day68-filter-method

December 18, 2023

Day 68 Filter Method By: Loga Aswin

[19]:
```
import pandas as pd
from sklearn.feature_selection import SelectKBest, f_classif,␣
 ↪mutual_info_classif, chi2
from sklearn.datasets import load_breast_cancer  # Changed dataset to breast␣
 ↪cancer

# Load breast cancer dataset
breast_cancer = load_breast_cancer()
df = pd.DataFrame(breast_cancer.data, columns=breast_cancer.feature_names)
target = breast_cancer.target
```

Various Methods Used in Filter Method

1. Correlation-based Feature Selection: Identifies top features based on their correlation with the target variable.

[13]:
```
corr_scores = df.corrwith(pd.Series(target, name="Target")).abs().
 ↪sort_values(ascending=False)
k_corr = 3
selected_corr = corr_scores.index[:k_corr]

print("Top features:")
print(selected_corr)
```

```
Top features:
Index(['worst concave points', 'worst perimeter', 'mean concave points'],
dtype='object')
```

2. Mutual Information-based Feature Selection: Selects features with the highest mutual information with the target.

[14]:
```
k_mi = 3
selected_mi = SelectKBest(score_func=mutual_info_classif, k=k_mi).fit(df,␣
 ↪target)
selected_mi = df.columns[selected_mi.get_support()]

print("Top features:")
print(selected_mi)
```

```
Top features using mutual information-based selection:
Index(['worst radius', 'worst perimeter', 'worst area'], dtype='object')
```

3. Chi-square Test:Uses chi-square test to find features most related to the target in categorical data.

[15]:
```
chi2_feat = SelectKBest(chi2, k=3)
X_kbest = chi2_feat.fit_transform(df, target)

print("Shape before and after chi-square test:")
print(df.shape)
print(X_kbest.shape)
```

```
Shape before and after chi-square test:
(569, 30)
(569, 3)
```

4. Fisher's Score: Utilizes Fisher's Score to pick the most discriminative features for classification.

[20]:
```
k_fisher = 2
fisher_selector = SelectKBest(score_func=f_classif, k=k_fisher)
X_new = fisher_selector.fit_transform(df, target)

# get indices
sel_indices = fisher_selector.get_support(indices=True)
selected_fisher = [breast_cancer.feature_names[i] for i in sel_indices]

print("Top features using Fisher's Score:")
print(selected_fisher)
```

```
Top features using Fisher's Score:
['worst perimeter', 'worst concave points']
```

5. Missing Value Ratio:

Filters features based on a threshold for the ratio of missing values.

[21]:
```
from sklearn.impute import SimpleImputer

thresh_missing = 0.3
missing_ratio = df.isnull().mean()

selected_missing = df.columns[missing_ratio < thresh_missing]

# Impute missing values if needed
imputer = SimpleImputer(strategy='mean')
df[selected_missing] = imputer.fit_transform(df[selected_missing])

print("Selected features after handling missing values:")
print(selected_missing)
```

```
Selected features after handling missing values:
Index(['mean radius', 'mean texture', 'mean perimeter', 'mean area',
       'mean smoothness', 'mean compactness', 'mean concavity',
       'mean concave points', 'mean symmetry', 'mean fractal dimension',
       'radius error', 'texture error', 'perimeter error', 'area error',
       'smoothness error', 'compactness error', 'concavity error',
       'concave points error', 'symmetry error', 'fractal dimension error',
       'worst radius', 'worst texture', 'worst perimeter', 'worst area',
       'worst smoothness', 'worst compactness', 'worst concavity',
       'worst concave points', 'worst symmetry', 'worst fractal dimension'],
      dtype='object')
```