**Machine Learning Algorithms in Python**

There are some code examples of most of the algorithms in this website:

<https://machinelearningmastery.com/spot-check-regression-machine-learning-algorithms-python-scikit-learn/>

* **Linear Regression**
* **Ridge Regression:** is an extension of linear regression where the loss function is modified to minimize the complexity of the model measured as the sum squared value of the coefficient values (also called the l2-norm). RidgeCV is the same but implements Cross Validaton.
* **LASSO Regression:** the Least Absolute Shrinkage and Selection Operator is a modification of linear regression, like ridge regression, where the loss function is modified to minimize the complexity of the model measured as the sum absolute value of the coefficient values (also called the l1-norm).
* **ElasticNet Regression:** combines the properties of both Ridge Regression and LASSO regression. It minimizes the complexity of the regression model (magnitude and number of regression coefficients) by penalizing the model using both the l2-norm (sum squared coefficient values) and the l1-norm (sum absolute coefficient values).
* **K-Nearest Neighbors:** (or KNN) locates the K most similar instances in the training dataset for a new data instance. From the K neighbors, a mean or median output variable is taken as the prediction.
* **Classification and Regression Trees:** (CART as they are known) use the training data to select the best points to split the data in order to minimize a cost metric **(DecisionTreeRegression)**
* Support Vector Machines extended for the prediction real-valued problems called **Support Vector Regression (SVR).**
* **Kernel Ridge Regression:** combines Ridge Regression with the kernel trick. It thus learns a linear function in the space induced by the respective kernel and the data. Here is the function in Python and one example: <http://scikit-learn.org/stable/modules/generated/sklearn.kernel_ridge.KernelRidge.html#sklearn.kernel_ridge.KernelRidge>
* **Gaussian Process Regression (GPR):** one disadvantage is that loose efficiency in high dimensional spaces – namely when the number of features exceeds a few dozens, but in our case, we have a small set of features, so we can try it. Here is the function in Python and one example: <http://scikit-learn.org/stable/modules/generated/sklearn.gaussian_process.GaussianProcessRegressor.html#sklearn.gaussian_process.GaussianProcessRegressor>
* Logistic Regression: <http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html>