**Autor: Rodrigo Duarte Xavier da Costa**

**Road Map**

1 - entender o problema e ler o artigo de referência;  
2 - entender o significado de cada coluna e valor;

3 - fazer a análise descritiva dos dados;

4 - desenvolver os algoritmos (random forest, SVM, logistic regression, NN);

5 - condensar os resultados em gráficos;

6 - condensar os pontos principais num word;

7 - gravar o vídeo.

**Objetivo**

Detectar sites maliciosos que roubam dados dos usuários a partir de algoritmos de classificação.

**Dados**

são 1353 exemplos e 10 features

**Legitimate** (valor = 1): São 548 exemplos de 1353, **40,5%**  
**Suspicious** (valor = 0) São 702 exemplos de 1353, **7,61%**

**Phishy** (valor = -1) São 103 exemplos de 1353, **51,88%**

**não há dados faltantes.**

As informações acima indicam que a distribuição das proporções de cada classe são satisfatórias para um bom modelo, visto que há poucos valores suspeitos (o que não é o objetivo principal do modelo) e a proporção dos valores são “próximos” entre legítimos e maliciosos, evitando um viés para o modelo.

No artigo de referência é utilizado uma técnica de classificação que usa de ‘if’s e ‘then’s com limiares para montar o banco de dados com as classes -1, 0 e 1. Essa transformação simplifica os dados, porém é bem possível que ele prejudique a qualidade dos modelos, principalmente se o modelo for levado para ser aplicado na prática.

**Features**

Segundo o artigo de referência do data set, as colunas contém o seguinte significado:

**URL Anchor**: Links dentro da página, para acessar recursos ou outras partes da página podem apontar para um domínio diferente da do URL.

**Request URL**: Objetos da página que são carregadas em servidores/domínios diferentes da URL da página.

**SFH**: “*Once the user submitted his information; the webpage will transfer the information to a server so that it can process it. Normally, the information is processed from the same domain where the webpage is being loaded. Phishers resort to make the server form handler either empty or the information is transferred to somewhere different than the legitimate domain*.”

**URL length:** Tamanho da URL

**Having @:** O URL conter um ‘@’

**Prefix/Suffix**: Uso de um sufixo ou prefixo para legitimar a URL

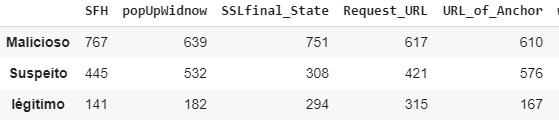
**IP:** Condição de que haja o IP na URL

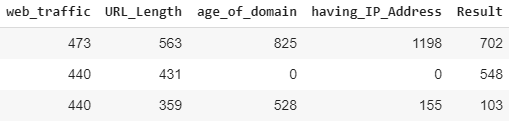
**Sub-domains:** Subdomínios nas URL

**Web traffic:** Quantidade de acessos ao site

**Domain age**: Tempo de presença online dos sites

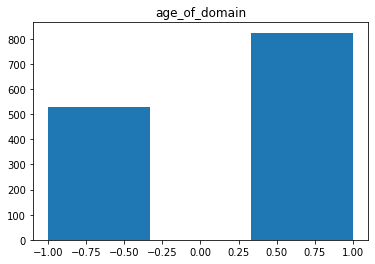
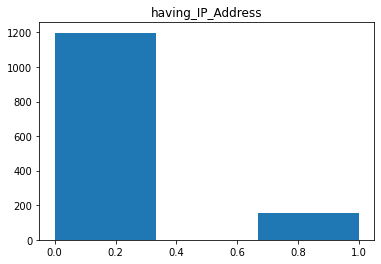
A distribuição das classes em cada feature pode ser observada a seguir





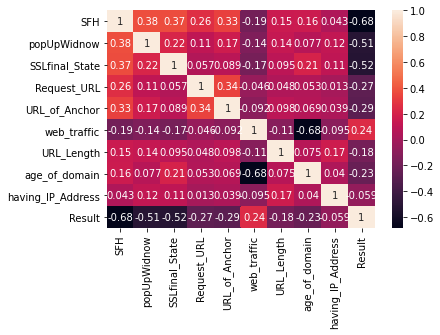
De forma visual agora:





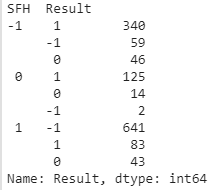
É possível inferir algumas suposições, as features que tem uma distribuição próxima entre as classes “atrapalhará” o nosso modelo a decidir, diferentemente da proporção da feature resultado, é desejável haver um viés de determinado feature para algum classe. Seguindo essa linha, podemos observar que as features **web\_traffic** e **URL\_Length** apresentam os número de exemplos de todos as classes próximos podendo ser features ruins para o modelo.Seguindo a linha de pensamento contrária, as classes que claramente tem um viés são **SFH**, **SLLfinalstate**, **Request URL**, **Age of domain**.

Outra observação que podemos fazer é a partir do mapa de correlação

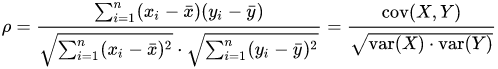


Numa análise superficial se vê que a maioria é uma correlação negativa que são fracas ou desprezíveis. Porém temos 3 features que tem uma correlação moderada com a feature resultado, são **SFH**, **popUpWindown**, **SSL\_finalstate**. há apenas uma correlação positiva,a feature **Web Trafic** mas é fraca.. **having\_ip\_adress** tem uma correlação desprezivel.

Como se tem apenas 3 valores em cada feature, é possível inferir que uma corr negativa indica que 1 leva a -1 e vice-versa, isso pode ser visto analisando a distribuição de valores da correlação entre **SFH**  e **Result**. O que é controverso, pois a indicação de que algo é legítimo em uma feature levar ao resultado de uma site malicioso.



Dito isto, o mapa de correlação é feito sobre o coeficiente de correlação pearson e é calculado com a seguinte fórmula abaixo:



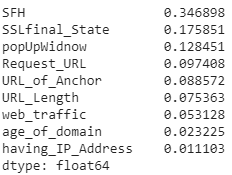
O que dá espaço para levantar a hipótese de que a quantidade grande de valores 0 nas features estejam levando o cálculo a valores mais negativos do que eles realmente deveriam ser. Então é possível que este mapa de correlação não seja totalmente confiável.

**Escolha dos modelos**

1. **Random Forest**

Eu escolhi o random forest por alguns motivos, o primeiro é que ele se apresenta um algoritmo de classificação melhor do que a árvore de decisão, já que o conjunto de dados é pequeno, é esperado que seu desempenho seja bom, principalmente pelas diversas ramificações dos valores das features entre -1, 0 e 1, podendo ser trabalhado cada um em uma “árvore”. Outro fator importante é que no artigo original, um dos objetivos é identificar features que impactam a detecção dos sites e é possível fazer isso com o random forest.

Como é possível ver abaixo, o algoritmo identificou que **age\_of\_domain** e **having\_ip\_adress** tem uma importância desprezível para o modelo, portanto o estudo será realizado a partir da análise do dataset com e sem essas features.



1. **SVM**

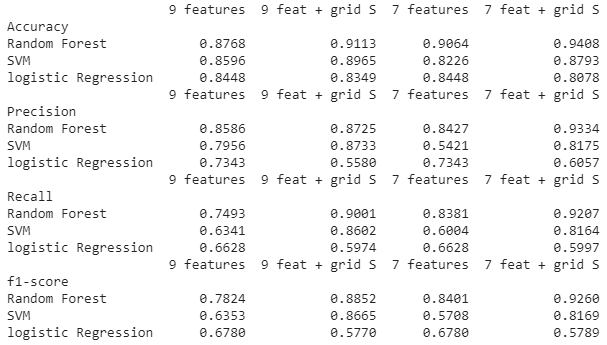
A escolha do SVM, além de ser conhecido que seu desempenho é bom, se deu justamente por conta que os dados são de apenas 3 classes, facilitando a escolha do hiperplano que abrange os pontos, um aspecto que poderia facilitar seu desempenho é o número baixo de features.

1. **Logistic Regression**

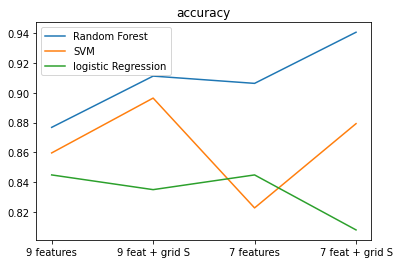
Outro importante e poderoso classificador é a regressão logística, a escolha dele é óbvia pelo seu peso em classificação binária ou multi classe, como só temos 3 classes, é uma tarefa fácil para esse modelo.

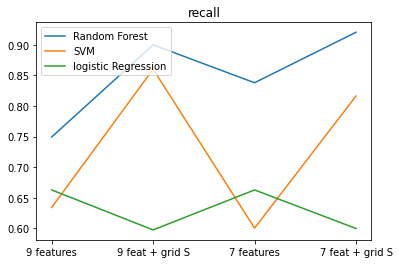
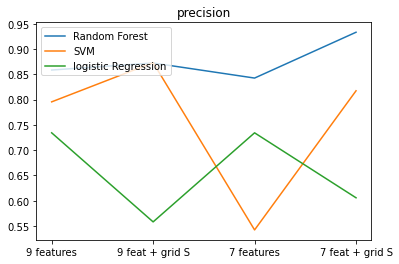
**Resultados**

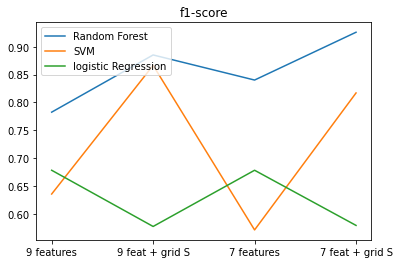
Foi-se utilizado a técnica de grid search para encontrar os melhores parâmetros dentro de um range pré-definido, isso demonstrou uma melhora substancial entre todos os modelos construídos.



A análise se torna mais valiosa se for visual, portanto eu organizei esses resultados em 4 gráficos.





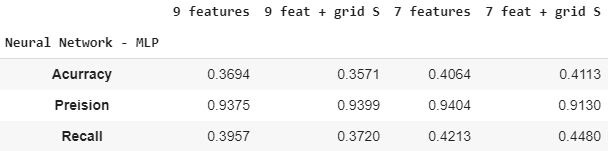


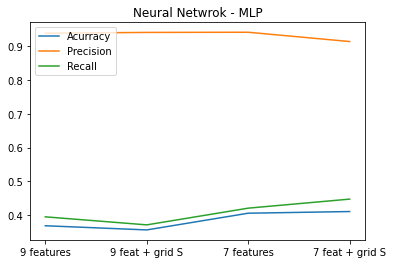
É possível ver que o Random Forest teve um desempenho superior aos outros, a tendência dos valores dos modelos se tornou igual para cada métrica de desempenho. É interessante analisar também como se deu os outros modelos escolhidos.

O modelo SVM teve uma melhora com as 9 features e grid search, porém ao ser retirado as 2 últimas features, ele teve uma queda na qualidade, o resultado recupera novamente utilizando com grid search, mas esse movimento nos mostra que estávamos enganados ao fazer a mesma abordagem de se retirar 2 features igual ao Random Forest.

O modelo da regressão logística, teve um desempenho abaixo do esperado, o uso do grid search no modelo não foi satisfatório, então um ponto para se poder aprofundar é do porque que o grid search não fez efeito no modelo.

Se construiu também um modelo de rede neural (MLP), porém seus resultados foram os mais estranhos do estudo, segue abaixo os resultados.





Como é possível ver, a métrica precision é praticamente uma constante, enquanto as outras duas que deveriam ser boas, mostram um péssimo desempenho, existe algumas hipóteses sobre isso.

1. A escolha das funções de ativação, função perda e o otimizador não foram as mais adequadas para o problema;
2. O conjunto de dados contendo apenas 3 valores dificulta o modelo a reconhecer os melhores pesos dos neurônios;
3. A métrica Precision tenha resultado nesse valor constante por uma questão da forma como ela é calculada, podendo assim não ser uma boa métrica para avaliar esse problema.

Em resumo, é visto que para o problema proposto o modelo **Random forest** mostrou um melhor desempenho que os outros, a utilização de menos features levou à uma súbita melhora, seu método ensemble de árvores de decisão trabalha bem as diversas combinações de valores entre features. Outra portanto questão é que a qualidade do modelo depende de como o conjunto de dados foi criado, vimos que houve uma normalização, caso os dados não fossem normalizados antes, talvez os modelos como NN e logistic regression tivessem um desempenho melhor, inclusive, melhor até do que o Random forest.

**Links utilizados**

1. <https://datascience.stackexchange.com/questions/64441/how-to-interpret-classification-report-of-scikit-learn>
2. <https://www.datacamp.com/community/tutorials/random-forests-classifier-python#advantages>
3. <https://towardsdatascience.com/hyperparameter-tuning-the-random-forest-in-python-using-scikit-learn-28d2aa77dd74>
4. <https://medium.com/airbnb-engineering/confidence-splitting-criterions-can-improve-precision-and-recall-in-random-forest-classifiers-ad2d4ba696a4#:~:text=Both%20in%20terms%20of%20classifying,Random%20Forest%20Classifier%20(RFC).>
5. <https://medium.com/analytics-vidhya/evaluating-a-random-forest-model-9d165595ad56>
6. <https://towardsdatascience.com/understanding-support-vector-machine-part-1-lagrange-multipliers-5c24a52ffc5e>
7. <https://towardsdatascience.com/understanding-support-vector-machine-part-2-kernel-trick-mercers-theorem-e1e6848c6c4d>
8. <https://www.datacamp.com/community/tutorials/svm-classification-scikit-learn-python>
9. <https://scikit-learn.org/stable/modules/svm.html>
10. <https://www.dezyre.com/recipes/optimize-hyper-parameters-of-logistic-regression-model-using-grid-search-in-python>
11. <https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html>
12. <https://towardsdatascience.com/logistic-regression-model-tuning-with-scikit-learn-part-1-425142e01af5>
13. <https://keras.io/api/metrics/>
14. <https://keras.io/api/models/model/>