Linear and nonlinear regression

Using Python to Differentiate Linear and Non-Linear Regression

Aniket Adhikari

2023-11-15

On this page

Linear Regression **2**
- What is Linear Regression . . . 2
- When is Linear Regression Used? . . . 2
- Requirements for Linear Regression . . . 3
- Pros and Cons of Linear Regression . . . 3
  - Pros . . . 3
  - Cons . . . 3
- Simple Linear Regression . . . 4

Non-linear Regression **7**
- What is Non-Linear Regression . . . 7
- When is Non-Linear Regression Used? . . . 7
- Requirements for Non-Linear Regression . . . 8
- Pros and Cons of Non-Linear Regression . . . 8
  - Pros . . . 8
  - Cons . . . 8

Regression analysis is a powerful statistical method for estimating the relationship between a dependent and independent variable. Regression analysis leverages historical data to make predictions and comes in 2 forms: linear and non-linear regression.

> **Tip with Title**
>
> an **independent variable** is the value that affects the outcome of a dependent variable and is often referred to as the explanatory variable because it explains the reason for a certain outcome, or **dependent variable**. Dependent variables are usually represented by y and independent varaibles are represented by x.

Linear Regression

What is Linear Regression

Linear regression is the more commonly used regression technique and assumes that there is a linear relationship between the independent variable (x) and dependent variable (y). Linear regression features a *best-fitting line*, which is a straight line that goes through all of the data points and it determined by minimizing the residual sum of squares (RSS) between the predicted and actual values. The general equation for linear regression looks like this:

$Y = mx + b$

- y = dependent variable
- m = slope of the line
- x = independent variable
- b = slope intercept

When is Linear Regression Used?

With the use of linear regression, we have a relatively simple and easy-to-interpret formula for making predictions based on historical data. As a result, it is heavily used in industries such as:

Finance

Using past financial statements, economic indicators, and historical prices to predict:

- Predicting stock prices
- Predicting bond yields
- Predicting future exchange rates

Economics

Using information like policy changes, taxes, and spending to predict:

- GDP
- Inflation
- Unemployment Rates

Sports

Using past team/player performance and rankings to predict:

- Outcome of games
- Make power rankings for teams
- Come up with betting lines

- Future player performance

Transportation

Using weight, distance, route, and fuel costs to predict:

- Transportation costs
- Delivery times
- Gas consumption

Requirements for Linear Regression

- Make sure to use a scatterplot to quickly determine whether there is a linear relationship between 2 variables. Even though there might be a relationship between 2 variables, we have to ensure that the relationship is truly linear. Otherwise, the variables might be better suites for a non-linear regression, which I'll talk about later.
- Remove outliers that may disproportionately influence the regression line
- Independent variables should be scaled/standardized to have a normal distribution
- Dependent variable should be a continuous numeric variable

Pros and Cons of Linear Regression

Pros

1. Simple to implement, easy to interpret
2. Fast to train and predict
3. Less prone to overfitting
4. Provides directly interpretable coefficients

Cons

1. Pretty useless if the relationship is non-linear
2. Bad at learning complex relationships in data
3. Assumes linear relationship even when there isn't one

Simple Linear Regression

Simple linear regression is used to model the linear relationship between a single independent variable and a dependent variable.

We start off by importing the required libraries, including the `LinearRegression` package.

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
import pandas as pd
```

We import a dataset that aggregates a World Happiness Report, which is a survey of the state of global happiness. It contains surveyed information regarding 156 countries around the world and their overall happiness. Additionally, it contains information like`GDP per capita`, `Social support`, `Healthy life expectancy`, `Freedom to make life choices`, `Generosity`, and `Perceptions of corruption`. Information from here can be used to inform policy-making decisions.

In our case, we want to see how much do these listed factors influence overall happiness.

```
data = pd.read_csv("../../datasets/happiness_report.csv")
```

So which one of these factors might be interesting to look like? I thought it might be interesting to look at how `GDP per capita`, `Social support`, `Healthy life expectancy`, and `Freedom to make life choices` potentially influence the `Score`.

As a result, we're going to create DataFrames for each of these columns of data that we want to designate as independent variables:

1. `gdp` represents `GDP per capita`
2. `social_support` represents `Social support`
3. `life_expectancy` represents `Healthy life expectancy`
4. `freedom` represents `Freedom to make life chocies`

We also need to create a DataFrame for the `Score` column so that we can designate it as a dependent variable.

```
# x values
gdp = data['GDP per capita'].values.reshape(-1, 1)
social_support = data['Social support'].values.reshape(-1, 1)
life_expectancy = data['Healthy life expectancy'].values.reshape(-1, 1)
freedom = data['Freedom to make life choices'].values.reshape(-1, 1)
# y values
score = data['Score'].values
```

I wanted to do a linear regression model that examine the effects of GDP per capita on happiness score.

To do this, we have to create a LinearRegression object that we can assign to `gdpModel`. This object can then be used to fit two arrays, `gdp` and `score`. This is done using the `fit()` method, which fits the 2 arrays based on minimizing the sum of squared residuals. From there, we can use the model to make predictions of what the `score` values will be based on `gdp` values. The values are stored into `gdp_predict_score`.

```
gdpModel = LinearRegression()

gdpModel.fit(gdp, score)

gdp_predict_score = gdpModel.predict(gdp)
```

We repeat this process to that examines the effects of

- `Social support`
- `Healthy life expectancy`
- `Freedom to make life choices`

upon `Score`

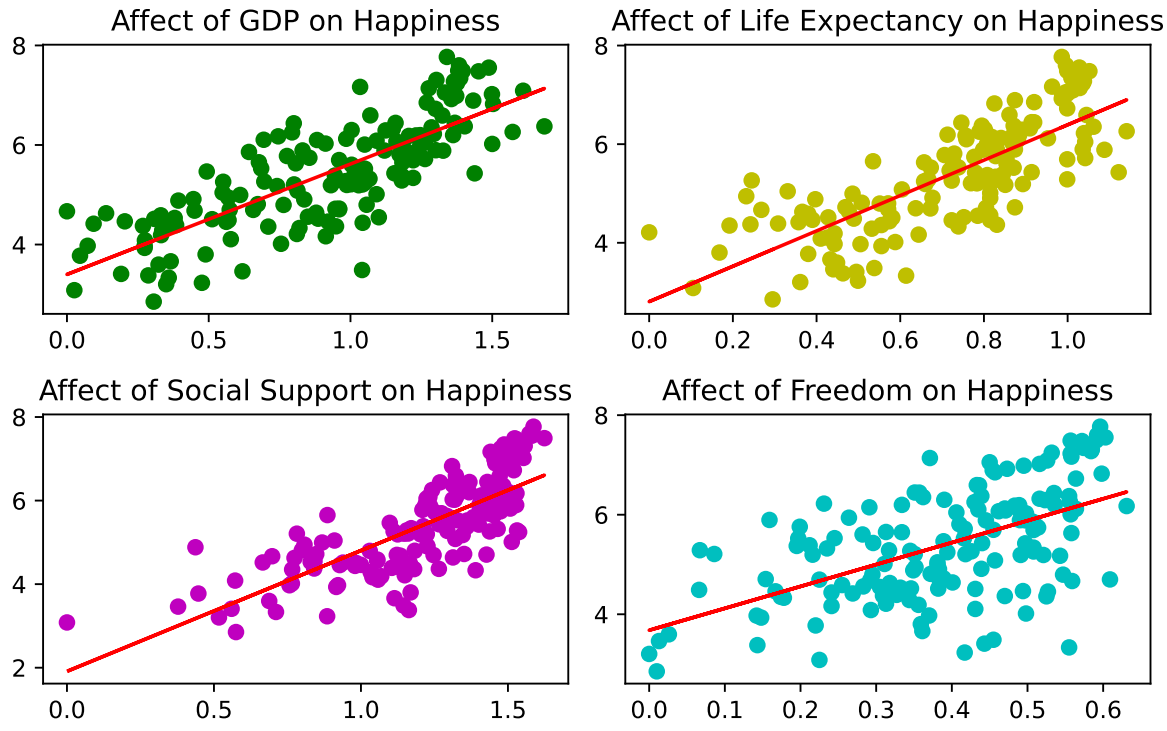
```
socialSupportModel = LinearRegression()

# Train the model
socialSupportModel.fit(social_support, score)

# Make predictions
social_support_predict_score = socialSupportModel.predict(social_support)
lifeExpectancyModel = LinearRegression()

# Train the model
lifeExpectancyModel.fit(life_expectancy, score)

# Make predictions
life_expectancy_predict_score = lifeExpectancyModel.predict(life_expectancy)
freedomModel = LinearRegression()

# Train the model
freedomModel.fit(freedom, score)

# Make predictions
freedom_predict_score = freedomModel.predict(freedom)
```

Now that we have the predictions for each of the independent variables, we can plot the data and run a best-fit line.

In this example, I wanted to create subplots so I could show the relationship between all 4 independent variables with `score` The first subplot is a scatterplot showing the relationship between `GDP per capita` and `Score`. The data is first scattered on the grid and then we run a best-fit line through this data. This line shows predictions of the `score` based on `GDP per capita`, with the assumption that their relationship is linear. We then create subplots for all of the other relationships and label them with the appropriate titles.

All of them seem to have a positive relationship, meaning as the independent value x increases, the dependent values y also increases

```
fig, axs = plt.subplots(2, 2, figsize=(7, 4.5), tight_layout=True)

axs[0, 0].scatter(gdp, score, c="g")
axs[0, 0].plot(gdp, gdp_predict_score, c="r")
axs[0, 0].set_title("Affect of GDP on Happiness")

axs[1, 0].scatter(social_support, score,  c="m")
axs[1, 0].plot(social_support, social_support_predict_score, c="r")
axs[1, 0].set_title("Affect of Social Support on Happiness")

axs[0, 1].scatter(life_expectancy, score, c="y")
axs[0, 1].plot(life_expectancy, life_expectancy_predict_score, c="r")
axs[0, 1].set_title("Affect of Life Expectancy on Happiness")

axs[1, 1].scatter(freedom, score, c="c")
axs[1, 1].plot(freedom, freedom_predict_score, c="red")
axs[1, 1].set_title("Affect of Freedom on Happiness")

plt.show()
```



Non-linear Regression

What is Non-Linear Regression

Non-linear regression is a regression technique that assumes non-linear relationship between one or more independent variables and a dependent variable. This assumption allows for more complex relationships to be modeled that can't be modeled with the straight line in linear regression.

When is Non-Linear Regression Used?

- Modeling growth rates
- Modeling diminishing returns
- Probabilistic outcomes
  - Modeling binary outcomes like pass/fail, win/loss
- Forecasting with trends
  - Time series with trends and seasonal patterns

Requirements for Non-Linear Regression

- Ensure the relationship between the independent and dependent variables is non-linear by examining the way the data is scattered
- Figure out which type of non-linear model we want to use. We have a couple of options like polynomial, logarithmic, exponential, and logistic
- There needs to be a sufficient number of data points to fit the model since non-linear regression is more complex.
- Minimize the number outliers in order to reduce disproportionate influence on models

Pros and Cons of Non-Linear Regression

Pros

1. Captures complex relationships
2. Flexible
3. Improved accuracy

Cons

1. Complex and difficult to implement
2. Risk of overfitting
3. Less generalizability