This can all be done with the BayesianRidge object (part of the linear_model module). Like all the previous regression objects, this one can be initialized with no required arguments.

```
print('Data shape: {}\n'.format(data.shape))
    print('Labels shape: {}\n'.format(labels.shape))
    from sklearn import linear model
    reg = linear_model.BayesianRidge()
    reg.fit(data, labels)
    print('Coefficients: {}\n'.format(repr(reg.coef_)))
    print('Intercept: {}\n'.format(reg.intercept_))
10 print('R2: {}\n'.format(reg.score(data, labels)))
11 print('Alpha: {}\n'.format(reg.alpha_))
    print('Lambda: {}\n'.format(reg.lambda_))
    RUN
                                                                                                SAVE
                                                                                                            RESET
                                                                                                                 Close
                                                                                                                 1.388s
Output
 Data shape: (150, 4)
 Labels shape: (150,)
 Coefficients: array([-0.11174619, -0.03900476, 0.24330537, 0.57343721])
 Intercept: 0.17022693722601356
 R2: 0.9303454031271241
```

We can manually specify the α_1 and α_2 gamma parameters for α with the <code>alpha_1</code> and <code>alpha_2</code> keyword arguments when initializing <code>BayesianRidge</code>. Similarly, we can manually set λ_1 and λ_2 with the <code>lambda_1</code> and <code>lambda_2</code> keyword arguments. The default value for each of the four gamma parameters is 10^{-6} .

So far, we've discussed hyperparameter optimization through cross-validation. Another way to optimize the hyperparameters of a regularized regression model is with Bayesian techniques.

In Bayesian statistics, the main idea is to make certain assumptions about the probability distributions of a model's parameters *before* being fitted on data. These initial distribution assumptions are called *priors* for the model's parameters.

In a Bayesian ridge regression model, there are two hyperparameters to optimize: α and λ . The α hyperparameter serves the same exact purpose as it does for regular ridge regression; namely, it acts as a scaling factor for the penalty term.

The λ hyperparameter acts as the precision of the model's weights. Basically, the smaller the λ value, the greater the variance between the individual weight values.

B. Hyperparameter priors

Both the α and λ hyperparameters have gamma distribution priors, meaning we assume both values come from a gamma probability distribution.

There's no need to know the specifics of a gamma distribution, other than the fact that it's a probability distribution defined by a shape parameter and scale parameter.

Specifically, the α hyperparameter has prior:

$$\Gamma(\alpha_1, \alpha_2)$$

and the λ hyperparameter has prior:

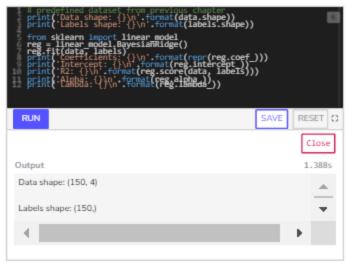
$$\Gamma(\lambda_1, \lambda_2)$$

where $\Gamma(k, \theta)$ represents a gamma distribution with shape parameter k and scale parameter θ .

C. Tuning the model

When finding the optimal weight settings of a Bayesian ridge regression model for an input dataset, we also concurrently optimize the α and λ hyperparameters based on their prior distributions and the input data.

This can all be done with the BayesianRidge object (part of the 1 inear_model module). Like all the previous regression objects, this one can be initialized with no required arguments.



We can manually specify the α_1 and α_2 gamma parameters for α with the <code>alpha_1</code>