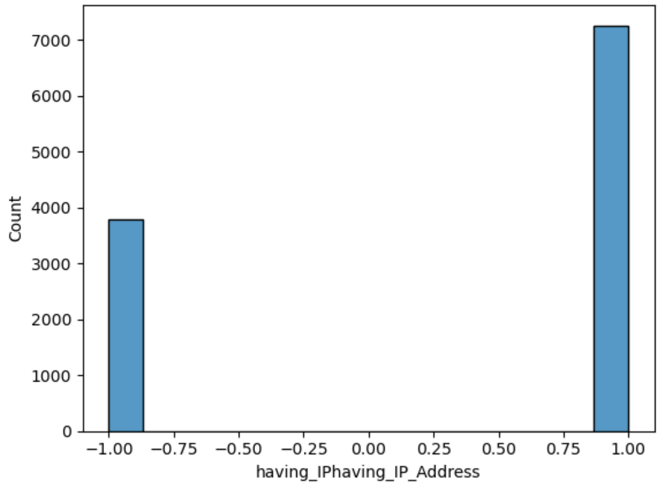
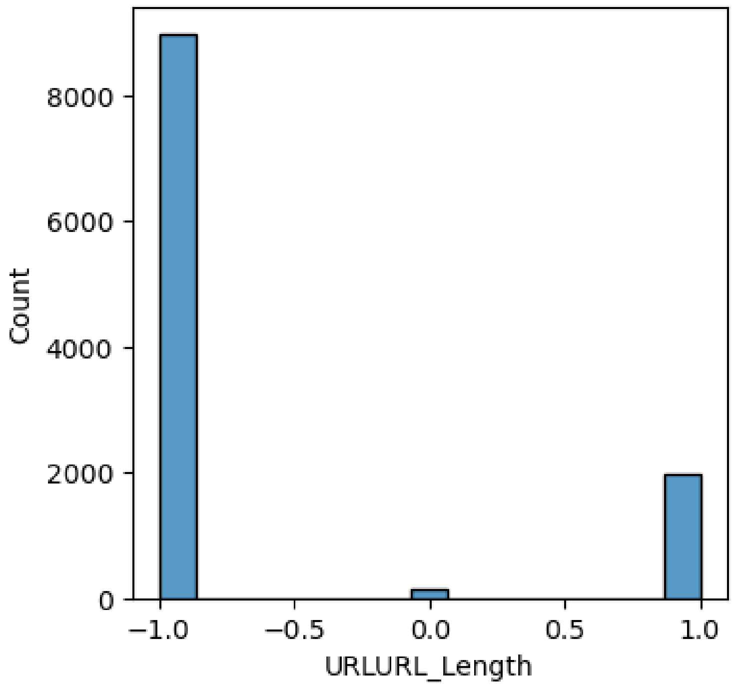
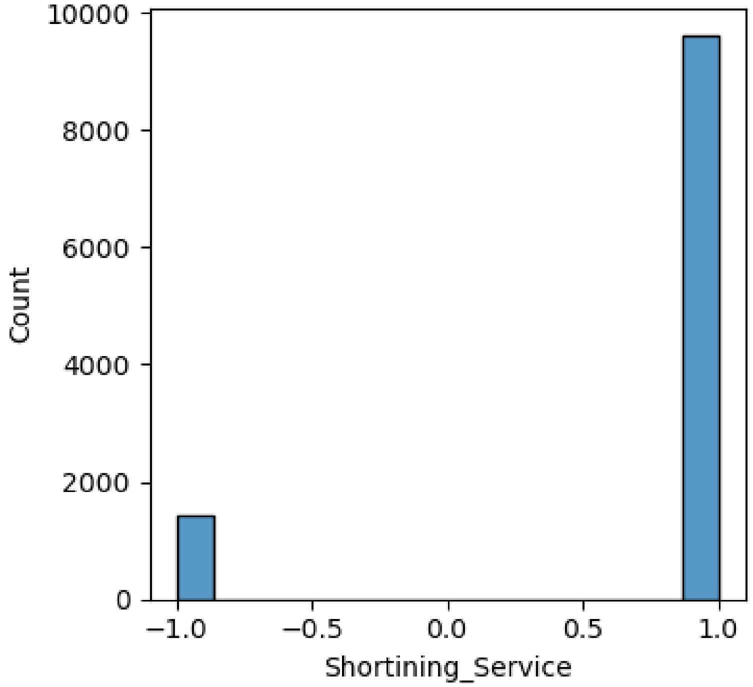
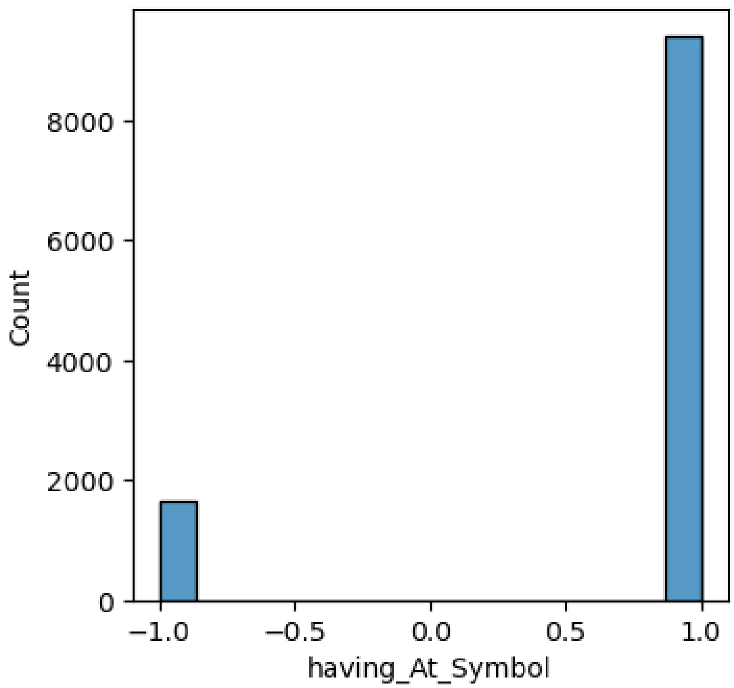
1. Each sample has 32 features ranging from -1,0,1. Explore the data using histogram, heatmaps.



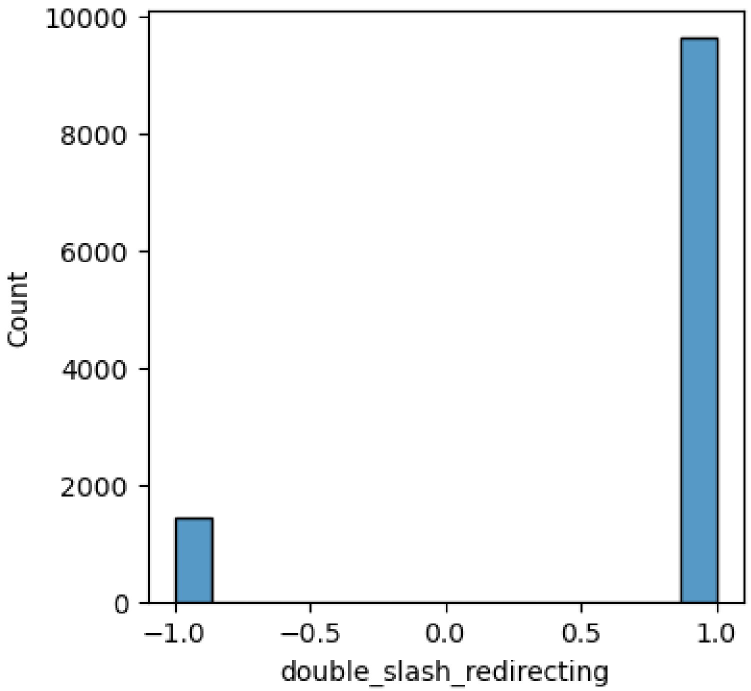



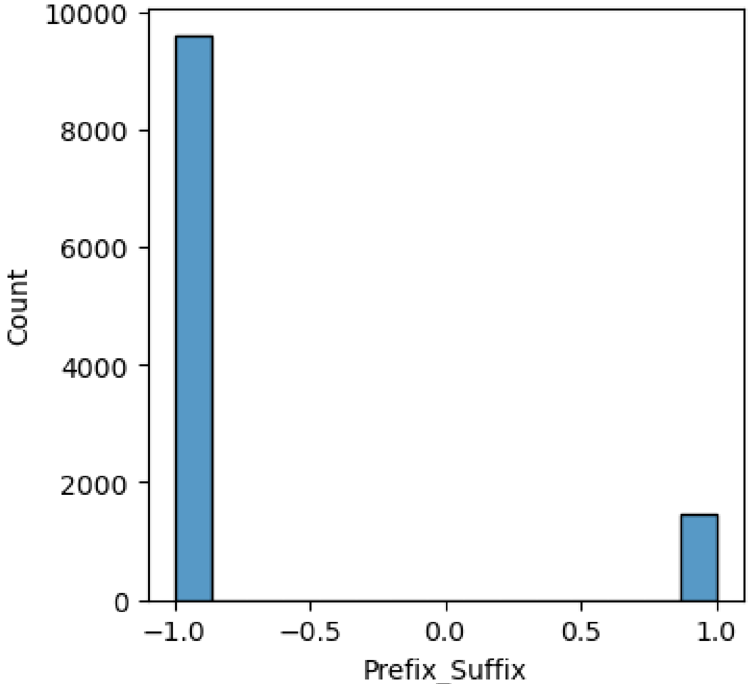


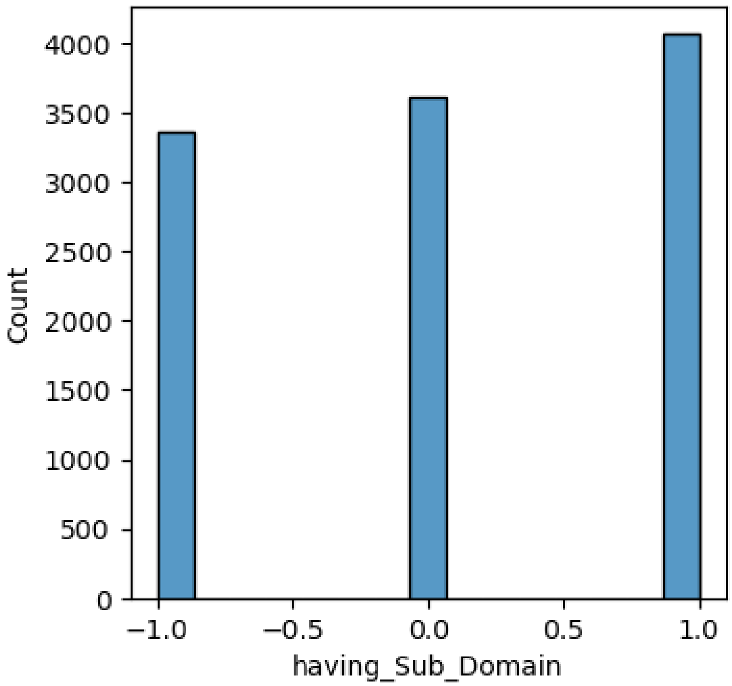
10000
8000
6000
4000
2000
0
Count
−1.0
−0.5
0.0
0.5
1.0
Shortining_Service

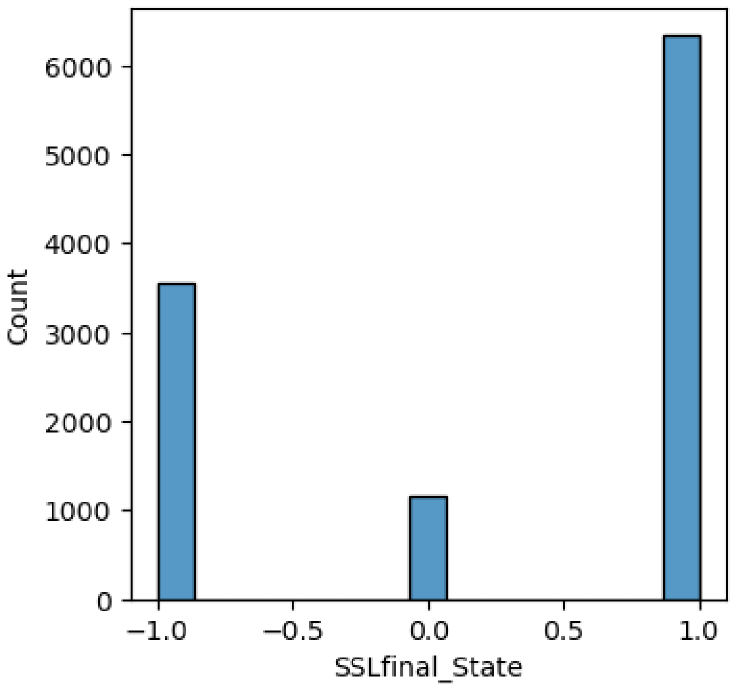



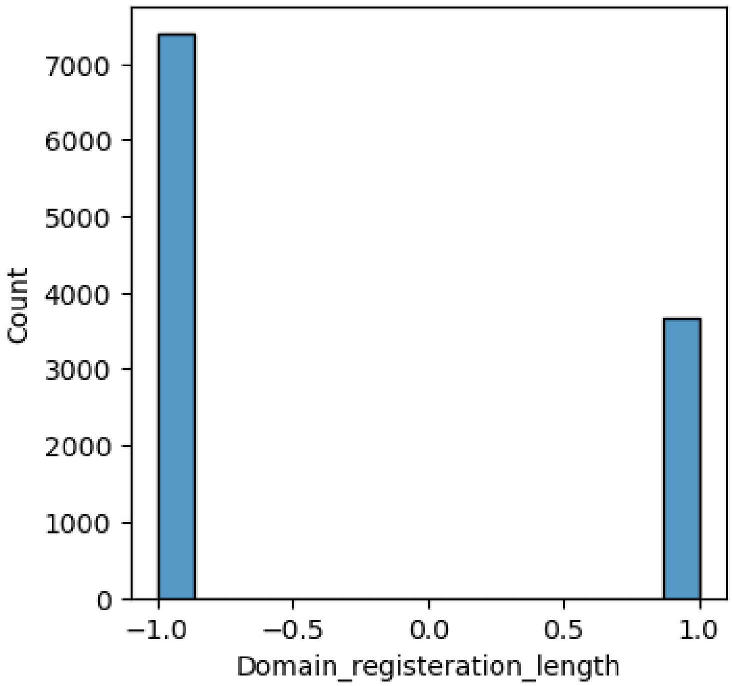
8000
6000
4000
2000
0
Count
−1.0
−0.5
0.0
0.5
1.0
having_At_Symbol


10000
8000
6000
4000
2000
0
Count
−1.0
−0.5
0.0
0.5
1.0
double_slash_redirecting

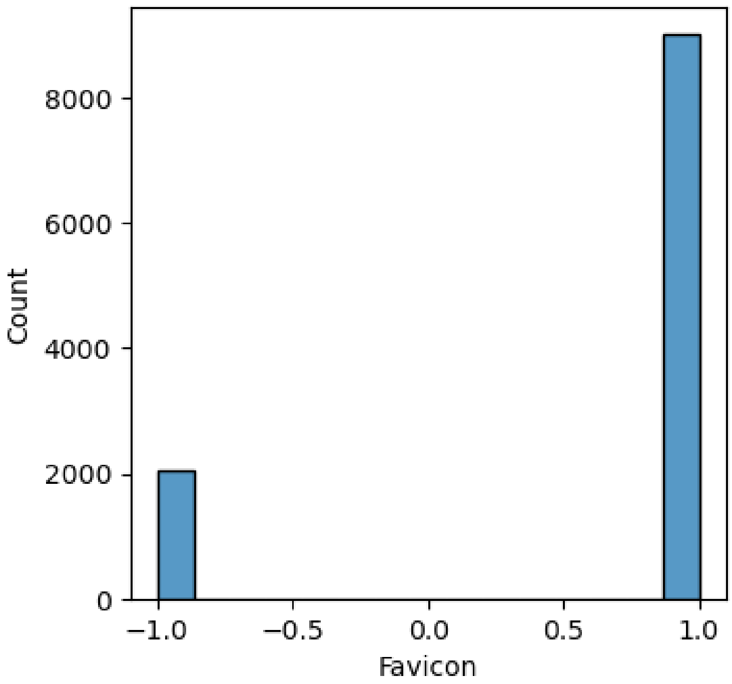

10000
8000
6000
4000
2000
0
Count
−1.0
−0.5
0.0
0.5
1.0
Prefix_Suffix


Count
having_Sub_Domain
4000
3500
3000
2500
2000
1500
1000
500
0
−1.0
−0.5
0.0
0.5
1.0

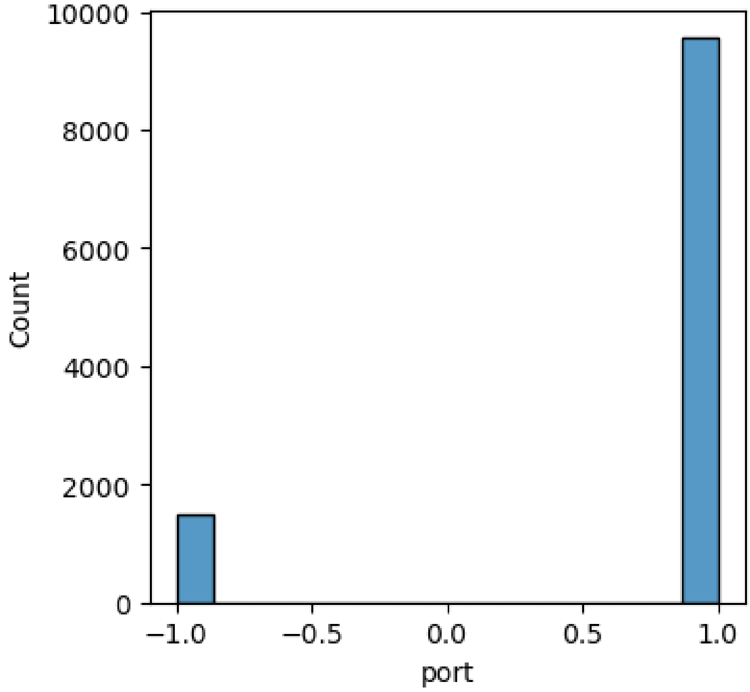

Count
SSLfinal_State
6000
5000
4000
3000
2000
1000
0
−1.0
−0.5
0.0
0.5
1.0

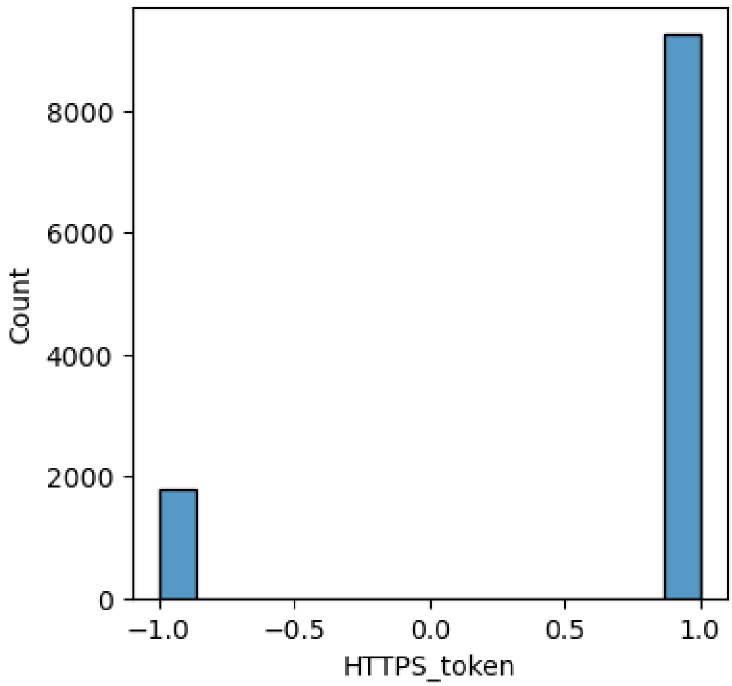


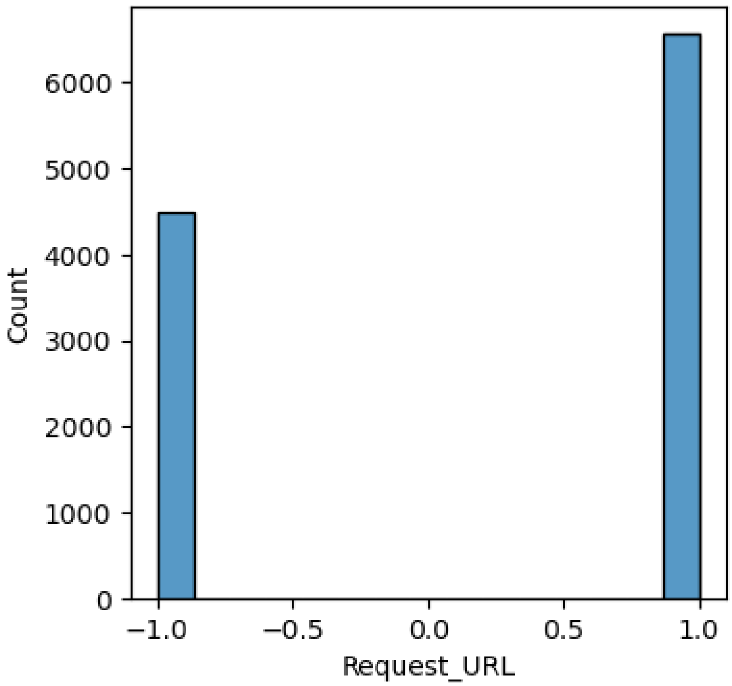
7000
6000
5000
4000
3000
2000
1000
0
Count
−1.0
−0.5
0.0
0.5
1.0
Domain_registeration_length

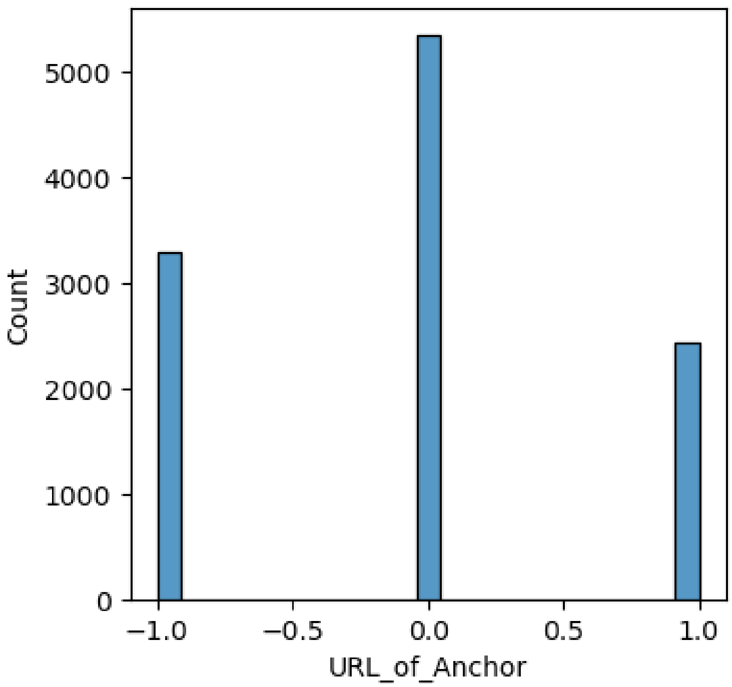
8000
6000
4000
2000
0
Count
−1.0
−0.5
0.0
0.5
1.0
Favicon


10000
8000
6000
4000
2000
0
Count
−1.0
−0.5
0.0
0.5
1.0
port

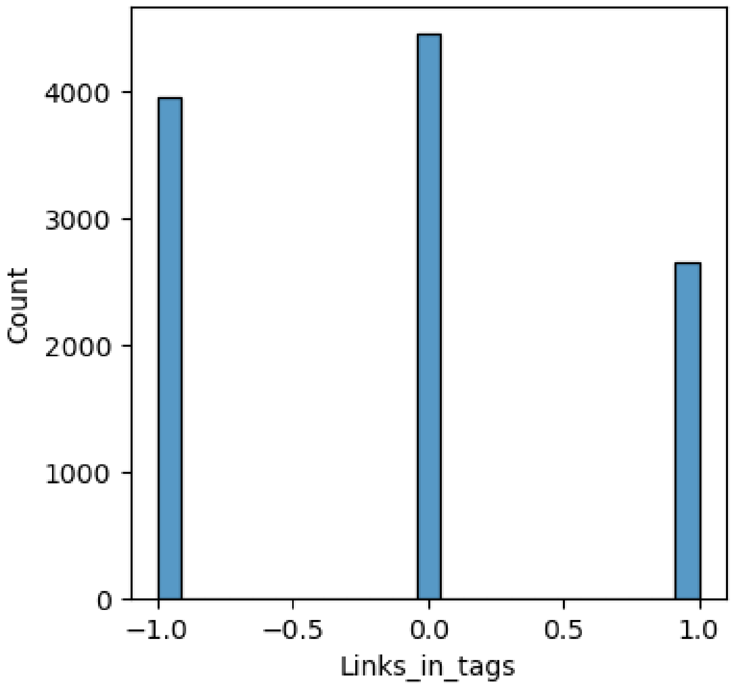

8000
6000
4000
2000
0
Count
−1.0
−0.5
0.0
0.5
1.0
HTTPS_token

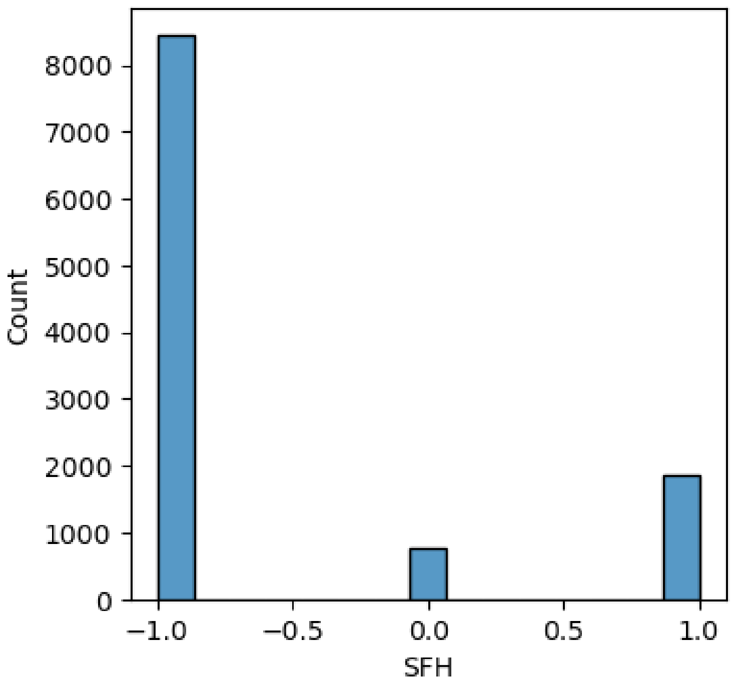


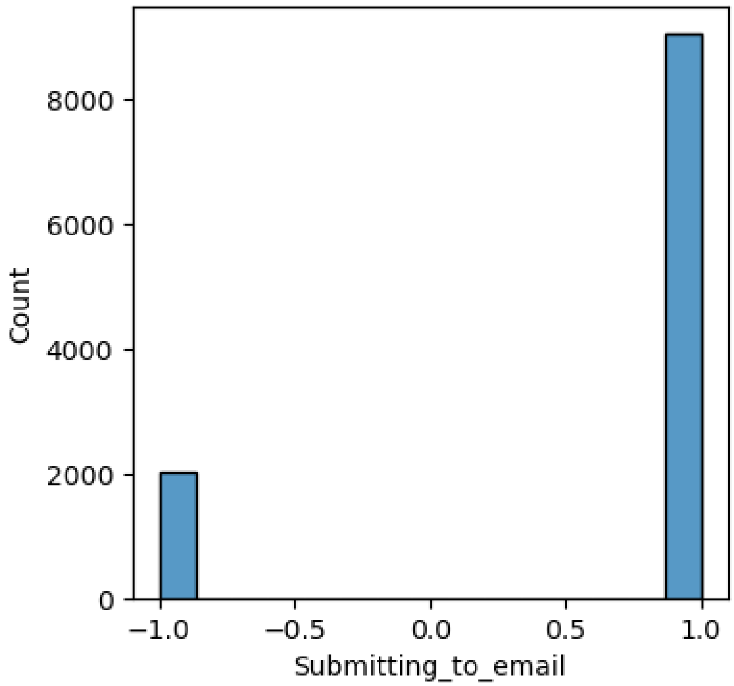
Count
Request_URL
0
1000
2000
3000
4000
5000
6000
−1.0
−0.5
0.0
0.5
1.0

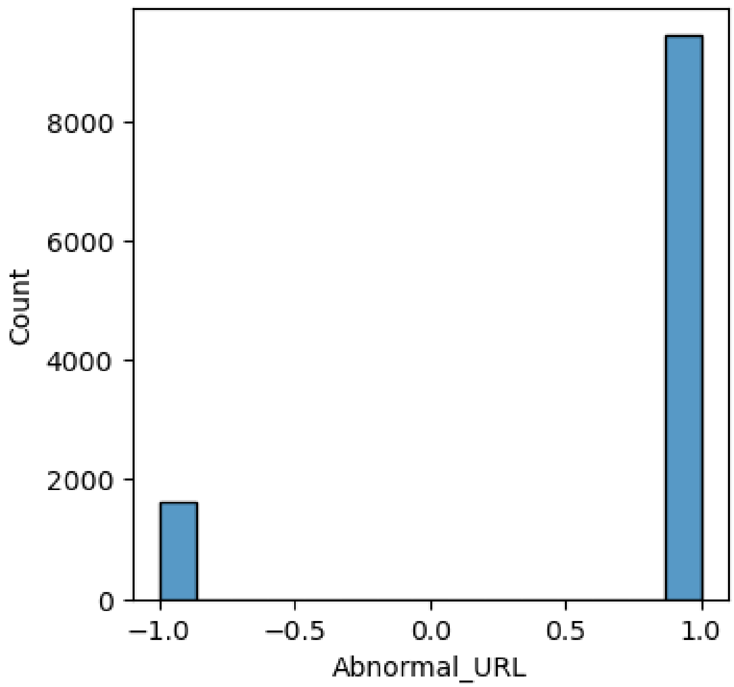



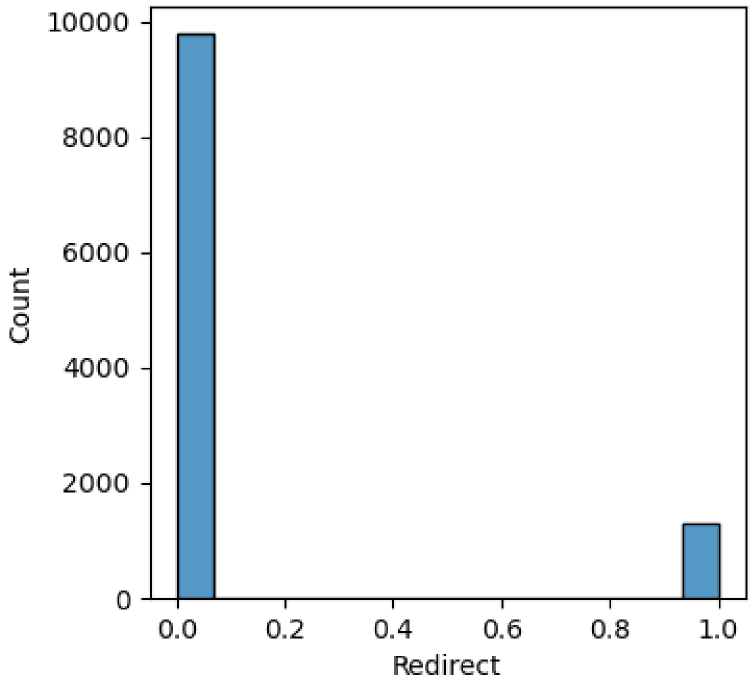
Count
URL_of_Anchor
0
1000
2000
3000
4000
5000
−1.0
−0.5
0.0
0.5
1.0

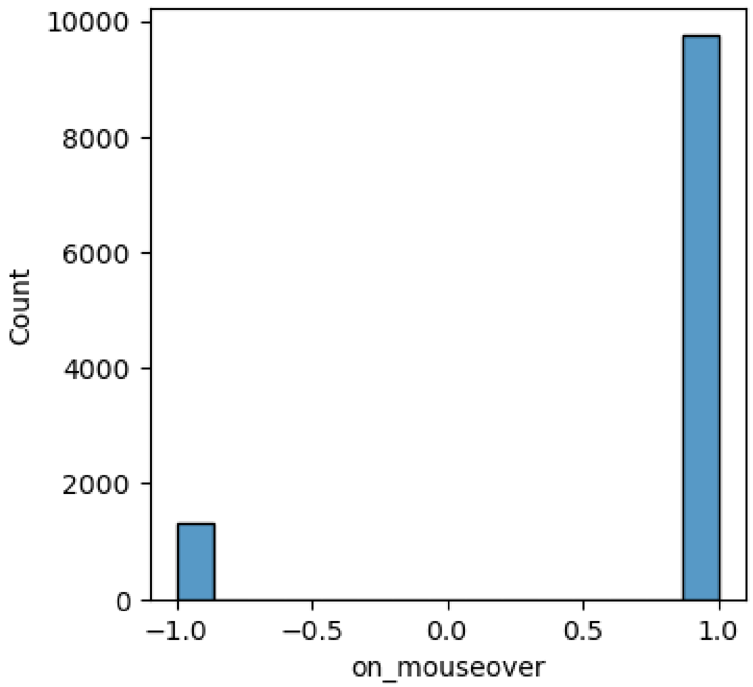

4000
3000
2000
1000
0
Count
−1.0
−0.5
0.0
0.5
1.0
Links_in_tags

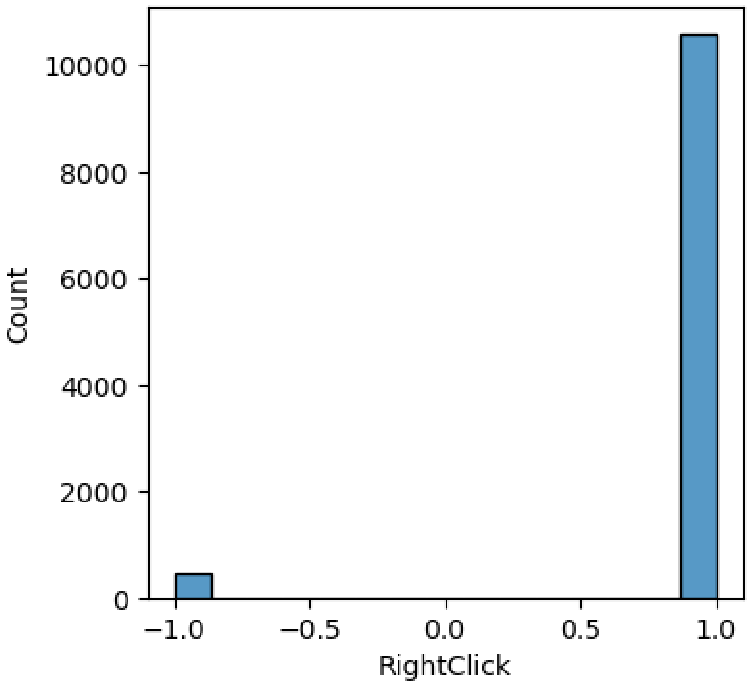

8000
7000
6000
5000
4000
3000
2000
1000
0
Count
−1.0
−0.5
0.0
0.5
1.0
SFH

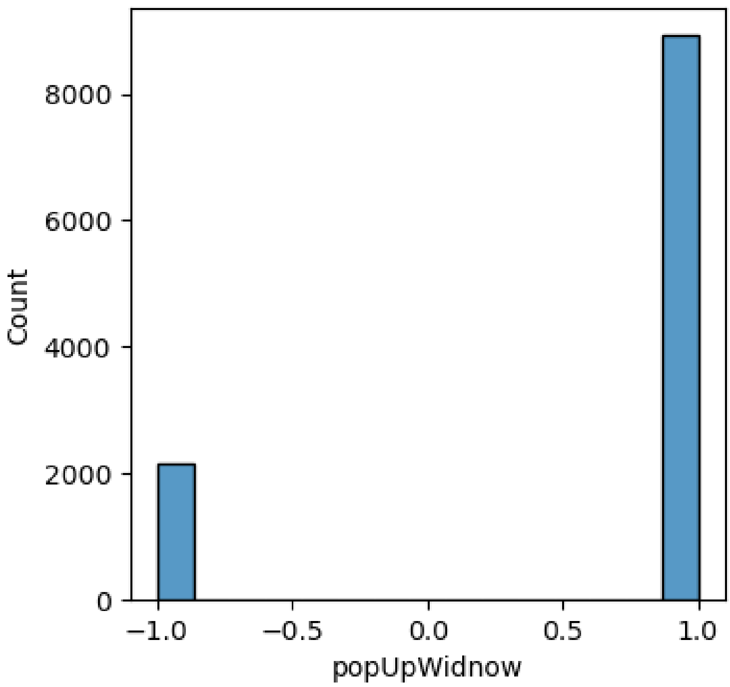

8000
6000
4000
2000
0
Count
−1.0
−0.5
0.0
0.5
1.0
Submitting_to_email

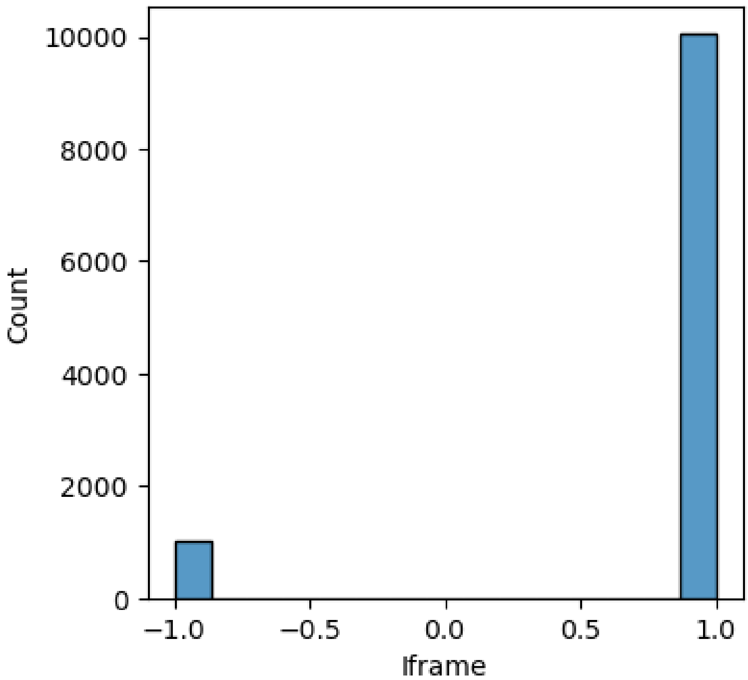

8000
6000
4000
2000
0
Count
−1.0
−0.5
0.0
0.5
1.0
Abnormal_URL

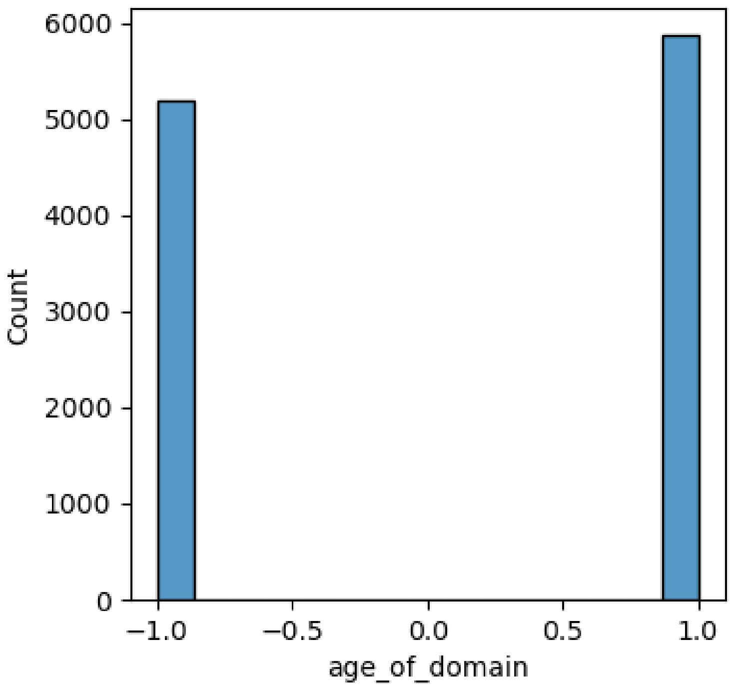

10000
8000
6000
4000
2000
0
Count
0.0
0.2
0.4
0.6
0.8
1.0
Redirect

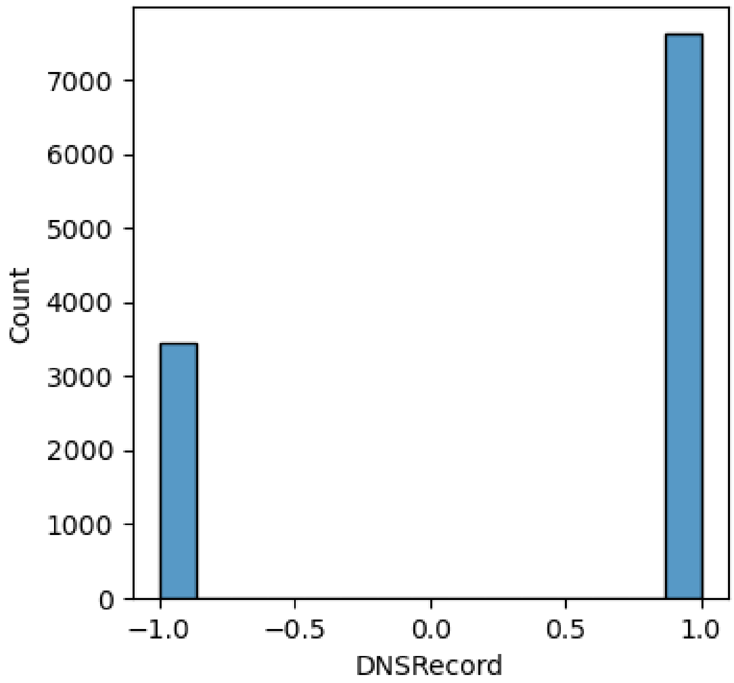

10000
8000
6000
4000
2000
0
Count
−1.0
−0.5
0.0
0.5
1.0
on_mouseover

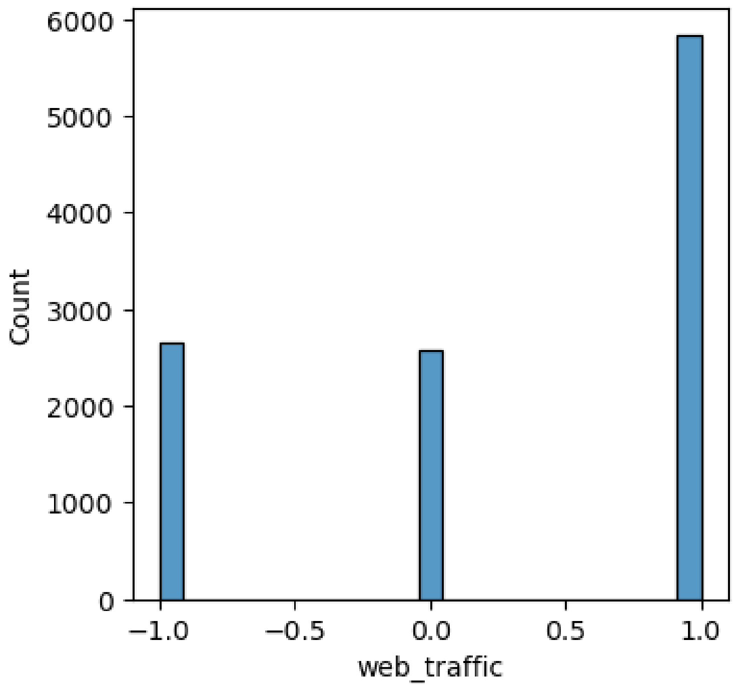

10000
8000
6000
4000
2000
0
Count
−1.0
−0.5
0.0
0.5
1.0
RightClick

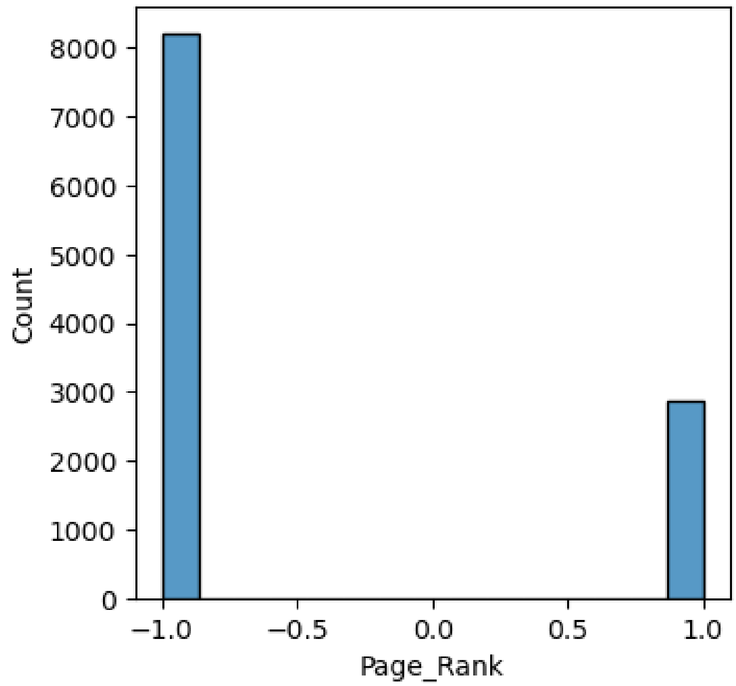

8000
6000
4000
2000
0
Count
−1.0
−0.5
0.0
0.5
1.0
popUpWidnow

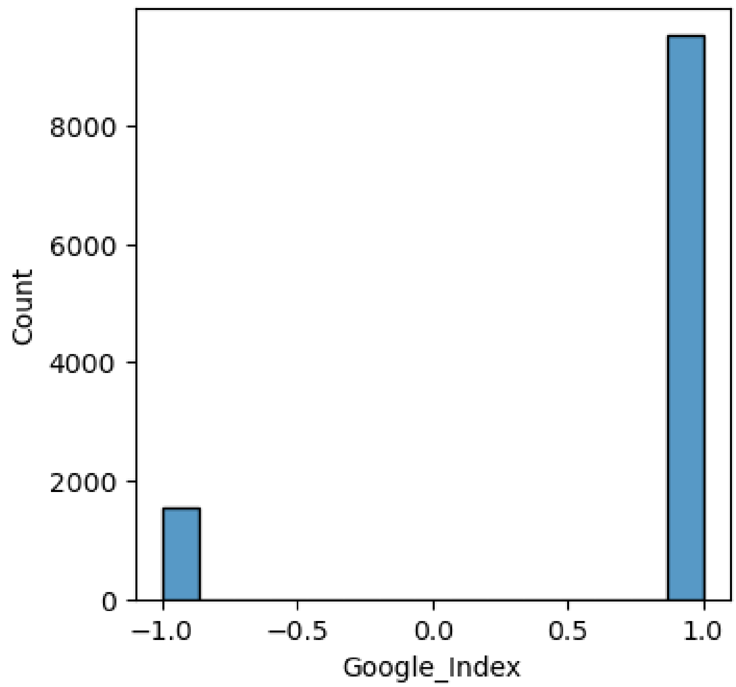

10000
8000
6000
4000
2000
0
Count
−1.0
−0.5
0.0
0.5
1.0
Iframe

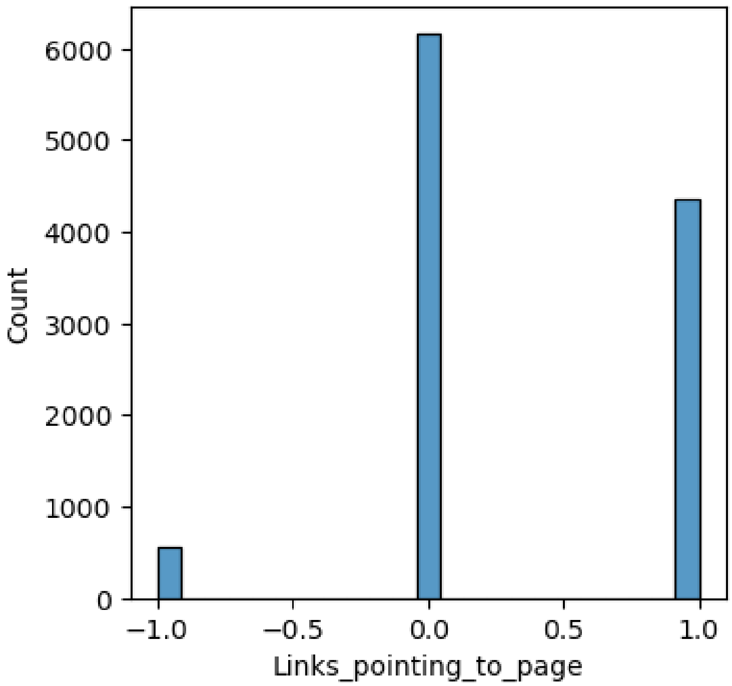

6000
5000
4000
3000
2000
1000
0
Count
−1.0
−0.5
0.0
0.5
1.0
age_of_domain

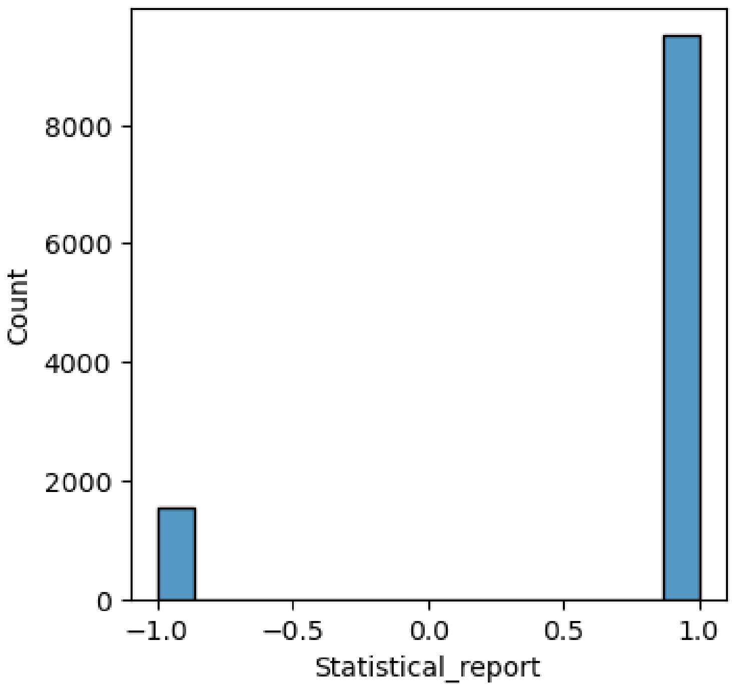

7000
6000
5000
4000
3000
2000
1000
0
Count
−1.0
−0.5
0.0
0.5
1.0
DNSRecord

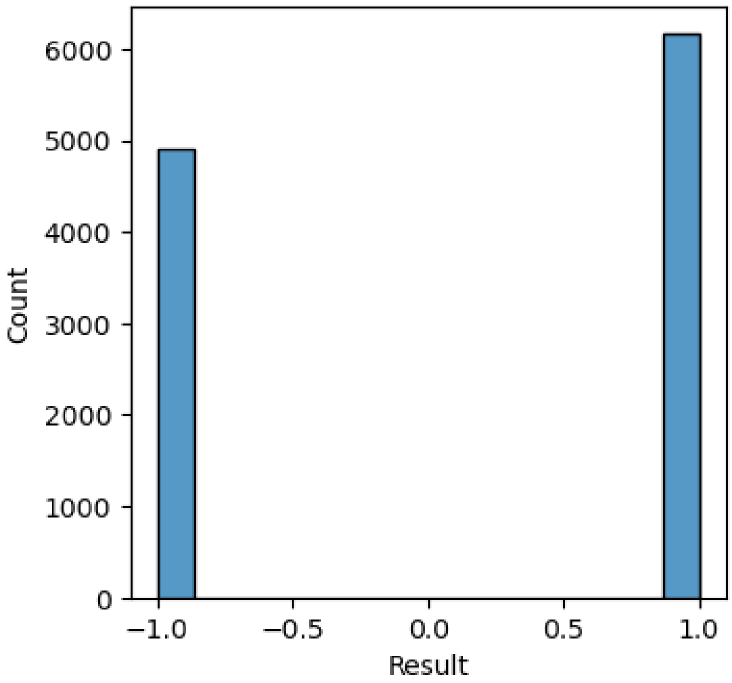

6000
5000
4000
3000
2000
1000
0
Count
−1.0
−0.5
0.0
0.5
1.0
web_traffic


8000
7000
6000
5000
4000
3000
2000
1000
0
Count
−1.0
−0.5
0.0
0.5
1.0
Page_Rank

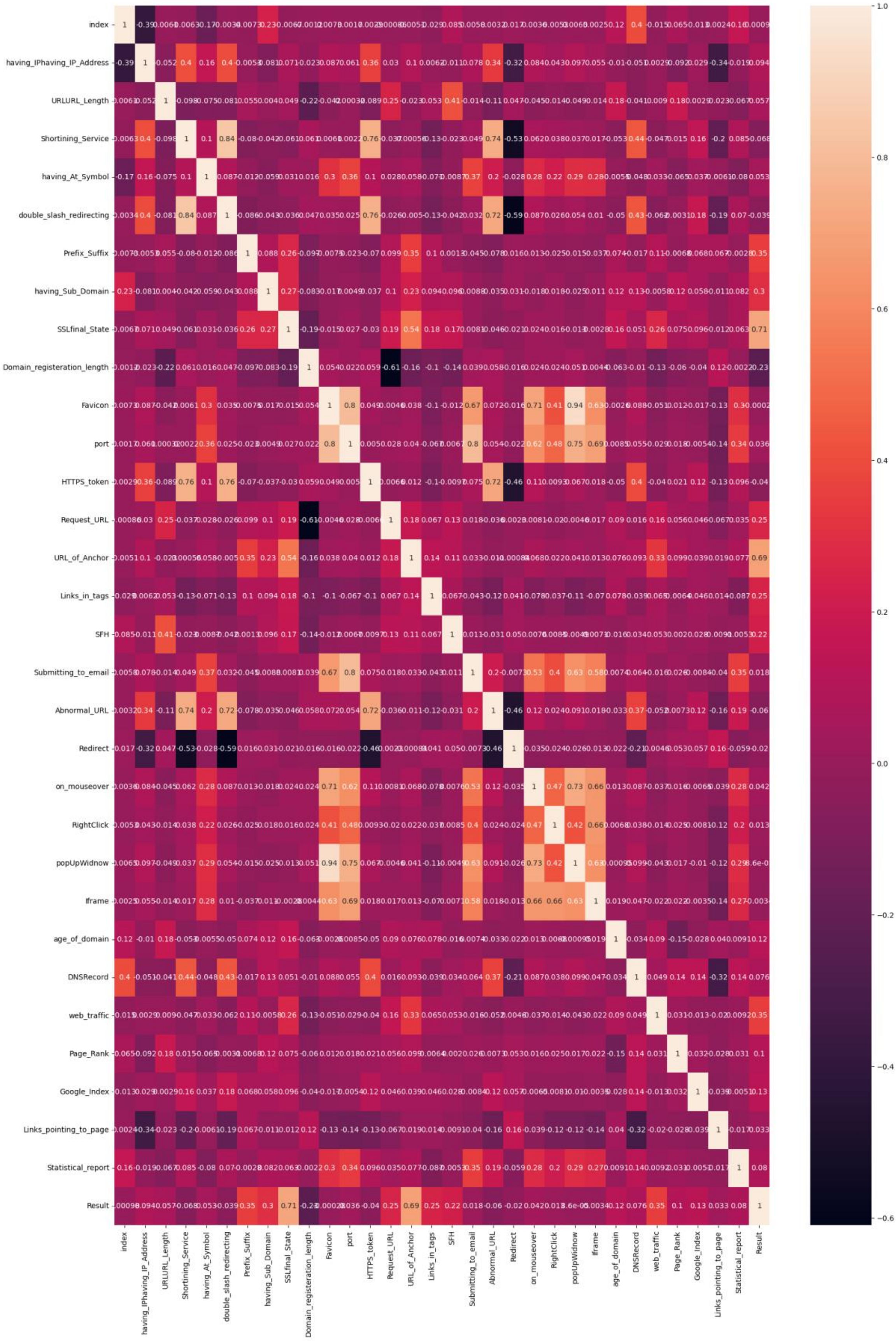

8000
6000
4000
2000
0
Count
−1.0
−0.5
0.0
0.5
1.0
Google_Index

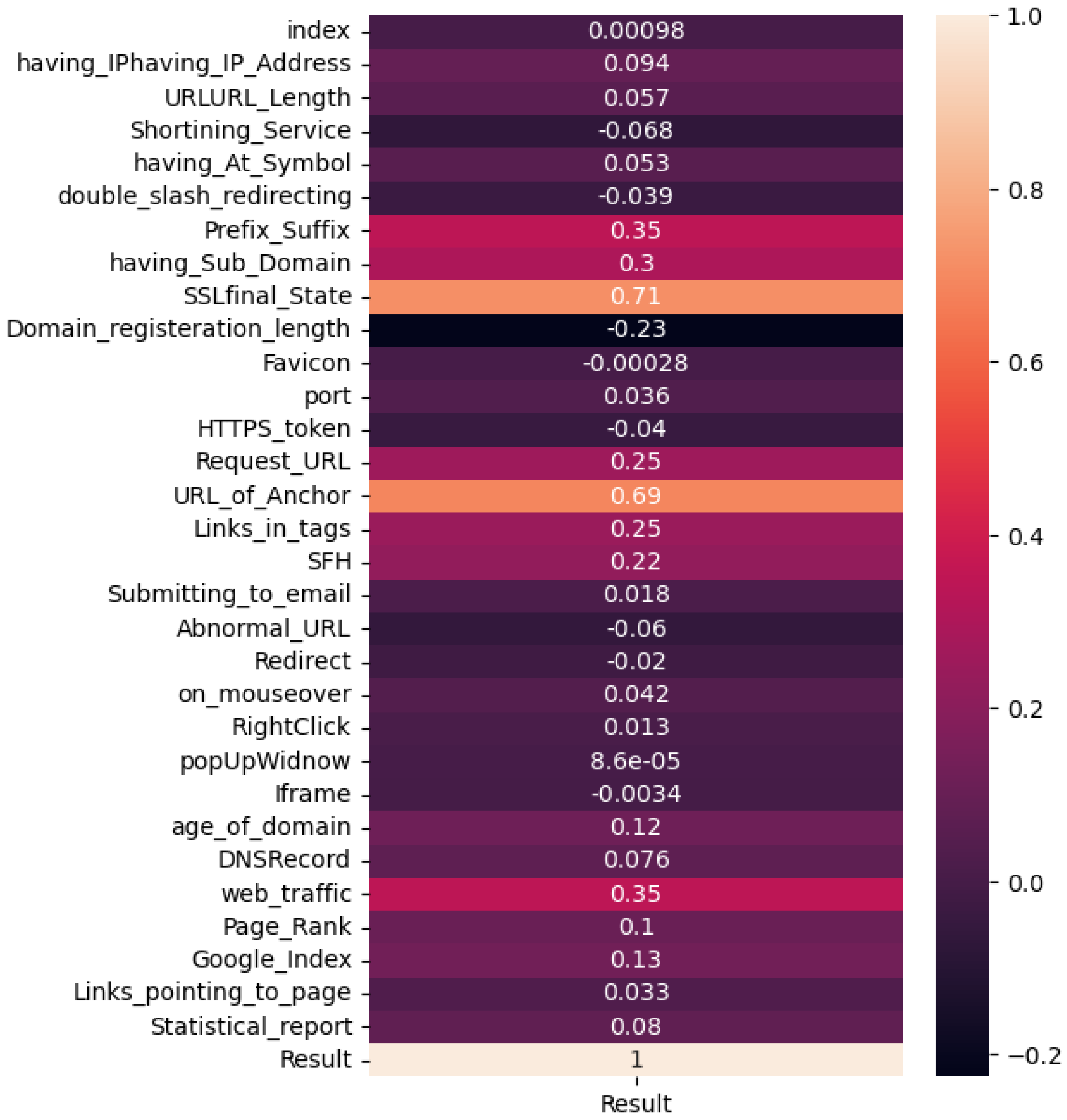

6000
5000
4000
3000
2000
1000
0
Count
−1.0
−0.5
0.0
0.5
1.0
Links_pointing_to_page


8000
6000
4000
2000
0
Count
−1.0
−0.5
0.0
0.5
1.0
Statistical_report



6000
5000
4000
3000
2000
1000
0
Count
−1.0
−0.5
0.0
0.5
1.0
Result


index
having_IPhaving_IP_Address
URLURL_Length
Shortining_Service
having_At_Symbol
double_slash_redirecting
Prefix_Suffix
having_Sub_Domain
SSLfinal_State
Domain_registeration_length
Favicon
port
HTTPS_token
Request_URL
URL_of_Anchor
Links_in_tags
SFH
Submitting_to_email
Abnormal_URL
Redirect
on_mouseover
RightClick
popUpWidnow
Iframe
age_of_domain
DNSRecord
web_traffic
Page_Rank
Google_Index
Links_pointing_to_page
Statistical_report
Result
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6




2. Determine the number of samples present in the data, unique elements in all the features.

```
df.shape
```

```
(11055, 32)
```

```
df.describe().T
```

| | count | mean | std | min | 25% | 50% | 75% | max |
|---|---|---|---|---|---|---|---|---|
| **index** | 11055.0 | 5528.000000 | 3191.447947 | 1.0 | 2764.5 | 5528.0 | 8291.5 | 11055.0 |
| **having_IPhaving_IP_Address** | 11055.0 | 0.313795 | 0.949534 | -1.0 | -1.0 | 1.0 | 1.0 | 1.0 |
| **URLURL_Length** | 11055.0 | -0.633198 | 0.766095 | -1.0 | -1.0 | -1.0 | -1.0 | 1.0 |
| **Shortining_Service** | 11055.0 | 0.738761 | 0.673998 | -1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| **having_At_Symbol** | 11055.0 | 0.700588 | 0.713598 | -1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| **double_slash_redirecting** | 11055.0 | 0.741474 | 0.671011 | -1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| **Prefix_Suffix** | 11055.0 | -0.734962 | 0.678139 | -1.0 | -1.0 | -1.0 | -1.0 | 1.0 |
| **having_Sub_Domain** | 11055.0 | 0.063953 | 0.817518 | -1.0 | -1.0 | 0.0 | 1.0 | 1.0 |
| **SSLfinal_State** | 11055.0 | 0.250927 | 0.911892 | -1.0 | -1.0 | 1.0 | 1.0 | 1.0 |
| **Domain_registeration_length** | 11055.0 | -0.336771 | 0.941629 | -1.0 | -1.0 | -1.0 | 1.0 | 1.0 |
| **Favicon** | 11055.0 | 0.628584 | 0.777777 | -1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| **port** | 11055.0 | 0.728268 | 0.685324 | -1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| **HTTPS_token** | 11055.0 | 0.675079 | 0.737779 | -1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| **Request_URL** | 11055.0 | 0.186793 | 0.982444 | -1.0 | -1.0 | 1.0 | 1.0 | 1.0 |
| **URL_of_Anchor** | 11055.0 | -0.076526 | 0.715138 | -1.0 | -1.0 | 0.0 | 0.0 | 1.0 |
| **Links_in_tags** | 11055.0 | -0.118137 | 0.763973 | -1.0 | -1.0 | 0.0 | 0.0 | 1.0 |
| **SFH** | 11055.0 | -0.595749 | 0.759143 | -1.0 | -1.0 | -1.0 | -1.0 | 1.0 |
| **Submitting_to_email** | 11055.0 | 0.635640 | 0.772021 | -1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| **Abnormal_URL** | 11055.0 | 0.705292 | 0.708949 | -1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| **Redirect** | 11055.0 | 0.115694 | 0.319872 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 |
| **on_mouseover** | 11055.0 | 0.762099 | 0.647490 | -1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| **RightClick** | 11055.0 | 0.913885 | 0.405991 | -1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| **popUpWidnow** | 11055.0 | 0.613388 | 0.789818 | -1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| **Iframe** | 11055.0 | 0.816915 | 0.576784 | -1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| **age_of_domain** | 11055.0 | 0.061239 | 0.998168 | -1.0 | -1.0 | 1.0 | 1.0 | 1.0 |
| **DNSRecord** | 11055.0 | 0.377114 | 0.926209 | -1.0 | -1.0 | 1.0 | 1.0 | 1.0 |
| **web_traffic** | 11055.0 | 0.287291 | 0.827733 | -1.0 | 0.0 | 1.0 | 1.0 | 1.0 |
| **Page_Rank** | 11055.0 | -0.483673 | 0.875289 | -1.0 | -1.0 | -1.0 | 1.0 | 1.0 |
| **Google_Index** | 11055.0 | 0.721574 | 0.692369 | -1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| **Links_pointing_to_page** | 11055.0 | 0.344007 | 0.569944 | -1.0 | 0.0 | 0.0 | 1.0 | 1.0 |
| **Statistical_report** | 11055.0 | 0.719584 | 0.694437 | -1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| **Result** | 11055.0 | 0.113885 | 0.993539 | -1.0 | -1.0 | 1.0 | 1.0 | 1.0 |

```
df['Result'].unique()
```

```
array([-1,  1], dtype=int64)
```

```
df['Result'].nunique()
```

```
2
```

```
for i in df.columns:
 print (i , ":", df[i].unique(),"\n len :",df[i].nunique())
 print ("\n")
```

```
index : [    1     2     3 ... 11053 11054 11055]
 len : 11055


having_IPhaving_IP_Address : [-1  1]
 len : 2


URLURL_Length : [ 1  0 -1]
 len : 3


Shortining_Service : [ 1 -1]
 len : 2
```


```
having_IPhaving_IP_Address : [-1  1]
 len : 2


URLURL_Length : [ 1  0 -1]
 len : 3


Shortining_Service : [ 1 -1]
 len : 2
```


```
having_At_Symbol : [ 1 -1]
 len : 2


double_slash_redirecting : [-1  1]
 len : 2


Prefix_Suffix : [-1  1]
 len : 2


having_Sub_Domain : [-1  0  1]
 len : 3
```

```
SSLfinal_State : [-1  1  0]
 len : 3


Domain_registeration_length : [-1  1]
 len : 2


Favicon : [ 1 -1]
 len : 2


port : [ 1 -1]
 len : 2


HTTPS_token : [-1  1]
 len : 2


Request_URL : [ 1 -1]
 len : 2


URL_of_Anchor : [-1  0  1]
 len : 3

Links_in_tags : [ 1 -1  0]
 len : 3


SFH : [-1  1  0]
 len : 3


Submitting_to_email : [-1  1]
 len : 2


Abnormal_URL : [-1  1]
 len : 2


Redirect : [0 1]
 len : 2


on_mouseover : [ 1 -1]
 len : 2


RightClick : [ 1 -1]
 len : 2
```

```
popUpWidnow : [ 1 -1]
 len : 2


Iframe : [ 1 -1]
 len : 2


age_of_domain : [-1  1]
 len : 2


DNSRecord : [-1  1]
 len : 2


web_traffic : [-1  0  1]
 len : 3


Page_Rank : [-1  1]
 len : 2


Google_Index : [ 1 -1]
 len : 2


Links_pointing_to_page : [ 1  0 -1]
 len : 3


Statistical_report : [-1  1]
 len : 2


Result : [-1  1]
 len : 2
```

## 3.Check if there is any null value in any features.

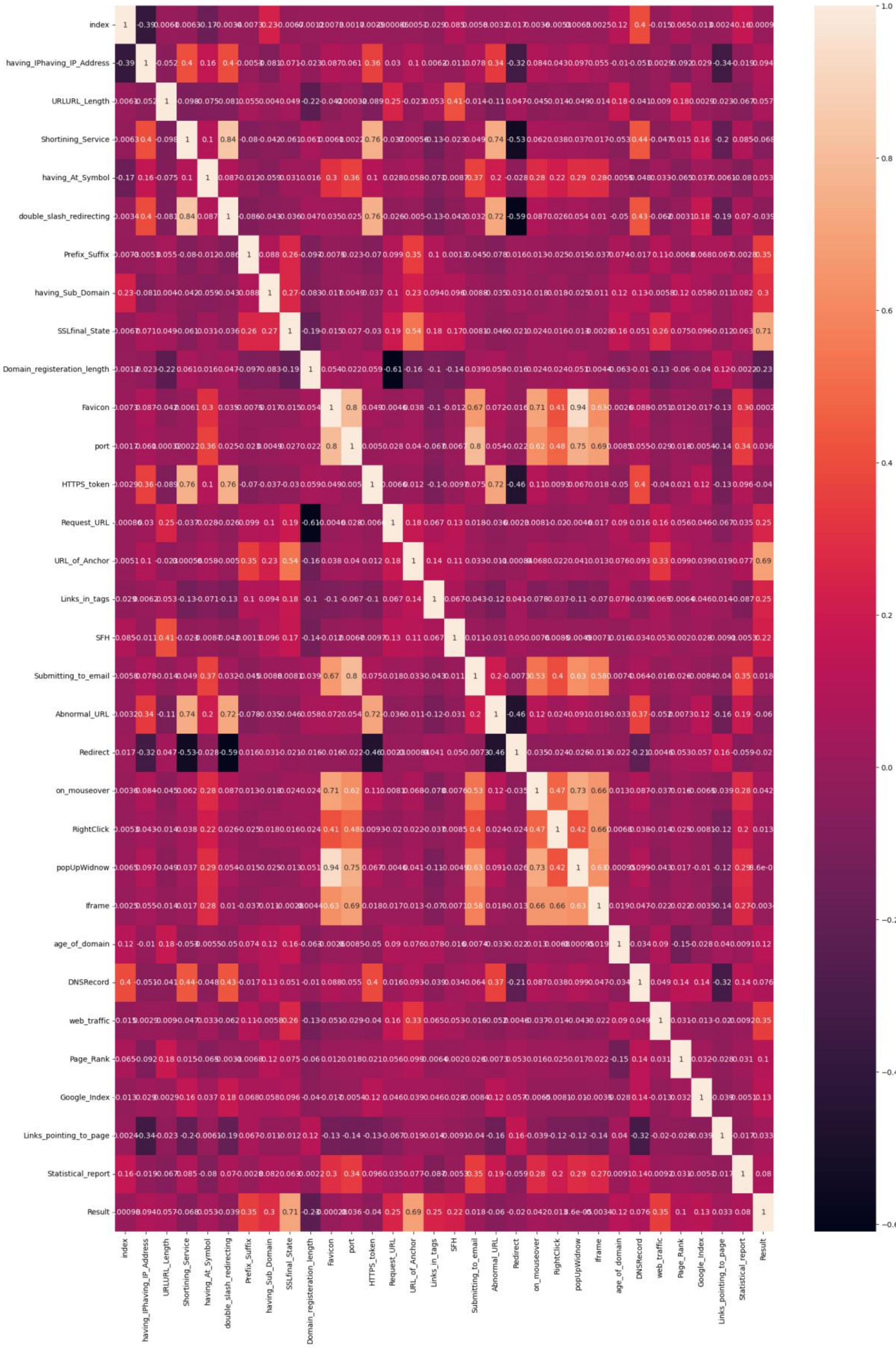
```
index                          0
having_IPhaving_IP_Address     0
URLURL_Length                  0
Shortining_Service             0
having_At_Symbol               0
double_slash_redirecting       0
Prefix_Suffix                  0
having_Sub_Domain              0
SSLfinal_State                 0
Domain_registeration_length    0
Favicon                        0
port                           0
HTTPS_token                    0
Request_URL                    0
URL_of_Anchor                  0
Links_in_tags                  0
SFH                            0
Submitting_to_email            0
Abnormal_URL                   0
Redirect                       0
on_mouseover                   0
RightClick                     0
popUpWidnow                    0
Iframe                         0
age_of_domain                  0
DNSRecord                      0
```

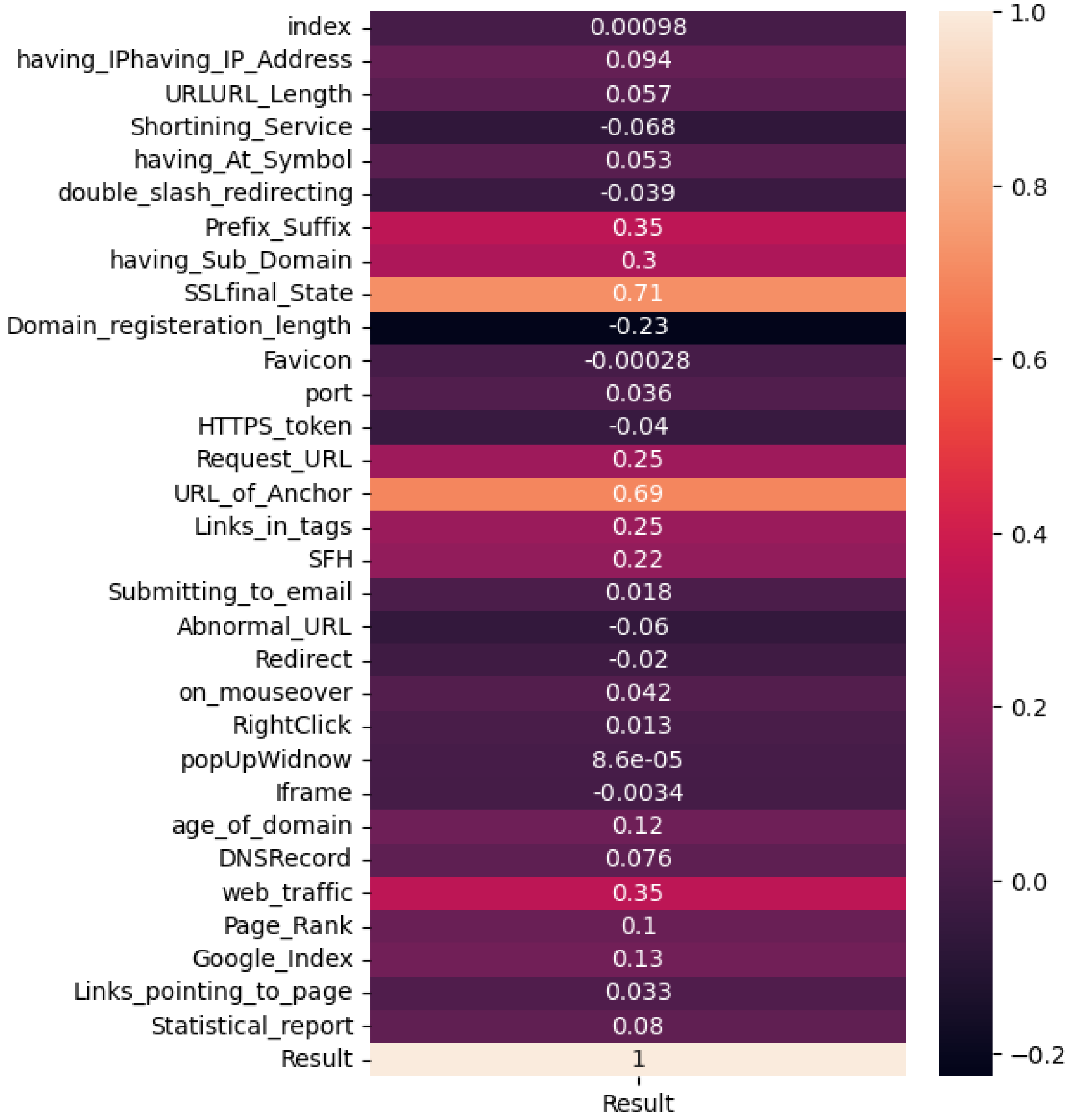
```
web_traffic                0
Page_Rank                  0
Google_Index               0
Links_pointing_to_page     0
Statistical_report         0
Result                     0
dtype: int64
```

4.Next, we have to find if there are any correlated features present in the data. Remove the feature which might be correlated with some threshold.

```
df.corr()
```

| | index | having_IPhaving_IP_Address | URLURL_Length | Shortining_Service | having_At_Symbol | double_slash_redirecting | Prefix_Su |
|---|---|---|---|---|---|---|---|
| **index** | 1.000000 | -0.388317 | 0.006105 | -0.006281 | -0.169478 | -0.003363 | -0.007 |
| **having_IPhaving_IP_Address** | -0.388317 | 1.000000 | -0.052411 | 0.403461 | 0.158699 | 0.397389 | -0.005 |
| **URLURL_Length** | 0.006105 | -0.052411 | 1.000000 | -0.097881 | -0.075108 | -0.081247 | 0.055 |
| **Shortining_Service** | -0.006281 | 0.403461 | -0.097881 | 1.000000 | 0.104447 | 0.842796 | -0.080 |
| **having_At_Symbol** | -0.169478 | 0.158699 | -0.075108 | 0.104447 | 1.000000 | 0.086960 | -0.011 |
| **double_slash_redirecting** | -0.003363 | 0.397389 | -0.081247 | 0.842796 | 0.086960 | 1.000000 | -0.085 |
| **Prefix_Suffix** | -0.007340 | -0.005257 | 0.055247 | -0.080471 | -0.011726 | -0.085590 | 1.000 |
| **having_Sub_Domain** | 0.234091 | -0.080745 | 0.003997 | -0.041916 | -0.058976 | -0.043079 | 0.087 |
| **SSLfinal_State** | -0.006682 | 0.071414 | 0.048754 | -0.061426 | 0.031220 | -0.036200 | 0.261 |
| **Domain_registeration_length** | -0.001180 | -0.022739 | -0.221892 | 0.060923 | 0.015522 | 0.047464 | -0.096 |
| **Favicon** | 0.007293 | 0.087025 | -0.042497 | 0.006101 | 0.304899 | 0.035100 | -0.007 |
| **port** | 0.001656 | 0.060979 | 0.000323 | 0.002201 | 0.364891 | 0.025060 | -0.022 |
| **HTTPS_token** | 0.002916 | 0.363534 | -0.089383 | 0.757838 | 0.104561 | 0.760799 | -0.070 |
| **Request_URL** | -0.000862 | 0.029773 | 0.246348 | -0.037235 | 0.027909 | -0.026368 | 0.098 |


index
having_IPhaving_IP_Address
URLURL_Length
Shortining_Service
having_At_Symbol
double_slash_redirecting
Prefix_Suffix
having_Sub_Domain
SSLfinal_State
Domain_registeration_length
Favicon
port
HTTPS_token
Request_URL
URL_of_Anchor
Links_in_tags
SFH
Submitting_to_email
Abnormal_URL
Redirect
on_mouseover
RightClick
popUpWidnow
Iframe
age_of_domain
DNSRecord
web_traffic
Page_Rank
Google_Index
Links_pointing_to_page
Statistical_report
Result
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6




5.Finally, build a robust classification system that classifies whether the URL sample is a phishing site or not.

- Build classification models using a binary classifier to detect malicious or phishing URLs.

```
logreg.fit(X_train,y_train)
```

```
▾ LogisticRegression
LogisticRegression()
```

```
[63]: #testing
from sklearn.metrics import confusion_matrix, classification_report
confusion_matrix(y_test, pred)
```

```
[63]: array([[1421,  144],
       [ 148, 1936]], dtype=int64)
```

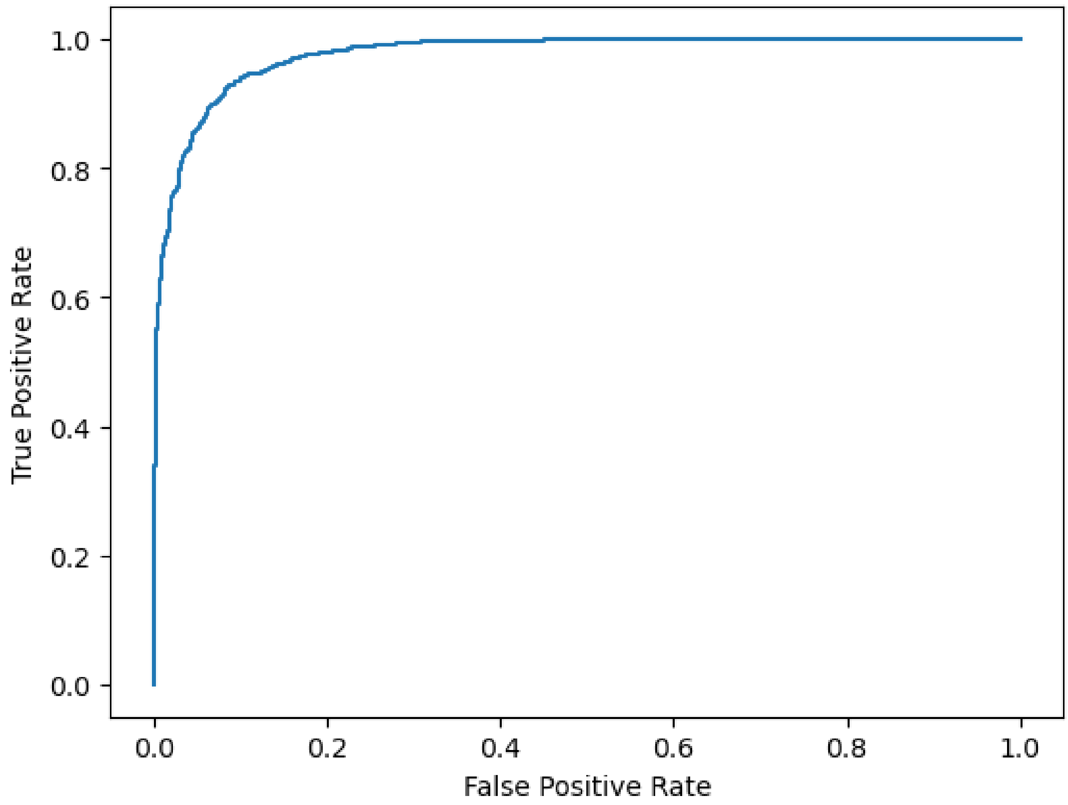
```
[64]: print(classification_report(y_test, pred))
```

```
              precision    recall  f1-score   support

          -1       0.91      0.91      0.91      1565
           1       0.93      0.93      0.93      2084

    accuracy                           0.92      3649
   macro avg       0.92      0.92      0.92      3649
weighted avg       0.92      0.92      0.92      3649
```

- Illustrate the diagnostic ability of this binary classifier by plotting the ROC curve.




- Validate the accuracy of data by the K-Fold cross-validation technique.

```
#view mean absolute error
mean(absolute(scores))
```

```
0.15413630301195455
```

```
# average absolute error between the model prediction and the actual observed data is 0.154
```

```
#view RMSE
sqrt(mean(absolute(scores)))
```

```
0.39260196511473877
```

```
# root mean squared error (RMSE) was 0.392
```

- The final output consists of the model, which will give maximum accuracy on the validation dataset with selected attributes.

```
[63]: #testing
      from sklearn.metrics import confusion_matrix, classification_report
      confusion_matrix(y_test, pred)
```

```
[63]: array([[1421,  144],
             [ 148, 1936]], dtype=int64)
```

```
[64]: print(classification_report(y_test, pred))
```

```
              precision    recall  f1-score   support

          -1       0.91      0.91      0.91      1565
           1       0.93      0.93      0.93      2084

    accuracy                           0.92      3649
   macro avg       0.92      0.92      0.92      3649
weighted avg       0.92      0.92      0.92      3649
```