feature_selection

Guangyao Zhao

2022-11-12

Contents

特征选择的方法主要有三种:

- Filter method: 通过一定的度量方法来计算出数据集中的每一个 feature 的 score，然后根据 score 的大小来筛选 feature。比如 Pearson correlation coefficient 和决策树中的 Information gain 或者 Gini。
- Wrapper method: 将数据集中的特征进行组合，然后用模型对特征组合进行评估，选出最优。
- Embedded method: 不同于之前的两种特征筛选方法，此方法是在模型训练的过程中隐式地选择特征，比如正则化 Lasso and Ridge。

本文的特征选择在一定程度上可以提取出一些有用的特征，但是在我的实际使用中，更推荐 decomposition 的手段。

```
import sklearn.datasets as ds
import numpy as np
from sklearn import linear_model
from sklearn.feature_selection import VarianceThreshold, SelectKBest, f_regression, mutual_info_regress
from sklearn.svm import SVR
```

移除低方差特征

方差代表着信息差，如果一个特征方差很小，则表明该特征的信息差很小，即倾向于是一个对于机器学习用处不大的特征。该方法是特征选择的一种基本方法，可以删除所有不满足某些阈值的特征。

```
X = [[0, 0, 1], [0, 1, 0], [1, 0, 0], [0, 1, 1], [0, 1, 0], [0, 1, 1]]
X = VarianceThreshold(threshold=(0.8 * (1 - 0.8))).fit_transform(X)

print('X:\n ', X)
```

```
X:
  [[0 1]
 [1 0]
 [0 0]
 [1 1]
 [1 0]
 [1 1]]
```

单变量特征选择

单变量特征选择是基于单变量统计检验选择最优特征实现的，在此介绍 SelectKBest 和 SelectPercentile，前者是指定特定的数值，后者则是百分位数，原理一样。

SelectKBest 和 SelectPercentile 可以选择一系列评价指标，比如回归中可用的：

- f_regression: 根据 F 值选择重要特征
- mutual_info_regression: 根据互信息选择重要特征

```
X, y = ds.fetch_california_housing(return_X_y=True)

print('X.shape: ', X.shape)

f_reg = SelectKBest(score_func=f_regression, k=3).fit(X, y)
# f_reg = SelectKBest(score_func=mutual_info_regression, k=3).fit(X, y)
# f_reg = SelectPercentile(score_func=f_regression,percentile=30).fit(X, y) # 百分比

X_new = f_reg.transform(X)
```

```
print('X_new.shape: ', X_new.shape)

f_scores = f_reg.scores_  # F value
f_pval = f_reg.pvalues_  # P value

print('f_scores: ', f_scores)
print('f_pval: ', f_pval)
```

```
X.shape:  (20640, 8)
X_new.shape:  (20640, 3)
f_scores:  [1.85565716e+04 2.32841479e+02 4.87757462e+02 4.51085756e+01
 1.25474103e+01 1.16353421e+01 4.38005453e+02 4.36989761e+01]
f_pval:  [0.00000000e+000 2.76186068e-052 7.56924213e-107 1.91258939e-011
 3.97630785e-004 6.48344237e-004 2.93985929e-096 3.92332207e-011]
```

递归特征消除 (Recursive feature elimination, RFE)

给定将权重分配给特征（例如线性模型的系数）的外部估计器。

递归特征消除的主要思想是反复构建模型，然后选出最好的（或者最差的）特征（根据系数来选），把选出来的特征放到一边，然后在剩余的特征上重复这个过程，直到遍历了所有的特征。在这个过程中被消除的次序就是特征的排序。

```
X, y = ds.fetch_california_housing(return_X_y=True)

X = X[:100, :]
y = y[:100]
print('X.shape: ', X.shape)

f_imp = RFE(estimator=SVR(kernel='linear'), n_features_to_select=3,
            step=1).fit(X, y)

X_new = f_imp.transform(X)
print('X_new.shape: ', X_new.shape)

f_ranking = f_imp.ranking_  # 特征排序，使 ranking_[i] 对应第 i 个特征的排序位置。选择的（即估计的最佳）特征
```

```
f_support = f_imp.support_  # 所选特征的掩码。

print('f_ranking: ', f_ranking)
print('f_support: ', f_support)
```

```
X.shape:  (100, 8)

X_new.shape:  (100, 3)
f_ranking:  [1 5 2 3 6 4 1 1]
f_support:  [ True False False False False False  True  True]
```