Preprocessing Data for Machine Learning

Many machine learning algorithms rely on the quality of the features to perform well. One approach to improve existing features is to preprocess and normalize them before feeding them into the machine learning algorithms. In this lab, you are going to get familiar with a few popular ways to improve your features.

Prerequisites:

It is assumed that you will install and take a look at the following packages in python before heading to activities:

- sklearn (http://scikit-learn.org/stable/)
- (http://flask.pocoo.org/) pandas (https://pandas.pydata.org/)

This lab makes use of the exposure dataset (https://www.kaggle.com/mpwolke/cusersmarildownloadsexposurecsv) .

Activity-1:

Description : Encoding categorical features by OrdinalEncoder. The 'GHRP' column is a categorical column that can be "No", "Yes", or null. Encode this column with OrdinalEncoder and build a machine learning model to detect a country's 'Income classification according to WB'.

Steps :

1. Load the exposure dataset (use the right eliminator and encoding)
2. Keep the following columns in the data frame and ignore the rest

```
'GHRP',
'Aid dependence',
'Remittances',
'food import dependence ',
'primary commodity export dependence',
'tourism as percentage of GDP',
'tourism dependence',
'Foreign currency reserves',
'Foreign direct investment, net inflows percent of GDP',
'Foreign direct investment',
'Covid_19_Economic_exposure_index',
'Income classification according to WB'
```

3. Remove the rows without values for "Income classification according to WB".
4. Fill the null values in the "GHRP" column with the "Unknown" keyword

5. Clean the numeric columns by: (1) replacing 'x' values with "numpy.nan", (2) replacing "," in all numeric columns with floating-point (".").
6. Use "OrdinalEncoder" to convert the "GHRP" into a numeric column (https://scikit-learn.org/stable/modules/preprocessing.html#encoding-categorical-features)
7. Fill all nan with 0 in the dataframe
8. Make assure that the datatype for all numeric columns is float
9. Split the dataset into test and train datasets; 70% of the data should be used for training the classifier and the rest for testing
10. Train a DecisionTreeClassifier to detect a country's Income Level (Income classification according to WB), and report precision, recall, and accuracy of the trained model on the testing dataset

 (https://github.com/mysilver/COMP9321-Data-Services/blob/master/Week11_Preprocessing/activity_1.py)

Activity-2:

Description : Dealing with missing values. In this activity, you are going to deal with null values by replacing them with proper values (e.g., the most frequent value, a constant value) to improve your machine learning models

Steps :

1. Follow the first six steps mentioned in Activity-1.
2. In Activity-1, you filled the nan values with zero; in this activity, you should replace the nan values in all numeric columns with the following cases using SimpleImputer (https://scikit-learn.org/stable/modules/impute.html#impute) :

```
"mean": the average of values in the column, ignoring the NaN values
"median": the median value, ignoring the NaN values
"most_frequent": the most frequent value, ignoring the NaN values
```

3. For each case, split the dataset into test and train datasets; 70% of the data should be used for training the classifier and the rest for testing, train a DecisionTreeClassifier to detect a country's Income Level (Income classification according to WB), and report precision, recall, and accuracy of the trained model on the testing dataset
4. Which case does improve the accuracy of your model the most?

    (https://github.com/mysilver/COMP9321-Data-Services/blob/master/Week11_Preprocessing/activity_2.py)

Activity-3:

Description : Adding nonlinear features of the input data. A simple method to add nonlinearity to the features is to use polynomial features (deriving new features from the existing features)

Steps :

1. Follow the first six steps mentioned in Activity-1.
2. Use the most effective imputer as you found out in Activity-2 to fill the missing values in numeric columns

3. Use PolynomialFeatures to add complexity to your feature set. (https://scikit-learn.org/stable/modules/preprocessing.html#generating-polynomial-features) (https://scikit-learn.org/stable/modules/preprocessing.html#generating-polynomial-features)
4. Split the dataset into test and train datasets; 70% of the data should be used for training the classifier and the rest for testing, train a DecisionTreeClassifier to detect a country's Income Level (Income classification according to WB), and report precision, recall, and accuracy of the trained model on the testing dataset
5. Tune the parameters of the PolynomialFeatures (https://scikit-learn.org/stable/modules/preprocessing.html#generating-polynomial-features) untill your new model outperforms the one you build in Activity-2.

 (https://github.com/mysilver/COMP9321-Data-Services/blob/master/Week11_Preprocessing/activity_3.py)

Resource created about a month ago (Monday 14 March 2022, 03:06:06 PM), last modified 21 days ago (Tuesday 05 April 2022, 03:31:17 PM).

Comments

(/COMP9321/22T1/forums/search?forum_choice=resource/74092)

(/COMP9321/22T1/forums/resource/74092)

Add a comment

There are no comments yet.