Linear Regression Model

AMAN KHARWAL / JULY 3, 2020 / MACHINE LEARNING

Let's train and run a Linear regression model to make Predictions, In this article, I will load the data, prepare it, create a scatter plot for visualization, and then train a linear regression model to make a prediction.

I will first create a function that will join our two datasets to be used in training our linear regression model. It's the boring pandas code that will join the life satisfaction data from the OECD with GDP per capita data from the IMF.

You can download both these datasets below:

gdp_per_capita Download

oecd_bli_2015 Download

Now let's, create a function, to move further with training our linear regression model.

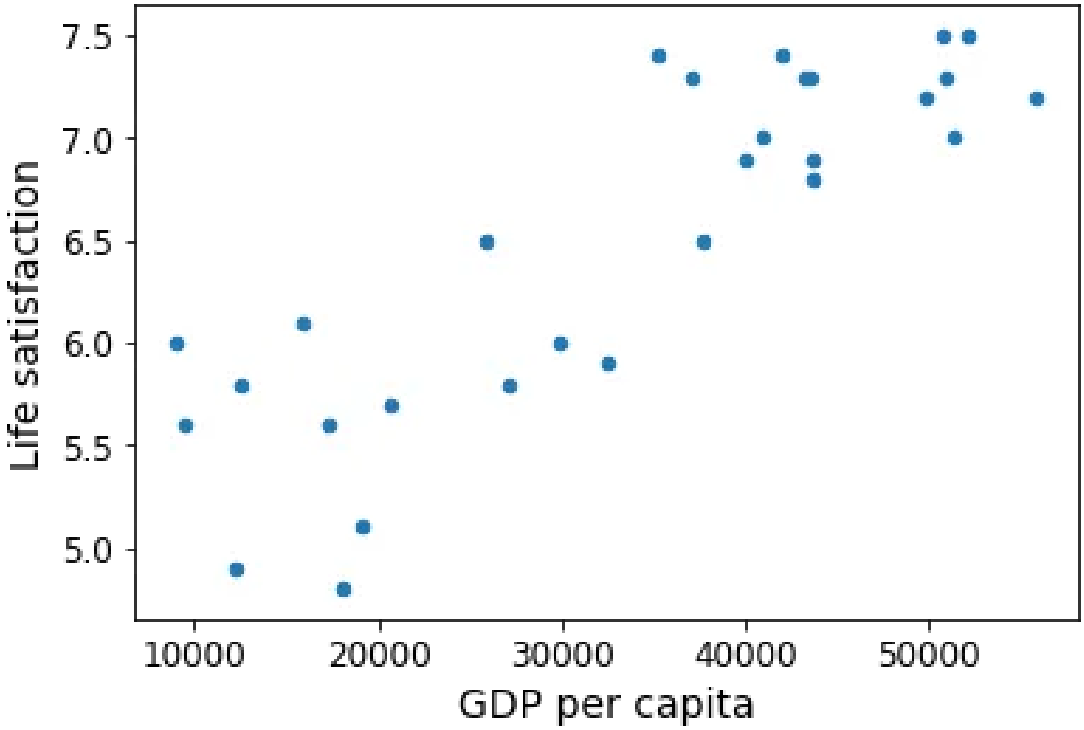
```
# To support both python 2 and python 3
from __future__ import division, print_function, unicode_literals

# Common imports
import numpy as np
import os

# to make this notebook's output stable across runs
np.random.seed(42)

# To plot pretty figures
%matplotlib inline
import matplotlib
import matplotlib.pyplot as plt
plt.rcParams['axes.labelsize'] = 14
plt.rcParams['xtick.labelsize'] = 12
plt.rcParams['ytick.labelsize'] = 12

def prepare_country_stats(oecd_bli, gdp_per_capita):
    oecd_bli = oecd_bli[oecd_bli["INEQUALITY"]=="TOT"]
    oecd_bli = oecd_bli.pivot(index="Country", columns="Indicator",
    gdp_per_capita.rename(columns={"2015": "GDP per capita"}, inplac
    gdp_per_capita.set_index("Country", inplace=True)
    full_country_stats = pd.merge(left=oecd_bli, right=gdp_per_capit
                                  left_index=True, right_index=True)
    full_country_stats.sort_values(by="GDP per capita", inplace=True
    remove_indices = [0, 1, 6, 8, 33, 34, 35]
    keep_indices = list(set(range(36)) - set(remove_indices))
    return full_country_stats[["GDP per capita", 'Life satisfaction'
```

Train and Run a Linear Regression Model

Now you are finally ready for training and running a linear regression model to make predictions. For example, say you want to know how happy Cypriots are, and the OECD data does not have the answer.

Fortunately, you can use your linear regression model to make a good prediction. Let's train a Linear Regression Model:

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

# Load the data
oecd_bli = pd.read_csv("oecd_bli_2015.csv", thousands=',')
gdp_per_capita = pd.read_csv("gdp_per_capita.csv",thousands=',',deli
                             encoding='latin1', na_values="n/a")

# Prepare the data
country_stats = prepare_country_stats(oecd_bli, gdp_per_capita)
X = np.c_[country_stats["GDP per capita"]]
y = np.c_[country_stats["Life satisfaction"]]

# Visualize the data
country_stats.plot(kind='scatter', x="GDP per capita", y='Life satis
plt.show()

# Select a linear model
model = sklearn.linear_model.LinearRegression()

# Train the model
model.fit(X, y)

# Make a prediction for Cyprus
X_new = [[22587]]  # Cyprus' GDP per capita
print(model.predict(X_new)) # outputs [[ 5.96242338]]
```



If you had used an instance-based learning algorithm instead, of a linear regression model, you have found that Slovenia has the closest GDP per capita to that of Cyprus, and since the Linear Regression Model tells you that Slovenians' life satisfaction is 5.7, you would have predicted a life satisfaction pf 5.7 for Cyprus.

If you zoom out a bit and look at the two next-closest countries, you will find Portugal and Spain with life satisfaction of 5.1 and 6.5, respectively.

Averaging these three values, you get 5.77, which is very close to your Linear Model prediction. This simple algorithm is called k-Nearest Neighbors Regression.

Replacing the Linear Regression model with k-Nearest Neighbors regression in the above code is as simple as

replacing these two lines:

```
import sklearn.linear_model
model = sklearn.linear_model.LinearRegression()
```

with these two:

```
import sklearn.neighbors
model = sklearn.neighbors.KNeighborsRegressor(n_neighbors=3)
```

Also, read – 10 Machine Learning Projects to Boost your Portfolio

If all went well, your model will make good predictions. If not, you may need to use more attributes (employment rate, health, air pollution, etc), get more or better quality training data, or perhaps select a more powerful model(e.g., a Polynomial Regression model).

Follow Us:

Aman Kharwal

Coder with the ♥ of a Writer || Data Scientist | Solopreneur | Founder

ARTICLES: 1228

 PREVIOUS

NEXT 

Recommended For You

Applications of Data Science in Finance

February 23, 2022

Best Programming Languages for Data Structures and Algorithms

February 19, 2022

Websites using Python

February 18, 2022

Difference Between Variance Covariance and Correlation

February 16, 2022

Leave a Reply

Enter your comment here...

FACEBOOK INSTAGRAM MEDIUM LINKEDIN