
LINEAR REGRESSION
LINEAR REGRESSION EVERYWHERE
imgflip.com

Welcome to Week 6 Lecture 1!

Data Science in Python & Machine Learning

Warm-Up

What is the difference between

Regression And Classification?

Put your answer in chat.

Learning Goals

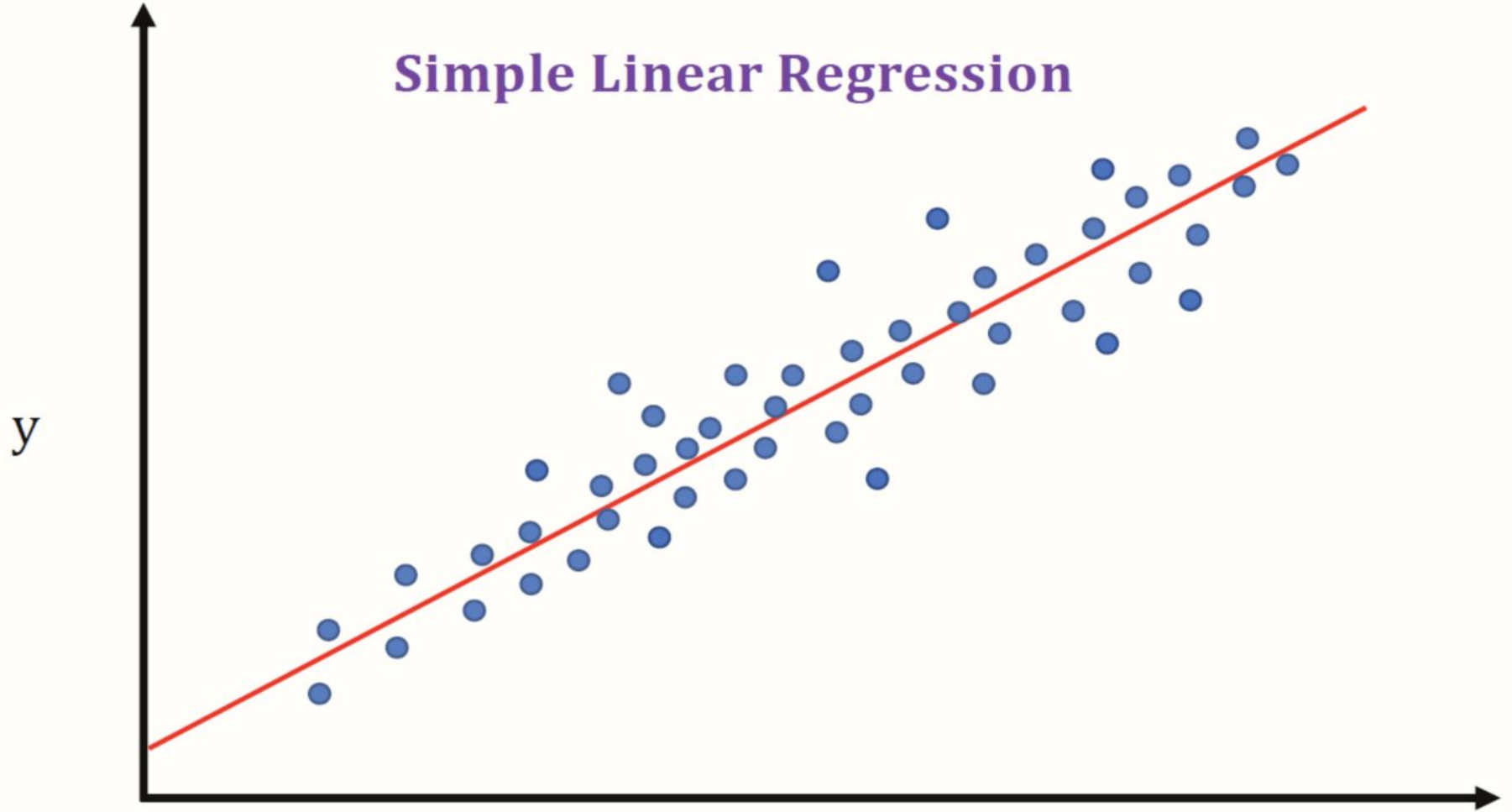
After this class you will be able to:

1. Use a linear regression model with a pipeline
2. Evaluate a regression model using multiple appropriate metrics.
3. Compare a fitted model's performance to a baseline.

Linear Regression

- Most commonly used machine learning algorithm
- $Y = b_1x + b_0$
- Model assumes a linear relationship between the input and output

Simple Linear Regression



Using a Model with a Pipeline

```
# imports
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.compose import make_column_selector, make_column_transformer
from sklearn.pipeline import make_pipeline
```

Using a Model with a Pipeline
Continued

```
#Instantiate the make column selectors to select the appropriate columns for the preprocessing steps
cat_selector = make_column_selector(dtype_include='object')
num_selector = make_column_selector(dtype_include='number')

#Instantiate the OneHotEncoder and the Standard Scaler
ohe = OneHotEncoder(sparse=False, handle_unknown='ignore')
scaler = StandardScaler()

#Create Tuples for the OneHotEncoder and Standard Scaler
ohe_tuple = (ohe, cat_selector)
scaler_tuple = (scaler, num_selector)

#Instantiate the Make Column Transformer to transform the data
transformer = make_column_transformer(scaler_tuple, ohe_tuple, remainder='passthrough')
```

Using a Model with a Pipeline
Continued

```
1 # instantiate a linear regression model
2 # put your ColumnTransformer and linear regression model into a pipeline
3 # fit your pipe on the training data
4
5 lin_reg_pipe = make_pipeline(transformer, LinearRegression())
6 lin_reg_pipe.fit(X_train, y_train);
```

Evaluating Regression Models

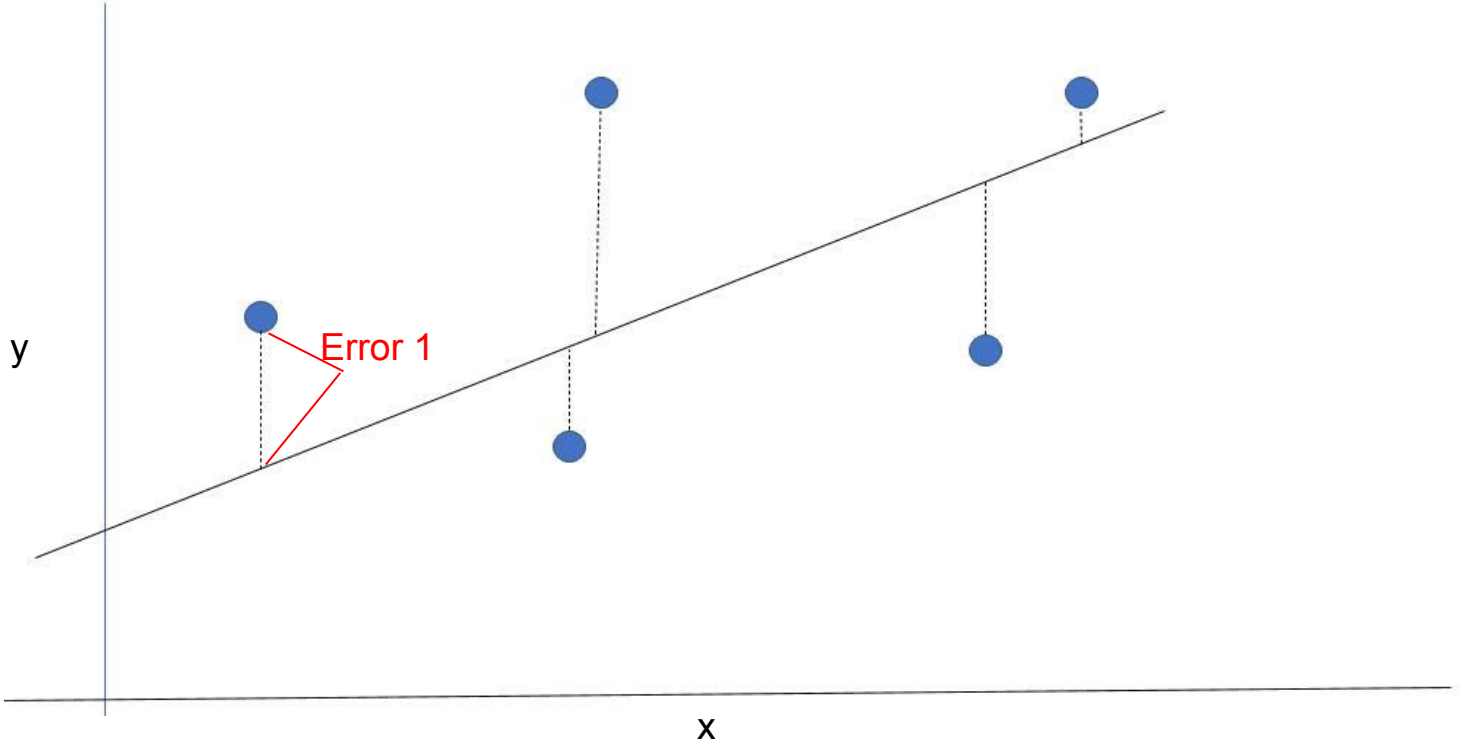
How do we know if our model is any good?

Regression Evaluation Metrics

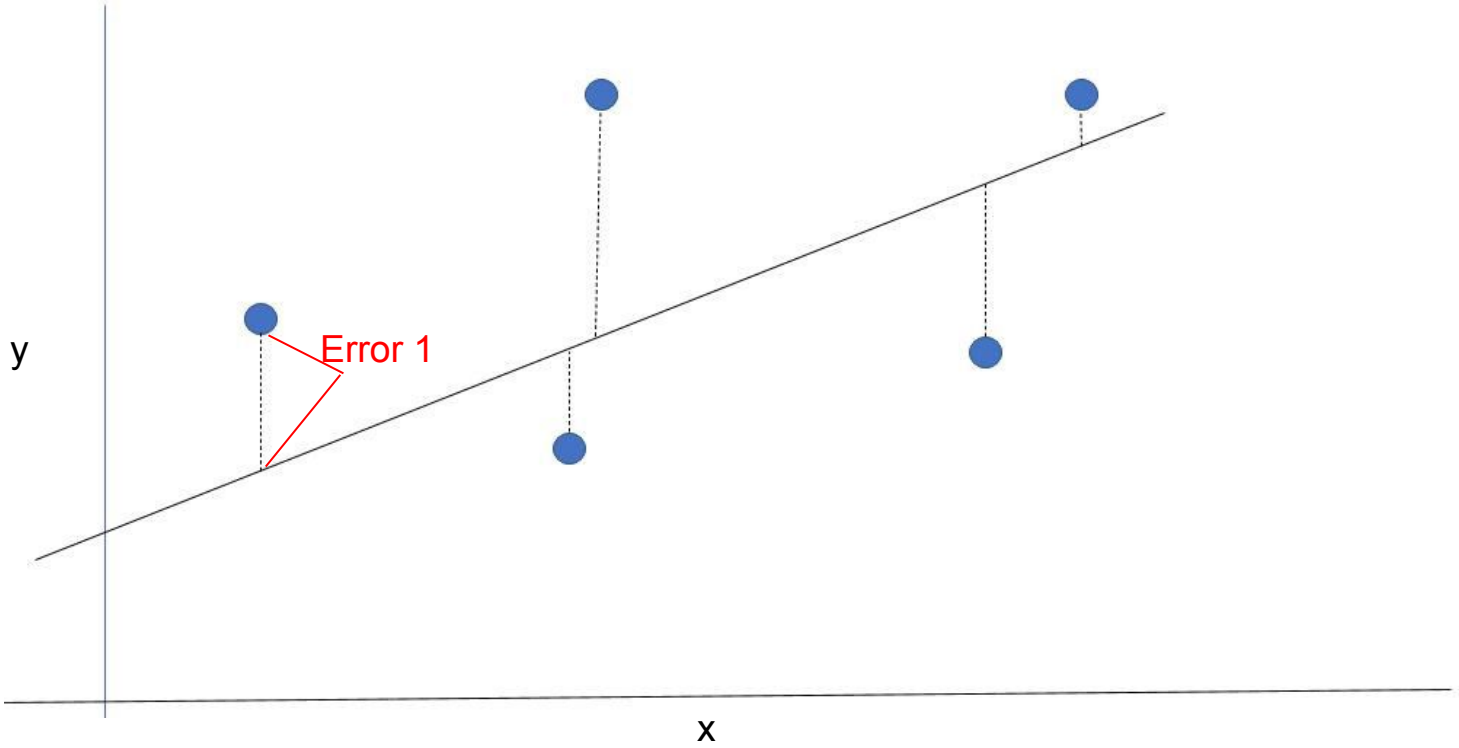
- Mean Absolute Error (MAE)
- Mean Squared Error (MSE)
- Root Mean Squared Error (RMSE)
- R-Square (R^2) or (R2) or (R^2)

What does ‘Error’ mean in predictive modeling?

- The line represents the prediction.
- The blue dots represent the actual values.
- The distance along the y axis between each dot and the line is the error.



➢ Notice that sometimes the prediction is over and sometimes it is under



Let’s look at a regression problem where we are predicting price in dollars.

True Value	Prediction
2	5
6	6
7	4
8	3
10	14
9	10

Note: These number are just examples for easy calculating, they don’t match the previous graph

- Error is the difference between the prediction and actual value
- If we wanted to add up each of the errors, we would have positive errors and negative errors cancelling each other out!

True Value	Prediction	Error
2	5	-3
6	6	0
7	4	3
8	3	5
10	14	-4
9	10	-1
Total		0

0/6 = 0

Mean Error	0

Adding the errors results in an average error of 0!

We know our predictions weren't perfect, so this won't work!

➢ One way to avoid positive and negative values cancelling is take the absolute value of each error before finding the average

True Value	Prediction	Error	Absolute Error
2	5	-3	3
6	6	0	0
7	4	3	3
8	3	5	5
10	14	-4	4
9	10	-1	1
Total:		0	16

16/6 ≈ 2.67

Mean Error	0
Mean Absolute Error	≈ 2.67

MAE is in the same units as the original data!

- Another way to deal with positive and negative errors is to square each error before finding the average. The result is always positive

True Value	Prediction	Error	Absolute Error	Squared Error
2	5	-3	3	9
6	6	0	0	0
7	4	3	3	9
8	3	5	5	25
10	14	-4	4	16
9	10	-1	1	1
Total:		0	16	60

16/6 ≈ 2.67

60/6 = 10

Mean Error	0
Mean Absolute Error	≈ 2.67
Mean Squared Error	10

Mean Squared Error is NOT In the same units as the true values.

➢ In order to return the squared error to the same units as the true values, we can take the square root.

True Value	Prediction	Error	Absolute Error	Squared Error
2	5	-3	3	9
6	6	0	0	0
7	4	3	3	9
8	3	5	5	25
10	14	-4	4	16
9	10	-1	1	1
Total:		0	16	60

16/6 ≈ 2.67

60/6 = 10

√10 ≈ 3.16

Mean Error	0
Mean Absolute Error	≈ 2.67
Mean Squared Error	10
Root Mean Squared Error	≈ 3.16

Root Mean Squared Error is in the same units as the true values.

Benefits and Drawbacks

MAE: Uses same units, but does not punish large errors.

MSE: Punishes large errors, but not in same units

RMSE: Punishes large errors AND in same units.

R^2 value

- While MAE, MSE, and RMSE scores are all specific to the particular data set, there IS a metric that can be interpreted similarly on any data set.
- This is the R^2 value. (The coefficient of determination.)
- It is usually given as a decimal value, but it is usually interpreted as a percent. For example, an R^2 of 0.95 means that the model explains 95% of the variation in y using the features
- A mean value baseline will always score R^2 of 0.
- R^2 will never be higher than 1 no matter what the units of the data set.
- Notice that unlike our other metrics, a higher R^2 indicates a better result!

- **REGRESSION METRICS ARE ONLY FOR REGRESSION MODELS**

Baseline Models

- Using the three metrics discussed so far, how do you know if you have a “good” model?
- For example, the same RMSE might be “good” for one data set and horrible for another.
- It is very specific to the particular data set.
- Using a baseline model is a great way to help you interpret MAE, MSE, or RMSE.

- There is NO set rule for how to develop a baseline model.
- For regression problems, one common approach is to see how well your model compares to simply predicting the mean of y for every value in your training set.

True Value	Baseline Prediction	Error	Absolute Error	Squared Error
2	7	-5	5	25
6	7	-1	1	1
7	7	0	0	0
8	7	1	1	1
10	7	3	3	9
9	7	2	2	4
Total:		0	12	40

Metric	Model	Baseline
Mean Error	0	0
Mean Absolute Error	≈ 2.67	2
Mean Squared Error	10	≈ 6.67
Root Mean Squared Error	≈ 3.16	≈ 2.58

Our model does not beat the baseline!
It's a bad model

Regression Metrics in Python

```
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
import numpy as np

Train_pred = model.predict(X_train)
```

MAE:

```
mae = mean_absolute_error(true, predicted)
```

Ex: `mae = mean_absolute_error(y_train, Train_pred)`

MSE:

```
mse = mean_squared_error(true, predicted)
```

RMSE:

```
rmse = np.sqrt(mean_squared_error(true, predicted))
```

R2:

```
r2 = r2_score(true, predicted)
```

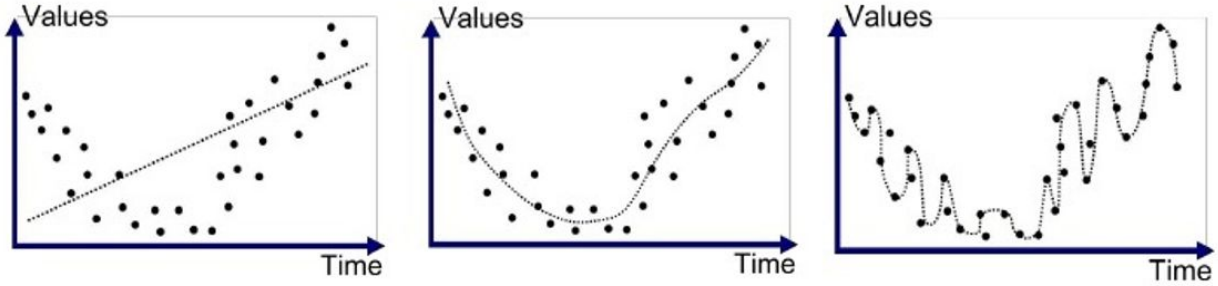
Dummy Regressor Baseline Model in Python

Just like any other model:

```
#Import
from sklearn.dummy import DummyRegressor

#Instantiate
dummy = DummyRegressor(strategy='mean')

#Fit
dummy.fit(X_train, y_train)

#Predict
train_pred = dummy.predict(X_train)
test_pred = dummy.predict(X_test)
```

Overfitting

- Model does well on data it has seen
- Poorly on new data
- Model fit too tightly to the specific data in the **training set**
- It ‘memorized’ the answers in the training set, but did not find a generalized function for the problem



Underfitted | Good Fit/Robust | Overfitted

Image Source

Predict the sale price of a house

CodeAlong Colab Notebook

Solutions will be posted tomorrow. Remind me if I forget!

ChallengeNotebook