vaksakalli Add files via upload

1 contributor

192 lines (192 sloc) | 8.3 KB

Practice Exercise: Scikit-Learn 2

Feature Selection and Ranking

Objectives

As in the SK2 Tutorial (https://www.featureranking.com/tutorials/machine-learning-tutorials/sk-part-2-feature-selection-and-ranking/), the goal of this practice notebook is to illustrate how you can perform feature selection (FS) and ranking using the relevant methods within `Scikit-Learn`. You will be using the cleaned "income data" from previous data preparation practices you will take a cross-validation (CV) approach.

In the previous practices, you cleaned and transformed the raw `income data` and renamed the `income` column as `target` (with high income being the positive class). Including `target`, the cleaned data consists of 42 columns and 45,222 rows. Each column is numeric and between 0 and 1.

For FS methods other than the ones illustrated here, please refer to the official `Scikit-Learn` documentation here (https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectKBest.html#sklearn.feature_selection.Sel

Machine Learning: The Art and The Science

If you notice things in our practice exercises that are different from those in the corresponding tutorials, it's OK. There are usually multiple ways of doing the right thing in ML, and more importantly, always keep in mind that machine learning is as much **art** as it is **science**!

Instructions

- For all the questions below, you will use **stratified 5-fold cross-validation with no repetitions** and set the random state to 999 when applicable.
- As the wrapper (that is, the intended classifier), you will use a decision tree with a `max_depth` of 5. Don't forget to set the `random_state` so that your results do not change from run to run.
- For ease of computation, you will first randomly sample 5,000 rows from this dataset.
- You will then split this sample into two equal-sized datasets.
- You will use the first set for **training**: you will find the best 10-features using different methods using this train data.
- You will use the second set for **testing**: you will perform cross-validation in a paired fashion using the test data and then you will compare the results using a paired t-test.
- For scoring, you will use AUC, that is, "area under the ROC curve".

Hint: For a list of scorers as a **string** that you can pass into `cross_val_score()` or `GridSearchCV()` methods, please try this:

```
from sklearn import metrics
metrics.SCORERS.keys()
```

Some Bookkeeping

- Define a variable called `num_samples` and set it to 5000. You will use this variable when sampling a smaller subset of the full set of instances.
- Define a variable called `num_features` and set it to 10. You will perform all feature selection tasks by making use of this `num_features` variable.
- Define a variable called `scoring_metric` and set it to to `'roc_auc'`. You will set `scoring` option in all `cross_val_score()` functions to this `scoring_metric` variable.
- Define an object called `clf` and set its value to `DecisionTreeClassifier(max_depth=5, random_state=999)`. You will use this classifier as your wrapper when comparing performance of feature selection (FS) methods.

You can achieve these by running the code chunk below:

```
import numpy as np
num_samples = 5000
num_features = 10
scoring_metric = 'roc_auc'
from sklearn.tree import DecisionTreeClassifier
clf = DecisionTreeClassifier(max_depth=5, random_state=999)
```

Exercise 0: Modeling Preparation

- Read in the clean data `us_census_income_data_clean_encoded.csv` on GitHub here (https://github.com/vaksakalli/datasets).
- Randomly sample the rows.
- Split the sampled data as 50% training set and the remaining 50% test set using a random seed of 999.
- Remember to separate `target` during the splitting process.

Exercise 1

Assess the cross-validated performance of your DT classifier using the **test** data with all the features.

Exercise 2

- Select the top 10 features via the **F-Score** method using the **train** data.
- Evaluate the cross-validated performance of these features using your DT classifier on the **test** data.

NOTE: For this particular dataset, the F-Score will be "NaN" for one of the features due to some technical reasons (related to the nature of the F-distribution). For this reason, when you pass the `fs_fit_fscore.scores_` object in to the `np.argsort()` function, you will need to apply the `np.nan_to_num()` function first. This way, you will convert that "NaN" value to zero for a correct result. Specifically, you will need the following line:

```
fs_indices_fscore = np.argsort(np.nan_to_num(fs_fit_fscore.scores_))[::-1
```

```
][0:num_features]
```

Exercise 3

- Select the top 10 features using the **Mutual Information** method using the **train** data.
- Evaluate the cross-validated performance of these features using your DT classifier on the **test** data.