regression hypothesis

May 11, 2021

[92]:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
%matplotlib inline
from sklearn.model_selection import train_test_split,cross_val_score
```

[93]:

```
dataset = pd.read_csv('dataset.csv')
X= dataset.drop(columns=['url','status'])
Y= dataset['status']
X.head()
```

[93]:

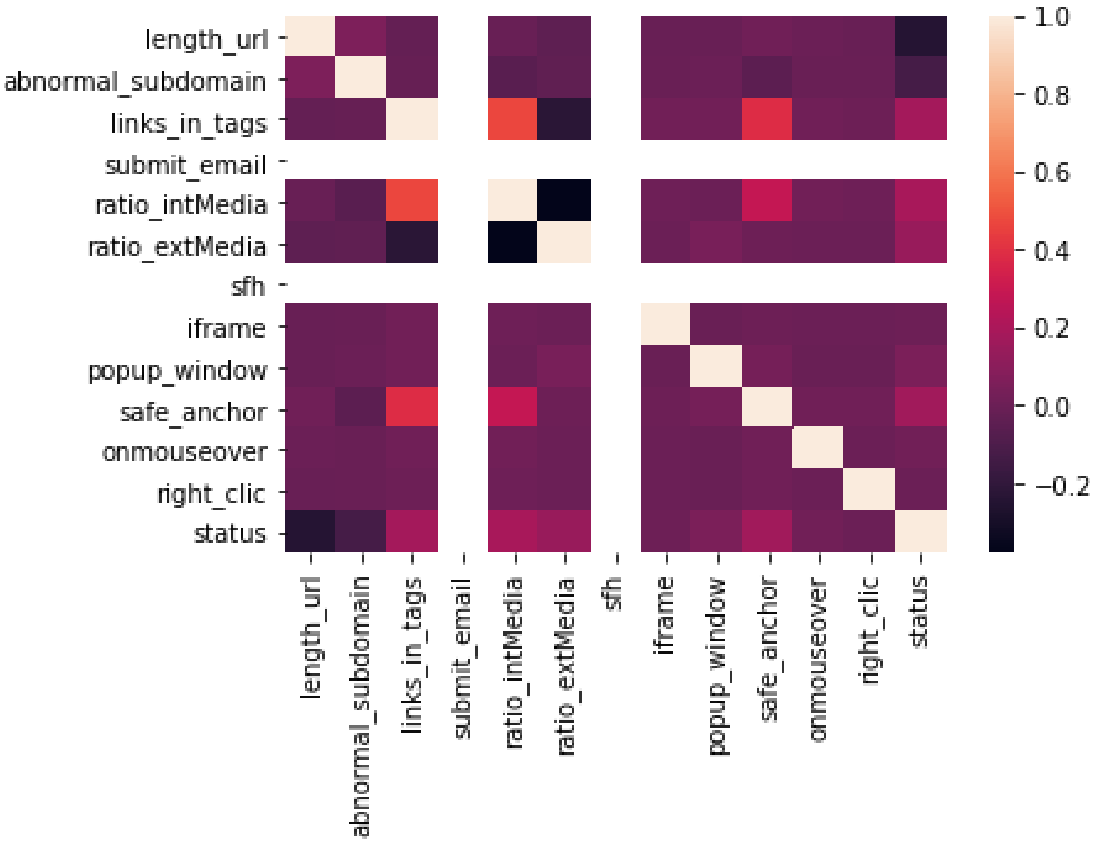
```
   length_url  abnormal_subdomain  links_in_tags  submit_email  \
0          46                   0      73.913043             0
1         128                   0       0.000000             0
2          52                   0     100.000000             0
3          21                   0     100.000000             0
4          28                   0      55.555556             0

   ratio_intMedia  ratio_extMedia  sfh  iframe  popup_window  safe_anchor  \
0      100.000000        0.000000    0       0             0    77.777778
1        0.000000        0.000000    0       0             0     0.000000
2        0.000000        0.000000    0       0             0     0.000000
3       92.307692        7.692308    0       0             0    82.539683
4       50.000000       50.000000    0       0             0    81.081081

   onmouseover  right_clic
0            0           0
1            0           0
2            0           0
3            0           0
4            0           0
```

[94]:

```
df = dataset.drop(columns='url')
sns.heatmap(dataset.corr())
```

[94]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x13d3d1cfc08>
```



```
[95]: # df1 = pd.DataFrame(10*np.random.randn(10, 3))
      # df1.iloc[0, 0] = 0 # So we can check the == 0 condition  use X insted of df1
      df1 = pd.DataFrame(X['length_url'])
      # df1.iloc[0, 0] = 0
      # 1 => legitimate
      # -1 => physhing
      # 0 => suspecious

      conds = [df1.values < 44 , df1.values > 55]
      choices = ['1', '-1']

      col = pd.DataFrame(np.select(conds, choices, default='0'),
                   index=df1.index,
                   columns=df1.columns)

      col.head()
```

```
[95]:   length_url
      0          0
      1         -1
```

```
2          0
3          1
4          1
```

[96]:
```
# df = pd.DataFrame(X)
X = X.assign(length_url=col['length_url'])
X.head()
```

[96]:
```
   length_url  abnormal_subdomain  links_in_tags  submit_email  ratio_intMedia  \
0           0                   0      73.913043             0      100.000000
1          -1                   0       0.000000             0        0.000000
2           0                   0     100.000000             0        0.000000
3           1                   0     100.000000             0       92.307692
4           1                   0      55.555556             0       50.000000

   ratio_extMedia  sfh  iframe  popup_window  safe_anchor  onmouseover  \
0        0.000000    0       0             0    77.777778            0
1        0.000000    0       0             0     0.000000            0
2        0.000000    0       0             0     0.000000            0
3        7.692308    0       0             0    82.539683            0
4       50.000000    0       0             0    81.081081            0

   right_clic
0           0
1           0
2           0
3           0
4           0
```

[97]:
```
train_X,test_X,train_Y,test_Y=train_test_split(X,Y,test_size=0.2,random_state=2)
```

[98]:
```
# test using logistic regression
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import␣
 ↪accuracy_score,confusion_matrix,classification_report
```

[99]:
```
logreg=LogisticRegression()
model_1=logreg.fit(train_X,train_Y)
```

[100]:
```
logreg_predict= model_1.predict(test_X)
```

[101]:
```
accuracy_score(logreg_predict,test_Y)
```

[101]:
```
0.68125
```

[102]:
```
print(classification_report(logreg_predict,test_Y))
```

```
              precision    recall  f1-score   support
```

```
           0       0.65      0.68      0.67       752
           1       0.71      0.68      0.69       848

    accuracy                           0.68      1600
   macro avg       0.68      0.68      0.68      1600
weighted avg       0.68      0.68      0.68      1600
```

[ ]:

[ ]:

[ ]:

[ ]:

[ ]: