Fitting a line through data

Linear regression is the first, and therefore, probably the most fundamental model—a straight line through data.

Description

The *boston* dataset is useful for learning about linear regression. The *boston* dataset has the median home price of several areas in Boston. It also has other factors that might impact housing prices, for example, crime rate.

Getting the data

Firstly, we import the datasets model, then we can load the dataset:

```
>>> from sklearn import datasets
>>> boston = datasets.load_boston()
```

Implementation

Using linear regression in scikit-learn is very simple. First, import the `LinearRegression` object and create an object (lets call it `lr`):

```
>>> from sklearn.linear_model import LinearRegression
>>> lr = LinearRegression()
```

To fit the model, supply the independent and dependent variables to the `fit` method of `LinearRegression`:

```
>>> lr.fit(boston.data, boston.target)

LinearRegression(copy_X=True, fit_intercept=True, normalize=False)
```

Let's take a look at the regression coefficients:

```
>>> lr.coef_
array([ -1.07170557e-01,  4.63952195e-02, 2.08602395e-02,
2.68856140e+00, -1.77957587e+01, 3.80475246e+00,
7.51061703e-04, -1.47575880e+00, 3.05655038e-01,
-1.23293463e-02, -9.53463555e-01, 9.39251272e-03,
-5.25466633e-01])
```

A common pattern to express the coefficients of the features and their names is `zip(boston.feature_names, lr.coef_)`.

- We can see which factors have a negative relationship with the outcome, and also the factors that have a positive relationship.
- The per capita crime rate is the first coefficient in the regression. An increase in the per capita crime rate by town has a negative relationship with the price of a home in Boston.

Making Predictions

Now, to get the predictions, use the `predict` method of `LinearRegression`:

```
>>> predictions = lr.predict(boston.data)
```

Residuals

The next step is to look at how close the predicted values are to the actual data. These differences are known as **residuals**. We can use a histogram to look at these residuals.

Other Remarks

The `LinearRegression` object can automatically normalize (or scale) the inputs:

```
>>> lr2 = LinearRegression(normalize=True)
>>> lr2.fit(boston.data, boston.target)

LinearRegression(copy_X=True, fit_intercept=True, normalize=True)
>>> predictions2 = lr2.predict(boston.data)
```