**Rescale Data -** When your data is comprised of attributes with varying scales, many machine learning algorithms

can bene\_t from rescaling the attributes to all have the same scale. Often this is referred to

as normalization and attributes are often rescaled into the range between 0 and 1. This is

useful for optimization algorithms used in the core of machine learning algorithms like gradient

descent. It is also useful for algorithms that weight inputs like regression and neural networks

and algorithms that use distance measures like k-Nearest Neighbors. You can rescale your data

using scikit-learn using the MinMaxScaler

**Standardization** is a useful technique to transform attributes with a Gaussian distribution and

differing means and standard deviations to a standard Gaussian distribution with a mean of

0 and a standard deviation of 1. It is most suitable for techniques that assume a Gaussian

distribution in the input variables and work better with rescaled data, such as linear regression,

logistic regression and linear discriminate analysis. You can standardize data using scikit-learn

with the StandardScaler

**Normalizing** in scikit-learn refers to rescaling each observation (row) to have a length of 1 (called

a unit norm or a vector with the length of 1 in linear algebra). This pre-processing method

can be useful for sparse datasets (lots of zeros) with attributes of varying scales when using

algorithms that weight input values such as neural networks and algorithms that use distance

measures such as k-Nearest Neighbors. You can normalize data in Python with scikit-learn

using the Normalizer class

**Feature Selection**

Feature selection is a process where you automatically select those features in your data that

contribute most to the prediction variable or output in which you are interested. Having

irrelevant features in your data can decrease the accuracy of many models, especially linear

algorithms like linear and logistic regression. Three benefits of performing feature selection

before modelling your data are:

* Reduces Overfitting: Less redundant data means less opportunity to make decisions

based on noise.

* Improves Accuracy: Less misleading data means modelling accuracy improves.
* Reduces Training Time: Less data means that algorithms train faster.

**Ways to do feature selection:**

1. **Univariate Selection** - Statistical tests can be used to select those features that

have the strongest relationship with the output variable. The scikit-learn library

provides the SelectKBest class2 that can be used with a suite of different statistical

tests to select a specific number of features.

1. **Recursive Feature Elimination  -** The Recursive Feature Elimination (or RFE) works by

recursively removing attributes and building a model on those attributes that remain.

It uses the model accuracy to identify which attributes (and combination of attributes)

contribute the most to predicting the target attribute.

You can learn more about the RFE class in the scikit-learn documentation.

1. **Principal Component Analysis -** Principal Component Analysis (or PCA) uses linear

algebra to transform the dataset into a compressed form. Generally this is called a

data reduction technique. A property of PCA is that you can choose the number of

dimensions or principal components in the transformed result.

1. **K-fold Cross-Validation:** Cross-validation is an approach that you can use to estimate the

performance of a machine learning algorithm with less variance than a single train-test set split.

It works by splitting the dataset into k-parts (e.g. k = 5 or k = 10). Each split of the data is called a fold. The algorithm is trained on k 􀀀 1 folds with one held back and tested on the held back fold. This is repeated so that each fold of the dataset is given a chance to be the held back test set. After running cross-validation you end up with k different performance scores that you can summarize using a mean and a standard deviation. The result is a more reliable estimate of the performance of the algorithm on new data. It is more accurate because the algorithm is trained and evaluated multiple times on different data. The choice of k must allow the size of each test partition to be large enough to be a reasonable sample of the problem, whilst allowing enough repetitions of the train-test evaluation of the algorithm to provide a fair estimate of the algorithms performance on unseen data. For modest sized datasets in the thousands or tens of thousands of records, k values of 3, 5 and 10 are common.

1. **Mean Absolute Error:** The Mean Absolute Error (or MAE) is the sum of the absolute differences between predictions and actual values. It gives an idea of how wrong the predictions were. The measure gives an idea of the magnitude of the error, but no idea of the direction. A value of 0 indicates no error or perfect predictions.
2. **Mean Squared Error** The Mean Squared Error (or MSE) is much like the mean absolute error in that it provides a gross idea of the magnitude of error. Taking the square root of the mean squared error converts the units back to the original units of the output variable and can be meaningful for description and presentation. This is called the Root Mean Squared Error (or RMSE).
3. **R2 Metric** The R2 (or R Squared) metric provides an indication of the goodness of fit of a set of predictions to the actual values. In statistical literature this measure is called the coefficient of determination. This is a value between 0 and 1 for no-fit and perfect fit respectively.
4. **Spot-checking** is a way of discovering which algorithms perform well on your machine learning problem. You cannot know which algorithms are best suited to your problem beforehand. You

must trial a number of methods and focus attention on those that prove themselves the most

promising.