Feature Importance

Definition:

Feature importance is a step in building a machine learning model that involves calculating the score for all input features in a model to establish the importance of each feature in the decision-making process.

The higher the score for a feature, the larger effect it has on the model to predict a certain variable.

Benefits:

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.

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.

Different techniques:

Coefficients

When we fit a generalized linear model (for example, a linear or logistic regression), we estimate coefficients for each predictor.

If the original features were standardized, these coefficients can be used to estimate relative feature importance; larger absolute value coefficients are more important.

This method is computationally inexpensive because coefficients are calculated when we fit the model. It is also useful for both classification and regression problems (i.e., categorical and continuous outcomes).

However, similar to the other methods described above, these coefficients do not take highly correlated features into account.

Gini impurity

Gini impurity is related to the extent to which observations are well separated based on the outcome variable at each node of the decision tree.
High Gini Gain considered as important feature

Permutation-based methods

Another way to test the importance of particular features is to essentially remove them from the model (one at a time) and see how much predictive accuracy suffers. One way to "remove" a feature is to randomly permute the values for that feature, then refit the model. This can be implemented with any machine learning model, including non-tree-based- methods.

This feature selection model to overcome from over fitting which is most common among **tree** based feature selection techniques.

It is Best for those algorithms which natively does not support feature importance . It calculate relative importance score **independent of model used.**It is one of the best technique to do feature selection.lets' understand it

Cons:

However, one potential drawback is that it is computationally expensive because it requires us to refit the model many times.

Steps we are following in Permutation Feature Importance:

Step 1 : - It randomly take one feature and shuffles the variable present in that feature and does prediction

Step 2 :- In this step it finds the loss using loss function and checks the variability between predicted and actual output.

Step 3:- Returns the variable of feature into original order or undo reshuffle.

Step 4 :- Does the above three procedures with all the features present in the dataset.

Step 5 :- Final important features will be calculated by comparing individual score with mean importance score.

Dataset:

```
# print the updated data
df2
```

	age	bp	al	su	bgr	bu	sc	sod	pot	hrmo	...	pc_normal	pcc_present	ba_present	htn_yes
0	2.000000	76.459948	3.0	0.0	148.112676	57.482105	3.077356	137.528754	4.627244	12.518156	...	False	False	False	False
1	3.000000	76.459948	2.0	0.0	148.112676	22.000000	0.700000	137.528754	4.627244	10.700000	...	True	False	False	False
2	4.000000	76.459948	1.0	0.0	99.000000	23.000000	0.600000	138.000000	4.400000	12.000000	...	True	False	False	False
3	5.000000	76.459948	1.0	0.0	148.112676	16.000000	0.700000	138.000000	3.200000	8.100000	...	True	False	False	False
4	5.000000	50.000000	0.0	0.0	148.112676	25.000000	0.600000	137.528754	4.627244	11.800000	...	True	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
394	51.492308	70.000000	0.0	0.0	219.000000	36.000000	1.300000	139.000000	3.700000	12.500000	...	True	False	False	False
395	51.492308	70.000000	0.0	2.0	220.000000	68.000000	2.800000	137.528754	4.627244	8.700000	...	True	False	False	True
396	51.492308	70.000000	3.0	0.0	110.000000	115.000000	6.000000	134.000000	2.700000	9.100000	...	True	False	False	True
397	51.492308	90.000000	0.0	0.0	207.000000	80.000000	6.800000	142.000000	5.500000	8.500000	...	True	False	False	True
398	51.492308	80.000000	0.0	0.0	100.000000	49.000000	1.000000	140.000000	5.000000	16.300000	...	True	False	False	False

Here, we are passing k values like 3, 4, 5, 6 and the permutation importance will select best features for each algorithm based on k values and we enabled the dataset by picking the unique best features from all the algorithm's featuresánd finally providing the best features that are suitable for all the algorithms

Classification Analysis using permutation feature importance:

```
# get all the accuracy of each algorithm and print it in a dataframe
result=fi_classification(acclog,accsvml,accsvmnl,accknn,accnav,accdes,accrf)
```

```
# result of  k = 4
result
```

	Logistic	SVMl	SVMnl	KNN	Navie	Decision	Random
Accuracy	0.96	0.96	0.95	0.96	0.9	0.92	0.94

```
# result of  k = 5
result
```

	Logistic	SVMl	SVMnl	KNN	Navie	Decision	Random
Accuracy	0.98	0.98	0.98	0.99	0.9	0.98	0.99

```
# result of  k = 6
result
```

	Logistic	SVMl	SVMnl	KNN	Navie	Decision	Random
Accuracy	0.97	0.97	0.98	0.97	0.9	0.89	0.99

```
# result of  k = 7
result
```

	Logistic	SVMl	SVMnl	KNN	Navie	Decision	Random
Accuracy	0.98	0.97	0.99	0.98	0.91	0.91	0.97

As per the analysis the accuracy is high when we select top 5 features for all the algorithms

Regression Analysis using permutation feature importance:

```
# result of  k = 3
result
```

	Linear	SVMl	SVM_NL	Decision	Random
R2 score	0.705335	0.667254	0.931694	0.826389	0.866753

```
# result of  k = 4
result
```

	Linear	SVMl	SVM_NL	Decision	Random
R2 score	0.704799	0.674446	0.926379	0.869792	0.885417

```
# result of  k = 5
result
```

	Linear	SVMl	SVM_NL	Decision	Random
R2 score	0.705372	0.672303	0.944377	0.956597	0.863715

```
# result of  k = 6
result
```

	Linear	SVMl	SVM_NL	Decision	Random
R2 score	0.713986	0.679654	0.93555	0.956597	0.858941

As per the analysis the R2 score value is high when we select top 5 features for each algorithms

In conclusion, KNN and Random forecast are having 99% of accuracy by using the features ['pcv', 'bp', 'sod', 'bu', 'hrmo', 'bgr', 'sc', 'al'] out of top 5 features for each algorithms

Decision Tree is having 95% of accuracy by using the features ['sg_c', 'hrmo', 'al', 'sg_b','htn_yes', 'sc', 'bgr', 'dm_yes', 'pcv','rc', 'pc_normal', 'sg_d'] out of top 5 features for each algorithms