Linear Regression in Python

Jose Chavez

July 26, 2023

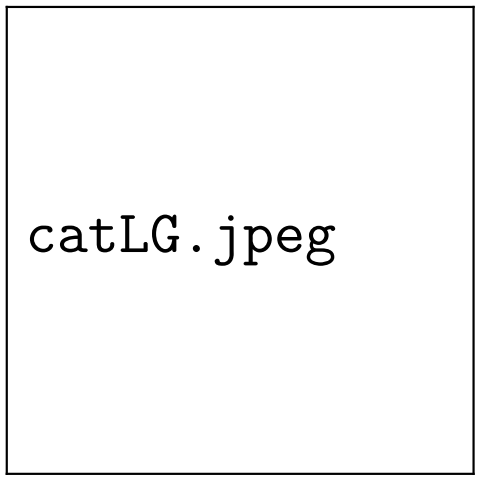


Table of Contents

Recap and Introduction

- Linear Regressions: Dependent variable (y) and one or more independent variables ($x_1, x_2, ..., x_n$).
  Fitting a linear equation to the observed data.
  Minimize differences between. real and predicted
  The linear regression model is represented by the equation:

$$y = b_0 + b_1 x_1 + b_2 x_2 + ... + b_n x_n$$

  where:
  - y is the dependent variable (the one we want to predict),
  - b_0 is the intercept (the value of y when all x values are zero),
  - $b_1, b_2, ..., b_n$ are the coefficients (slope) for each independent variable $x_1, x_2, ..., x_n$.

Linear Regression in Python

Recap and Introduction

3 Hi class. How are you all doing? Recall that last class we coverd the main concepts behind linear regression and we saw a simple example with ages and heights as features. I promised that today we would see how to do all of this with Python.

Recap and Introduction

- Linear Regressions: Dependent variable (y) and one or more independent variables ($x_1, x_2, ..., x_n$).
  Fitting a linear equation to the observed data.
  Minimize differences between. real and predicted
  The linear regression model is represented by the equation:

$$y = b_0 + b_1x_1 + b_2x_2 + ... + b_nx_n$$

  where:
  - y is the dependent variable (the one we want to predict),
  - b_0 is the intercept (the value of y when all x values are zero),
  - $b_1, b_2, ..., b_n$ are the coefficients (slope) for each independent variable $x_1, x_2, ..., x_n$.

Hypothetical Project

Question:
Are older Twitters user's less likely to use the app throughout the week? What's the relationship between age usage?

Hypothetical Project

Question:
Are older Twitters user's less likely to use the app throughout the week? What's the relationship between age usage?

let's assume that we are being asked to find a relationship between the age of twitter users and their average rate, per week of twitter usage. We suspect that there's probably some relationship, in fact its probably safe to say that most people would expect older users to tweet less often, right? Let's get a feel for that.

Data: Fake Twitter Data

- Hypothetical Data Scientist scenario:
  You are asked to investigate the connection between age and tweet rates.
  You are given an access token to an API.
- Dataset for our demonstration: Python scripts you can find at: github were used to generate hundreds of records of consisting of records that include: age, date, tweet (a string);
- Why not the real thing?
  Privacy:)

Linear Regression in Python

Data: Fake Twitter Data

Data: Fake Twitter Data

- Hypothetical Data Scientist scenario:
  You are asked to investigate the connection between age and tweet rates.
  You are given an access token to an API.
- Dataset for our demonstration: Python scripts you can find at: github were used to generate hundreds of records of consisting of records that include: age, date, tweet (a string);
- Why not the real thing?
  Privacy:)

Let's talk about what data we're going to be using. I took the liberty of dveloping some python scripts that generate three years of "fake data" for a database of twitter users. I used Poissson distributions and an assortment of custom scripts to simulate many records. You can find the scripts and colloborate with me potentially on them by visiting the link provided.
Now imagine this. You are a data scientist with access to an api providing the data in csv format and are being asked to quickly establish some sort of usable mathematical relationship between user age and the amount that they tweet per week.
Now the data that we are going to be working with was created by me via Python. It's fairly simple. It's a essentially csv data consisting of two columns corresponding to age and average number of tweets per week. You can find the code used to generate it as well as some more flexible but complex code at the github link provided in our slides.

I didn't use real data because the real data available from twitter online doesn't contain any user information to protect privacy not even ages. I

The Data Science Process

- **Understand the Problem:** Define problem and objective.
  Here: The problem is finding a relationship between the ages and weekly tweet rates.
- **Data Collection:** Gather data.
- **Exploratory Data Analysis (EDA):** Explore the data to get a feel for its strucute.
  Here: We're going to view a scatter plot to get a rough idea whether or not we should expect any success.

Linear Regression in Python

The Data Science Process

Before we move on, let's remember to

- **Understand the Problem:** Define problem and objective.
  Here: The problem is finding a relationship between the ages and weekly tweet rates.
- **Data Collection:** Gather data.
- **Exploratory Data Analysis (EDA):** Explore the data to get a feel for its strucute.
  Here: We're going to view a scatter plot to get a rough idea whether or not we should expect any success.

The Data Science Processs (2)

- **Feature Selection:** Choosing relevant features (given in our problem).
  Here: This is straighforward in our scenario.
- **Data Splitting:** split to train and to test.
- **Model Building:** Use data to create a model.
- **Model Evaluation:** Assess the model's performance using evaluation metrics like Mean Squared Error (MSE), R-squared, or Root Mean Squared Error (RMSE).

The Data Science Processs (2)

- **Feature Selection:** Choosing relevant features (given in our problem).
  Here: This is straighforward in our scenario.
- **Data Splitting:** split to train and to test.
- **Model Building:** Use data to create a model.
- **Model Evaluation:** Assess the model's performance using evaluation metrics like Mean Squared Error (MSE), R-squared, or Root Mean Squared Error (RMSE).

Page: 12Continuing on we need to choose features, recall that this is the part of the data that we choose to potentially explain what we're interested in. In this case we're interested in the rate of tweets and we only have one features so our task is simple this time. We will use Python to split the set. More on that when we get there.We are again going to benefit from Python when it comes to this but we will have use our knowledge to interpret the values.12

1. If this was last week we'd be doing the linear regression ourselves. This morning we will be using python's scikit-learn library.

The Data Science Process(3)

- **Model Validation:** Validate the final model on the testing data to ensure it generalizes well to new, unseen data.
- **Interpretation:** Interpret your findings.
- **Deployment:** Deploy the trained model to make predictions on new data or integrate it into a larger application.
- **Monitoring and Maintenance:** Monitor the model's performance, retrain, tune, etc.

Python and Linear Regression

- Idea: We will use Python and some helpful libraries to create linear regression , then we will test it's accuracy and consider next steps.
- Python Libraries:

  Importing and cleaning the data:
  - pandas
  - python scripts for detecting and removing outliers and null values

  Exploring the data:
  - matplotlib

  Model training and evaluation:
  - `import sklearn as sk`
  - sk.LinearRegression;
  -

  Visualing the model's results

1. matplotlib
2. tableau?

Collect Data from our API

How does text interact with this. Our training data we set aside seems to "align" with the line. . .

??

Linear Regression in Python

Collect Data from our API

How does text interact with this. Our training data we set aside seems to "align" with the line. . .

??

Let' assume that we have a token and get access to our twitter data and that it looks something like what's shown. Again the idea is that you would use your token and access the API endpoint which would in turn send you the csv. Now we need to hydrate a python object that lets us work with this. In effect we want to be able to use the pandas library to clean up the data at a minimum so that we might have data structures that can be used in tandem with the sk learn tools we will be bringing in.

Collect Data from our API (2)

The JavaScript Node.js Application serving the csv file we generated with Python.

pics/server.png

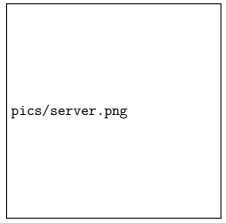
To access: `curl -i http://csv.bluewfjc.online/download?token=playtoken`

Linear Regression in Python

Collect Data from our API (2)

Collect Data from our API (2)

The JavaScript Node.js Application serving the csv file we generated with Python.



To access: `curl -i http://csv.bluewfjc.online/download?token=playtoken`

Let' assume that we have a token and get access to our twitter data and that it looks something like what's shown. Again the idea is that you would use your token and access the API endpoint which would in turn send you the csv. Now we need to hydrate a python object that lets us work with this. In effect we want to be able to use the pandas library to clean up the data at a minimum so that we might have data structures that can be used in tandem with the sk learn tools we will be bringing in.

Examine Visuals of Data

Plotting ages versus rates per week Does it look linear?

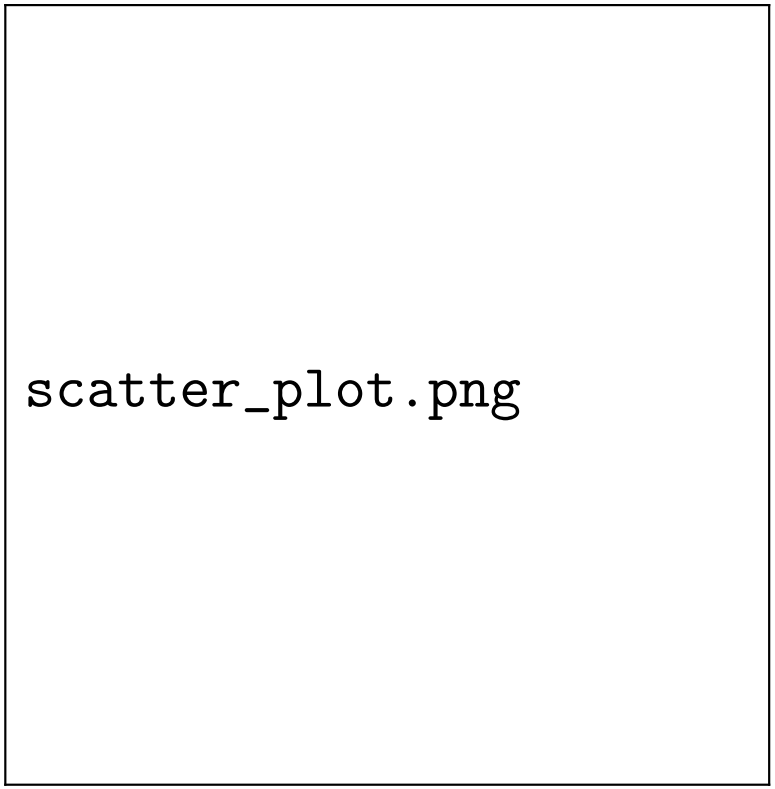


Figure: Scatter Plot of Data Produced with Matplotib

Cleanup

Removes negative 'rates' values

Drops rows with null values

Displays initial cleaned data

Calculates quartiles and IQR

Removes outliers from data
Displays data without outliers

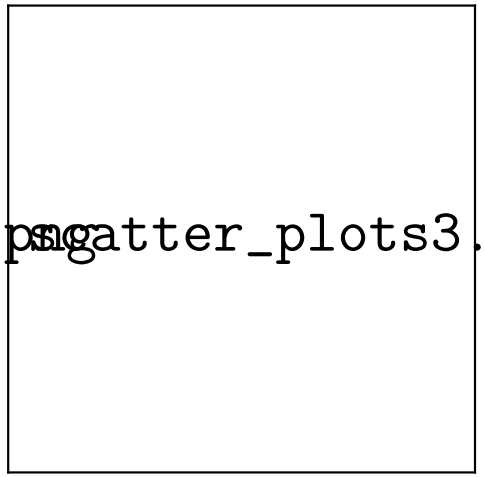
```
  # Drop negative values
df = df[df['rates'] >= 0]
# Drop null values
df = df.dropna()

# Display the first few rows of the dataframe
df.head()
Q1 = df.quantile(0.25)
Q3 = df.quantile(0.75)
IQR = Q3 - Q1

# Remove outliers
df_no_outliers = df[~((df < (Q1 - 1.5 * IQR)) | (df > (Q3 + 1.5 * IQR))).any(axis=1)]

# Display the first few rows of the dataframe without outliers
df_no_outliers.head()
```

Figure: Pyt Data Scatter Plot With Line

Clean Up

Code run and differences:

scatter_plots1.png

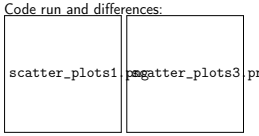


Linear Regression in Python

Clean Up

Clean Up

Code run and differences:



Now if one were to examine the records in their entirety, one would discover that there's some negative values and some outliers. In particular there are outliers outside of 3 standard deviations. So we need to run some scripts to get rid of these. Then we can commence divying up the data into training and testing portions, train our model and evaluate the situation.

Library Imports and Splitting the Data

```
1  from sklearn.model_selection import train_test_split
2  from sklearn.linear_model import LinearRegression
3  from sklearn.metrics import mean_squared_error, r2_score
4    # Split the data into training and test sets
5  X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
       random_state=42)
6
7  # Create a linear regression object
8  regr = LinearRegression()
9
10 # Train the model using the training sets
11 regr.fit(X_train, y_train)
```

.

Linear Regression in Python

Library Imports and Splitting the Data


```
1  from sklearn.model_selection import train_test_split
2  from sklearn.linear_model import LinearRegression
3  from sklearn.metrics import mean_squared_error, r2_score
4    # Split the data into training and test sets
5  X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
           random_state=42)
6
7  # Create a linear regression object
8  regr = LinearRegression()
9
10 # Train the model using the training sets
11 regr.fit(X_train, y_train)
```

Okay create the model. Now we have clean data and Python is ready with its scikit-learn models but before we use those models we need to do an additional step with our data. Does anyone recall? Are we going to use all the dat? Right, so recall that first of all we need to divide our data into two parts. In particular we need to take a small fraction of our data and set it aside for testing and we are free to to use the rest for the actual training of the model. So that's what you are seeing here and again this is all in our classes' python notebook so no worries you can find it there. Now I let's finally get to the exact statements and we will see that one of the lines is all about this splitting we just mentioned. We first see the import of sci kit learn's train_test_split split method. This method is going to help us divy up the data as we mentioned and I will be explaining that shortly. Next we see an import of sci kit learns linearRegressio method from its linear models module. Recall that we access modules with packagename.module name and we add the name of a method if we're jsust grabbing one method. Next we see that i'm importing some methods from the metrics module. In particualr tools to help us get the maximum error squared and r2 score. Theres are going to help us evaluate the model. We will also take advantage of residuals as you will see. Any questiosn?

Okay next we see that we are using the train_test_split method to actually take the data in and assign 20% of the data both features and outputs as test data and the rest as training data. Notice the nifty destructuring going on there. The method returns four objects and we asign them all at once. Next we instantiate an instance of a linear regression model. This is the wwhere the magic sort of starts right.

But its just a template of a linear regression model. It doesn't have any adata. Well in the next line we see that we are loading the training data into th emodel and subsequently fitting the model.

Okay let's pause a moment. So we have the model now. Its parts are blackboxed for now but will soon see its details. Before doing so let's go ahead and make some predictions. Recall that we divied up our data into training asets and testing sets. Its tim eto use that 20% to get a feel for how the model will perform.

With that in mind we see that on line 11 we call the models predict method on the model with the training features fed in and the

Gathering the Model's Parameters and Testing

```
12 # Make predictions using the testing set
13 y_pred = regr.predict(X_test)
14 # Print the coefficients
15 #print('Coefficients:', regr.coef_)
16 model_cos=regr.coef_
17
18 # Print the mean squared error
19 #print('Mean squared error:', mean_squared_error(y_test, y_pred))
20 mse=mean_squared_error(y_test, y_pred)
21
22 coefficients = regr.coef_[0]
23 intercept = regr.intercept_
24 # Print the coefficient of determination (R^2 score)
25 #print('Coefficient of determination (R^2 score):', r2_score(y_test,
       y_pred))
26 r2score=r2_score(y_test,y_pred)
```

The Model We Have Created...

??

and MSE (means squared error)

?? ??

and intercept ...

??

and as a formula:

??

Evaluating the Model Visually

Our training data we set aside seems to "align" with the line. . .

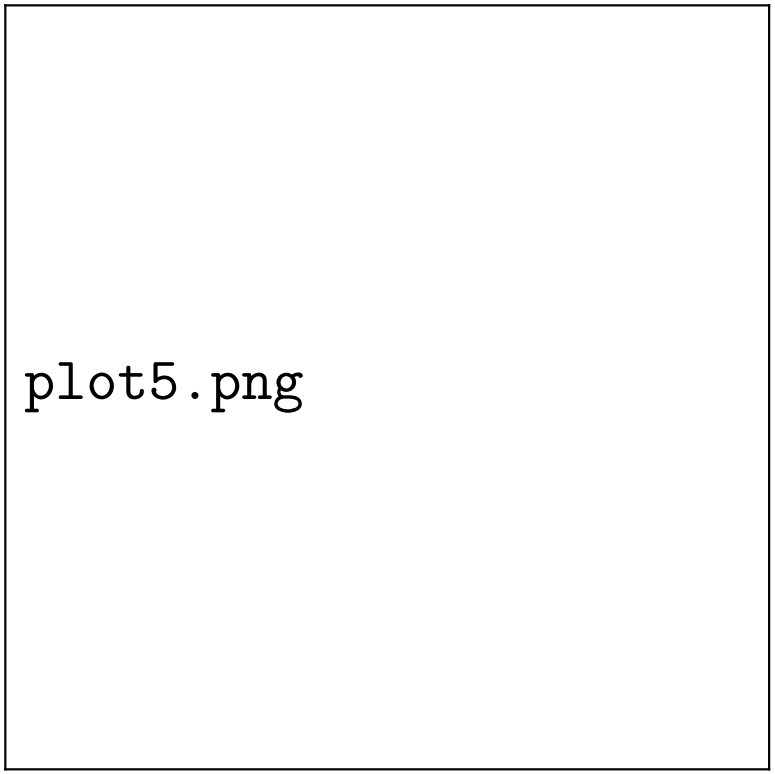


Figure: Training Data Scatter Plot With Line

18 Now let's take a look at the model's capabilities visually.

Evaluating the Model Visually

Our training data we set aside seems to "align" with the line...

Figure: Training Data Scatter Plot With Line

Evaluating the Model Visually (2)