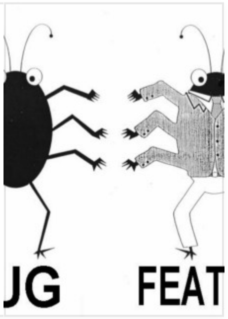
Getting Data ready for modelling: Feature engineering, Feature Selection, Dimension Reduction (Part two)

Akash Desarda Follow
Dec 13, 2018 · 10 min read

This is part two of Series of Getting Data ready for modelling. If you have not read Part 1, then I suggest you go first through it. As Feature Engineering is the generally the first step.

Getting Data ready for modelling:
Feature engineering, Feature Selection, Dimension Reduction
...
Feature engineering, Feature Selection, Dimension Reductionmedium.com



Once you have sufficient, less or no missing data or outliers next comes is Feature Selection or Feature Extraction(both of them mostly do the same job and can be used interchangeably). There are generally two approaches:

1. **Feature Extraction/Selection**

2. **Dimension Reduction or Feature Reduction**

Let's figure out them one by one, step by step.

. . .

Part 2 : Feature Extraction/Selection

So what is Feature Selection? Feature Extraction? Their Difference?

→ In machine learning and statistics, feature selection, also known as variable selection, is the process of selecting a subset of relevant features (variables, predictors) for use in model construction.

→ Feature extraction is for creating a new, smaller set of features that still captures most of the useful information.

→ Again, feature selection keeps a subset of the original features while feature extraction creates new ones.

Importance of Feature Selection/Extraction

→ This becomes even more important when the number of features are very large.

→ You need not to use every feature at your disposal for creating an algorithm.

→ You can assist your algorithm by feeding in only those features that are really important.

Where to use feature selection?

→ It enables the machine learning algorithm to train faster, reduces the complexity and makes it easier to interpret.

→ It improves the accuracy of a model if the right subset is chosen.

→ It reduces overfitting.

> *It can be broadly divided into two techniques (though this is not " **the just method out there**")*

i. Univariate Feature Selection

ii. Multivariate Feature Selection

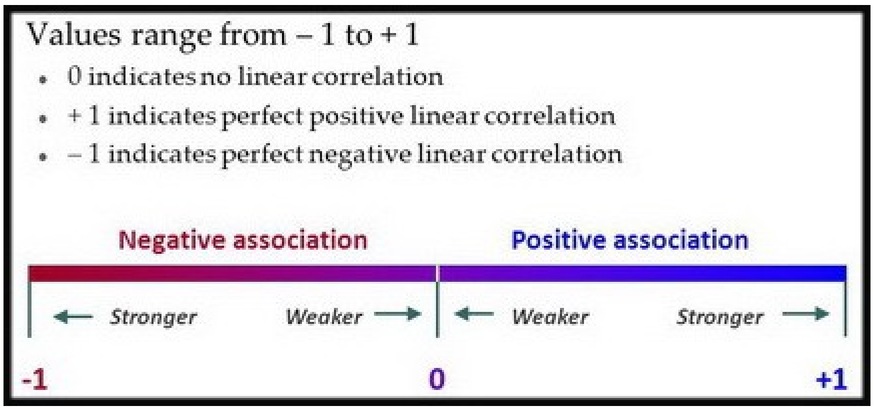
U***nivariate Feature Selection:** This technique involves more of a manual kind of work. Visiting every feature and checking its importance with the target. There are some great tricks that you should keep under your sleeve to implement* Univariate Feature Selection.

→ If you have proper **domain knowledge** and trust your judgement call, then always start with this step. Analyze all the features and remove all unwanted. Yes this is time and effort consuming step, but hey, whom would you trust more, **“the machine or yourself”**

→ **Checking the variance** (yes the ever confusing bias-variance trade-off :) of all features. The thumb rule here is set a threshold value (say a feature with 0 variance means it has the same value for every sample so such a feature would not bring any predictive power to the model) remove feature accordingly.

→ **Use of Pearson Correlation:** It might be the most applicable technique of three. If you don’t know or confused with it, then first read this article.

- So, in a nutshell, it gives us the inter-dependence between the target variable and a feature.


Values range from – 1 to + 1
0 indicates no linear correlation
+ 1 indicates perfect positive linear correlation
– 1 indicates perfect negative linear correlation
Negative association
Positive association
Stronger
Weaker
Weaker
Stronger
-1
0
+1

Thump Rule To analyse Pearson Correlation

- Thumb rule to use the Pearson Correlation:

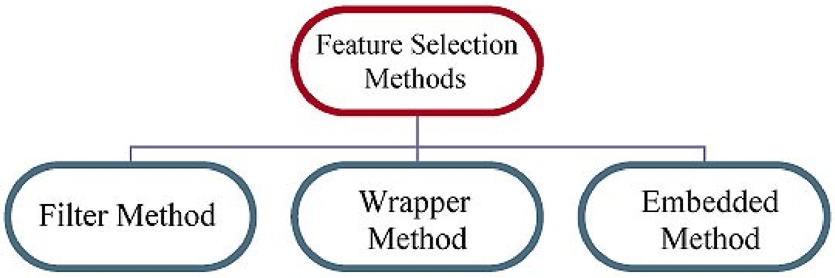
i. Selecting only moderate to strong relations with target variable. (See the image above).

ii. When two feature themselves have a strong relation with each other along with the target variable, then pick any one of them (picking both will not add any value). Use '**seaborn.heatmap()**' to visualize and pick, it greatly helps.

iii. There is a catch here 😢 . It works best with linear data and performs poorly with non-linear data (so just avoid with it).

Multivariate Feature Selection: When you have a lot of features (like hundreds or thousands of them), then it really becomes impossible to go & manually check for every one of them or if you don't have enough domain knowledge then you got to trust this following technique. So put in layman's term it is nothing but selecting multiple features at once.

Multivariate Feature Selection is broadly divided into three categories:



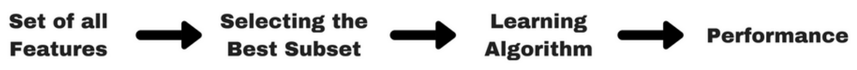
Let's check them out (We'll discuss most widely used technique in each category)

Filter Method :

→ Filter methods are generally used as a preprocessing step. The selection of features is independent of any machine learning algorithms.

→ Filter methods apply some ranking over features. The ranking denotes how 'useful' each feature is likely to be for classification. Once this ranking has been computed, a feature set composed of the best N features is created.

→ Features are selected on the basis of their scores in various statistical tests for their correlation with the outcome variable. (The correlation is a subjective term here).



Blueprint of Filter Method

1. **Pearson's Correlation:** Oh yes! Pearson Correlation is Filter method. We have already discussed it.
2. **Variance Thresholds:** This too, we have already discussed.
3. **Linear discriminant analysis:** The goal is to project a dataset onto a lower-dimensional space with good class-separability in order avoid overfitting ("***curse of dimensionality***") and also reduce computational costs.

→ Not going into maths, LDA brings all the higher dimensional variable (which we can't plot and analyse) onto 2D graph & while doing so removes the useless feature.

→ LDA is also 'S***upervised Dimension Reduction'*** technique and more kind of **Feature Extraction** than **Selection** (as it is creating kind of a new variable by reducing its dimension). So it works only on labelled data.

→ It maximizes the *separability* between classes. (Too much technical jargon, right. Don't worry see the video).

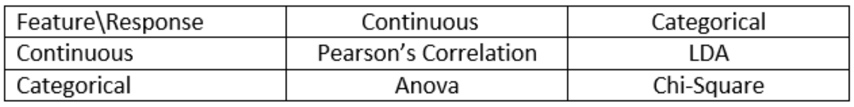
Creator: Josh Starmer

Other:

ANOVA: Analysis of variance It is similar to LDA except for the fact that it is operated using one or more categorical independent features and one continuous dependent feature. It provides a statistical test of whether the means of several groups are equal or not.

Chi-Square: It is a is a statistical test applied to the groups of categorical features to evaluate the likelihood of correlation or association between them using their frequency distribution.

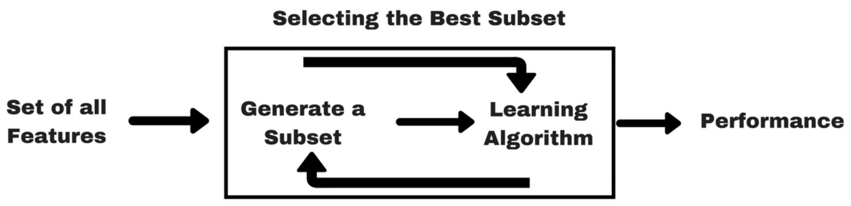
> One thing that should be kept in mind is that filter method do not remove multicollinearity. So, you must deal with multicollinearity of features as well before training models for your data.



Feature\Response	Continuous	Categorical
Continuous	Pearson's Correlation	LDA
Categorical	Anova	Chi-Square

What to choose when

Wrapper Method:



Blueprint Of Wrapper Method

→ Based on the inferences that we draw from the previous model, we decide to add or remove features from your subset.

→ Wrapper methods are so called because they wrap a classifier up in a feature selection algorithm. Typically a set of features is chosen; the efficiency of this set is determined; some perturbation is made to change the original set and the efficiency of the new set is evaluated.

→ The problem with this approach is that feature space is vast and looking at every possible combination would take a large amount of time and computation.

→ The problem is essentially reduced to a search problem. These methods are usually computationally very expensive.

1. **Forward Selection:** Forward selection is an iterative method in which we start with having no feature in the model. In each iteration, we keep adding the feature which best improves our model till an addition of a new variable does not improve the performance of the model.

2. **Backward Elimination:** In backward elimination, we start with all the features and removes the least significant feature at each iteration which improves the performance of the model. We repeat this until no improvement is observed on the removal of features.

3. **Recursive Feature Elimination (RFE):** It works by recursively removing attributes and building a model on those attributes that remain. It uses an external estimator that assigns weights to features (for example, the coefficients of a linear model) to identify which attributes (and the combination of attributes) contribute the most to predicting the target attribute.

→ It is a greedy optimization algorithm which aims to find the best performing feature subset.

→ It repeatedly creates models and keeps aside the best or the worst performing feature at each iteration.

→ It constructs the next model with the left features until all the features are exhausted. It then ranks the features based on the order of their elimination.

```
# Recursive Feature Elimination
from sklearn.feature_selection import RFE
from sklearn.linear_model import LinearRegression

# create a base classifier used to evaluate a subset of
attributes
model = LinearRegression()

X, y = iowa.iloc[:,:-1], iowa.iloc[:,-1]
# create the RFE model and select 3 attributes
rfe = RFE(model, 10)
rfe = rfe.fit(X, y)

# summarize the selection of the attributes
print(rfe.support_)
print(rfe.ranking_)
Output:
[False False  True  True False False False False False False
False False
 False False False  True  True  True False  True  True  True
True False
```

```
  True False False False False False False False False
False]
[16 24  1  1  4  9 18 13 14 15 11  6  7 12 10  1  1  1  2  1
1  1  1  5  1
 23 17 20 22 19  8 21 25  3]
```

→ Here is what is happening in the above example,

i. '**rfe.support_**' gives the result (based on the selected model and no of requirement, obviously) with respect to features sequentially.

ii. '**rfe.ranking_**' gives the rank to all features respectively. This really comes handy when you need more features than you gave input to '**n_features_to_select**' (in above eg it was 10). So, you can set a threshold value & select all the features above it respectively.

4.Sequential Feature Selector: Sequential feature selection algorithms are a family of greedy search algorithms that are used to reduce an initial *d*-dimensional feature space to a *k*-dimensional feature subspace (where $k < d$).

→ Step forward feature selection starts with the evaluation of each individual feature and selects that which results in the best performing selected algorithm model.

→ Step backward feature selection is closely related, and as you may have guessed, it starts with the entire set of features and works backwards from there, removing features to find the optimal subset of a predefined size.

→ What's the "best?"

That depends entirely on the defined evaluation criteria (AUC, prediction accuracy, RMSE, etc.). Next, all possible combinations of the that selected feature and a subsequent feature are evaluated, and a second feature is selected, and so on until the required predefined number of features is selected.

→ In a nutshell, SFAs remove or add one feature at the time based on the classifier performance until a feature subset of the desired size *k* is reached.

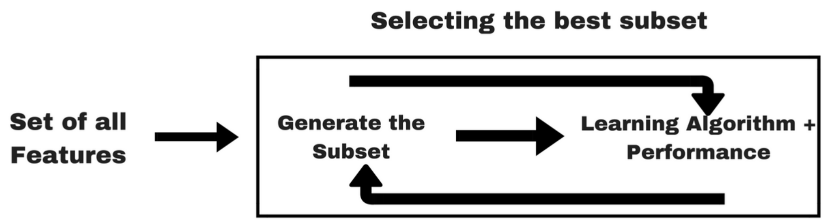
> *Note: I suggest you to visit official docs to understand it in more details with example*

Sequential Feature Selector — mlxtend

A library consisting of useful tools and extensions for the day-to-day data science tasks.

rasbt.github.io

Embbeded Method:



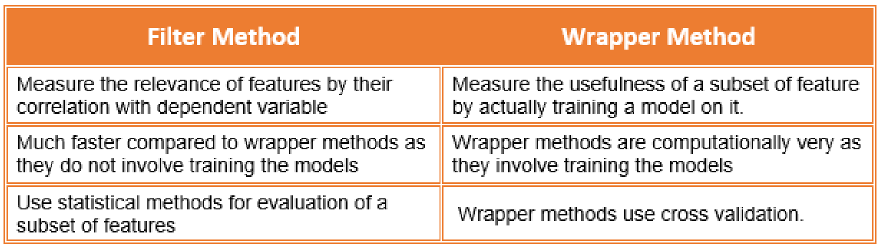
Blueprint of Embbeded Methods

→ Embedded methods combine the qualities' of filter and wrapper methods. It's implemented by algorithms that have their own built-in feature selection methods.

→ So this are not any kind of special feature selection or extraction techniques and they also help in avoiding overfitting.

1. Lasso Regularization in Linear Regression
2. Select k-best in Random Forest
3. Gradient boosting machine (GBM)

Difference between Filter and Wrapper method



Filter Method	Wrapper Method
Measure the relevance of features by their correlation with dependent variable	Measure the usefulness of a subset of feature by actually training a model on it.
Much faster compared to wrapper methods as they do not involve training the models	Wrapper methods are computationally very as they involve training the models
Use statistical methods for evaluation of a subset of features	Wrapper methods use cross validation.

. . .

Part 3: Dimension Reduction

So, again starting with the same question, What is Dimension Reduction?

In simple terms, to reduce an initial *d*-dimensional feature space to a *k*-dimensional feature subspace (where $k < d$).

So does the Feature selection & extraction then why this?

In a way, yes (but only '*In layman's term*'). To understand this we have to dive deeper.

> *In machine learning,* ***dimensionality*** *simply refers to the* ***number of features*** *(i.e. input variables) in your dataset. When the number of features is very large relative to the number of observations in your dataset,* certain *algorithms struggle to train effective models. This is called the "Curse of Dimensionality," and it's especially relevant for clustering algorithms that rely on distance calculations.*

(A Quora user has provided an excellent analogy for the Curse of Dimensionality, have a look)

So when you have let's say when you have 100s or even 1000s of features, that time you have only one choice **Dimension Reduction.** Let us discuss two extremely robust and popular techniques.

1. **Linear discriminant analysis (LDA):** Yes, along with Filter method (as discussed above) it is also used as Dimension Reduction technique.

→ We used LDA in Supervised Learning when features are labelled.

→ Please do up and understand LDA (if you had not already).

2. Principal Component Analysis(PCA): The main purposes of a PCA are the analysis of data to identify patterns and finding patterns to reduce the dimensions of the dataset with minimal loss of information.

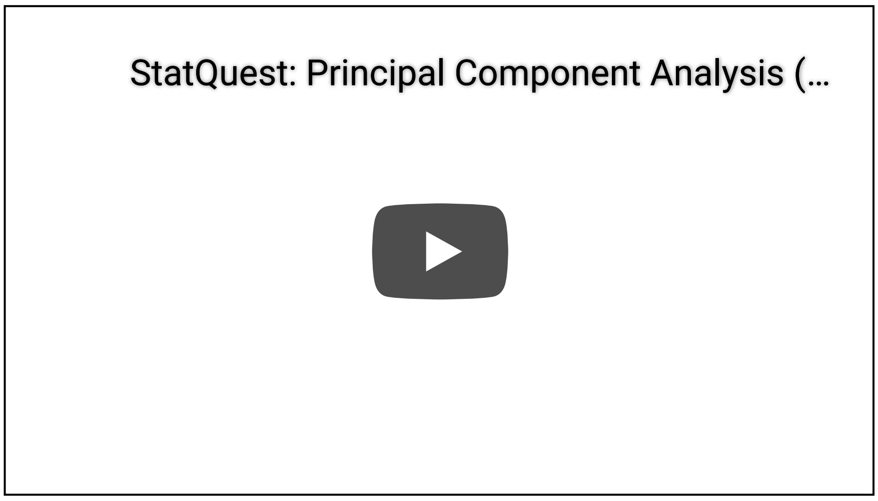
→ PCA will try to reduce dimensionality by exploring how one feature of the data is expressed in terms of the other features(linear dependency). Feature selection instead, takes the target into consideration.

→ PCA works best on dataset having 3 or higher dimensions. Because, with higher dimensions, it becomes increasingly difficult to make interpretations from the resultant cloud of data.

(PCA is a little complex, true. Explaining here will make this already lengthy blog more boring. So use this two excellent source to understand,

i. A One-Stop Shop for Principal Component Analysis

ii. Video explanation by Josh Starmer (the same guy from StatQuest)



Creator: Josh Starmer

. . .

Wrapping up: But not by the Wrapper method(Duh…😜). This comes to an end of series. Last (but not the list) minute tips:

i. Never ever ignore Feature engineering nor Feature Selection and keep everything upon the algorithm.

ii. On the endnotes, I am sharing two extremely useful and full of awesomeness tools (Mark my words, this will help you a lot)

→ ***FeatureSelector*** (A big thanks to William Koehrsen)

→ ***FeatureTools*** (or this)