Filter Methods

May 15, 2021

[2]:
```
import pandas as pd
```

[3]:
```
dataset = pd.read_csv('dataset.csv')
X= dataset.drop(columns='Result')
Y= dataset['Result']
X.head()
```

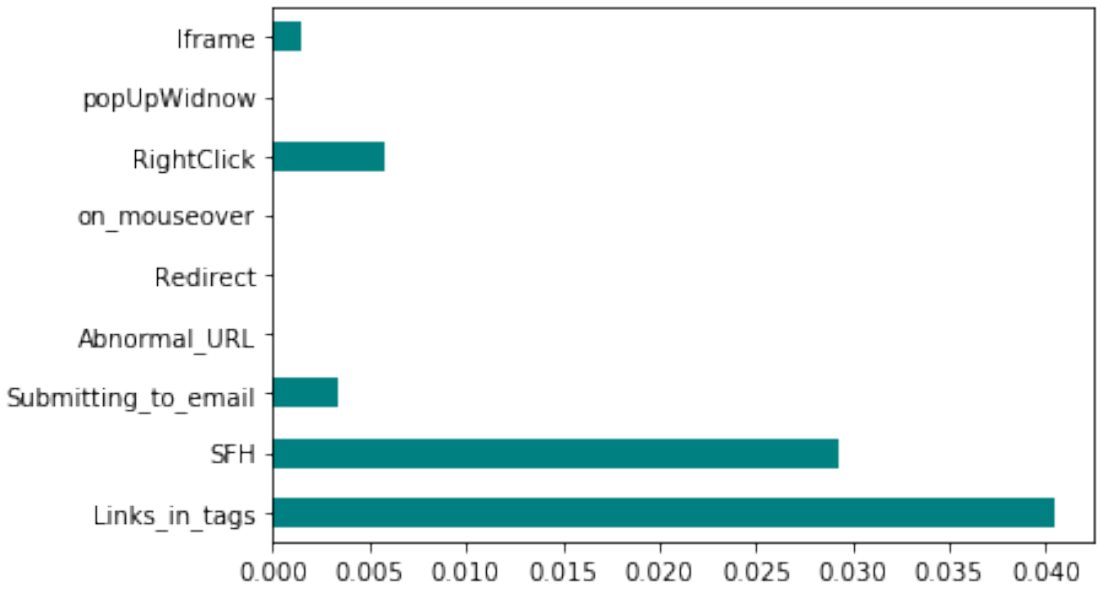
[3]:

	Links_in_tags	SFH	Submitting_to_email	Abnormal_URL	Redirect	on_mouseover	RightClick	popUpWidnow	Iframe
0	1	-1	-1	-1	0	1	1	1	1
1	-1	-1	1	1	0	1	1	1	1
2	-1	-1	-1	-1	0	1	1	1	1
3	0	-1	1	1	0	1	1	1	1
4	0	-1	1	1	0	-1	1	-1	1

[4]:
```
# Information Gain
from sklearn.feature_selection import mutual_info_classif
import matplotlib.pyplot as plt
%matplotlib inline
```

[5]:
```
importances = mutual_info_classif(X,Y)
feature_importances = pd.Series(importances, dataset.columns[0:len(dataset.
 ↪columns)-1])
feature_importances.plot(kind='barh', color='teal')
```

[5]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x24dfcd6cd88>
```



[6]:
```
# Chi-square test
# from sklearn.feature_selection import SelectKBest
# from sklearn.feature_selection import chi2
```

[7]:
```
# convert into catoagorical data by converting data into integers
# X_cat = X.astype(int)

#Three features with highest chi-sqaured statistics are selected
# chi2_features = SelectKBest(chi2, k=3)
# X_kbest_features = chi2_features.fit_transform(X_cat,Y)

# this method is not suitable for our dataset, because it cannot work with␣
 ↪negetive values
```

[9]:
```
# Fisher's Score
# from skfeature.function.similarity_based import fisher_score
# import pandas as pd
# from sklearn.model_selection import train_test_split


# dataset = pd.read_csv('dataset.csv')
# X= dataset.drop(columns='Result')
# Y= dataset['Result']
# X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.2)


# # calculating score
```

```
# # ranks = fisher_score.fisher_score(X, Y)
# score = fisher_score.fisher_score(X_train, Y_train)
# score

# rank features in descending order according to score
# idx = fisher_score.feature_ranking(score)


# plotting the ranks
# feature_imprtances = pd.series(idx, dataset.columns[0:len(dataset.columns)-1])
# feature_imprtances.plot(kind='barh', color='teal')
# plt.show()
```

```
␣
↪---------------------------------------------------------------------------

       ValueError                                Traceback (most recent call␣
↪last)

       <ipython-input-9-d5718c360bb9> in <module>
        13 # calculating score
        14 # ranks = fisher_score.fisher_score(X, Y)
   ---> 15 score = fisher_score.fisher_score(X_train, Y_train)
        16 score
        17


       c:
↪\users\lakru\appdata\local\programs\python\python37\lib\site-packages\skfeature\function\sim
↪py in fisher_score(X, y, mode)
        46     t2 = np.transpose(np.dot(Xt, L.todense()))
        47     # compute the numerator of Lr
   ---> 48     D_prime = np.sum(np.multiply(t1, X), 0) - np.multiply(tmp, tmp)/
↪D.sum()
        49     # compute the denominator of Lr
        50     L_prime = np.sum(np.multiply(t2, X), 0) - np.multiply(tmp, tmp)/
↪D.sum()


       c:
↪\users\lakru\appdata\local\programs\python\python37\lib\site-packages\pandas\core\ops\common
↪py in new_method(self, other)
        62         other = item_from_zerodim(other)
        63
   ---> 64         return method(self, other)
        65
```

```
     66     return new_method


c:
↪\users\lakru\appdata\local\programs\python\python37\lib\site-packages\pandas\core\ops\__init
↪py in wrapper(left, right)
    500         result = arithmetic_op(lvalues, rvalues, op, str_rep)
    501
--> 502         return _construct_result(left, result, index=left.index,␣
↪name=res_name)
    503
    504     wrapper.__name__ = op_name


c:
↪\users\lakru\appdata\local\programs\python\python37\lib\site-packages\pandas\core\ops\__init
↪py in _construct_result(left, result, index, name)
    473     # We do not pass dtype to ensure that the Series constructor
    474     #  does inference in the case where `result` has object-dtype.
--> 475     out = left._constructor(result, index=index)
    476     out = out.__finalize__(left)
    477


c:
↪\users\lakru\appdata\local\programs\python\python37\lib\site-packages\pandas\core\series.
↪py in __init__(self, data, index, dtype, name, copy, fastpath)
    290                     if len(index) != len(data):
    291                         raise ValueError(
--> 292                             f"Length of passed values is␣
↪{len(data)}, "
    293                             f"index implies {len(index)}."
    294                         )


ValueError: Length of passed values is 1, index implies 9.
```

[9]:

[9]:

	Links_in_tags	SFH	Submitting_to_email	Abnormal_URL	Redirect	\
0	1	-1	-1	-1	0	
1	-1	-1	1	1	0	
2	-1	-1	-1	-1	0	
3	0	-1	1	1	0	
4	0	-1	1	1	0	

```
   on_mouseover  RightClick  popUpWidnow  Iframe
0             1           1            1       1
1             1           1            1       1
2             1           1            1       1
3             1           1            1       1
4            -1           1           -1       1
```

[15]:
```
# calculating score
# ranks = fisher_score.fisher_score(X, Y)
# ranks

# plotting the ranks
# feature_imprtances = pd.series(ranks, dataset.columns[0:len(dataset.
 ↪columns)-1])
# feature_imprtances.plot(kind='barh', color='teal')
# plt.show()
```

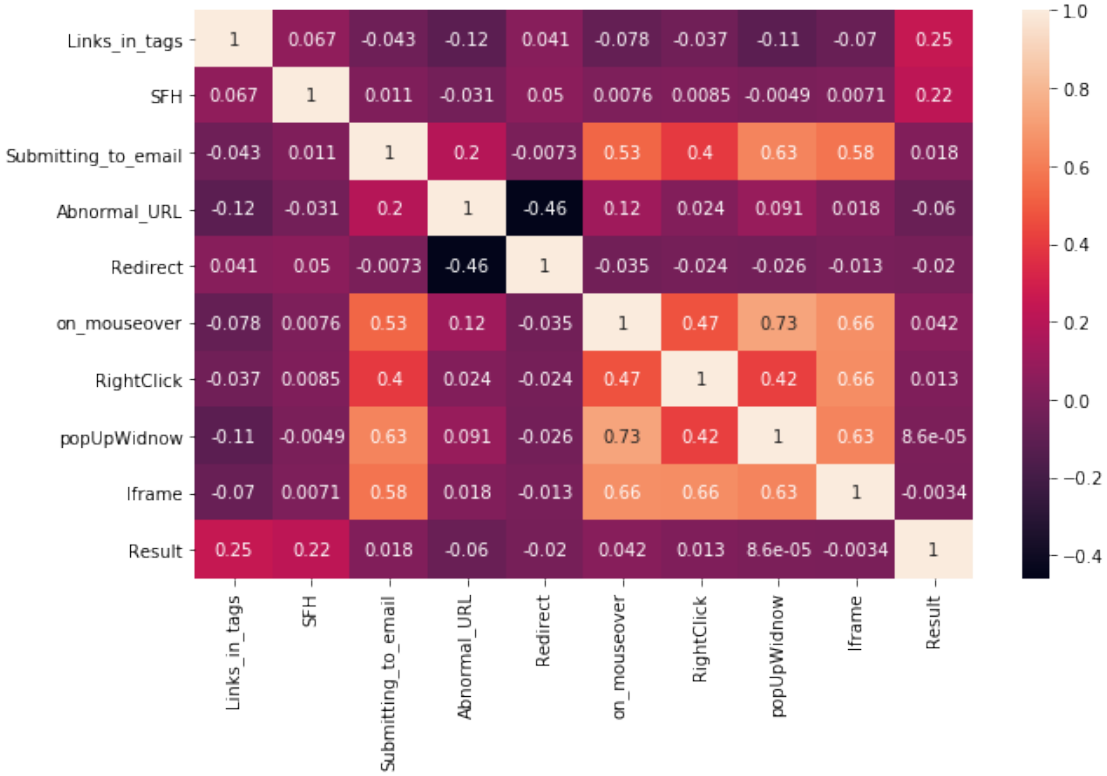
```
↪----------------------------------------------------------------------------

       ValueError                                Traceback (most recent call␣
↪last)

       <ipython-input-15-3cd1dd23521e> in <module>
         1 # calculating score
   ----> 2 ranks = fisher_score.fisher_score(X, Y)
         3 # ranks
         4
         5 # plotting the ranks


       c:
↪\users\lakru\appdata\local\programs\python\python37\lib\site-packages\skfeature\function\sim
↪py in fisher_score(X, y, mode)
        46     t2 = np.transpose(np.dot(Xt, L.todense()))
        47     # compute the numerator of Lr
   ---> 48     D_prime = np.sum(np.multiply(t1, X), 0) - np.multiply(tmp, tmp)/
↪D.sum()
        49     # compute the denominator of Lr
        50     L_prime = np.sum(np.multiply(t2, X), 0) - np.multiply(tmp, tmp)/
↪D.sum()


       c:
↪\users\lakru\appdata\local\programs\python\python37\lib\site-packages\pandas\core\ops\common
↪py in new_method(self, other)
```

```
        62         other = item_from_zerodim(other)
        63
   ---> 64         return method(self, other)
        65
        66     return new_method


      c:
↪\users\lakru\appdata\local\programs\python\python37\lib\site-packages\pandas\core\ops\__init
↪py in wrapper(left, right)
      500         result = arithmetic_op(lvalues, rvalues, op, str_rep)
      501
   --> 502         return _construct_result(left, result, index=left.index,␣
↪name=res_name)
      503
      504     wrapper.__name__ = op_name


      c:
↪\users\lakru\appdata\local\programs\python\python37\lib\site-packages\pandas\core\ops\__init
↪py in _construct_result(left, result, index, name)
      473     # We do not pass dtype to ensure that the Series constructor
      474     #  does inference in the case where `result` has object-dtype.
   --> 475     out = left._constructor(result, index=index)
      476     out = out.__finalize__(left)
      477


      c:
↪\users\lakru\appdata\local\programs\python\python37\lib\site-packages\pandas\core\series.
↪py in __init__(self, data, index, dtype, name, copy, fastpath)
      290                     if len(index) != len(data):
      291                         raise ValueError(
   --> 292                             f"Length of passed values is␣
↪{len(data)}, "
      293                             f"index implies {len(index)}."
      294                         )


      ValueError: Length of passed values is 1, index implies 9.
```

[19]:
```
# Correlation coefficient
import seaborn as sns

# correlation matrix
cor = dataset.corr()
```

```
# plotting heatmap
plt.figure(figsize=(10,6))
sns.heatmap(cor, annot = True)
```

[19]: <matplotlib.axes._subplots.AxesSubplot at 0x24f2d8fbe88>



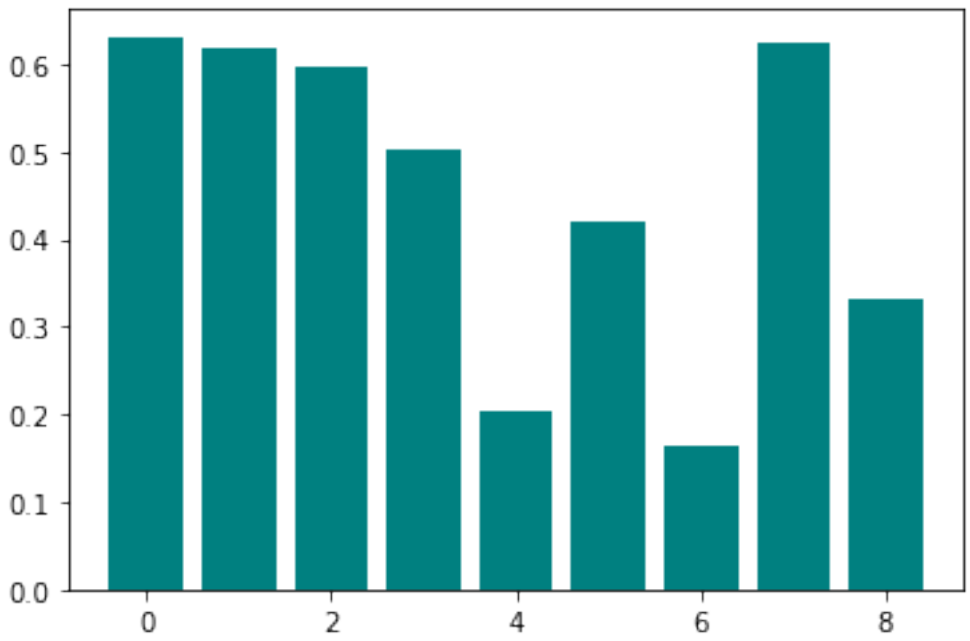
[5]:
```
# Detecting Multicollinearity with VIF
from statsmodels.stats.outliers_influence import variance_inflation_factor
import pandas as pd


dataset = pd.read_csv('dataset.csv')
X= dataset.drop(columns='Result')
X.head()

# VIF dataframe
vif_data = pd.DataFrame()
vif_data["feature"] = X.columns

# calculating VIF for each feature
vif_data["VIF"] = [variance_inflation_factor(X.values, i)
```

```
                          for i in range(len(X.columns))]

print(vif_data)
```

```
              feature       VIF
0       Links_in_tags  1.056889
1                 SFH  1.502931
2  Submitting_to_email  3.219025
3        Abnormal_URL  2.232462
4            Redirect  1.286968
5         on_mouseover  6.084780
6           RightClick  7.037169
7          popUpWidnow  4.344865
8              Iframe  8.383330
```

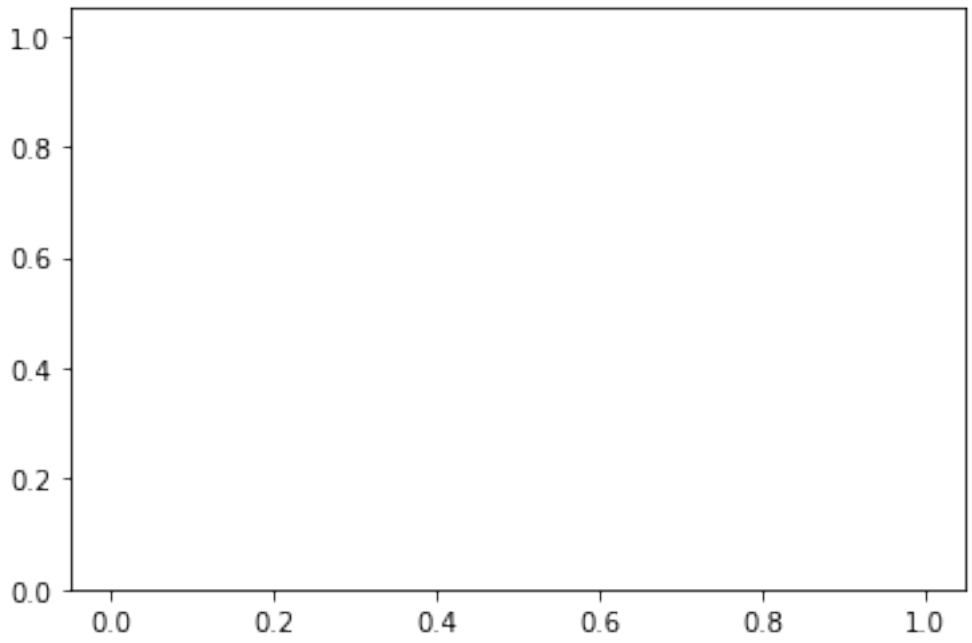
[2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline

dataset = pd.read_csv('dataset.csv')
X= dataset.drop(columns='Result')

# Mean absolute Difference (MAD)
mean_abs_diff = np.sum(np.abs(X -np.mean(X, axis=0)), axis=0)/X.shape[0]

plt.bar(np.arange(X.shape[1]), mean_abs_diff, color='teal')
```

[2]: <BarContainer object of 9 artists>



[7]:
```
# Dispersion ratio
dataset = pd.read_csv('dataset.csv')
X= dataset.drop(columns='Result')

X = X+1

# Arithmatic mean
am = np.mean(X, axis=0)

# Geomatric mean
gm = np.power(np.prod(X, axis=0), 1/X.shape[0])

# ratio of AM and GM
disp_ratio =am/gm

plt.bar(np.arange(X.shape[1]), disp_ratio, color='teal')
```

[7]:
```
<BarContainer object of 9 artists>
```



[ ]:
```

```