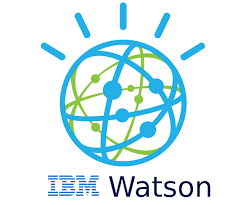
**Machine Learning Model Deployment with IBM Cloud Watson Studio**



**Innovation Phase 2**

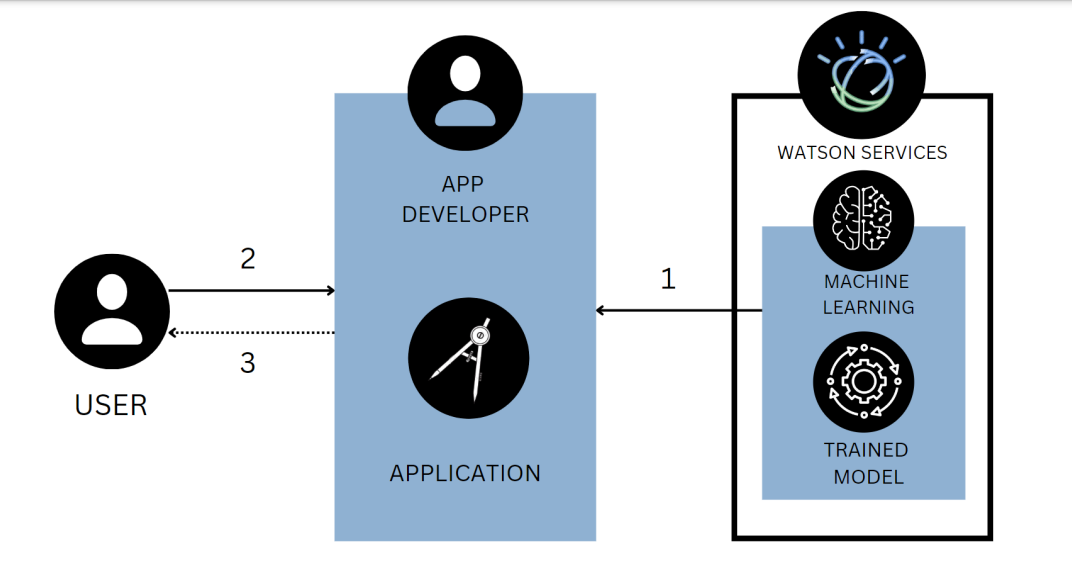
|  |  |
| --- | --- |
| **Date** | 10 October 2023 |
| **Project Name** | Machine Learning Model Deployment with IBM Cloud Watson Studio |
| **Team Member** | 1.HALISH RICHARD J (**TL**)  2. MOHAMED ATHIL M  3. ARUSAMY A  4. RAHUL DRAVID A |

**Innovations and Enhancements**

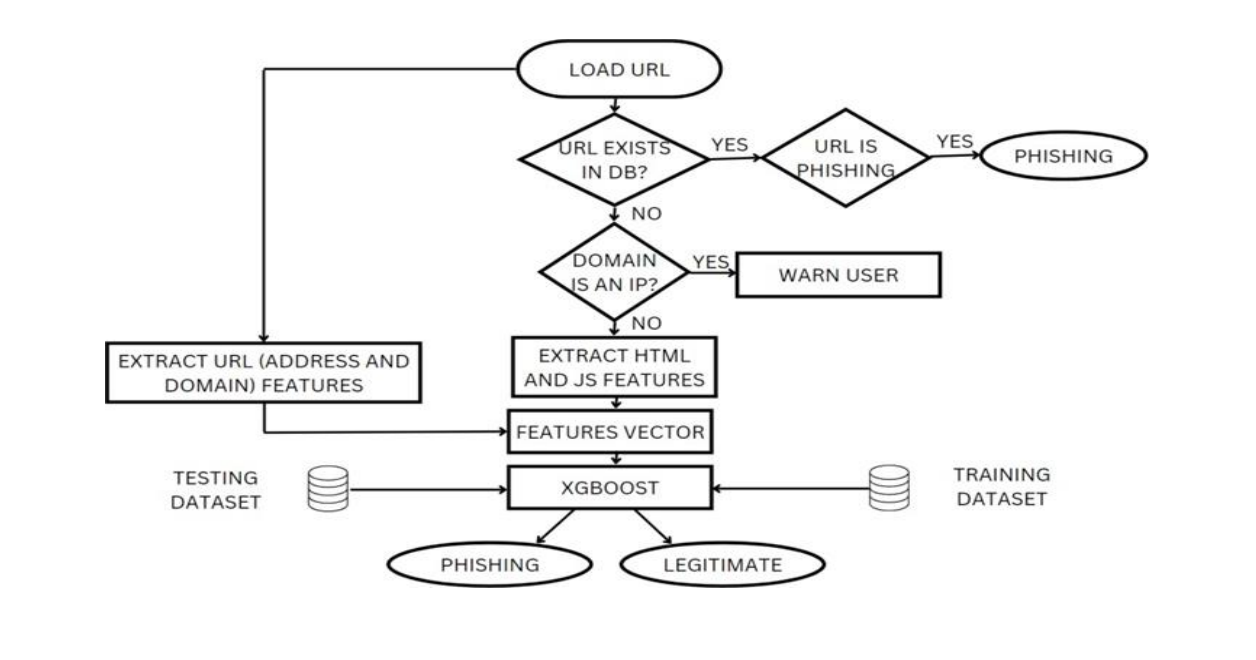
Our solution is to build an efficient and intelligent system to detect phishing sites by applying a machine learning algorithm which implements classification algorithms and techniques to extract the phishing datasets criteria to classify their legitimacy by carefully analysing and identifying various factors that could be used to detect a phishing site. These factors fall under the categories of address bar-based features, domain-based features, HTML & JavaScript based features. Using these features, we can identify a phishing site with high accuracy

**Technical Architecture:**

Technical architecture which is also often referred to as application architecture includes the major components of the system, their relationships, and the contracts that define the interactions between the components. The goal of technical architects is to achieve all the business needs with an application that is optimized for both performance and security.

****

**Fig 1**

****

**Fig 2**

1. The application developer builds a Python-based app and deploys it.

2. The user enters the URL of a website in the application to check for its genuineness.

3. The user submits the URL through the web-based application and gets back the result.

4. The user makes a decision whether to proceed surfing in that website or move to another one.

**CODING & SOLUTIONING**

**Classification of URL:**

The primary feature of this project is to classify the given URL as phishing or benign. Various classification algorithms are used to achieve this.

**Data collection:**

URL features of legitimate websites and phishing websites were collected. The data set consists of total 11,055 URLs which include 6,157 legitimate URLs and 4,898 phishing URLs. Legitimate URLs are labelled as “1” and phishing URLs are labelled as “-1”. The features that are present in the data set include:

• IP Address in URL

• Length of URL

• Using URL Shortening Services

• "@" Symbol in URL

• Redirection "//" in URL

• Prefix or Suffix "-" in Domain

• Having Sub Domain

• Length of Domain Registration

• Favicon

• Port Number

• HTTPS Token

• Request URL

• URL of Anchor

• Links in Tags

• SFH

• Email Submission

• Abnormal URL

• Status Bar Customization (on mouse over)

• Disabling Right Click

• Presence of Popup Window

• IFrame Redirection

• Age of Domain

• DNS Record

• Web Traffic

• Page Rank

• Google Index

• Links pointing to the page

• Statistical Report

• Result

Using IBM Cloud Storage this data is accessed throughout the project. The code written below is used to import the dataset.

import os, types

import pandas as pd

from botocore.client import Config

import ibm\_boto3

def \_\_iter\_\_(self): return 0

# The following code accesses a file in your IBM Cloud Object

Storage. It includes your credentials.

# You might want to remove those credentials before you share the

notebook.

cos\_client = ibm\_boto3.client(service\_name='s3',

 ibm\_api\_key\_id='',

 ibm\_auth\_endpoint="https://iam.cloud.ibm.com/oidc/token",

 config=Config(signature\_version='oauth'),

 endpoint\_url='https://s3.private.us.cloud-objectstorage.appdomain.cloud')

bucket = 'webphishingdetection-donotdelete-pr-icmjtvktnzli2s'

object\_key = 'dataset\_website.csv'

body = cos\_client.get\_object(Bucket=bucket,Key=object\_key)['Body']

# add missing \_\_iter\_\_ method, so pandas accepts body as file-like

object

if not hasattr(body, "\_\_iter\_\_"): body.\_\_iter\_\_ = types.MethodType(

\_\_iter\_\_, body )

data0 = pd.read\_csv(body)

data0.head()

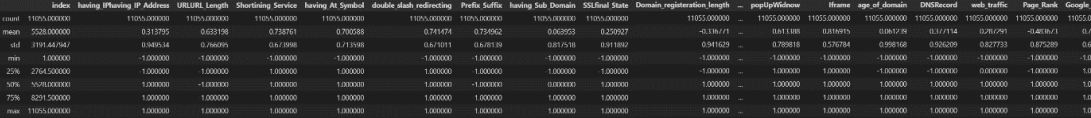
**Data pre-processing and Exploratory Data Analysis:**

Few plots and graphs were drawn to find how the data is distributed and the how features are related to each other.

**Univariate analysis:**

Univariate analysis provides an understanding in the characteristics of each feature in the data set. Different characteristics are computed for numerical and categorical data. For the numerical features characteristics are standard deviation, skewness, kurtosis, percentile, interquartile range (IQR) and range. For the categorical features characteristics are count, cardinality, list of unique values, top and freq.

data0.describe()

****

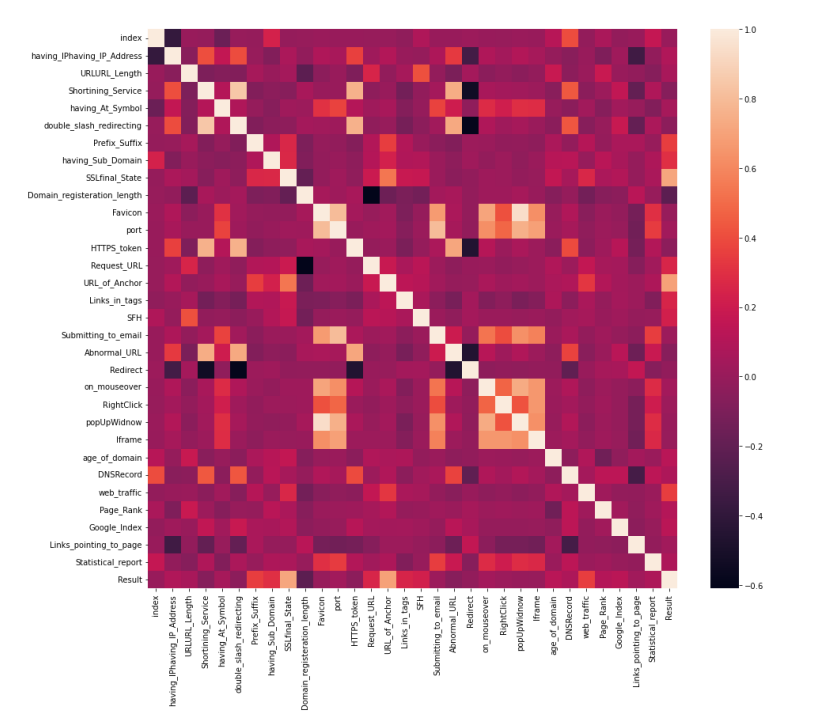
**Fig 3**

**Bivariate analysis:**

plt.figure(figsize=(15,13))

sns.heatmap(data0.corr())

plt.show()

****

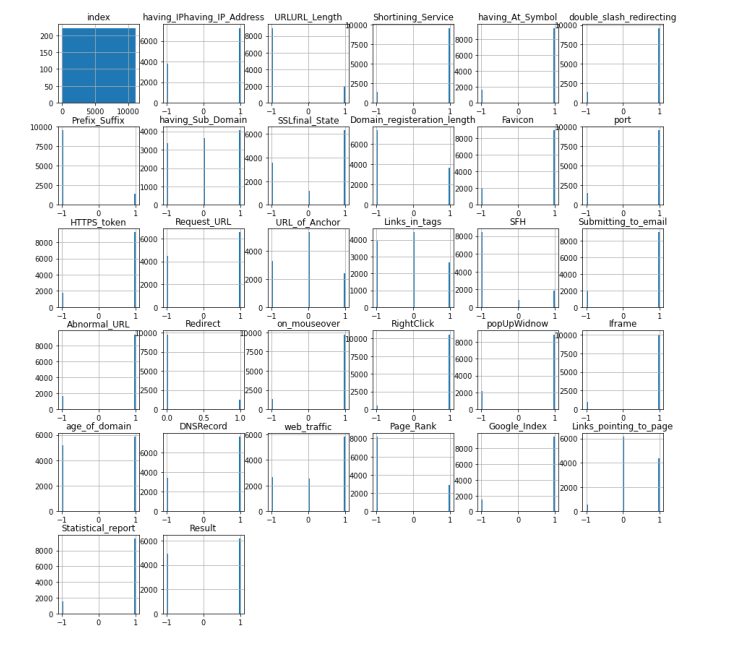
**Fig 4**

From this correlation matrix, it is evident that there is no correlation with many features. So, it is crucial to eliminate these features.

**Multivariate analysis:**

data0.hist(bins = 50,figsize = (15,15))

plt.show()

****

**Fig 5**

From data distribution graph and correlation matrix, we can conclude that the following features do not have much impact on the result:

• having\_Sub\_Domain

• Domain\_registeration\_length

• Favicon

• Request\_URL

• URL\_of\_Anchor 21

• Links\_in\_tags

• Submitting\_to\_email

• Redirect

• web\_traffic

• Page\_Rank

• Google\_Index

• Links\_pointing\_to\_page

All the above features will not be included in further processing

#Removing the features which do not have much impact on Result

data=data0.iloc[:,[1,2,3,4,5,6,12,20,21,22,23,24,25,30,31]]

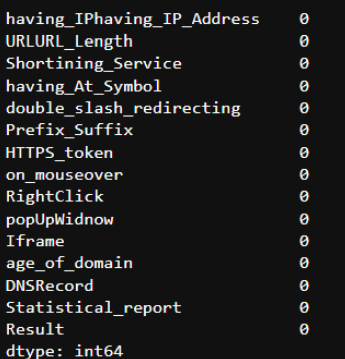
data.head()

**Checking for null values:**

This dataset doesn’t contain any null values.

#checking the data for null or missing values

data.isnull().sum()

****

**Fig 6**

**Model building:**

From the dataset above, it is clear that this is a supervised machine learning task. There are two major types of supervised machine learning problems, called classification and regression. 22 This data set comes under classification problem, as the input URL is classified as phishing (-1) or legitimate (1). The supervised machine learning models (classification) considered to train the dataset in XGBoost

**XGBoost:**

XGBoost is one of the most popular machine learning algorithms these days. XGBoost stands for eXtreme Gradient Boosting. Regardless of the type of prediction task at hand; regression or classification. XGBoost is an implementation of gradient boosted decision trees designed for speed and performance.

#XGBoost Classification model

from xgboost import XGBClassifier

import warnings

warnings.filterwarnings("ignore", category=UserWarning)

# instantiate the model

xgb = XGBClassifier(learning\_rate=0.4,max\_depth=7,verbosity = 0)

#fit the model

xgb.fit(X\_train, y\_train)

#predicting the target value from the model for the samples

y\_test\_xgb = xgb.predict(X\_test)

y\_train\_xgb = xgb.predict(X\_train)

#computing the accuracy of the model performance

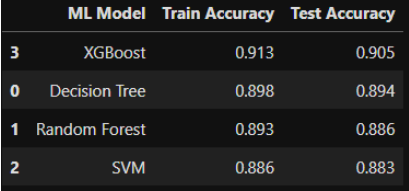
acc\_train\_xgb = accuracy\_score(y\_train,y\_train\_xgb)

acc\_test\_xgb = accuracy\_score(y\_test,y\_test\_xgb)

print("XGBoost: Accuracy on training Data:

{:.3f}".format(acc\_train\_xgb))

print("XGBoost : Accuracy on test Data: {:.3f}".format(acc\_test\_xgb))

****

**Fig 7**

Thus, accuracy for all the four used models were calculated and ranked. XGBoost performed better than other models.

**Github:**

<https://github.com/Halishrichard17/Cloud-Application-Development-NM>