ASWIN SUNDAR - 2015103006
RAMKUMAR B - 2015103042

PHISHING SITE DETECTION

CHROME EXTENSION

PHISHTOR

WHAT WE AIM TO DO

- A lite chrome plugin that aims to detect phishing websites and warn the user.
- Real time detection of phishing site.
- Offline detection and thus fast.
- Client side classification and thus better privacy.

WORKS THAT ARE SIMILAR TO THIS

TITLE	AUTHORS	METHODS USED	ADVANTAGES	LIMITATIONS
Intelligent phishing website detection using random forest classifier (IEEE-2017)	Abdulhamit Subasi, Esraa Molah, Fatin Almkallawi, Touseef J. Chaudhery	Random Forest Classifier	Random Forest has performed the best	-
PhishBox: An Approach for Phishing Validation and Detection (IEEE-2017)	Jhen-Hao Li Sheng-De Wang	Ensemble model and a detection model	False positive dropped	Low recall value
Real time detection of phishing websites (IEEE-2016)	Abdulghani Ali Ahmed Nurul Amirah Abdullah	URLs are inspected based on particular characteristics	Detect wide types of phishing	Compatibility with web browser
Certain investigation on web application security: Phishing detection and phishing target discovery (IEEE-2016)	R. Aravindhan R. Shanmugalakshmi K. Ramya	Various methodologies has been analysed.	An efficient mechanism was proposed	-

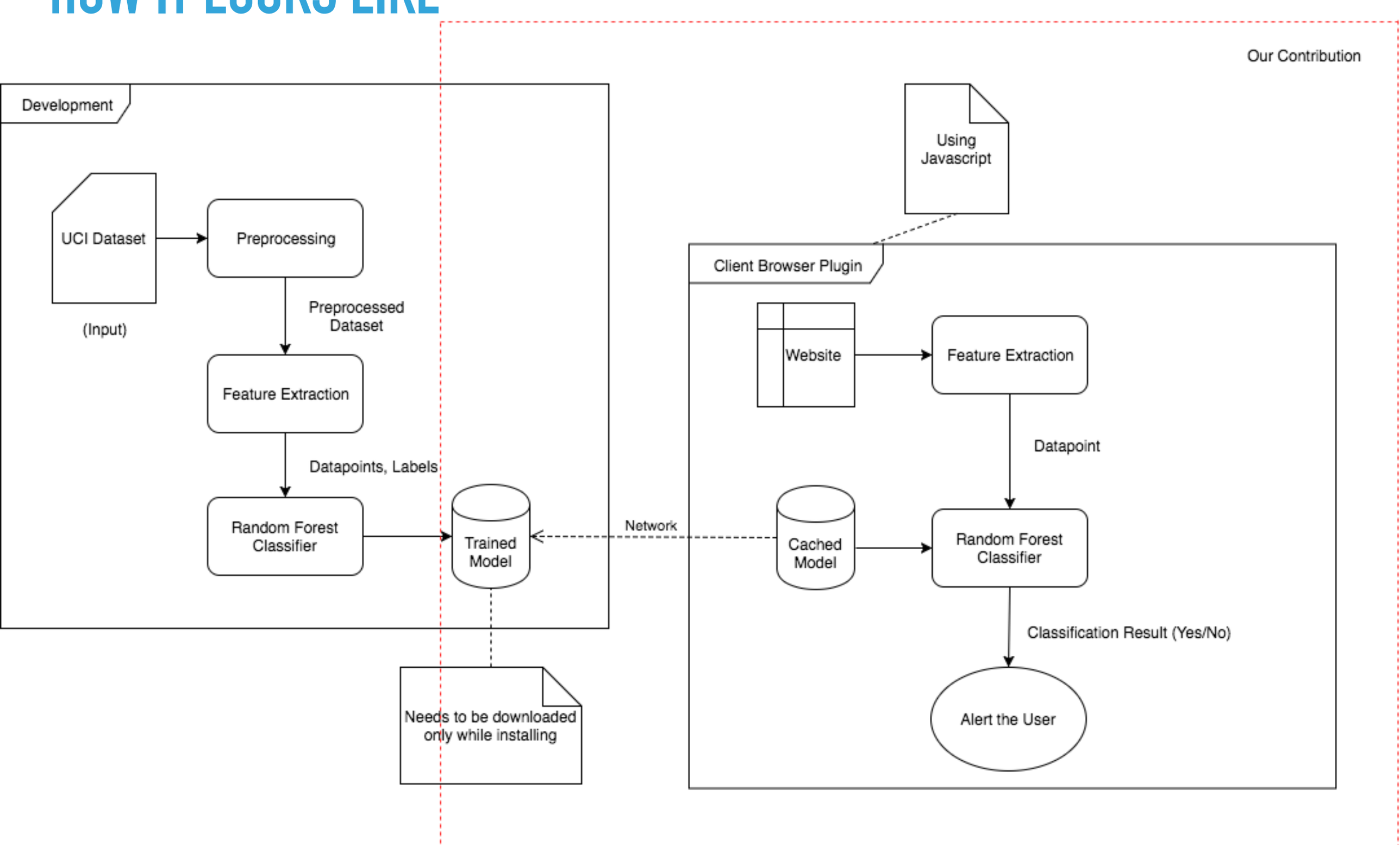
WORKS THAT ARE SIMILAR TO THIS

TITLE	AUTHORS	METHODS USED	ADVANTAGES	LIMITATIONS
Predicting Phishing Websites using Neural Networks (arXiv-2011)	A.Martin Na.Ba.Anutthama	Neural Networks	Framework with low error rate.	Not a significant improvement
Phishing Detection Using Neural Network (Stanford-2012)	Ningxia Zhang Yongqing Yuan	3 layer feedforward NN	Highest recall	Didn’t work for ham emails
Phishing Dynamic Evolving Neural Fuzzy Framework (arXiv-2013)	Ammar Almomani B.B.Gupta Tat Chee Wan	Neural Fuzzy framework	Effective with Zero days attack	Computationally heavy
branding a phishing website using advanced pattern matching (Google patent-2013)	Gary Warner Bradley Wardman	Pattern Matching	Efficient in real time	-

ISSUE THAT WE SOLVE

- Not implemented for day to day usage
- Too heavy to be done in client side
- Browser plugin with ML prediction mechanism
- No Real time and rapid detection
- No Privacy and network dependent

HOW IT LOOKS LIKE



PREPROCESSING

- **INPUT:** Dataset in arff format (attribute relation file format)
- **OUTPUT:** Training and testing data in npy format (numpy)
- **METHODOLOGY:** python arff library to load dataset and scikit learn train_test_split for splitting the dataset.
- **EVALUATION METRIC:** cross validation on training set

PSEUDOCODE

```
start
load dataset into numpy array
examine dataset characteristics
split training and test set
export to npy file
end
```

TRAINING

- **INPUT:** Training and testing data npy format (numpy)
- **OUTPUT:** Trained random forest classifier model
- **METHODOLOGY:** Scikit learn Random forest classifier trained on training set and exported in json format.
- **EVALUATION METRIC:** F1 score on testing set

PSEUDOCODE

```
start
load training set from file
calculate cross validation score
train random forest classifier
load testing set from file
Calculate f1 score on testing set
end
```

EXPORT LEARNED PARAMETERS

- **INPUT:** Learned Classifier python object
- **OUTPUT:** JSON file with model parameters
- **METHODOLOGY:** Loop through all estimators and nodes in each.
- **EVALUATION METRIC:** NIL-

PSEUDOCODE

```
start
get no of estimators
For each estimator (decision tree)
  Loop through it's nodes
  Get node threshold or label
  write to JSON stream
write JSON stream to file
end
```

FEATURE RETRIEVAL – PLUGIN

- **INPUT:** Website viewed by the user.
- **OUTPUT:** 17 features needed for classification.
- **METHODOLOGY:** Javascript DOM API to extract direct features and indirect features are calculated .
- **EVALUATION METRIC:** -

PSEUDOCODE

```
start
obtain URL and DOM of the website
extract url features
extract direct DOM features
calculate indirect features
end
```

PREDICTION – PLUGIN

- **INPUT:** 17 features extracted from the website
- **OUTPUT:** Alert/Popup to the user.
- **METHODOLOGY:** Manual Javascript implementation of the scikit-learn random forest predict function.
- **EVALUATION METRIC:** -

PSEUDOCODE

```
start
load model parameters
load features
reimplement sklearn predict
if seems like phishing
    alert the user
end
```

PREPROCESSING

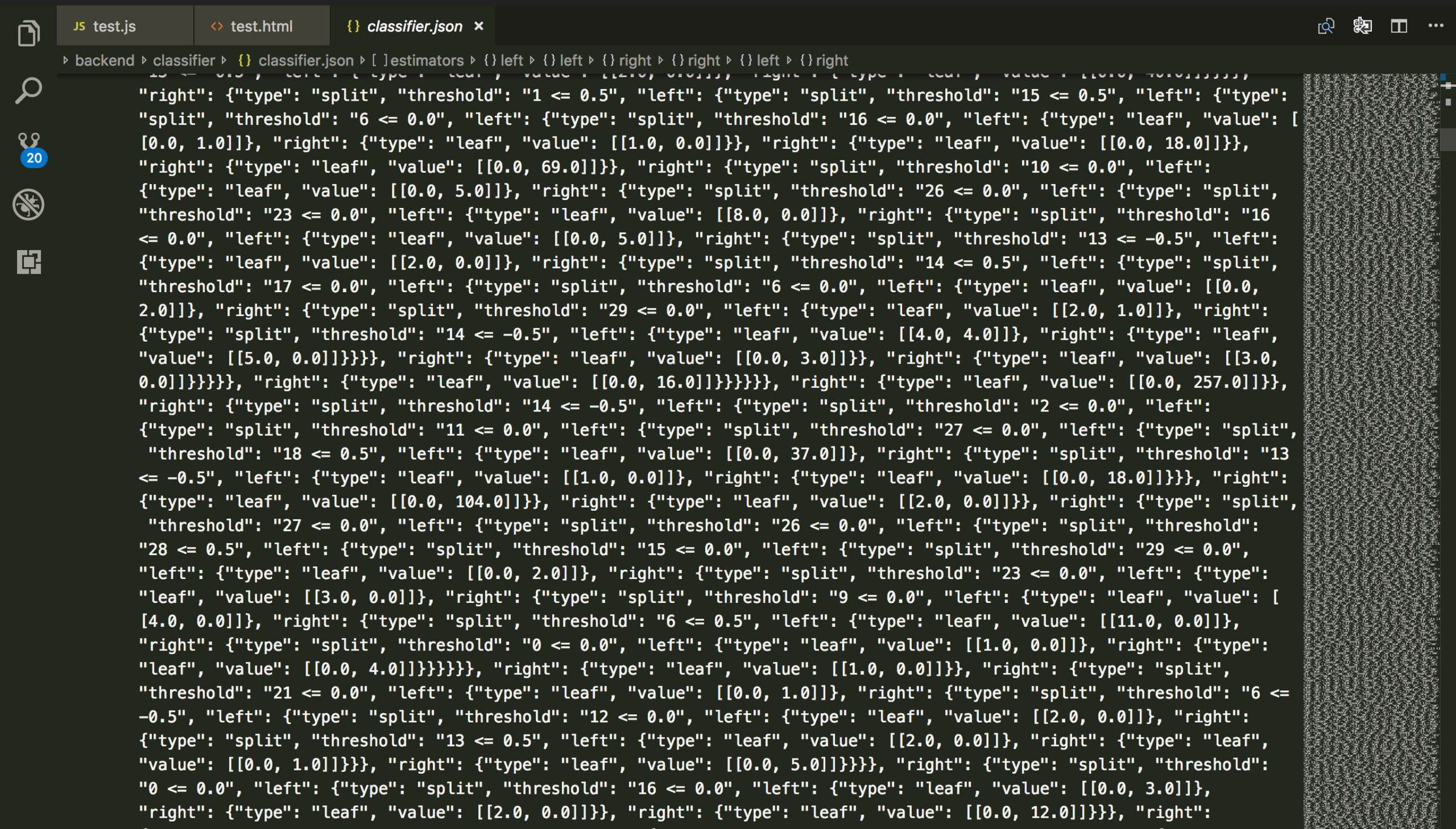
```
~/D/m/phishing_detector  *+  backend/dataset  python3 preprocess.py                Sun Sep 16 19:31:05 2018
The dataset has 11055 datapoints with 30 features
Features: ['having_IP_Address', 'URL_Length', 'Shortining_Service', 'having_At_Symbol', 'double_slash_redirecting', 'Pre
fix_Suffix', 'having_Sub_Domain', 'SSLfinal_State', 'Domain_registeration_length', 'Favicon', 'port', 'HTTPS_token', 'Re
quest_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']
Before spliting
X:(11055, 17), y:(11055,)
After spliting
X_train:(7738, 17), y_train:(7738,), X_test:(3317, 17), y_test:(3317,)
Saved!
Test Data written to testdata.json
```

TRAINING

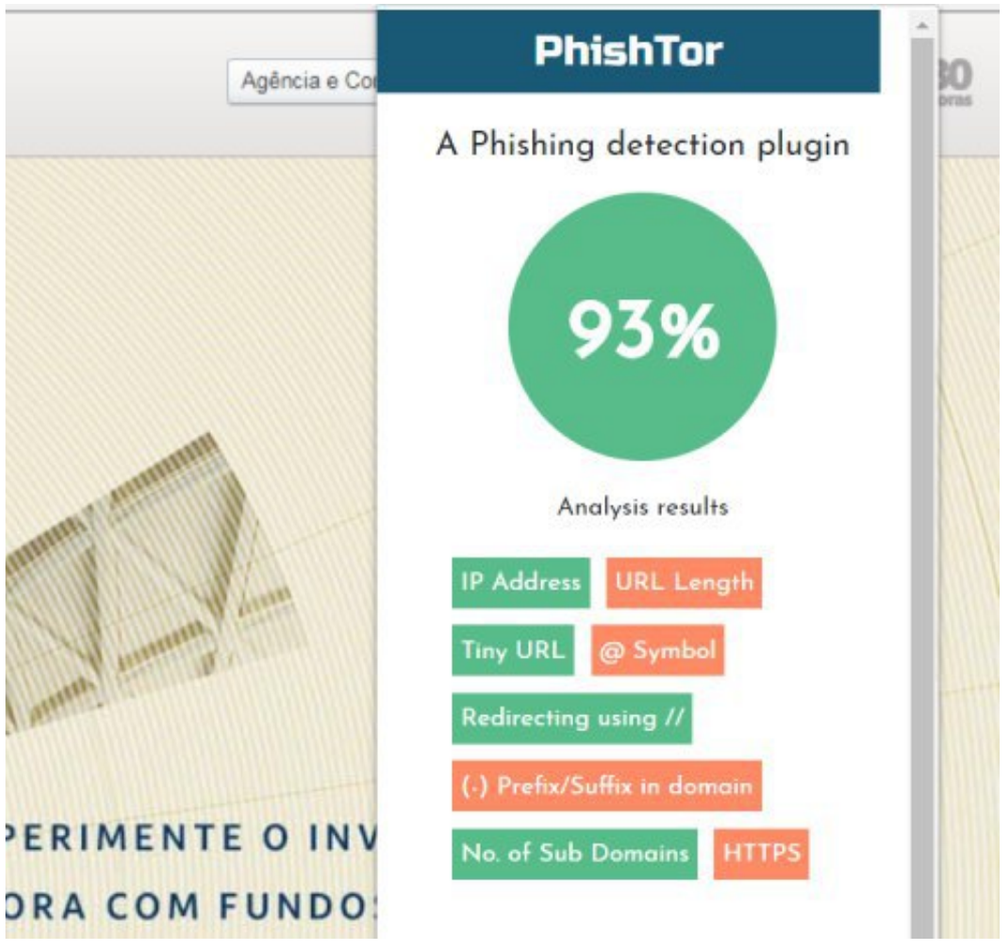
```
~/D/m/phishing_detector  *+  backend/classifier  python3 training.py              Fri Oct 26 13:03:31 2018
/usr/local/lib/python3.7/site-packages/sklearn/ensemble/weight_boosting.py:29: DeprecationWarning: numpy.core.umath_test
s is an internal NumPy module and should not be imported. It will be removed in a future NumPy release.
  from numpy.core.umath_tests import inner1d
X_train:(7738, 17), y_train:(7738,)
Cross Validation Score: 0.9455923597113163
Accuracy: 0.9469400060295448
```

OUTPUT

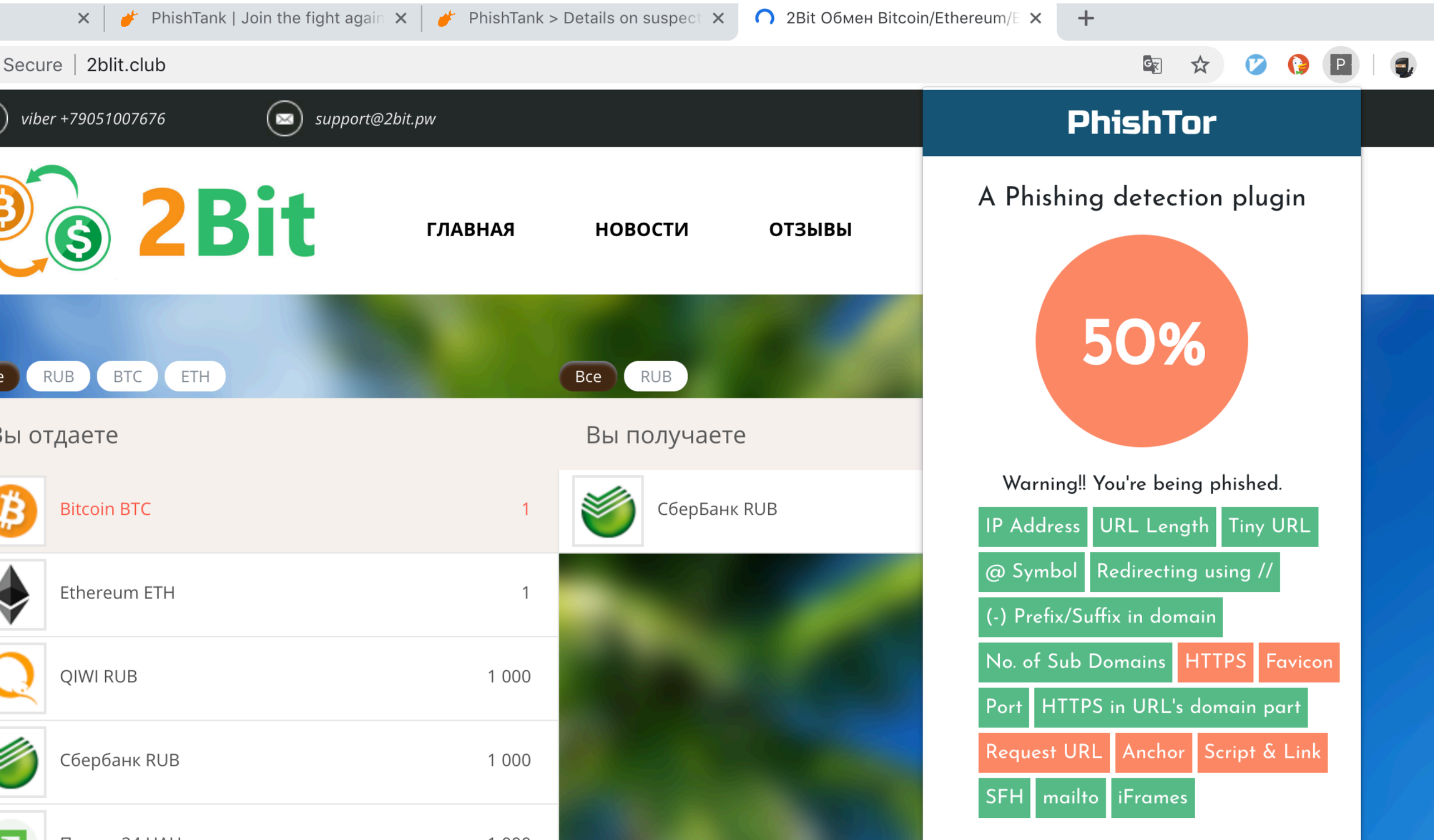
CLASSIFIER DUMP



FEATURE EXTRACTION – PLUGIN



FINAL OUTPUT

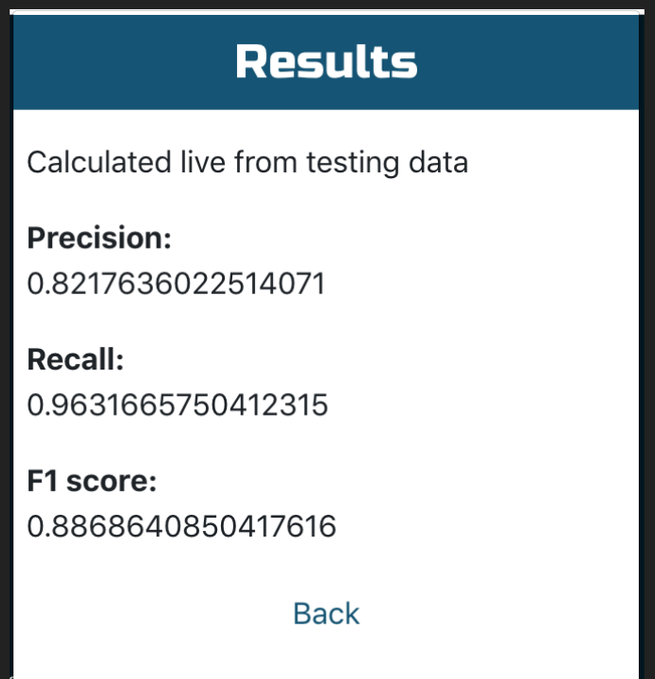


HOW IT PERFORMED?

- The score on 10 Fold cross validation is as below

```
Cross Validation Score: 0.9455923597113163
```

- The precision, recall and F1 score of the phishing classifier is calculated manually using javascript on the test data set. The results are shown.



REFERENCES

- A. Subasi, E. Molah, F. Almkallawi, and T. J. Chaudhery, “Intelligent phishing website detection using random forest classifier,” 2017 International Conference on Electrical and Computing Technologies and Applications (ICECTA), Nov. 2017.
- “UCI Machine Learning Repository: Phishing Websites Data Set,” UCI Machine Learning Repository: Breast Cancer Wisconsin (Diagnostic) Data Set. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/phishing websites.
- J.-H. Li and S.-D. Wang, “PhishBox: An Approach for Phishing Validation and Detection,” 2017 IEEE 15th Intl Conf on Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence and Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/ CyberSciTech), 2017.