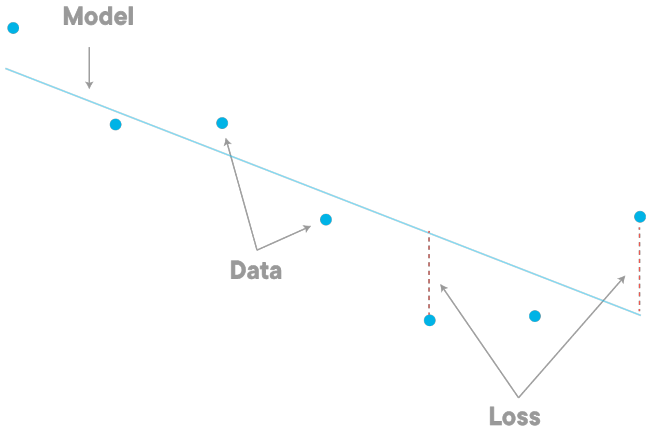


In statistical learning, if the model has learned well from the training data, it will perform well on both training data **and** test data. You can then use the test data to calculate the **accuracy**, which is assessed based on how close it has estimated the output to the actual value.

Model Loss

> **A loss function evaluates how well your model represents the relationship between data variables**.

If the model is unable to identify the underlying relationship between the independent and dependent variable(s), the loss function will output a very high number. Consider the age vs. income example above. You can see that the linear model is not exactly touching each data point because these points do not exist in a line. the individual distance of each point from the line is the **loss** that the model exhibits.



These individual losses, which is essentially the **vertical distance between the individual data points and the line** are taken into account to calculate the overall model loss.

If the relationship is well modeled, the loss will be low. As we change the parameters of our model to try and improve results, our loss function is our best friend, telling us if we are on the right track.

You'll learn about loss in further detail in upcoming lessons.

Additional Resources

- Youtube: Introduction to Statistical Learning Theory
- An Overview of Statistical Learning Theory with examples

Summary

Summary

In this lesson, you briefly looked at statistical learning theory and its main components. You looked at what a statistical model is and what the model parameters represent. You also got a feel for the differences between independent and dependent variables plus learned about loss and its role in model creation. You looked at all of this in the context of a simple model, a straight line. Next, you'll see the "learning" part of statistical learning theory by learning slope and intercept parameters of a straight line.

Simple Linear Regression

Introduction

Regression analysis is often the first real learning application that aspiring data scientists will come across. It is one of the simplest techniques to master, but it still requires some mathematical and statistical understanding of the underlying process. This lesson will introduce you to the regression process based on the statistical ideas we have discovered so far.

Objectives

You will be able to:

- Perform a linear regression using self-constructed functions
- Interpret the parameters of a simple linear regression model in relation to what they signify for specific data

Linear Regression

Regression analysis is one of the most important statistical techniques for business applications. It's a statistical methodology that helps estimate the strength and direction of the relationship between two (or more) variables. Regression results show whether the relationship is valid or not. It also helps to *predict* an unknown value based on the derived relationship.

> Regression Analysis is a **parametric** technique meaning a set of parameters are used to **predict** the value of an unknown target variable (or dependent variable) y based on one or more of known input features (or independent variables, predictors), often denoted by x.

Let's consider another example. Someone's height and foot size are generally considered to be related. Generally speaking, taller people tend to have bigger feet (and, obviously, shoe size).

We can use a linear regression analysis here to predict foot size (dependent variable), given height (independent variable) of an individual. Regression is proven to give credible results if the data follows some assumptions which will be covered in upcoming lessons in detail. In general, regression analysis helps us in the following ways:

- Finding an **association** or relationship between certain phenomena or variables
- Identifying **which variables contribute** more towards the outcomes
- **Prediction** of future observations

Why "linear" regression?

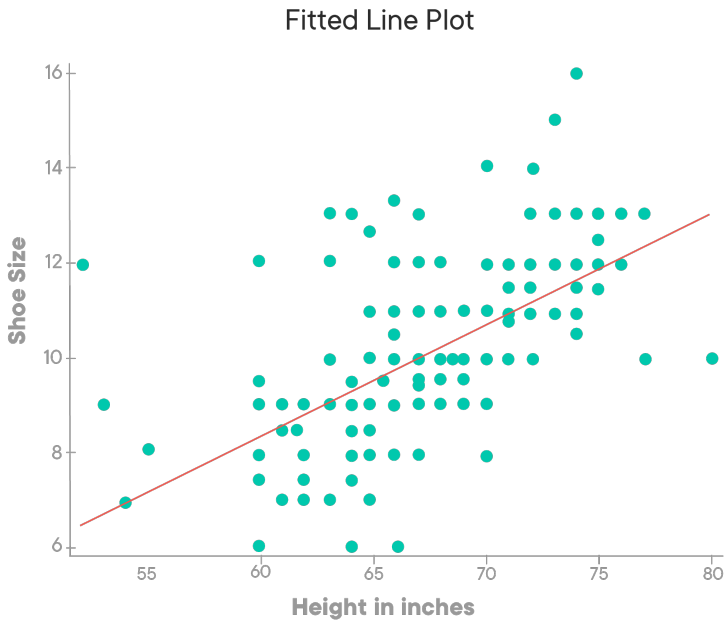
The term **linear** implies that the model functions along with a straight (or nearly straight) line. **Linearity**, one of the assumptions of this approach, suggests that the relationship between dependent and independent variables can be expressed as a straight line.

Simple Linear Regression uses a single feature (one independent variable) to model a linear relationship with a target (the

dependent variable) by fitting an optimal model (i.e. the best straight line) to describe this relationship.

Multiple Linear Regression uses more than one feature to predict a target variable by fitting the best linear relationship.

In this section, we will mainly focus on simple regression to build a sound understanding. For the example shown above i.e. height vs foot size, a simple linear regression model would fit a line to the data points as follows:



This line can then be used to describe the data and conduct further experiments using this fitted model. So let's move on and see how to calculate this "best-fit line" in a simple linear regression context.

Calculating Regression Coefficients: Slope and Intercepts

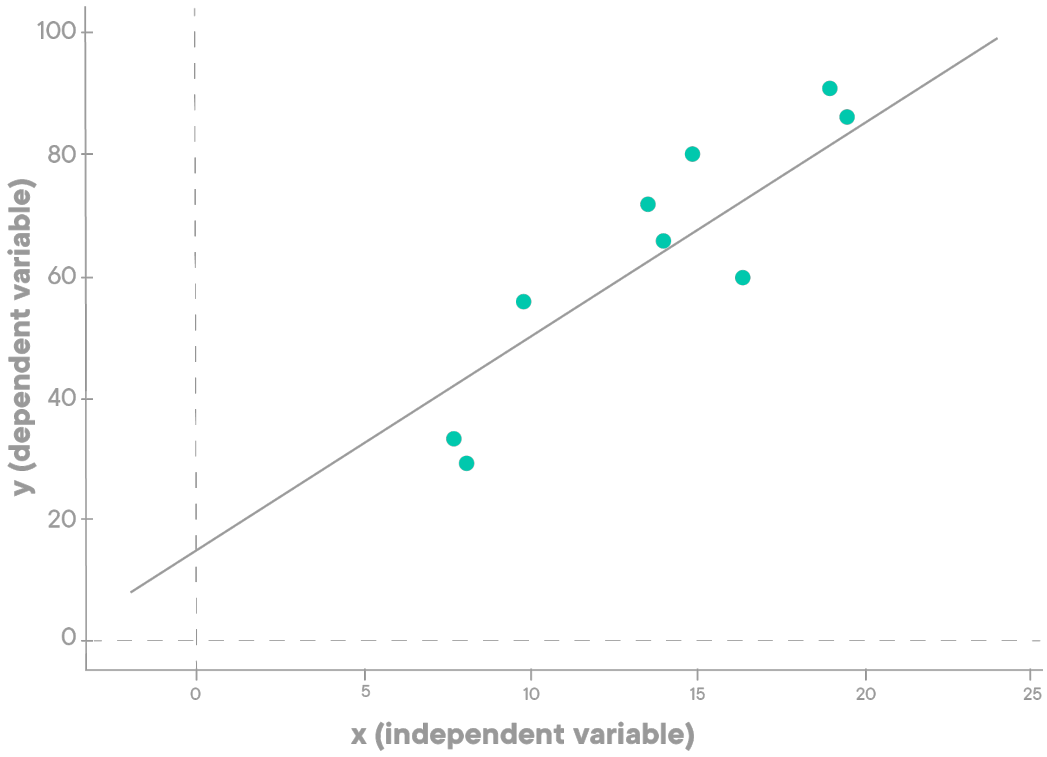
A straight line can be written as :

$$y = mx + c$$

or, alternatively

$$y = \beta_0 + \beta_1 x$$

You may come across other ways of expressing this straight line equation for simple linear regression. Yet there are **four key components** you'll want to keep in mind:



- A **dependent variable** that needs to estimated and predicted (here: y)

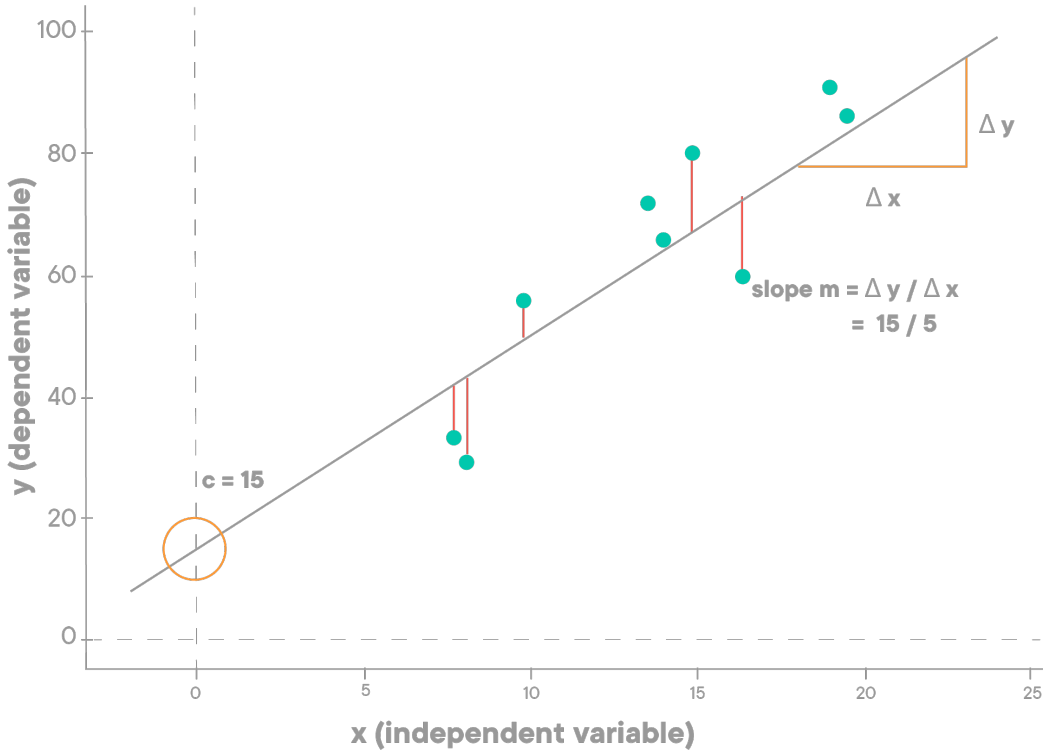
- An **independent variable**, the input variable (here: x)
- The **slope** which determines the angle of the line. Here, the slope is denoted as m, or β_1.
- The **intercept** which is the constant determining the value of y when x is 0. We denoted the intercept here as c or β_0.

> *Slope* and *Intercept* are the **coefficients** or the **parameters** of a linear regression model. Calculating the regression model simply involves the calculation of these two values.

Linear regression is simply a manifestation of this simple equation! So this is as complicated as our linear regression model gets. The equation here is the same one used to find a line in algebra, but in statistics, the actual data points don't necessarily lie on a line!

> The real challenge for regression analysis is to fit a line, out of an infinite number of lines that best describes the data.

Consider the line below to see how we calculate slope and intercept.



In our example:

c is equal to 15, which is where our line intersects with the y-axis.

m is equal to 3, which is our slope.

You can find a slope by taking an arbitrary part of the line, looking at the differences for the x-value and the y-value for that part of the line, and dividing Δy by Δx. In other words, you can look at the **change in y over the change in x** to find the slope!

Important note on notation

Now that you know how the slope and intercept define the line, it's time for some more notation.

Looking at the above plots, you know that you have the green dots that are our observations associated with x- and y-values.

Now, when we draw our regression line based on these few green dots, we use the following notations:

$$\hat{y} = \hat{m}x + \hat{c}$$

or

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x$$

As you can see, you're using a "hat" notation which stands for the fact that we are working with **estimations**.

- When trying to draw a "best fit line", you're **estimating** the most appropriate value possible for your intercept and your slope, hence $\hat{c}$ /$\hat{\beta}_0$ and $\hat{m}$ /$\hat{\beta}_1$.

- Next, when we use our line to predict new values y given x, your estimate is an **approximation** based on our estimated parameter values. Hence we use $\hat{y}$ instead of y. $\hat{y}$ lies *ON* your regression line, y is the associated y-value for each of the green dots in the plot above. The **error** or the **vertical offset** between the line and the actual observation values is denoted by the red vertical lines in the plot above. Mathematically, the vertical offset can be written as $|\hat{y} - y|$.

So how do you find the line with the best fit? You may think that you have to try lots and lots of different lines to see which one fits best. Fortunately, this task is not as complicated as in may seem. Given some data points, the best-fit line always has a distinct slope and y-intercept that can be calculated using simple linear algebraic approaches. Let's quickly visit the required formulas.

Best-Fit Line Ingredients

Before we calculate the best-fit line, we have to make sure that we have calculated the following measures for variables X and Y:

- The mean of the X ($\bar{X}$)
- The mean of the Y ($\bar{Y}$)
- The standard deviation of the X values (S_X)
- The standard deviation of the y values (S_Y)
- The correlation between X and Y (often denoted by the Greek letter "Rho" or ρ - Pearson Correlation)

Calculating Slope

With the above ingredients in hand, we can calculate the slope (shown as b below) of the best-fit line, using the formula:

$$\hat{m} = \rho \frac{S_Y}{S_X}$$

This formula is also known as the **least-squares method**.

You can visit this Wikipedia link to get take a look into the math behind the derivation of this formula.

The slope of the best-fit line can be a negative number following a negative correlation. For example, if an increase in police officers is related to a decrease in the number of crimes in a linear fashion, the correlation and hence the slope of the best-fitting line in this particular setting is negative.

Calculating Intercept

So now that we have the slope value (\hat m), we can put it back into our formula ($\hat{y} = \hat{m}x + \hat{c}$) to calculate intercept. The idea is that

$$\bar{Y} = \hat{c} + \hat{m}\bar{X}$$

$$\hat{c} = \bar{Y} - \hat{m}\bar{X}$$

Recall that $\bar{X}$ and $\bar{Y}$ are the mean values for variables X and Y. So, in order to calculate the $\hat{y}$-intercept of the best-fit line, we start by finding the slope of the best-fit line using the above formula. Then to find the $\hat{y}$-intercept, we multiply the slope value by the mean of x and subtract the result from the mean of y.

Predicting from the model

As mentioned before, when you have a regression line with defined parameters for slope and intercept as calculated above, you can easily predict the $\hat{y}$ (target) value for a new x (feature) value using the estimated parameter values:

$$\hat{y} = \hat{m}x + \hat{c}$$

Remember that the difference between y and $\hat{y}$ is that $\hat{y}$ is the value predicted by the fitted model, whereas y carries actual values of the variable (called the truth values) that were used to calculate the best fit.

Next, let's move on and try to code these equations to fit a regression line to a simple dataset to see all of this in action.

Additional Reading

Visit the following series of blogs by Bernadette Low for details on topics covered in this lesson.

- Super Simple Machine Learning—Simple Linear Regression Part 1
- Super Simple Machine Learning—Simple Linear Regression Part 2

Summary

In this lesson, you learned the basics of a simple linear regression. Specifically, you learned some details about performing the actual technique and got some practice interpreting regression parameters. Finally, you saw how the parameters can be used to make predictions!

Coefficient of Determination

Introduction

For linear regression analysis, as we saw earlier, the straight line does not **fully** describe the relationship between variables and there is always some error. In general, you'll want to determine a "goodness of fit"-measure of the fitted line. In this lesson, you'll learn about the "R-Squared"(R^2) measure, also known as the Coefficient of Determination.

Objectives

You will be able to:

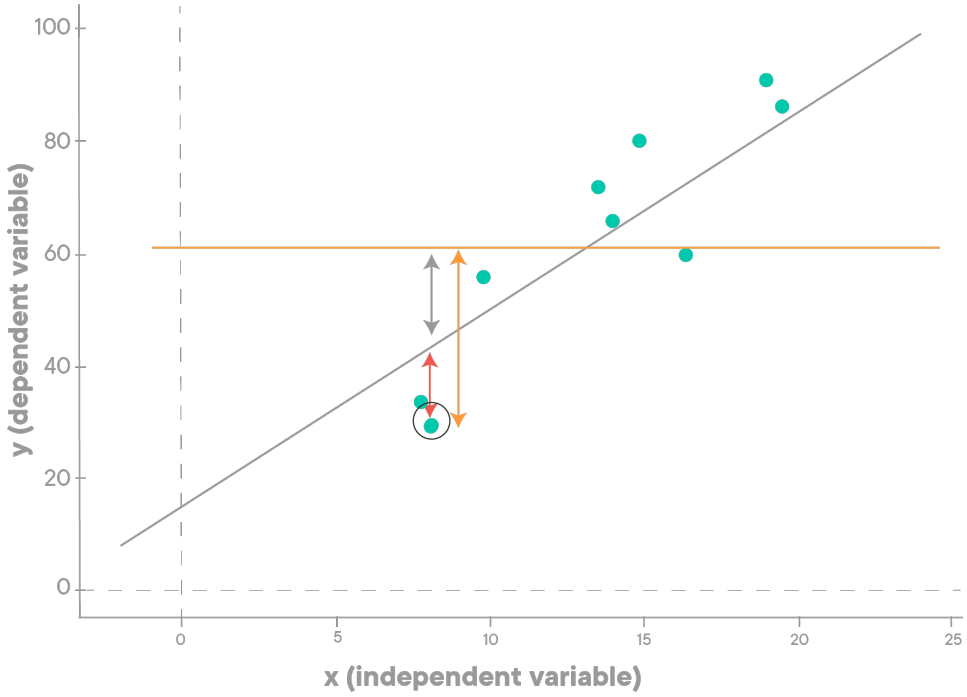
- Calculate the coefficient of determination using self-constructed functions
- Use the coefficient of determination to determine model performance

R-Squared

> **The R^2 or Coefficient of determination is a statistical measure that is used to assess the goodness of fit of a regression model**

Here is how it works.

R-Squared uses a so-called "baseline" model which is a very simple, naive model. This baseline model does not make use of any independent variables to predict the value of dependent variable Y. Instead, it uses the **mean** of the observed responses of the dependent variable y and always predicts this mean as the value of y for any value of x. In the image below, this model is given by the straight orange line.



You can see that, in this plot, the baseline model always predicts the mean of y **irrespective** of the value of the x. The gray line, however, is our fitted regression line which makes use of x values to predict the values of y. Looking at the plot above, R-Squared simply asks the question:

> **Is our fitted regression line better than our baseline model?**

Any regression model that we fit is compared to this baseline model to understand its **goodness of fit**. Simply put, R-Squared just explains how good your model is when compared to the baseline model. That's about it.

Great, so how do you calculate R-Squared?

The mathematical formula to calculate R-Squared for a linear regression line is in terms of **squared errors** for the fitted model and

the baseline model. It's calculated as :

$$R^2 = 1 - \frac{SS_{RES}}{SS_{TOT}} = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y}_i)^2}$$

- SS_{RES} (also called RSS) is the **Residual** sum of squared errors of our regression model also known as $\boldsymbol{SSE}$ (Sum of Squared Errors). SS_{RES} is the squared difference between y and $\hat{y}$. For the one highlighted observation in our graph above, the SS_{RES} is denoted by the red arrow. This part of the error is not explained by our model.

- SS_{TOT} (also called TSS) is the **Total** sum of squared error. SS_{TOT} is the squared difference between y and $\bar{y}$. For the one highlighted observation in our graph above, the SS_{TOT} is denoted by the orange arrow.

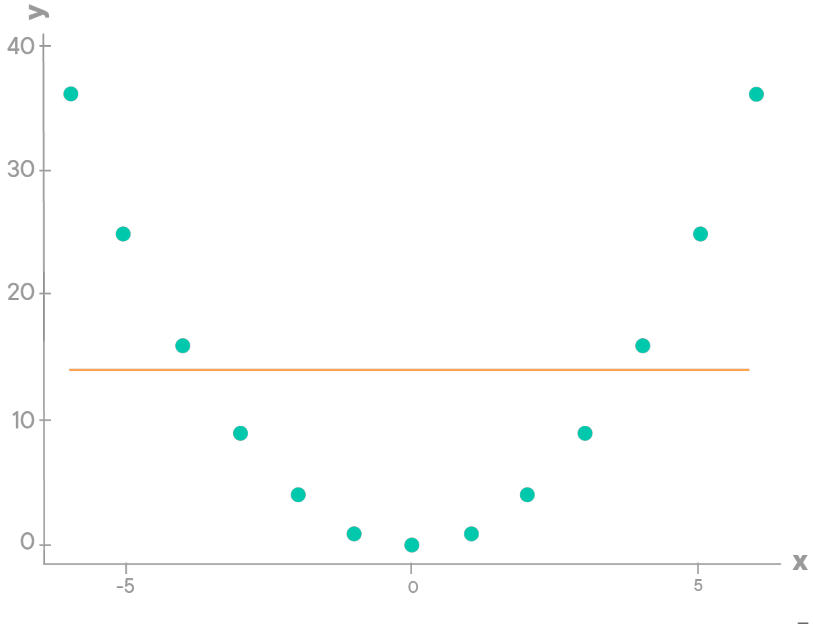
Looking at this, you'll understand that you can interpret R-Squared as "1 - the proportion of the variance *not* explained by the model", which means as much as "the variation explained by the model". As a result, you'll want to maximize the R-Squared.

For completion,

- SS_{EXP} (also called ESS) is the **Explained** sum of squared error. SS_{EXP} is the squared difference between $\hat{y}$ and $\bar{y}$. For the one highlighted observation in our graph above, the SS_{EXP} is denoted by the gray arrow.

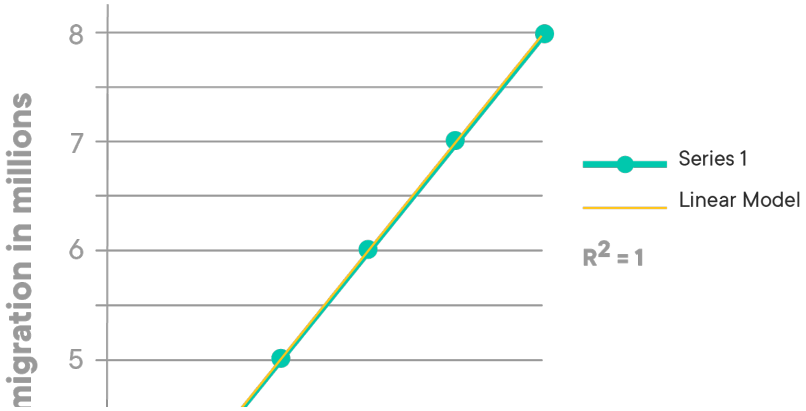
Let's interpret the outcome of R^2

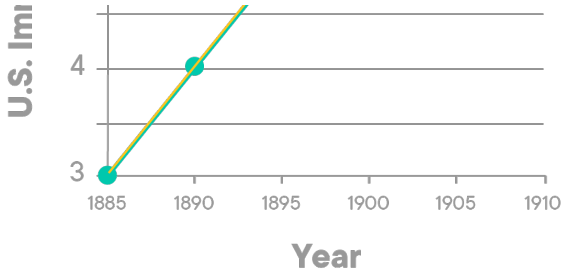
Our worst possible regression model could be the baseline model itself. In that case, the RSS is equal to TSS looking at the graph above. Then your R_squared value is 0. Look at the plot below, where you notice that $\hat{y}$ is simply a straight line.



Due to this particular relationship between x and y, the regression line $\hat{y}$ is the same as the mean line for $\bar{y}$. The R-Squared for this model is 0. It's clear that a straight line is probably not the right fit for this data.

On the other extreme, the best model could also be one that fits all the data points perfectly. Because the unexplained part of the variation is 0, R-Squared is 1–0, so 1 in this case, which indicates a perfect model. Below is an example of this (know that this will rarely happen with real world data).





> **R-Squared can take a value between 0 and 1 where values closer to 0 represent a poor fit and values closer to 1 represent an (almost) perfect fit**

Phrasing R-Squared values

An R-squared value of say 0.85 can be described conceptually as:

> ***85% of the variations in dependent variable y are explained by the independent variable in our model.***

Summary

In this lesson, you looked at the R-Squared, or the Coefficient of Determination to evaluate the goodness of fit for a regression line. You saw how R-Squared is calculated by comparing a given model to a baseline model and learned that it must be a value between 0 and 1. In the next lab, you'll move on to calculating R-Squared in Python.

Assumptions for Linear Regression

Introduction

Least Squares is one of the most common regression techniques for linear models. As long as our model satisfies the least squares regression assumptions, we can get the best possible estimates. In this lesson, you will learn about these assumptions.

Objectives

You will be able to:

- List the assumptions of linear regression
- Determine if a particular set of data exhibits the assumptions of linear regression

About Regression Assumptions

Regression is a powerful analysis technique that is routinely used to answer complex analytical questions. However, if some of the necessary assumptions are not satisfied, you may not be able to get good and trustworthy results!

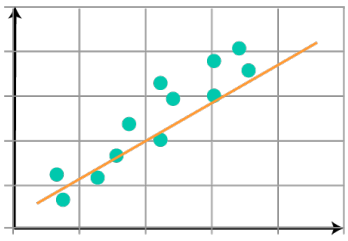
In this lesson, you'll dig deeper into the topic of ordinary least squares (OLS) regression assumptions. Additionally, you'll learn about their importance as well as some techniques to help us determine whether your model satisfies the assumptions.

Regression is "Parametric"

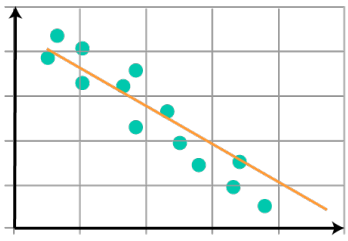
Regression is a parametric technique, which means that it uses parameters learned from the data. Because of that, certain assumptions must be made. These assumptions define the complete scope of regression analysis and it is **mandatory** that the underlying data fulfills these assumptions. If violated, regression makes biased and unreliable predictions. Luckily, we have measures to check for these assumptions.

1. Linearity

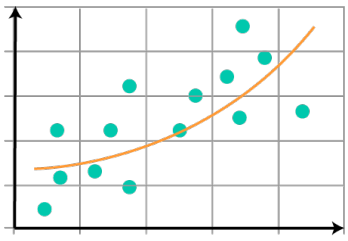
> The linearity assumptions requires that there is a **linear relationship** between the response variable (Y) and predictor (X). Linear means that the change in Y by 1-unit change in X, is constant.



Linear Relationship



Linear Relationship



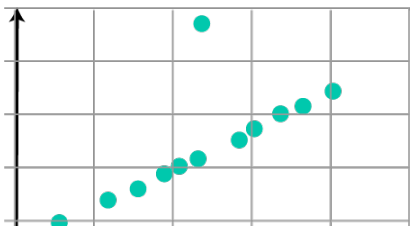
No Linear Relationship

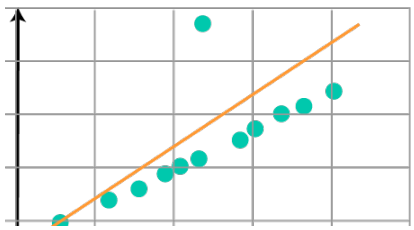
As shown above, If we try to fit a linear model to a non-linear data set, OLS will fail to capture the trend mathematically, resulting in an inaccurate relationship. This will also result in erroneous predictions on an unseen data set.

> The linearity assumption can best be tested with scatter plots

For non-linear relationships, you can use non-linear mathematical functions to fit the data e.g. polynomial and exponential functions. You'll come across these later.

Note: As an extra measure, it is also important to check for outliers as the presence of outliers in the data can have a major impact on the model.





In the above example, we can see that an outlier prohibits the model to estimate the true relationship between variables by introducing bias.

2. Normality

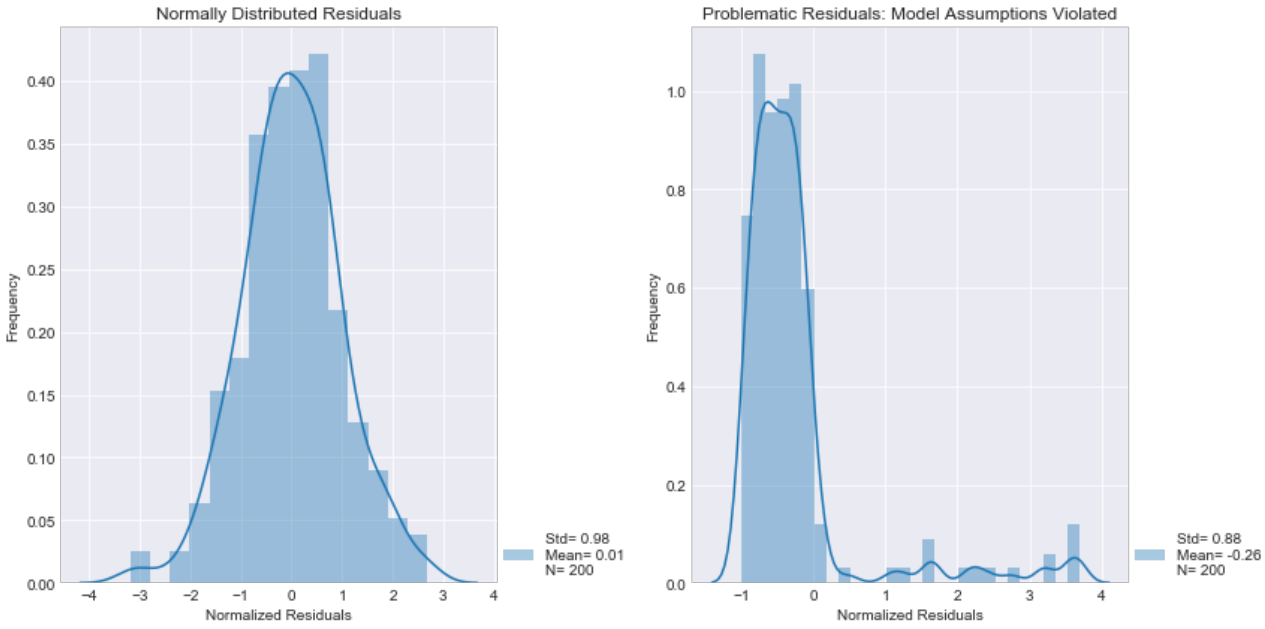
> The normality assumption states that the **model residuals** should follow a normal distribution

Note that the normality assumption talks about the **model residuals** and *not* about the distributions of the **variables**! In general, data scientists will often check the distributions of the variables as well. Keep in mind that the normality assumption is mandatory for the residuals, and it is useful to check normality of your variables to check for weirdness (more on data distributions later), but OLS works fine for non-normal data distributions in the context of prediction.

The easiest way to check for the normality assumption is with histograms or a Q-Q-Plots.

Histograms

We have already seen quite a few histograms and also know how to build them. You can use histograms to check the errors generated by the model and see if the plot shows a so-called "normal distribution" (bell curve shape). As the error term follows a normal distribution, we can develop better confidence in the results and calculate the statistical significance. An example of a regression error histogram is shown below:

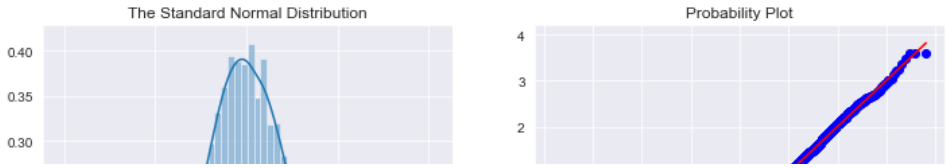


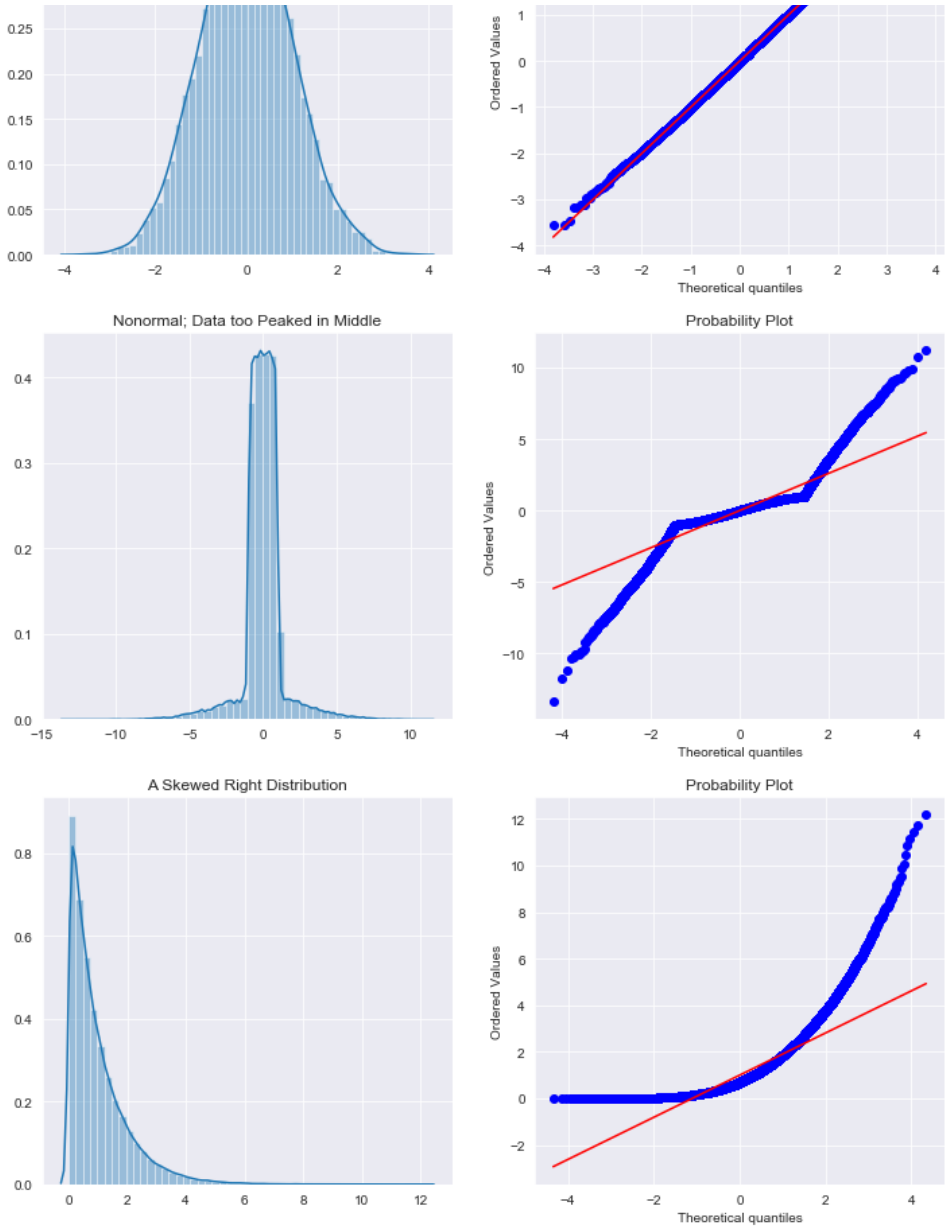
Q-Q Plots

> In statistics, a Q–Q (quantile-quantile) plot is a probability plot, which is a graphical method for comparing two probability distributions by plotting their quantiles against each other.

The Q-Q plot (quantile-quantile plot) is used to help assess if a sample comes from a known distribution such as a normal distribution. For regression, when checking if the data in this sample is normally distributed, we can use a Normal Q-Q plot to test that assumption. Remember that this is just a visual check, so the interpretation remains subjective. However, it is a good first check to see the overall shape of your data against the required distribution. If you can reject normality through Q-Q plots, you have saved yourself from a lot of statistical testing. You have to be careful, however, when deciding that data is totally normal just by looking at a Q-Q plot.

Below, you can find a few examples of comparing histograms and corresponding plots. You can see how the quantiles of normal data appear as a straight line along the diagonal when plotted against a standard normal distribution's quantiles. The skewness and kurtosis of data can also be inspected this way





In the context of normality of residuals, Q-Q plots can help you validate the assumption of normally distributed residuals. It uses standardized values of residuals to determine the normal distribution of errors. Ideally, this plot should show a straight line. A curved, distorted line suggests residuals have a non-normal distribution.Here is a good article explaining the interpretation of Q-Q plots in detail.

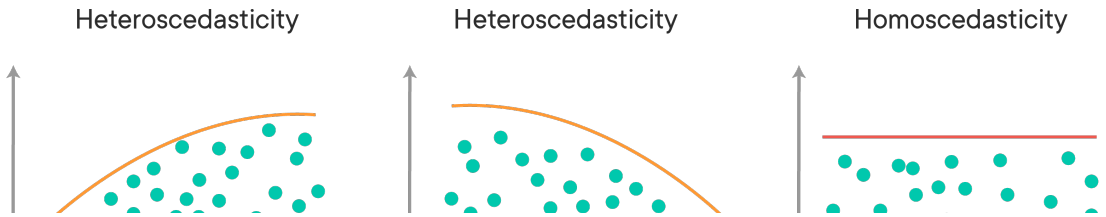
Normality can also be checked with goodness of fit tests such as the Kolmogorov-Smirnov test. When the data is not normally distributed, there are some ways to fix that, such as a non-linear transformation (e.g., log-transformation).

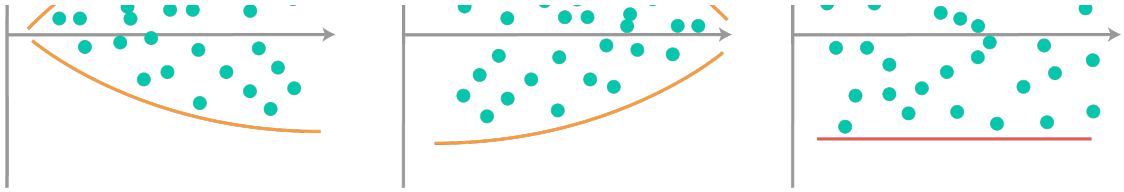
3. Homoscedasticity

> *Heteroscedasticity* (also spelled heteroskedasticity) refers to the circumstance in which the dependent variable is unequal across the range of values of the predictor(s).

When there is heteroscedasticity in the data, a scatterplot of these variables will often create a cone-like shape. The scatter of the dependent variable widens or narrows as the value of the independent variable increases.

The inverse of heteroscedasticity is *homoscedasticity*, which indicates that a dependent variable's variability is equal across values of the independent variable. **Homoscedasticity is the third assumption necessary when creating a linear regression model.**



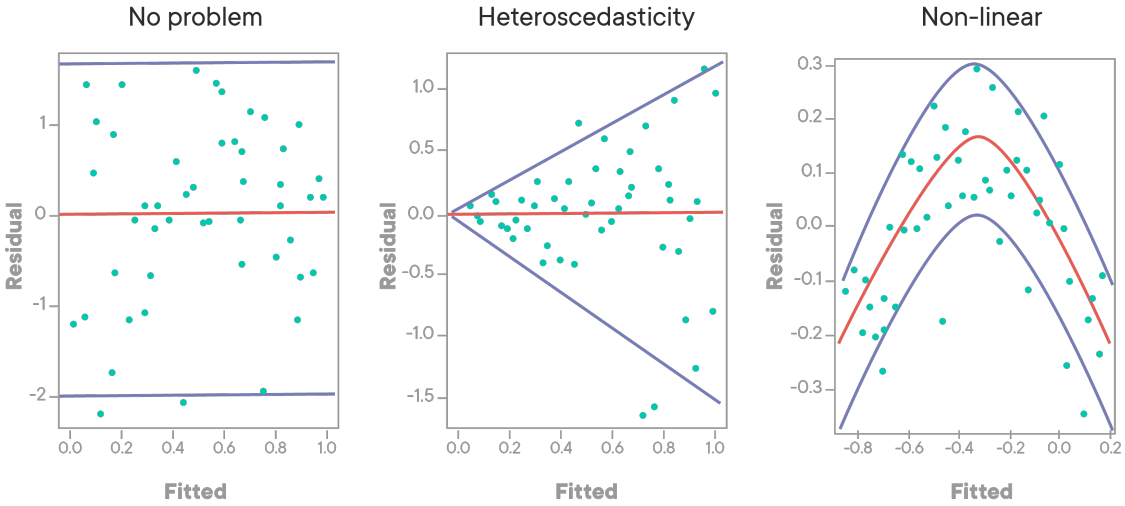


A scatter plot is good way to check whether the data are homoscedastic (meaning the residuals are equal across the regression line). The scatter plots shown here are examples of data that are heteroscedastic (except the plot far right). You can also use significance tests like Breusch-Pagan / Cook-Weisberg test or White general test to detect this phenomenon. You will learn about p-values later, but for now, you can remember that, if these tests give you a p-value < 0.05, the null hypothesis can rejected, and you can assume the data is heteroscedastic.

What Else?

There are other assumptions for linear regression that apply to more complicated cases, but for now these three assumptions are sufficient.

As a first check, always looks at plots of the residuals. If you see anything similar to what is shown below, you are violating one or more assumptions and the results will not be reliable.



Summary

In this lesson, you learned about some assumptions for a simple linear regression that must be held in order to interpret the results reliably. As mentioned earlier, once these assumptions are confirmed, you can run your regression model. Next, you'll be exposed to some examples!

Ordinary Least Squared (OLS) in Statsmodels

Introduction

So far, you learned how to create code for running linear regression experiments along with checking their goodness of fit. Python provides us with many libraries to automate this process and to enhance the efficiency of computation. In this lesson, you'll be introduced to the `statsmodels` library to run OLS regression experiments.

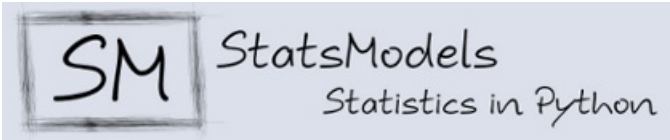
Objectives

You will be able to:

- Perform a linear regression using statsmodels
- Evaluate a linear regression model by using statistical performance metrics pertaining to overall model and specific parameters
- Determine if a particular set of data exhibits the assumptions of linear regression

What is Statsmodels?

Statsmodels is a powerful Python package for many types of statistical analyses. If you installed Python via Anaconda, then the module was installed at the same time. In statistics, ordinary least square (OLS) regression is a method for estimating the unknown parameters in a linear regression model. It minimizes the sum of squared vertical distances between the observed values and the values predicted by the linear approximation. The OLS method in statsmodels is widely used for regression experiments in all fields of study.



For simple linear regression, Statsmodels builds a regression model where y is a $(n * 1)$-vector and x is a $(n * 1)$-vector. The method returns a vector of size n, where n is the number of observations.

Importing Necessary Libraries

The next code cell shows you how to import statsmodels OLS method into your working Python environment. You'll also import Pandas for data handling and Matplotlib for visualizations.

In [1]:

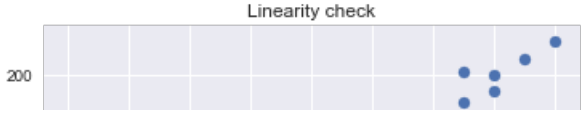
```
import pandas as pd
import statsmodels.api as sm
from statsmodels.formula.api import ols
import matplotlib.pyplot as plt
plt.style.use('seaborn')
```

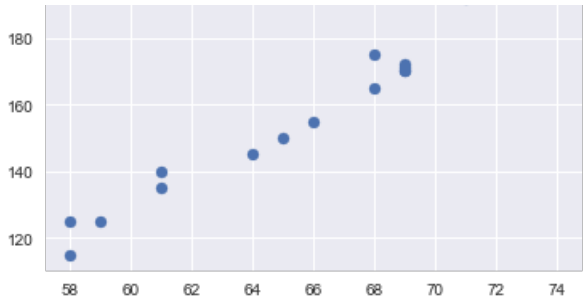
Load the data - Initial Checks

Let's load a simple dataset for the purpose of understanding the process first. You can use the weight-height dataset used before. Let's try to identify the relationship between height as independent and weight and dependent variables. You will also use Pandas visualizations to check for your linearity assumption.

In [2]:

```
df = pd.read_csv('heightWeight.csv')
plt.scatter(df.height, df.weight)
plt.title("Linearity check")
plt.show()
```





Next, let's look at the distributions of the dependent and independent variables.

NOTE: Observing normality here does *NOT* necessarily mean your normality assumption is fulfilled! You just generally want to check out the distribution of your variables before starting to build a model. You'll verify the normality assumption later by checking the distribution of the residuals **after** building the model.

In [3]:

```
df.plot.kde()
plt.title("distribution check for dependent and independent variable")
plt.show()
```

Regression formula

Looks like you're good for the linearity assumption, and additionally, the distributions for height and weight look reasonable (and even pretty normal!). Now, let's run the regression. Statsmodels allows users to fit statistical models using R-style **formulas**. The formula framework is quite powerful and for simple regression it is written using a **~** as `Y ~ X`.

The formula gives instruction for a general structure for a regression call. For a statsmodels ols calls, you'll need a Pandas dataframe with column names that you will add to your formula.

In [4]:

```
f = 'weight~height'
```

You can now pass the formula with variable names to `ols` along with `fit()` to fit a linear model to given variables.

In [5]:

```
model = ols(formula=f, data=df).fit()
```

Great, that was fast (remember we have only 20 observations). Now, you can go ahead and inspect your fitted model in many ways. First, let's get a summary of what the model contains using `model.summary()`

In [6]:

```
model.summary()
```

Out[6]:

OLS Regression Results

Dep. Variable:	weight	**R-squared:**	0.955
Model:	OLS	**Adj. R-squared:**	0.953

Model:	OLS	**Adj. R-squared:**	0.953
Method:	Least Squares	**F-statistic:**	384.8
Date:	Fri, 10 Apr 2020	**Prob (F-statistic):**	1.35e-13
Time:	17:22:00	**Log-Likelihood:**	-64.112
No. Observations:	20	**AIC:**	132.2
Df Residuals:	18	**BIC:**	134.2
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	-204.4834	18.929	-10.802	0.000	-244.252	-164.714
height	5.5390	0.282	19.616	0.000	4.946	6.132

Omnibus:	2.588	**Durbin-Watson:**	2.053
Prob(Omnibus):	0.274	**Jarque-Bera (JB):**	1.245
Skew:	0.202	**Prob(JB):**	0.537
Kurtosis:	1.846	**Cond. No.**	902.

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Wow, that's a lot of information. Statsmodels performs a ton of tests and calculates measures to identify goodness of fit.

- You can find the R-Squared, which is 0.95 i.e. the data are very linearly related
- You can also look at the coefficients of the model for intercept and slope (next to "height")
- Kurtosis and Skew values are shown here
- A lot of significance testing is being done here

Here is a brief description of these measures:

The left part of the first table gives some specifics on the data and the model:

- **Dep. Variable**: Singular. Which variable is the point of interest of the model
- **Model**: Technique used, an abbreviated version of Method (see methods for more).
- **Method**: The loss function optimized in the parameter selection process. Least Squares since it picks the parameters that reduce the training error. This is also known as Mean Square Error [MSE].
- **No. Observations**: The number of observations used by the model, or size of the training data.
- **Degrees of Freedom Residuals**: Degrees of freedom of the residuals, which is the number of observations – number of parameters. Intercept is a parameter. The purpose of Degrees of Freedom is to reflect the impact of descriptive/summarizing statistics in the model, which in regression is the coefficient. Since the observations must "live up" to these parameters, they only have so many free observations, and the rest must be reserved to "live up" to the parameters' prophecy. This internal mechanism ensures that there are enough observations to match the parameters.
- **Degrees of Freedom Model**: The number of parameters in the model (not including the constant/intercept term if present)
- **Covariance Type**: Robust regression methods are designed to be not overly affected by violations of assumptions by the underlying data-generating process. Since this model is Ordinary Least Squares, it is non-robust and therefore highly sensitive to outliers.

The right part of the first table shows the goodness of fit:

- **R-squared**: The coefficient of determination, the Sum Squares of Regression divided by Total Sum Squares. This translates to the percent of variance explained by the model. The remaining percentage represents the variance explained by error, the E term, the part that model and predictors fail to grasp.
- **Adj. R-squared**: Version of the R-Squared that penalizes additional independent variables.
- **F-statistic**: A measure of how significant the fit is. The mean squared error of the model divided by the mean squared error of the residuals. Feeds into the calculation of the P-Value.
- **Prob (F-statistic) or P-Value**: The probability that a sample like this would yield the above statistic, and whether the model's verdict on the null hypothesis will consistently represent the population. Does not measure effect magnitude, instead measures the integrity and consistency of this test on this group of data.
- **Log-likelihood**: The log of the likelihood function.
- **AIC**: The Akaike Information Criterion. Adjusts the log-likelihood based on the number of observations and the complexity of the model. Penalizes the model selection metrics when more independent variables are added.
- **BIC**: The Bayesian Information Criterion. Similar to the AIC, but has a higher penalty for models with more parameters.

Penalizes the model selection metrics when more independent variables are added.

The second table shows the coefficient report:

- **coef**: The estimated value of the coefficient. By how much the model multiplies the independent value by.
- **std err**: The basic standard error of the estimate of the coefficient. Average distance deviation of the points from the model, which offers a unit relevant way to gauge model accuracy.
- **t**: The t-statistic value. This is a measure of how statistically significant the coefficient is.
- **P > |t|**: P-value that the null-hypothesis that the coefficient = 0 is true. If it is less than the confidence level, often 0.05, it indicates that there is a statistically significant relationship between the term and the response.
- **[95.0% Conf. Interval]**: The lower and upper values of the 95% confidence interval. Specific range of the possible coefficient values.

The third table shows information about the residuals, autocorrelation, and multicollinearity:

- **Skewness**: A measure of the symmetry of the data about the mean. Normally-distributed errors should be symmetrically distributed about the mean (equal amounts above and below the line). The normal distribution has 0 skew.
- **Kurtosis**: A measure of the shape of the distribution. Compares the amount of data close to the mean with those far away from the mean (in the tails), so model "peakiness". The normal distribution has a Kurtosis of 3, and the greater the number, the more the curve peaks.
- **Omnibus D'Angostino's test**: Provides a combined statistical test for the presence of skewness and kurtosis.
- **Prob(Omnibus)**: The above statistic turned into a probability
- **Jarque-Bera**: A different test of the skewness and kurtosis
- **Prob (JB)**: The above statistic turned into a probability
- **Durbin-Watson**: A test for the presence of autocorrelation (that the errors are not independent), which is often important in time-series analysis
- **Cond. No**: A test for multicollinearity (if in a fit with multiple parameters, the parameters are related to each other).

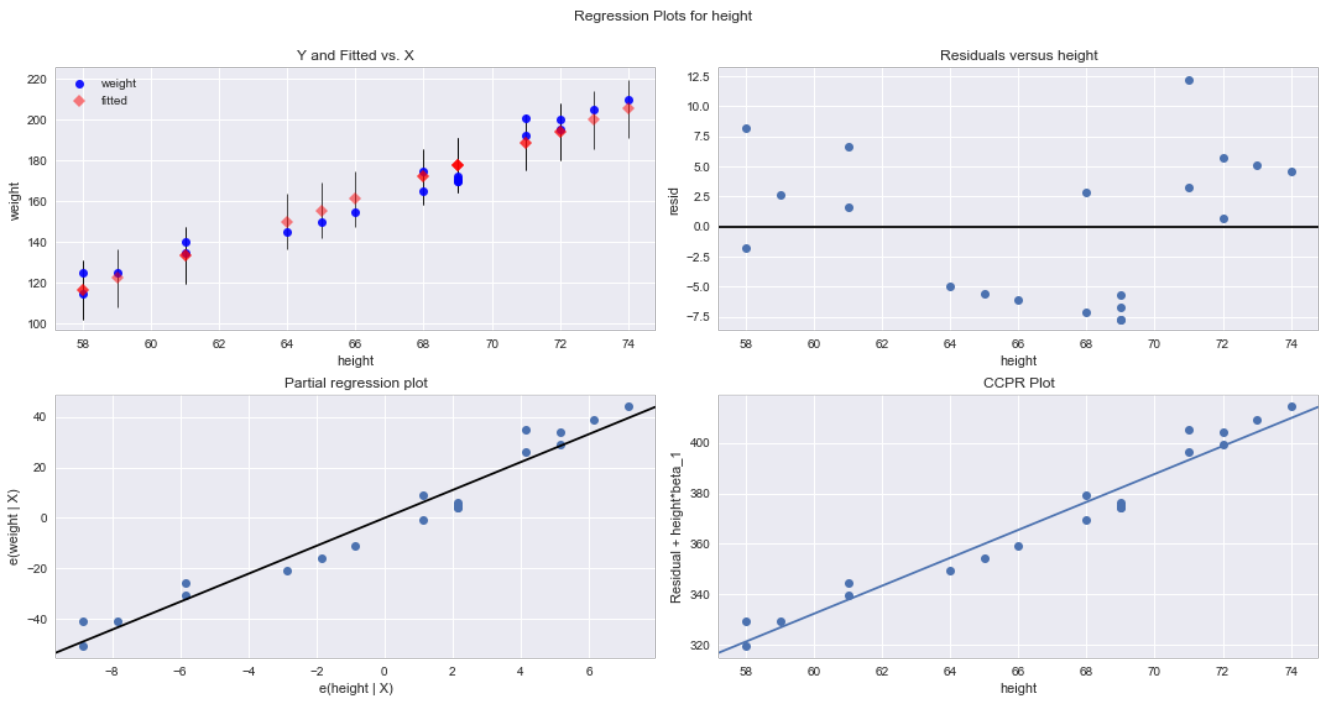
The interpretation of some of these measures will be explained in the next lessons. For others, you'll get a better insight into them in the lessons on statistics.

Visualize error terms

You can also plot some visualizations to check the regression assumptions with respect to the error terms. You'll use `sm.graphics.plot_regress_exog()` for some built-in visualization capabilities of statsmodels. Here is how to do it:

In [7]:

```
fig = plt.figure(figsize=(15,8))
fig = sm.graphics.plot_regress_exog(model, "height", fig=fig)
plt.show()
```



For the four graphs we see above:

- The **Y and Fitted vs. X** graph plots the dependent variable against our predicted values with a confidence interval. The positive relationship shown that height and weight are correlated, i.e., when one variable increases the other increases

positive relationship shows that height and weight are correlated, i.e., when one variable increases the other increases.

- The **Residuals versus height** graph shows our model's errors versus the specified predictor variable. Each dot is an observed value; the line represents the mean of those observed values. Since there's no pattern in the distance between the dots and the mean value, the OLS assumption of homoskedasticity holds.
- The **Partial regression plot** shows the relationship between height and weight, taking in to account the impact of adding other independent variables on our existing height coefficient. You'll later learn how this same graph changes when you add more variables.
- The **Component and Component Plus Residual (CCPR)** plot is an extension of the partial regression plot. It shows where the trend line would lie after adding the impact of adding our other independent variables on the weight.

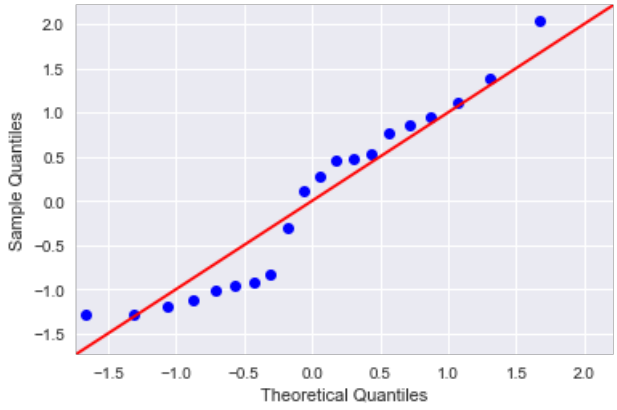
Q-Q Plots

To check for the normality assumption, you can obtain error terms (residuals) from the model and draw Q-Q Plot against a standard normal distribution as shown below. While the residuals do not seem to match up perfectly with the red line, there seem to be no super clear deviations from the red line. So you can assume that you're OK for the normality assumption.

In [8]:

```
import scipy.stats as stats
residuals = model.resid
fig = sm.graphics.qqplot(residuals, dist=stats.norm, line='45', fit=True)
fig.show()
```

```
C:\Anaconda3\envs\learn-env\lib\site-packages\ipykernel_launcher.py:4: UserWarning: Matplotlib is curre
ntly using module://ipykernel.pylab.backend_inline, which is a non-GUI backend, so cannot show the figu
re.
  after removing the cwd from sys.path.
```



So you now know how to run an OLS simple regression experiment in Statsmodels. In the next lesson, you'll learn how to interpret the diagnostics better and how to relate the outcome to the research question.

Summary

In this lesson, you learned how to run a simple regression experiment in Statsmodels. You learned about the format and conventions for running such an experiment. You also looked at regression diagnostics and how to interpret them, plus some visualizations to check for your regression assumptions.

Regression Diagnostics in Statsmodels

Introduction

So far, you have looked mainly at R-Squared values along with some visualization techniques to confirm that regression assumptions are met. Now, you'll look at some statistical procedures to further understand your model and results. You'll be looking at the results obtained in the regression analysis outcomes for the advertising dataset in the previous lab.

Note: Some of the terms in this lesson highlighting underlying statistical testing concepts will be new to you. These terms will be covered in detail in later sections. Here, the focus will be on running and interpreting the results of these tests in a regression context.

Objectives

You will be able to:

- Determine if a particular set of data exhibits the assumptions of linear regression

Let's get started

Regression diagnostics is a set of procedures available for regression analysis that assess the validity of a model in a number of different ways.

This could be:

- An exploration of the model's underlying statistical assumptions
- An examination of the structure of the model by considering formulations that have less, more or different explanatory variables
- A study of subgroups of observations, looking for those that are either poorly represented by the model (outliers) or that have a relatively large effect on the regression model's predictions

For a thorough overview, you can go and have a look at the wikipedia page on regression diagnostics.

Here we'll revisit some of the methods you've already seen, along with some new tests and how to interpret them.

Normality Check (Q-Q plots)

You've already seen Q-Q Plots as a measure to check for normality (or, by extension, any other distribution).

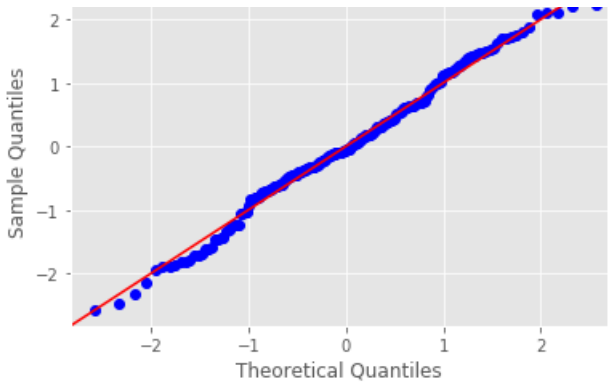
Q-Q plots are also referred to as normal density plots when used with standard normal quantiles. These plots are a good way to inspect the distribution of model errors. You saw this earlier with the small height-weight data set. Let's quickly generate a Q-Q plot for the residuals in the `sales ~ TV` and the `sales ~ radio` models again!

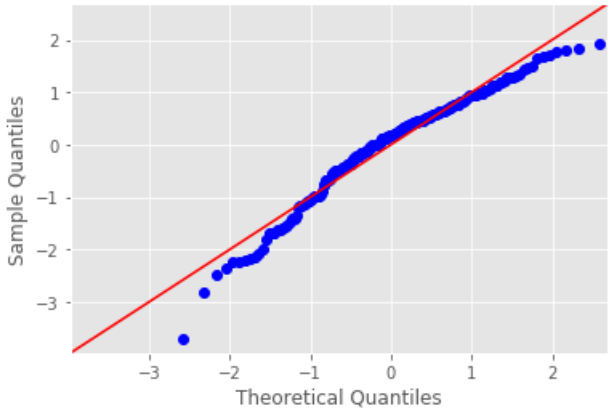
In [1]:

```
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import statsmodels.api as sm
import statsmodels.stats.api as sms
import statsmodels.formula.api as smf
import scipy.stats as stats
plt.style.use('ggplot')

data = pd.read_csv('advertising.csv', index_col=0)
f = 'sales~TV'
f2 = 'sales~radio'
model = smf.ols(formula=f, data=data).fit()
model2 = smf.ols(formula=f2, data=data).fit()

resid1 = model.resid
resid2 = model2.resid
fig = sm.graphics.qqplot(resid1, dist=stats.norm, line='45', fit=True)
fig = sm.graphics.qqplot(resid2, dist=stats.norm, line='45', fit=True)
```



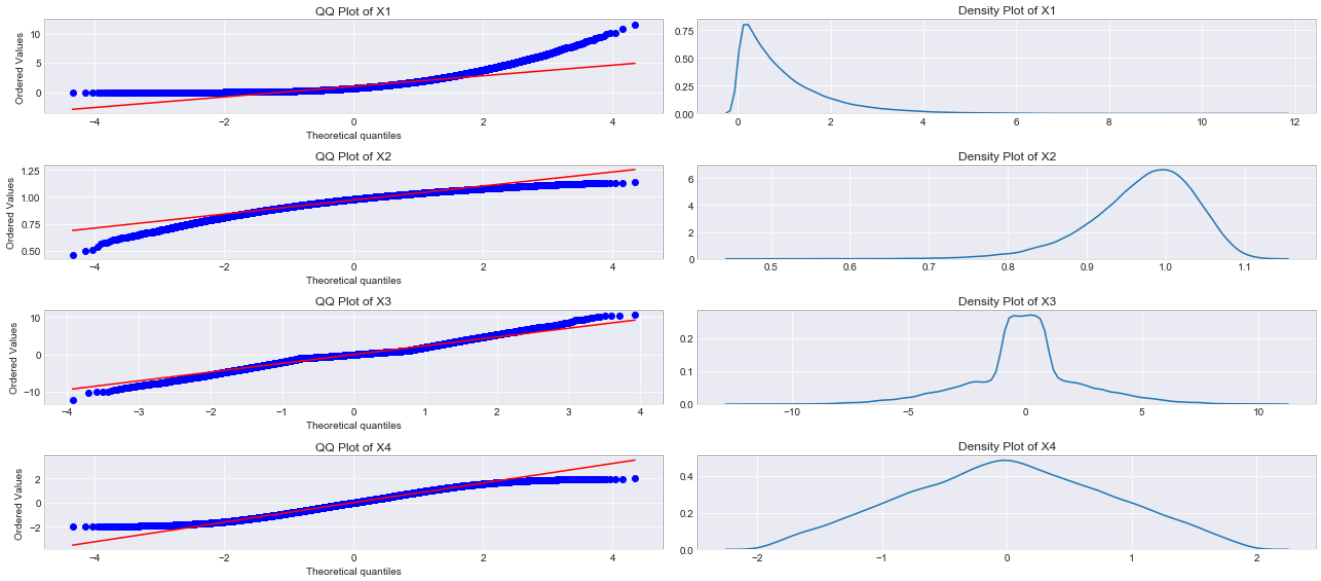


Normal Q-Q Plots are a direct visual assessment of how well our residuals match what we would expect from a normal distribution.

In the Q-Q plots above, you can see that residuals are better normally distributed for TV than for radio.

You can also spot an outlier in the left tail of radio residuals. Dealing with this might help improve the fit of the model. Outliers, skew, heavy and light-tailed aspects of distributions (all violations of normality) can be assessed from Q-Q plots. It might require a bit of practice before you can truly start to interpret them.

The images below show you how to relate a histogram to their respective Q-Q Plots.



Normality Check (Jarque-Bera Test)

The Jarque-Bera (JB) test is a test for normality. This test is usually used for large data sets, because other tests like Q-Q Plots can become unreliable when your sample size is large.

> The Jarque-Bera test inspects the skewness and kurtosis of data to see if it matches a normal distribution. It is a common method for inspecting errors distribution in regression as shown below.

$$\frac{S^2}{(\quad} \quad \frac{(K-3)^2}{24}$$

$$JB = n * \left(\frac{6}{} + \frac{}{24} \right)$$

Here, n is the sample size, S is the sample skewness coefficient and K is the sample kurtosis.

Here is how you use JB in statsmodels. A JB value of roughly 6 or higher indicates that errors are not normally distributed. In other words, this means that the normality null hypothesis has been rejected at the 5% significance level. A value close to 0 on the contrary, indicates the data is normally distributed. We have already seen the JB test using `model.summary()` . The code below shows you how to run this test on its own.

In [2]:

```
# JB test for TV
name = ['Jarque-Bera','Prob','Skew', 'Kurtosis']
test = sms.jarque_bera(model.resid)
list(zip(name, test))
```

Out[2]:

```
[('Jarque-Bera', 0.6688077048615518),
 ('Prob', 0.715764660551865),
 ('Skew', -0.08863202396577118),
 ('Kurtosis', 2.7790149735970555)]
```

We have a JB value = 0.67, which is pretty low (and in favor of normality), and the p-value of 0.71 is quite high to reject the null hypothesis for normality. Additionally, the kurtosis is below 3, where a kurtosis higher than 3 indicates heavier tails than a normal distribution. The skewness values however show that underlying data is moderately skewed. Let's see what happens if we look at the `radio` residuals.

In [3]:

```
# JB test for radio
name = ['Jarque-Bera','Prob','Skew', 'Kurtosis']
test2 = sms.jarque_bera(model2.resid)
list(zip(name, test2))
```

Out[3]:

```
[('Jarque-Bera', 21.90969546280269),
 ('Prob', 1.74731047370758e-05),
 ('Skew', -0.7636952540480038),
 ('Kurtosis', 3.5442808937621666)]
```

Where The TV residuals showed to be close to normality, the JB results for radio are considerably worse. More-over, a JB p-value much smaller than 0.05 indicates that the normality assumption should definitely be rejected.

These results show that even when in the Q-Q plots the results seemed moderately different, the JB test could shed new light on the normality assumption.

Checking Heteroscadasticity (Goldfeld-Quandt test)

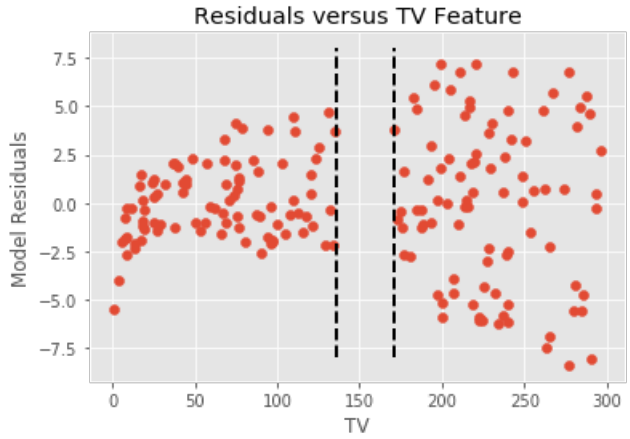
The Goldfeld Quandt (GQ) test is used in regression analysis to check for homoscedasticity in the error terms. The GQ test checks if you can define a point that can be used to **differentiate** the variance of the error term. It is a parametric test and uses the assumption that the data is normally distributed. So it is general practice to check for normality before going over to the GQ test!

In the image below, you can see how observations are split into two groups. Next, a test statistic is run through taking the ratio of mean square residual errors for the regressions on the two subsets. Evidence of heteroscedasticity is based on performing a hypothesis test (more on this later) as shown in the image.

In [4]:

```
lwr_thresh = data.TV.quantile(q=.45)
upr_thresh = data.TV.quantile(q=.55)
middle_10percent_indices = data[(data.TV >= lwr_thresh) & (data.TV<=upr_thresh)].index
# len(middle_10percent_indices)

indices = [x-1 for x in data.index if x not in middle_10percent_indices]
plt.scatter(data.TV.iloc[indices], model.resid.iloc[indices])
plt.xlabel('TV')
plt.ylabel('Model Residuals')
plt.title("Residuals versus TV Feature")
plt.vlines(lwr_thresh, ymax=8, ymin=-8, linestyles='dashed',linewidth=2)
plt.vlines(upr_thresh, ymax=8, ymin=-8, linestyles='dashed',linewidth=2);
```



Here is a brief description of the steps involved:

- Order the data in ascending order
- Split your data into *three* parts and drop values in the middle part.
- Run separate regression analyses on two parts. After each regression, find the Residual Sum of Squares.
- Calculate the ratio of the Residual sum of squares of two parts.
- Apply the F-test.

(F-test will be covered later in the syllabus. Here is a quick introduction)

For now, you should just remember that high F values typically indicate that the variances are different. If the error term is homoscedastic, there should be no systematic difference between residuals and F values will be small. However, if the standard deviation of the distribution of the error term is proportional to the x variable, one part will generate a higher sum of square values than the other.

Here is how you can run this test in statsmodels.

In [5]:

```
# Run Goldfeld Quandt test
name = ['F statistic', 'p-value']
test = sms.het_goldfeldquandt(model.resid.iloc[indices], model.model.exog[indices])
list(zip(name, test))
```

Out[5]:

```
[('F statistic', 1.1993147096678916), ('p-value', 0.19780602597731686)]
```

In [6]:

```
# Run Goldfeld Quandt test
import statsmodels.stats.api as sms
name = ['F statistic', 'p-value']
test = sms.het_goldfeldquandt(model2.resid.iloc[indices], model2.model.exog[indices])
list(zip(name, test))
```

Out[6]:

```
[('F statistic', 1.2189878283402957), ('p-value', 0.1773756718718901)]
```

The null hypothesis for the GQ test is homoscedasticity. The larger the F-statistic, the more evidence we will have against the homoscedasticity assumption and the more likely we have heteroscedasticity (different variance for the two groups).

The p-value for our tests above tells us whether or not to reject the null-hypothesis of homoscedasticity. Taking a confidence level of alpha = 0.05, we cannot reject the null hypothesis because for both TV and radio, p-values are larger than 0.05. So even though we visually inspected some heteroscedasticity previously, this cannot be confirmed by the GQ test.

Statsmodel also offers newer tests for heteroscadasticity including the Breush-Pagan Test and White's Test which may be advantageous in certain cases.

Summary

In this lesson, you learned a few methods to check for regression assumptions in addition to the visual methods learned earlier. An understanding and hands-on experience with visual as well as statistical techniques to check your regression analysis will provide you with a good set of tools to run more detailed regression experiments later.

you with a good set of tools to run more detailed regression experiments later.

Interpreting Significance and P-values

Introduction

In this lesson, you'll learn more about some of the ideas mentioned in earlier lessons around interpreting the significance of results. You'll learn that associating a confidence level to our model's output is very important during decision making and an essential part of data science.

Objectives

You will be able to:

- Evaluate a linear regression model by using statistical performance metrics pertaining to overall model and specific parameters

Let's get started

The ideas of hypothesis testing can be applied to regression, as well as statistical inference. For now, You'll learn about how it works in regression, and we'll spend time on formal hypothesis testing in a variety of contexts later on.

When performing hypothesis tests, a set of hypotheses are developed including a so-called **null hypothesis** and an **alternative hypothesis**. In regression, you will generally try to reject the null hypothesis, because that means that there is some relationship between the dependent and independent variable and your model at hand makes sense. Rejecting or not rejecting a null hypothesis always goes hand in hand with a so-called **confidence level**, often defined by the Greek letter alpha, α.

Hypothesis Testing in Regression

During regression, you try to measure the model parameters (coefficients). The null and alternative hypotheses are also set up in those terms. Think about the simple regression model you created using the advertising dataset. For a simple dataset like this, you can set up the hypotheses as follows:

> **Null Hypothesis (H_0)**: There is no relationship between the amount spent on TV advertisement and sales figures
>
> **Alternative Hypothesis (H_a):** There is "some" relation between the amount spent on TV advertisement and sales figures

If we reject the null hypothesis, it means that we believe that there is some sort of relationship between TV advertisement and sales. If we fail to reject the null hypothesis, it means that we believe that there is no *significant* relationship between TV advertisement and sales, and that our slope parameter m in $y = mx + c$ is not *significantly* different from zero.

You see that we used the word *significant* and *significantly*. What does this mean, and when is something *significant*?

P-value as a level of statistical significance

When performing statistical analyses, rejecting or not rejecting a null hypothesis always goes along with an associated **significance level** or **p-value**.

> The p-value represents a **probability of observing your results (or something more extreme) given that the null hypothesis is true**

Applied to a regression model, p-values associated with coefficients estimates indicate the probability of observing the associated coefficient given that the null-hypothesis is true. As a result, very small p-values indicate that coefficients are **statistically significant**. A very commonly used cut-off value for the p-value is 0.05. If your p-value is smaller than 0.05, you would say:

> The parameter is statistically significant at α level 0.05.

Just like for statistical significance, rejecting the null hypothesis at an alpha level of 0.05 is the equivalent for having a 95% confidence interval around the coefficient that does not include zero. In short

> The p-value represents the probability that the coefficient is actually zero.

Let's import the code from the previous lesson and let's have a look at the p-value.

In [1]:

```
import pandas as pd
import matplotlib.pyplot as plt
import statsmodels.api as sm
import statsmodels.formula.api as smf
import scipy.stats as stats
plt.style.use('fivethirtyeight')

data = pd.read_csv('Advertising.csv', index_col=0)
f = 'sales~TV'
f2 = 'sales~radio'
model = smf.ols(formula=f, data=data).fit()
```

Now, we can check for the p-value associated with our coefficient for TV:

In [2]:

```
model.pvalues
```

Out[2]:

```
Intercept    1.406300e-35
TV           1.467390e-42
dtype: float64
```

You'll notice that you also get a p-value for the intercept. Except if the intercept is actually 0, you should get a pretty low p-value here anytime. The tiny p-value for TV indicates that our coefficient for TV is definitely significant, and we can reject the null hypothesis. Next, there are two final things from the entire model summary we'd like to highlight:

In [3]:

```
model.summary()
```

Out[3]:

OLS Regression Results

Dep. Variable:	sales	**R-squared:**	0.612
Model:	OLS	**Adj. R-squared:**	0.610
Method:	Least Squares	**F-statistic:**	312.1
Date:	Fri, 10 Apr 2020	**Prob (F-statistic):**	1.47e-42
Time:	17:22:16	**Log-Likelihood:**	-519.05
No. Observations:	200	**AIC:**	1042.
Df Residuals:	198	**BIC:**	1049.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	7.0326	0.458	15.360	0.000	6.130	7.935
TV	0.0475	0.003	17.668	0.000	0.042	0.053

Omnibus:	0.531	**Durbin-Watson:**	1.935
Prob(Omnibus):	0.767	**Jarque-Bera (JB):**	0.669
Skew:	-0.089	**Prob(JB):**	0.716
Kurtosis:	2.779	**Cond. No.**	338.

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

1. Have a look at the probability for the F-statistic in the upper right plane. This is a p-value for the **overall model fit**. You can see how this probability value is the exact same as the p-value for TV. You'll see that this is always the case if there is only one

independent variable. Here, TV drives the entire model, so the model p-value is the same as the TV coefficient p-value.
2. In the plane with the coefficient estimates, p-values are in the column `P > |t|` and rounded to 3 digits (hence 0.000 for Intercept and TV). You can find the confidence intervals in the two last columns: The confidence interval for intercept is [6,13, 7.935] meaning that there is a 95% chance that the actual coefficient value is in that range. For TV, there is a 95% chance that the actual coefficient value is in the interval [0.042, 0.053]. Note that 0 is in none of these intervals as expected given the very low p-value.

Note that the alpha level of 0.05 is just a convention. You'll come across alpha levels of 0.1 and 0.001. Note that the confidence intervals change too as alpha levels change (to 90% and 99% confidence intervals, yet the standard output of statsmodels is a 95% confidence interval).

Try running the above and check for p-values for `radio` and see how would interpret the outcome.

Additional Resources

We encourage you to visit the following links to see more examples and explanations around hypothesis testing and p values in regression!

Hypothesis Test for Regression Slope

Regression Continued

Summary

In this lesson, you learned how to apply the ideas of hypothesis testing in regression analysis to associate significance and confidence level with your model. You used this with your previous regression model to check the association. In the next lab, you'll combine all the ideas around simple linear regression with OLS on a slightly more complex dataset with more features.

Introduction to Linear Regression - Recap

Introduction

This short lesson summarizes the topics we covered in this section and why they'll be important to you as a data scientist.

Key Takeaways

In this section, the nominal focus was on how to perform a linear regression, but the real value was learning how to think about the application of machine learning models to data sets. Key takeaways include:

- Statistical learning theory deals with the problem of finding a predictive function based on data
- A loss function calculates how well a given model represents the relationship between data values
- A linear regression is simply a (straight) line of best fit for predicting a continuous value ($y = mx + c$)
- The Coefficient of Determination (R Squared) can be used to determine how well a given line fits a given data set
- Certain assumptions must hold true for a least squares linear regression to be useful - linearity, normality and heteroscedasticity
- Q-Q plots can check for normality in residual errors
- The Jarque-Bera test can be used to test for normality - especially when the number of data points is large
- The Goldfeld-Quant test can be used to check for homoscedasticity