**Feature importance** refers to techniques that assign a score to input features based on how useful they are at predicting a target variable.

There are many types and sources of feature importance scores, although popular examples include statistical correlation scores, coefficients calculated as part of linear models, decision trees, and permutation importance scores.

Feature importance scores play an important role in a predictive modeling project, including providing insight into the data, insight into the model, and the basis for [dimensionality reduction](https://machinelearningmastery.com/dimensionality-reduction-for-machine-learning/) and [feature selection](https://machinelearningmastery.com/rfe-feature-selection-in-python/) that can improve the efficiency and effectiveness of a predictive model on the problem.

Feature importance refers to a class of techniques for assigning scores to input features to a predictive model that indicates the relative importance of each feature when making a prediction.

Feature importance scores can be calculated for problems that involve predicting a numerical value, called regression, and those problems that involve predicting a class label, called classification.

The scores are useful and can be used in a range of situations in a predictive modeling problem, such as:

* Better understanding the data.
* Better understanding a model.
* Reducing the number of input features.

**Feature importance scores can provide insight into the dataset**. The relative scores can highlight which features may be most relevant to the target, and the converse, which features are the least relevant. This may be interpreted by a domain expert and could be used as the basis for gathering more or different data.

**Feature importance scores can provide insight into the model**. Most importance scores are calculated by a predictive model that has been fit on the dataset. Inspecting the importance score provides insight into that specific model and which features are the most important and least important to the model when making a prediction. This is a type of model interpretation that can be performed for those models that support it.

**Feature importance can be used to improve a predictive model**. This can be achieved by using the importance scores to select those features to delete (lowest scores) or those features to keep (highest scores). This is a type of feature selection and can simplify the problem that is being modeled, speed up the modeling process (deleting features is called dimensionality reduction), and in some cases, improve the performance of the model.

*Often, we desire to quantify the strength of the relationship between the predictors and the outcome. […] Ranking predictors in this manner can be very useful when sifting through large amounts of data.*

Feature importance scores can be fed to a wrapper model, such as the [SelectFromModel](https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectFromModel.html) class, to perform feature selection.

There are many ways to calculate feature importance scores and many models that can be used for this purpose.

Perhaps the simplest way is to calculate simple coefficient statistics between each feature and the target variable.

## Why Is Feature Importance Useful?

Feature importance is extremely useful for the following reasons:

### 1. Data Comprehension

Building a model is one thing, but understanding the data that goes into the model is another. Like a [correlation matrix](https://builtin.com/data-science/correlation-matrix), feature importance allows you to understand the relationship between the features and the target variable. It also helps you understand what features are irrelevant for the model.

### 2. Model Improvement

When training your model, you can use the scores calculated from feature importance to reduce the dimensionality of the model. The higher scores are usually kept and the lower scores are deleted as they are not important for the model. This simplifies the model and speeds up the model’s working, ultimately improving the performance of the model.

### 3. Model Interpretability

Feature importance is also useful for interpreting and communicating your model to other stakeholders. By calculating scores for each feature, you can determine which features attribute the most to the predictive power of your model.

* SelectKBest
* Linear Regression
* Random Forest
* XGBoost
* Recursive Feature Elimination
* Boruta

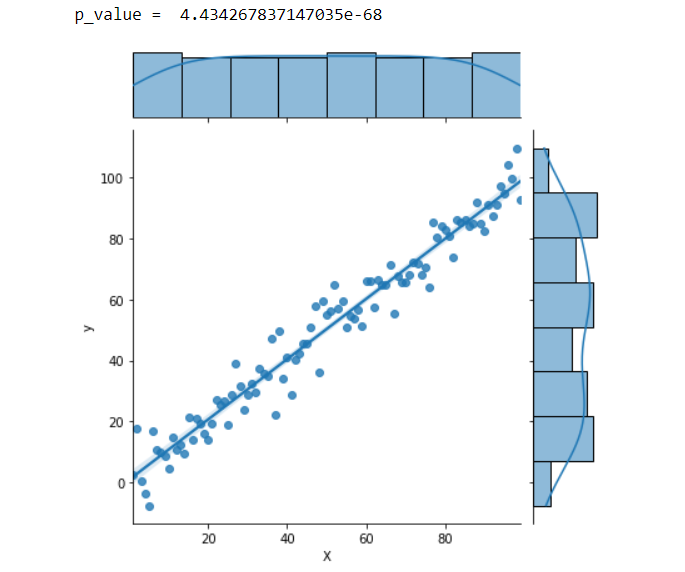
**SelectKBest**

SelectKbest is a method provided by sklearn to rank features of a dataset by their “importance ”with respect to the target variable. This “importance” is calculated using a score function which can be one of the following:

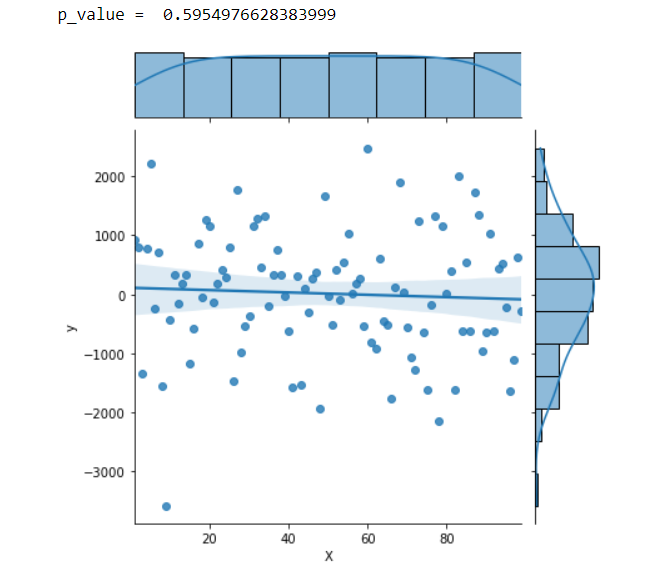
* **f\_classif**: ANOVA F-value between label/feature for classification tasks
* **f\_regression**: F-value between label/feature for regression tasks.
* **chi2**: Chi-squared stats of non-negative features for classification tasks.
* **mutaul\_info\_classif**: Mutual information for a discrete target.
* **SelectPercentile**: Select features based on the percentile of the highest scores.
* **SelectFpr**: Select features based on a false positive rate test.
* **SelectFdr**: Select features based on an estimated false discovery rate.
* **SelectFwe**: Select features based on the family-wise error rate.
* **GenericUnivariateSelect**: Univariate feature selector with configurable mode.

All of the above-mentioned scoring functions are based on statistics. For instance, the **f\_regression** function arranges the**p\_values** of each of the variables in increasing order and picks the best K columns with the least p\_value. Features with a p\_value of less than**0.05** are considered “significant” and only these features should be used in the predictive model.

**Significant Feature- P\_value lesser than 0.05**:



**Insignificant Features- P\_value more than 0.05**



This is one of the simplest methods as it is very computationally efficient and takes just a few lines of code to execute.

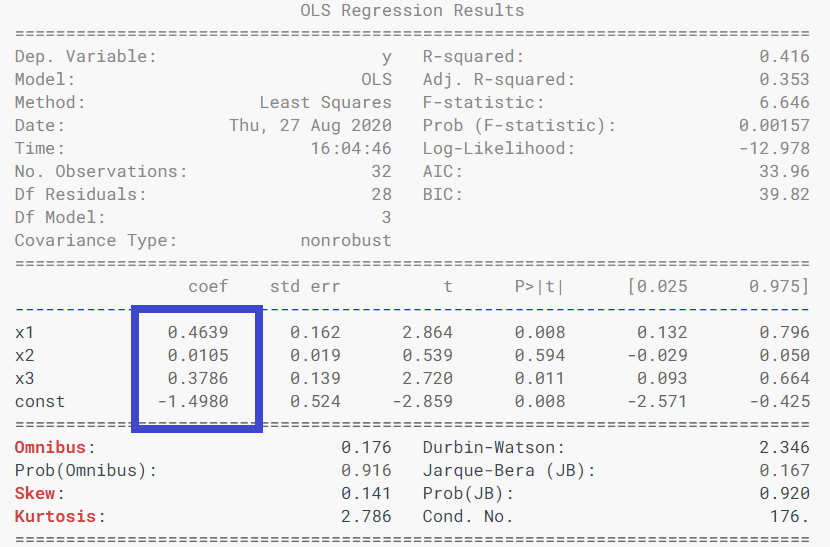
**Why P\_value is not the perfect feature selection technique?**

P\_value is an analysis of how each dependent variable is individually related to the target variable. However, this is not always the case. Let’s take an example to illustrate this

Consider a predictive regression model that tried to predict the price of a plot given the length and breadth of a plot. The p\_value of each of these variables might actually be very large since neither of these features is directly related to the price. However, a combination of these 2 variables, specifically their product, gives the land area of the plot. This product has a very strong relationship with the price. Thus both length and breadth are significant features that are overlooked during p\_value feature selection.

**Linear Regression- Comparing Coefficients**

By comparing the coefficients of linear models, we can make an inference about which features are more important than others. This method does not work well when your linear model itself isn't a good fit for the dataset given. This method can be used if your model’s accuracy is around 95%



In the case of the above example, the coefficient of x1 and x3 are much higher than x2, so dropping x2 might seem like a good idea here. This approach is valid in this example as this model is a very good fit for the given data.

**Random Forest**

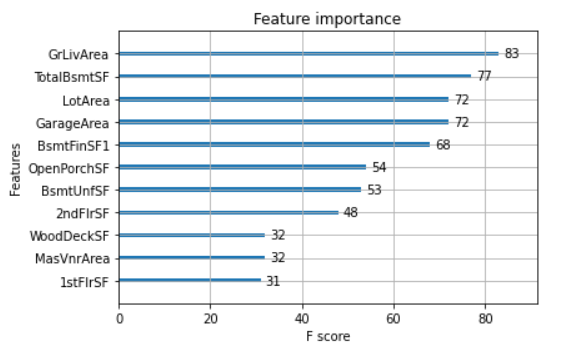
The Random Forest is a very elegant algorithm that usually gives highly accurate predictions, even with minimal hyperparameter tuning. However, this is not where its usefulness ends!

Random Forest, when imported from the sklearn library, provides a method where you can get the feature importance of each of the variables. This is a good method to gauge the feature importance on datasets where Random Forest fits the data with high accuracy.

**XGBoost**

Just like random forests, XGBoost models also have an inbuilt method to directly get the feature importance. XGBoost feature accuracy is much better than the methods that are mentioned above since:

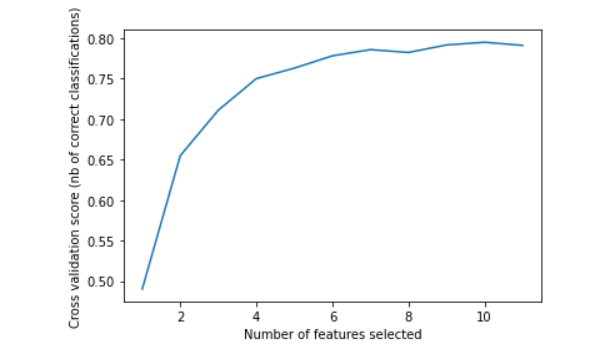
* Faster than Random Forests by far!
* It is way more reliable than Linear Models, thus the feature importance is usually much more accurate
* P\_value test does not consider the relationship between two variables, thus the features with p\_value > 0.05 might actually be important and vice versa. XGBoost usually does a good job of capturing the relationship between multiple variables while calculating feature importance



**RFE- Recursive Feature Elimination**

This algorithm recursively calculates the feature importances and then drops the least important feature. It starts off by calculating the feature importance for each of the columns. It then drops the column with the least importance score and proceeds to repeat the same.

NOTE: This algorithm assumes that none of the features are correlated. It is not advisable to use a feature if it has a Pearson correlation coefficient of more than 0.8 with any other feature.



[Image By Author]

**Boruta Feature Selection Algorithm**

Unlike the previously mentioned algorithms, Boruta is an all-relevant feature selection method while most algorithms are minimal optimal. What this means is that Boruta tries to find all features carrying useful information rather than a compact subset of features that give a minimal error.

**Steps to Build a Boruta Selector:**

* Install Bortua

!pip install boruta

* Make the necessary imports:
* Establish Base Score to build upon
* Train Boruta Feature Selector
* Calculate scores on the shortlisted features and compare them!

More often than not, using Boruta significantly reduces the dimension while also providing a minor boost to accuracy.

When trained on Housing Price Regression Dataset, Boruta reduced the dimensions from 80+ features to just 16 while it also provided an accuracy boost of 0.003%!

**Summary:**

* If the dataset is not too large, use Boruta for feature selection.
* If XGboost or RandomForest gives more than 90% accuracy on the dataset, we can directly use their inbuilt method “.feature\_importance\_”
* If you just want the relationship between any 2 variables and not the whole dataset itself, it’s ideal to go for p\_value score or person correlation.

[Understanding Feature Importance in Machine Learning | Built In](https://builtin.com/data-science/feature-importance)

[Best Practice to Calculate and Interpret Model Feature Importance | by Stacy Li | Towards Data Science](https://towardsdatascience.com/best-practice-to-calculate-and-interpret-model-feature-importance-14f0e11ee660)

[How to Calculate Feature Importance With Python - MachineLearningMastery.com](https://machinelearningmastery.com/calculate-feature-importance-with-python/)