Discovering the shades of Feature Selection Methods

Swetha Manoj
23 Apr, 2021 • 5 min read

This article was published as a part of the Data Science Blogathon.

Introduction

The data consists of a two-dimensional array of categorical or continuous data points for each field also called features that have a great impact on the dependent variables that need to be predicted using any Machine learning models. Mindless about the different features present in the dataset, how we make the best out of the data is what affects the performance of the forecasting model on a large basis.

So how do we exhibit our data to leverage it for a specific application? ……. Feature Engineering

Feature Engineering

Feature engineering is one of the important tasks that need to be performed in designing any data science-related problem statement because exhibiting the data in a proper way can have an enormous influence on the evaluation of the model. In other words, this method involves experimenting with all combinations of features to highlight the key ones thereby isolating those that are irrelevant. It requires domain knowledge for selecting the most appropriate features that will bring out an accurate result.

Feature Selection

Feature selection or variable selection is a cardinal process in the feature engineering technique which is used to reduce the number of dependent variables. This is achieved by picking out only those that have a paramount effect on the target attribute. By employing this method, the exhaustive dataset can be reduced in size by pruning away the redundant features that reduce the model's accuracy. Doing this will help curtail the computational expense of modeling and in some cases may also boost the accuracy of the implemented model.
For example let us consider, Heart Failure Clinical dataset containing medical records of patients who had left ventricular systolic dysfunction and with a history of heart failure.

```
heart=pd.read_csv('heart_failure_clinical_records_dataset.csv')
heart.head()
```

	age	anaemia	creatinine_phosphokinase	diabetes	ejection_fraction	high_blood_pressure	platelets	serum_creatinine	serum_sodium	sex	smoking	time
0	75.0	0	582	0	20	1	265000.00	1.9	130	1	0	4
1	55.0	0	7861	0	38	0	263358.03	1.1	136	1	0	6
2	65.0	0	146	0	20	0	162000.00	1.3	129	1	1	7
3	50.0	1	111	0	20	0	210000.00	1.9	137	1	0	7
4	65.0	1	160	1	20	0	327000.00	2.7	116	0	0	8

```
[ ]  heart.columns

Index(['age', 'anaemia', 'creatinine_phosphokinase', 'diabetes',
       'ejection_fraction', 'high_blood_pressure', 'platelets',
       'serum_creatinine', 'serum_sodium', 'sex', 'smoking', 'time',
       'DEATH_EVENT'],
      dtype='object')
```

Features to predict DEATH_EVENT

So here we have 12 features enlisted to carry out the forecasting process. Clearly using our expertise in this field we can tell that whether a person suffers from heart failure or not can be predicted using the Gender of the person. Nowadays a person irrespective of age and gender is suffering from CVD, so it may not be a key feature for prediction and hence can be isolated from training the model.
Increasing the features can make the model complex to learn and can also cause overfitting. In order to generalize the feature selection methods/models, we will look into a few feature selection techniques to reduce the dimensionality of the dataset. There are 3 basic approaches: Model-based approach(Extra-tree classifier), Iterative search (Forward stepwise selection), and Univariant statistics (Correlation and Chi-square test).
The feature selection methods we are going to discuss encompasses the following:

- Extra Tree Classifier

- Pearson correlation
- Forward selection
- Chi-square
- Logit (Logistic Regression model)

Extra Tree Classifier

This is a Model-based approach for selecting the features using the tree-based supervised models to make decisions on the importance of the features. The Extra Tree Classifier or the Extremely Random Tree Classifier is an ensemble algorithm that seeds multiple tree models constructed randomly from the training dataset and sorts out the features that have been most voted for. It fits each decision tree on the whole dataset rather than a bootstrap replica and picks out a split point at random to split the nodes.
The splitting of nodes occurring at every level of the constituent decision trees is based on the measure of randomness or entropy in the sub-nodes. The nodes are split on all variables available in the dataset and the split that results in the most homogenous sub-child is selected in the constituent tree models. This lowers the variance and makes the model less prone to overfitting.

```
X=heart.iloc[:,0:12]
Y=heart['DEATH_EVENT']
from sklearn.ensemble import ExtraTreesClassifier
model = ExtraTreesClassifier()
model.fit(X, Y)
print(model.feature_importances_)
feat_importances = pd.Series(model.feature_importances_, index=X.columns)
feat_importances.nlargest(13).plot.bar()
plt.show()
list1=feat_importances.keys().to_list()
```

Pearson Correlation

Pearson Correlation is used to construct a correlation matrix that measures the linear association between two features and gives a value between -1 and 1 indicating how related the two features are to one another. This measures the degree to which two features are interdependent by computing the association between each feature and the target variable, the one exerting high impact on the target can be picked out.

$$r = \frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2 \sum_{i=1}^{n}(y_i-\bar{y})^2}}$$

where x is the ith value of the variable x, xis the average value of sample attribute x, n is the number of records in the dataset and, x and y are the independent and target variables. A value of 1 indicates a positive correlation, -1 indicates a negative correlation and 0 indicates no correlation between the features.

```
#Correlation with the output variable
cor_target = abs(corr["DEATH_EVENT"])
#Selecting highly correlated features
relevant_features = cor_target[cor_target>0.2]
list4=relevant_features.keys().to_list()
list4
```


Chi-squared

A chi-square test is used in statistical models to check the independence of attributes. The model measures the degree of deviation between the expected and actual response. The lower the value of Chi-square, the less dependent the variables are to one another, and the higher the value more is their correlation. Initially, the attributes are assumed to be independent which forms the null hypothesis. The value of the expected outcome is computed using the following formula:

$$E_i = P(x_i \cap y_i) = P(x_i) \times P(y_i)$$

Chi-square is obtained from the expression that goes as follows:

$$\chi^2 = \sum_{i=1}^{n} \frac{O_i - E_i}{E_i}$$

Where, i is in the range (1,n), n is the number of dataset records, Oi is the actual outcome, and Eiis the expected outcome.

```
chi2_features = SelectKBest(chi2, k = 6)
X_kbest_features = chi2_features.fit_transform(X, Y)
mask=chi2_features.get_support()
new_feature=[]
for bool,feature in zip(mask,X.columns):
  if bool:
    new_feature.append(feature)
list3=new_feature
list3
```

Forward Selection

Forward selection is a wrapper model that evaluates the predictive power of the features jointly and returns a set of features that performs the best. It selects the predictors one by one and chooses that combination of features that makes the model perform the best based on the cumulative residual sum of squares. This process continues till there is no impact on the performance of the model with the incoming variable.

```
from sklearn.ensemble import RandomForestRegressor, RandomForestClassifier
from sklearn.metrics import roc_auc_score
from mlxtend.feature_selection import SequentialFeatureSelector
forward_feature_selector = SequentialFeatureSelector(RandomForestClassifier(n_jobs=-1),
          k_features=6,
          forward=True,
          verbose=2,
          scoring='roc_auc',
          cv=4)
fselector = forward_feature_selector.fit(X,Y)
```

Logit Model

This is the Logistic regression-based model which selects the features based on the p-value score of the feature. The features with p-value less than 0.05 are considered to be the more relevant feature.

```
import statsmodels.api as sm
logit_model=sm.Logit(Y,X)
result=logit_model.fit()
print(result.summary2())
```

```
                          Results: Logit
==========================================================================
Model:                  Logit              Pseudo R-squared:   0.406
Dependent Variable:     DEATH_EVENT        AIC:                246.9033
Date:                   2021-03-23 06:25   BIC:                291.3086
No. Observations:       299                Log-Likelihood:     -111.45
Df Model:               11                 LL-Null:            -187.67
Df Residuals:           287                LLR p-value:        4.7149e-27
Converged:              1.0000             Scale:              1.0000
No. Iterations:         7.0000
--------------------------------------------------------------------------
                           Coef.  Std.Err.    z    P>|z|   [0.025  0.975]
--------------------------------------------------------------------------
age                         0.0522   0.0154  3.3809 0.0007  0.0219  0.0825
anaemia                    -0.0368   0.3568 -0.1032 0.9178 -0.7361  0.6625
creatinine_phosphokinase    0.0002   0.0002  1.2257 0.2203 -0.0001  0.0006
diabetes                    0.2462   0.3448  0.7141 0.4751 -0.4296  0.9221
ejection_fraction          -0.0757   0.0160 -4.7236 0.0000 -0.1071 -0.0443
high_blood_pressure        -0.0771   0.3554 -0.2170 0.8282 -0.7737  0.6194
platelets                  -0.0000   0.0000 -0.6052 0.5450 -0.0000  0.0000
serum_creatinine            0.7226   0.1801  4.0114 0.0001  0.3695  1.0756
serum_sodium                0.0033   0.0092  0.3551 0.7225 -0.0147  0.0212
sex                        -0.4449   0.4056 -1.0969 0.2727 -1.2398  0.3500
smoking                    -0.0165   0.4085 -0.0405 0.9677 -0.8172  0.7841
time                       -0.0204   0.0029 -6.9462 0.0000 -0.0262 -0.0147
==========================================================================
```

Now we have applied different feature selection models to our dataset, we will see the features selected by each module,

Model	Features
Extra Tree classifier	time, serum_creatinine, ejection_fraction, age, serum_sodium
Pearson Correlation	age, ejection_fraction, serum_creatinine, time
Chi-squared test	age, creatinine_phosphokinase, ejection_fraction, platelets, serum_creatinine, time
Forward selection	age, creatinine_phosphokinase, ejection_fraction, high_blood_pressure, serum_creatinine, smoking
Logit model	age, ejection_fraction, serum creatinine, time

Now that the features have been selected, we are good to apply any supervised classification models to predict the outcome.
You can find the full code of this project here.
Happy learning !!

You can reach out to me through LinkedIn

The media shown in this article on feature selection methods are not owned by Analytics Vidhya and is used at the Author's discretion.