```
import pandas as pd
import numpy as np

df=pd.read_csv('dataset.csv')

df
```

```
       index  having_IPhaving_IP_Address  URLURL_Length
Shortining_Service  \
0          1                           -1              1
1
1          2                            1              1
1
2          3                            1              0
1
3          4                            1              0
1
4          5                            1              0
-1
...      ...                          ...            ...
...
11050  11051                            1             -1
1
11051  11052                           -1              1
1
11052  11053                            1             -1
1
11053  11054                           -1             -1
1
11054  11055                           -1             -1
1

       having_At_Symbol  double_slash_redirecting  Prefix_Suffix  \
0                     1                        -1             -1
1                     1                         1             -1
2                     1                         1             -1
3                     1                         1             -1
4                     1                         1             -1
...                 ...                       ...            ...
11050                -1                         1              1
11051                -1                        -1             -1
11052                 1                         1             -1
11053                 1                         1             -1
11054                 1                         1             -1

       having_Sub_Domain  SSLfinal_State  Domain_registeration_length
...  \
0                     -1              -1                           -1
...
1                      0               1                           -1
...
```

```
2                    -1              -1                           -1
...
3                    -1              -1                            1
...
4                     1               1                           -1
...
...                  ...             ...                          ...
...
11050                 1               1                           -1
...
11051                 1              -1                           -1
...
11052                 1              -1                           -1
...
11053                -1              -1                            1
...
11054                -1              -1                            1
...

       popUpWidnow  Iframe  age_of_domain  DNSRecord  web_traffic
Page_Rank  \
0                1       1             -1         -1           -1
-1
1                1       1             -1         -1            0
-1
2                1       1              1         -1            1
-1
3                1       1             -1         -1            1
-1
4               -1       1             -1         -1            0
-1
...            ...     ...            ...        ...          ...
...
11050           -1      -1              1          1           -1
-1
11051           -1       1              1          1            1
1
11052            1       1              1          1            1
-1
11053           -1       1              1          1            1
-1
11054            1       1             -1          1           -1
-1

       Google_Index  Links_pointing_to_page  Statistical_report
Result
0                 1                       1                  -1      -
1
1                 1                       1                   1      -
1
```

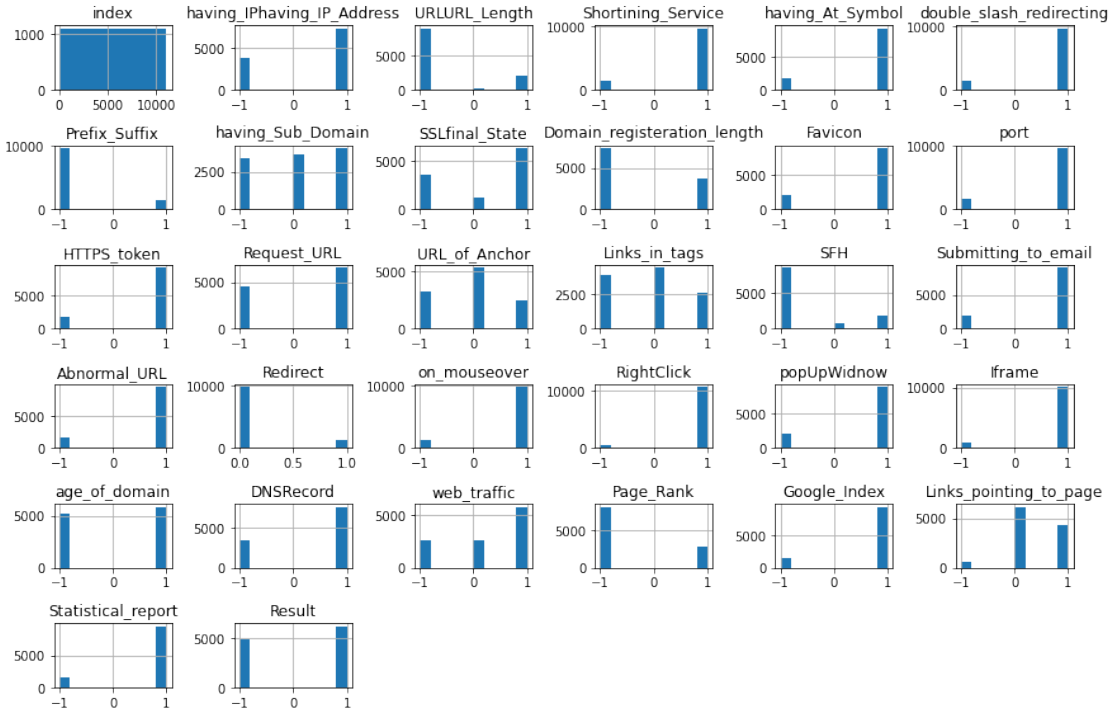
```
2                1                        0                   -1      -
1
3                1                       -1                    1      -
1
4                1                        1                    1
1
...            ...                      ...                  ...     ..
.
11050            1                        1                    1
1
11051            1                       -1                    1      -
1
11052            1                        0                    1      -
1
11053            1                        1                    1      -
1
11054           -1                        1                   -1      -
1

[11055 rows x 32 columns]
```

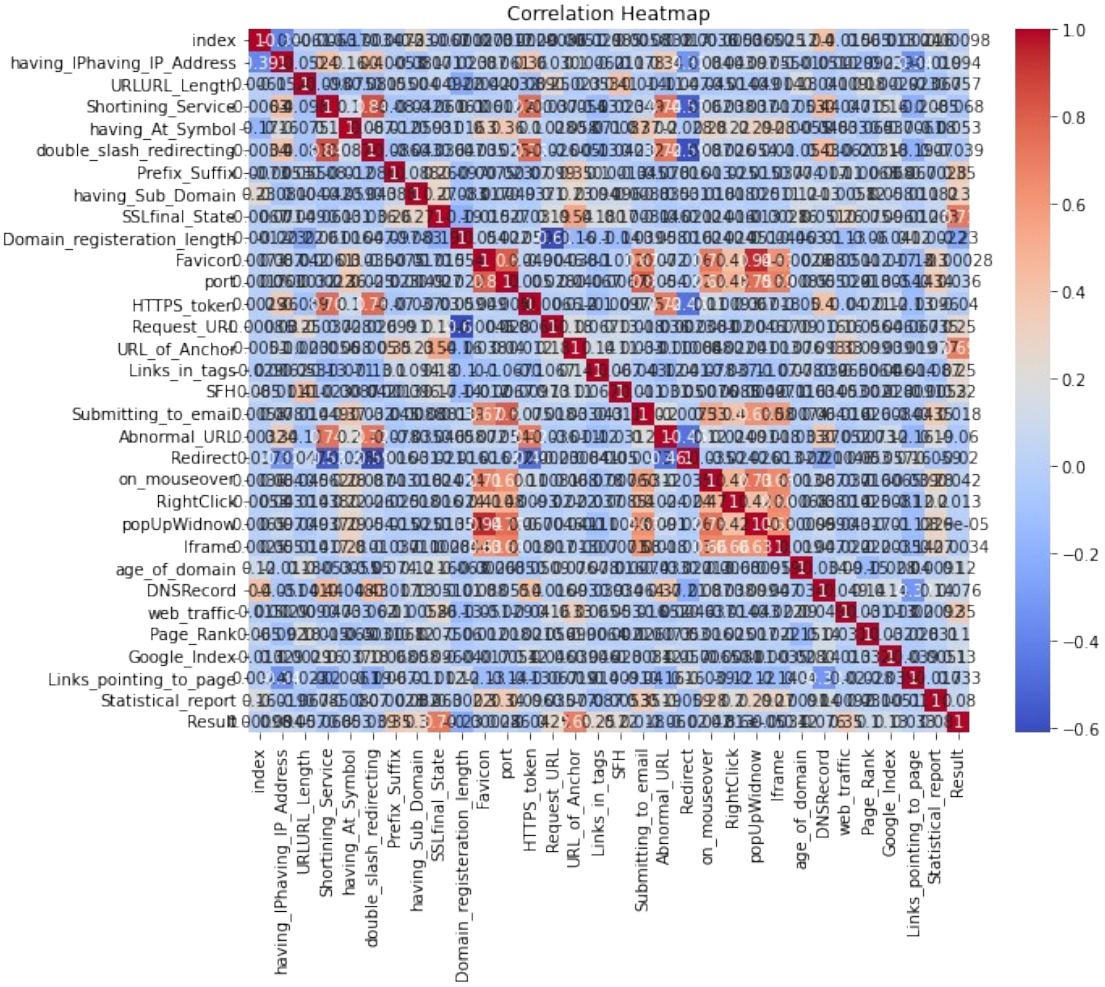
```
df.head()
```

```
   index  having_IPhaving_IP_Address  URLURL_Length
Shortining_Service  \
0      1                          -1              1
1
1      2                           1              1
1
2      3                           1              0
1
3      4                           1              0
1
4      5                           1              0                -
1

   having_At_Symbol  double_slash_redirecting  Prefix_Suffix  \
0                 1                        -1             -1
1                 1                         1             -1
2                 1                         1             -1
3                 1                         1             -1
4                 1                         1             -1

   having_Sub_Domain  SSLfinal_State  Domain_registeration_length  ...
\
0                 -1              -1                           -1  ...

1                  0               1                           -1  ...

2                 -1              -1                           -1  ...
```

```
3                 -1              -1                            1  ...

4                  1               1                           -1  ...


   popUpWidnow  Iframe  age_of_domain  DNSRecord  web_traffic
Page_Rank  \
0            1       1             -1         -1           -1
-1
1            1       1             -1         -1            0
-1
2            1       1              1         -1            1
-1
3            1       1             -1         -1            1
-1
4           -1       1             -1         -1            0
-1

   Google_Index  Links_pointing_to_page  Statistical_report  Result
0             1                       1                  -1      -1
1             1                       1                   1      -1
2             1                       0                  -1      -1
3             1                      -1                   1      -1
4             1                       1                   1       1

[5 rows x 32 columns]
```

```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
df = pd.read_csv('dataset.csv')
numeric_features = df.select_dtypes(include='number')
numeric_features.hist(figsize=(12, 8))
plt.tight_layout()
plt.show()
corr_matrix = df.corr()
plt.figure(figsize=(10, 8))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()
```


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




```
import pandas as pd
df = pd.read_csv('dataset.csv')
num_samples = len(df)
print("Number of samples:", num_samples)
for column in df.columns:
    unique_elements = df[column].nunique()
    print("Unique elements in", column, ":", unique_elements)
```

```
Number of samples: 11055
Unique elements in index : 11055
Unique elements in having_IPhaving_IP_Address : 2
Unique elements in URLURL_Length : 3
Unique elements in Shortining_Service : 2
Unique elements in having_At_Symbol : 2
Unique elements in double_slash_redirecting : 2
Unique elements in Prefix_Suffix : 2
Unique elements in having_Sub_Domain : 3
Unique elements in SSLfinal_State : 3
Unique elements in Domain_registeration_length : 2
Unique elements in Favicon : 2
```

```
Unique elements in port : 2
Unique elements in HTTPS_token : 2
Unique elements in Request_URL : 2
Unique elements in URL_of_Anchor : 3
Unique elements in Links_in_tags : 3
Unique elements in SFH : 3
Unique elements in Submitting_to_email : 2
Unique elements in Abnormal_URL : 2
Unique elements in Redirect : 2
Unique elements in on_mouseover : 2
Unique elements in RightClick : 2
Unique elements in popUpWidnow : 2
Unique elements in Iframe : 2
Unique elements in age_of_domain : 2
Unique elements in DNSRecord : 2
Unique elements in web_traffic : 3
Unique elements in Page_Rank : 2
Unique elements in Google_Index : 2
Unique elements in Links_pointing_to_page : 3
Unique elements in Statistical_report : 2
Unique elements in Result : 2
```

```
import pandas as pd
df = pd.read_csv('dataset.csv')
null_values = df.isnull().sum()
if null_values.any():
    print("Null values found in the following features:")
    print(null_values[null_values > 0])
else:
    print("No null values found in any features.")
```

```
No null values found in any features.
```

```
import pandas as pd
import numpy as np
df = pd.read_csv('dataset.csv')
corr_matrix = df.corr().abs()
correlation_threshold = 0.7
correlated_features = []
for i in range(len(corr_matrix.columns)):
    for j in range(i):
        if corr_matrix.iloc[i, j] > correlation_threshold:
            correlated_feature = corr_matrix.columns[i]
            correlated_features.append(correlated_feature)
df_filtered = df.drop(columns=correlated_features)
remaining_features = df_filtered.columns.tolist()
print("Remaining features after removing correlated features:")
print(remaining_features)
```

```
Remaining features after removing correlated features:
['index', 'having_IPhaving_IP_Address', 'URLURL_Length',
```

```
'Shortining_Service', 'having_At_Symbol', 'Prefix_Suffix',
'having_Sub_Domain', 'SSLfinal_State', 'Domain_registeration_length',
'Favicon', 'Request_URL', 'URL_of_Anchor', 'Links_in_tags', 'SFH',
'Redirect', 'RightClick', 'Iframe', 'age_of_domain', 'DNSRecord',
'web_traffic', 'Page_Rank', 'Google_Index', 'Links_pointing_to_page',
'Statistical_report']
```

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, precision_score,
recall_score, f1_score
df = pd.read_csv('dataset.csv')
url_text = df['URLURL_Length'].astype(str)
vectorizer = TfidfVectorizer()
url_features = vectorizer.fit_transform(url_text)
X = url_features.toarray()
y = df['_Shortining_Service']
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)
model = RandomForestClassifier()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)
print("Accuracy:", accuracy)
print("Precision:", precision)
print("Recall:", recall)
print("F1 Score:", f1)
```

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import roc_curve, auc
import matplotlib.pyplot as plt
df = pd.read_csv('dataset.csv')
url_text = df['URLURL_Length'].astype(str)
vectorizer = TfidfVectorizer()
url_features = vectorizer.fit_transform(url_text)
X = url_features.toarray()
y = df['Shortining_Service']
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)
model = RandomForestClassifier()
model.fit(X_train, y_train)
y_pred_prob = model.predict_proba(X_test)[:, 1]
fpr, tpr, thresholds = roc_curve(y_test, y_pred_prob)
```

```
auc_score = auc(fpr, tpr)
plt.plot(fpr, tpr, label='ROC Curve (AUC = {:.2f})'.format(auc_score))
plt.plot([0, 1], [0, 1], 'k--', label='Random')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.legend(loc='lower right')
plt.show()
```

```
import pandas as pd
from sklearn.model_selection import cross_val_score
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.ensemble import RandomForestClassifier
df = pd.read_csv('dataset.csv')
url_text = df['URLURL_Length'].astype(str)
vectorizer = TfidfVectorizer()
url_features = vectorizer.fit_transform(url_text)
X = url_features.toarray()
y = df['Shortining_Service']
model = RandomForestClassifier()
k = 5
accuracy_scores = cross_val_score(model, X, y, cv=k,
scoring='accuracy')
print("Accuracy scores for each fold:", accuracy_scores)
mean_accuracy = accuracy_scores.mean()
print("Mean accuracy:", mean_accuracy)
```

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
df = pd.read_csv('dataset.csv')
selected_attributes = ['URLURL_Length', 'Feature1', 'Feature2']
df_selected = df[selected_attributes]
url_text = df_selected['URLURL_Length'].astype(str)
vectorizer = TfidfVectorizer()
url_features = vectorizer.fit_transform(url_text)
X = url_features.toarray()
y = df['Shortining_service
X_train, X_val, y_train, y_val = train_test_split(X, y, test_size=0.2,
random_state=42)
models = [
    RandomForestClassifier(),
]

best_model = None
best_accuracy = 0.0

for model in models:
```

```
    model.fit(X_train, y_train)
    y_pred = model.predict(X_val)
    accuracy = accuracy_score(y_val, y_pred)
    if accuracy > best_accuracy:
        best_accuracy = accuracy
        best_model = model
print("Best Model:", best_model)
print("Accuracy on Validation Dataset:", best_accuracy)
```