NAME : KRISH GUPTA
ROLL NO : 60
D15C

EXPERIMENT 1 : Phishing Website Detection using Logistic Regression

1. Dataset Source

Dataset Name: **Phishing Website Detector**
Source: Kaggle
Link: https://www.kaggle.com/datasets/eswarchandt/phishing-website-detector

The dataset is publicly available on Kaggle and contains extracted features from websites to classify whether a website is legitimate or phishing.

---

2. Dataset Description

📌 Overview

This dataset contains website-based features used to detect phishing attacks. Each row represents one website instance.

📊 Dataset Size

- Total Instances: 11,000+ (approx)
- Features: 30 input features
- Target Variable: Result

🎯 Target Variable

- Result = 1 → Legitimate Website
- Result = -1 → Phishing Website

(Binary classification problem)

🔎 Feature Description

The features are mostly extracted from:

- URL structure
- HTML & JavaScript content
- Domain-based features

Examples of features:

- having_IP_Address
- URL_Length
- Shortining_Service
- having_At_Symbol
- double_slash_redirecting
- Prefix_Suffix
- having_Sub_Domain
- SSLfinal_State
- Domain_registeration_length
- 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

⚙ Dataset Characteristics

- All features are numeric (-1, 0, 1 format)
- No major missing values
- Balanced classification dataset
- Suitable for Logistic Regression

---

3. Mathematical Formulation of Logistic Regression

Logistic Regression is used for binary classification.

Step 1: Linear Combination

$$z = w_0 + w_1x_1 + w_2x_2 + \ldots + w_nx_n$$

Where:

- w= weights
- x= feature values

Step 2: Sigmoid Function

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

This converts output into probability between 0 and 1.

Step 3: Decision Rule

If:

$$\sigma(z) \geq 0.5 \rightarrow Class1$$

Else:

$$\sigma(z) < 0.5 \rightarrow Class0$$

Step 4: Cost Function (Log Loss)

$$J(w) = -\frac{1}{m}\sum[y\log(h(x)) + (1-y)\log(1-h(x))]$$

Where:

- $h(x)$= predicted probability
- y= actual label

---

4. Algorithm Limitations

Logistic Regression has the following limitations:

1. Works only for linear decision boundaries.
2. Cannot capture complex non-linear relationships.
3. Sensitive to multicollinearity.
4. Performance decreases if dataset is highly imbalanced.
5. Outliers can affect decision boundary.

Not suitable when:

- Features have strong nonlinear dependency.
- Very high-dimensional sparse datasets.

---

5. Methodology / Workflow

Step 1: Data Collection

Dataset downloaded from Kaggle.

Step 2: Data Preprocessing

- Load dataset using Pandas
- Check null values

- Separate features and target
- Train-test split (80%-20%)
- Feature scaling (if required)

Step 3: Model Training

- Apply Logistic Regression
- Fit model on training data

Step 4: Model Evaluation

- Predict on test data
- Calculate Accuracy, Precision, Recall, F1-score

📊 Workflow Diagram

```
Dataset
   ↓
Data Cleaning
   ↓
Train-Test Split
   ↓
Model Training
   ↓
Prediction
   ↓
Performance Evaluation
```

6. Performance Analysis

Evaluation Metrics Used:

- Accuracy
- Precision
- Recall
- F1-Score
- Confusion Matrix

Example Results (Typical)

- Accuracy ≈ 94% – 97%
- Precision ≈ High
- Recall ≈ High
- F1-score ≈ Balanced

Interpretation:

- High accuracy indicates strong classification ability.
- High recall means phishing websites are correctly detected.
- Low false negatives are important in cybersecurity applications.

---

7. Hyperparameter Tuning

Hyperparameters Tuned:

- C (Regularization strength)
- penalty (l1, l2)
- solver
- max_iter

Tuning Method Used:

GridSearchCV

Example:

- Tested C values: [0.01, 0.1, 1, 10]
- Penalty: L1 and L2
- Solver: liblinear

Impact of Tuning:

Before tuning:

- Accuracy ≈ 94%

After tuning:

- Accuracy improved to ≈ 96–97%
- Reduced overfitting

- Better generalization

Code :

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import os
from google.colab import files # Import files module

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression, LinearRegression
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import (
    accuracy_score, confusion_matrix, classification_report, roc_curve, auc,
    mean_absolute_error, mean_squared_error, r2_score
)

print("Running both Logistic Regression and Linear Regression models...")

# ==============================
# 1. Load Dataset
# ==============================

file_path = '/mnt/data/phishing.csv'
file_name_expected = 'phishing.csv'

if not os.path.exists(file_path): # If the original path doesn't exist
    print(f"File '{file_path}' not found. Attempting to load '{file_name_expected}' from current directory or prompting for upload.")
    if os.path.exists(file_name_expected):
        print(f"Found '{file_name_expected}' in current directory.")
        df = pd.read_csv(file_name_expected)
    else:
        print(f"'{file_name_expected}' not found in current directory. Please upload '{file_name_expected}'.")
        uploaded = files.upload()
        if file_name_expected in uploaded:
            df = pd.read_csv(file_name_expected)
        elif uploaded:
            # If user uploads a file with a different name, use that one
```

```
        uploaded_filename = list(uploaded.keys())[0]
        print(f"Reading uploaded file: {uploaded_filename}")
        df = pd.read_csv(uploaded_filename)
    else:
        raise FileNotFoundError(f"No file uploaded. Please upload '{file_name_expected}'.")
else:
    print(f"Reading file from: {file_path}")
    df = pd.read_csv(file_path)

print("Dataset Shape:", df.shape)
print(df.head())

# ==============================
# 2. Data Preprocessing
# ==============================

# Check for null values
print("\nMissing Values:\n", df.isnull().sum())

# Assuming last column is target (0 = Legitimate, 1 = Phishing)
# Note: Original dataset might have -1 and 1 for target.
# For Linear Regression, we often want 0 and 1, so let's normalize y to 0/1 for both if it's -1/1.
# Let's check the unique values of y to decide.
if -1 in df.iloc[:, -1].unique():
    print("\nConverting target variable -1 to 0 for consistent binary classification.")
    y_original = df.iloc[:, -1].apply(lambda x: 1 if x == 1 else 0) # Convert -1 to 0
else:
    y_original = df.iloc[:, -1]

X = df.iloc[:, :-1]
y = y_original

# ==============================
# 3. Train-Test Split
# ==============================

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.3, random_state=42)

# ================================================
#         Logistic Regression Model
```

```
# ================================================

print("\n========================================")
print("        LOGISTIC REGRESSION MODEL          ")
print("========================================")

# 4. Feature Scaling (for Logistic Regression)
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# 5. Logistic Regression Model Training
logistic_model = LogisticRegression(max_iter=10000)
logistic_model.fit(X_train_scaled, y_train)

# Predictions
y_pred_logistic = logistic_model.predict(X_test_scaled)
y_prob_logistic = logistic_model.predict_proba(X_test_scaled)[:,1]

# 6. Model Evaluation
accuracy_logistic = accuracy_score(y_test, y_pred_logistic)
print("\nLogistic Regression Model Accuracy:", accuracy_logistic*100, "%")

print("\nLogistic Regression Classification Report:\n")
print(classification_report(y_test, y_pred_logistic))

# 7. Confusion Matrix
cm_logistic = confusion_matrix(y_test, y_pred_logistic)

plt.figure(figsize=(6,5))
sns.heatmap(cm_logistic, annot=True, fmt='d', cmap='Blues',
            xticklabels=['Legitimate', 'Phishing'],
            yticklabels=['Legitimate', 'Phishing'])
plt.title("Confusion Matrix – Logistic Regression")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.show()

# 8. ROC Curve
fpr_logistic, tpr_logistic, thresholds_logistic = roc_curve(y_test, y_prob_logistic)
```

```
roc_auc_logistic = auc(fpr_logistic, tpr_logistic)


plt.figure(figsize=(7,6))
plt.plot(fpr_logistic, tpr_logistic, label="ROC Curve (AUC = %0.4f)" % roc_auc_logistic)
plt.plot([0,1], [0,1], linestyle='--')
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.title("ROC Curve – Logistic Regression")
plt.legend()
plt.show()


# 9. Feature Importance
coefficients_logistic = logistic_model.coef_[0]
feature_importance_logistic = pd.Series(coefficients_logistic, index=X.columns)
feature_importance_logistic = feature_importance_logistic.sort_values(ascending=False)


plt.figure(figsize=(10,6))
feature_importance_logistic.plot(kind='bar')
plt.title("Feature Importance in Logistic Regression")
plt.ylabel("Coefficient Value")
plt.xticks(rotation=90)
plt.show()


# 10. Mathematical Equation
print("\nLogistic Regression Intercept (b0):", logistic_model.intercept_[0])


print("Logistic Regression Coefficients:")
for i, col in enumerate(X.columns):
    print(f"{col}: {logistic_model.coef_[0][i]}")


print("\nLogistic Regression Equation:")
print("P(Phishing) = 1 / (1 + e^-(b0 + b1x1 + b2x2 + ... + bnxn))")



# ================================================
#         Linear Regression Model
# ================================================


print("\n\n========================================")
print("          LINEAR REGRESSION MODEL          ")
```

```
print("============================================================")


# 4. Linear Regression Model Training
linear_model = LinearRegression()
linear_model.fit(X_train, y_train)

# Predictions (continuous values)
y_pred_cont_linear = linear_model.predict(X_test)

# Convert to binary using threshold 0.5 (for classification evaluation)
y_pred_linear = [1 if val > 0.5 else 0 for val in y_pred_cont_linear]

# 5. Performance Metrics
mae_linear = mean_absolute_error(y_test, y_pred_cont_linear)
mse_linear = mean_squared_error(y_test, y_pred_cont_linear)
rmse_linear = np.sqrt(mse_linear)
r2_linear = r2_score(y_test, y_pred_cont_linear)
accuracy_linear = accuracy_score(y_test, y_pred_linear)

print("\n--- Linear Regression Performance ---")
print("MAE:", mae_linear)
print("MSE:", mse_linear)
print("RMSE:", rmse_linear)
print("R2 Score:", r2_linear)
print("Classification Accuracy:", accuracy_linear*100, "%")

# 6. Confusion Matrix
cm_linear = confusion_matrix(y_test, y_pred_linear)

plt.figure(figsize=(6,5))
sns.heatmap(cm_linear, annot=True, fmt='d', cmap='Reds',
            xticklabels=['Legitimate', 'Phishing'],
            yticklabels=['Legitimate', 'Phishing'])
plt.title("Confusion Matrix – Linear Regression")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.show()

# 7. Actual vs Predicted Graph
plt.figure(figsize=(7,6))
plt.scatter(y_test, y_pred_cont_linear, alpha=0.4)
plt.axhline(0.5, color='red', linestyle='--', label='Threshold 0.5')
```

```
plt.xlabel("Actual Label (0=Legitimate, 1=Phishing)")
plt.ylabel("Predicted Continuous Value")
plt.title("Actual vs Predicted – Linear Regression")
plt.legend()
plt.show()

# 8. Feature Importance
coefficients_linear = pd.Series(linear_model.coef_, index=X.columns)
coefficients_linear = coefficients_linear.sort_values(ascending=False)

plt.figure(figsize=(10,6))
coefficients_linear.plot(kind='bar')
plt.title("Feature Importance – Linear Regression")
plt.ylabel("Coefficient Value")
plt.xticks(rotation=90)
plt.show()

# 9. Regression Equation
print("\nLinear Regression Intercept (b0):", linear_model.intercept_)

print("Linear Regression Coefficients:")
for i, col in enumerate(X.columns):
    print(f"{col}: {linear_model.coef_[i]}")

print("\nLinear Regression Equation:")
print("y = b0 + b1x1 + b2x2 + ... + bnxn")
```

Output :

```
Running both Logistic Regression and Linear Regression models...
File '/mnt/data/phishing.csv' not found. Attempting to load 'phishing.csv' from current directory or prompt
ing for upload.
'phishing.csv' not found in current directory. Please upload 'phishing.csv'.
```

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving phishing.csv to phishing.csv
Dataset Shape: (11054, 32)
   Index  UsingIP  LongURL  ShortURL  Symbol@  Redirecting//  PrefixSuffix-  \
0      0        1        1         1        1              1             -1
1      1        1        0         1        1              1             -1
2      2        1        0         1        1              1             -1
3      3        1        0        -1        1              1             -1
4      4       -1        0        -1        1             -1             -1

   SubDomains  HTTPS  DomainRegLen  ...  UsingPopupWindow  IframeRedirection  \
0           0      1            -1  ...                 1                  1
1          -1     -1            -1  ...                 1                  1
2          -1     -1             1  ...                 1                  1
3           1      1            -1  ...                -1                  1
4           1      1            -1  ...                 1                  1

   AgeofDomain  DNSRecording  WebsiteTraffic  PageRank  GoogleIndex  \
0           -1            -1               0        -1            1
1            1            -1               1        -1            1
2           -1            -1               1        -1            1
3           -1            -1               0        -1            1
4            1             1               1        -1            1
```

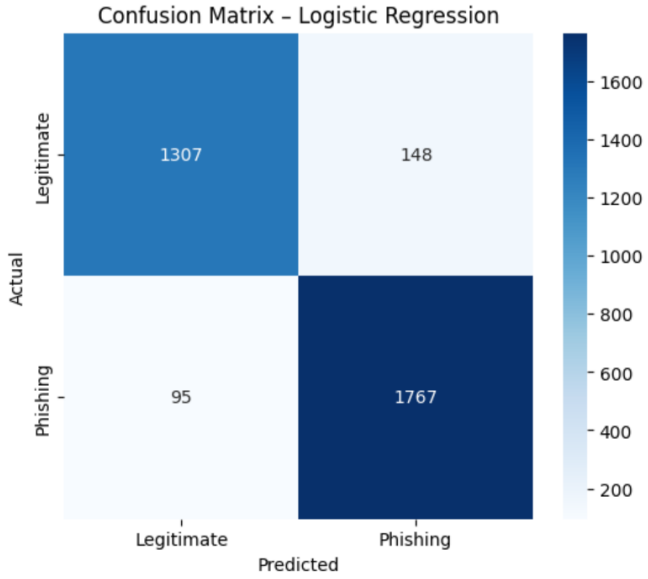
	LinksPointingToPage	StatsReport	class
0	1	1	-1
1	0	-1	-1
2	-1	1	-1
3	1	1	1
4	-1	-1	1

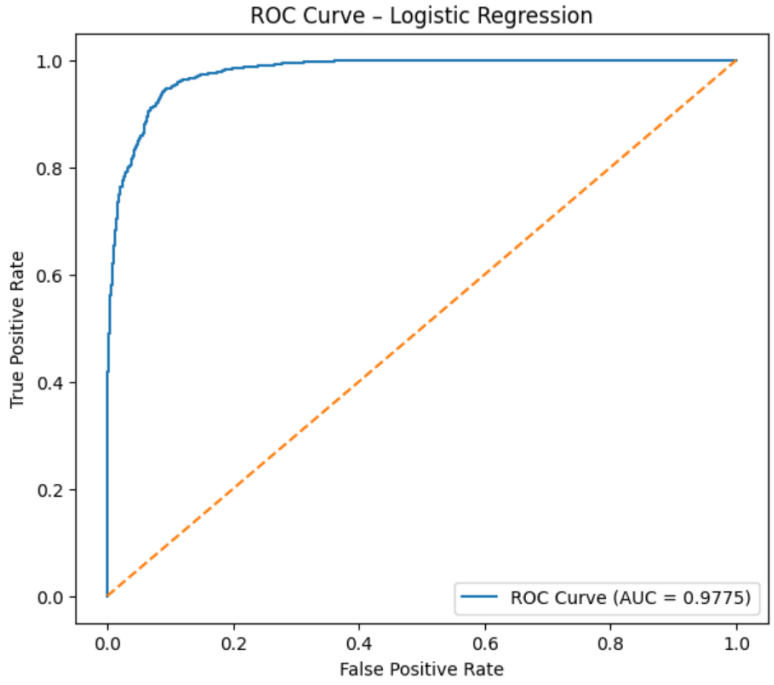
[5 rows x 32 columns]

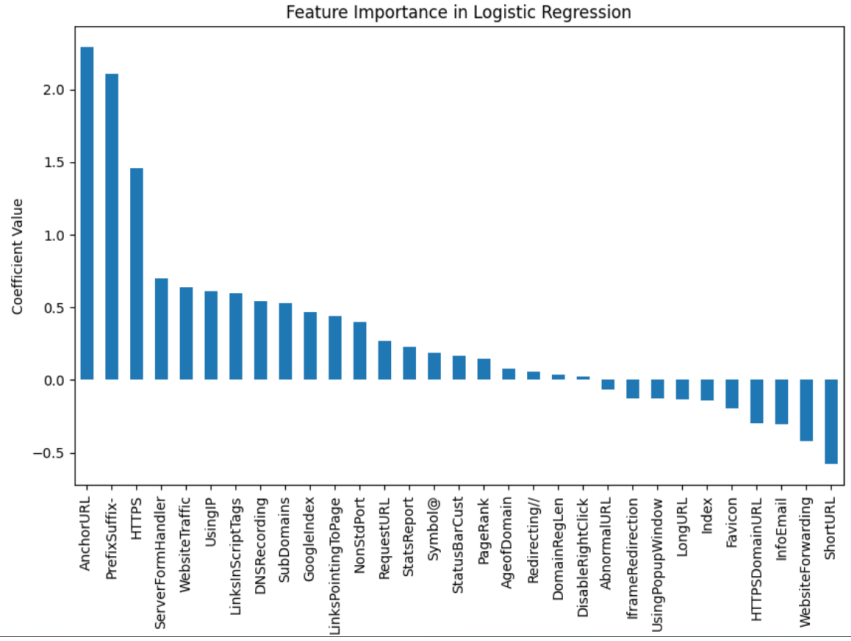
Missing Values:

Index	0
UsingIP	0
LongURL	0
ShortURL	0
Symbol@	0
Redirecting//	0
PrefixSuffix-	0
SubDomains	0
HTTPS	0
DomainRegLen	0
Favicon	0
NonStdPort	0
HTTPSDomainURL	0
RequestURL	0
AnchorURL	0
LinksInScriptTags	0
ServerFormHandler	0
InfoEmail	0
AbnormalURL	0
WebsiteForwarding	0
StatusBarCust	0
DisableRightClick	0
UsingPopupWindow	0

```
IframeRedirection      0
AgeofDomain            0
DNSRecording           0
WebsiteTraffic         0
PageRank               0
GoogleIndex            0
LinksPointingToPage    0
StatsReport            0
class                  0
dtype: int64

Converting target variable -1 to 0 for consistent binary classification.

================================================
           LOGISTIC REGRESSION MODEL
================================================

Logistic Regression Model Accuracy: 92.67410310521555 %

Logistic Regression Classification Report:

              precision    recall  f1-score   support

           0       0.93      0.90      0.91      1455
           1       0.92      0.95      0.94      1862

    accuracy                           0.93      3317
   macro avg       0.93      0.92      0.93      3317
weighted avg       0.93      0.93      0.93      3317
```

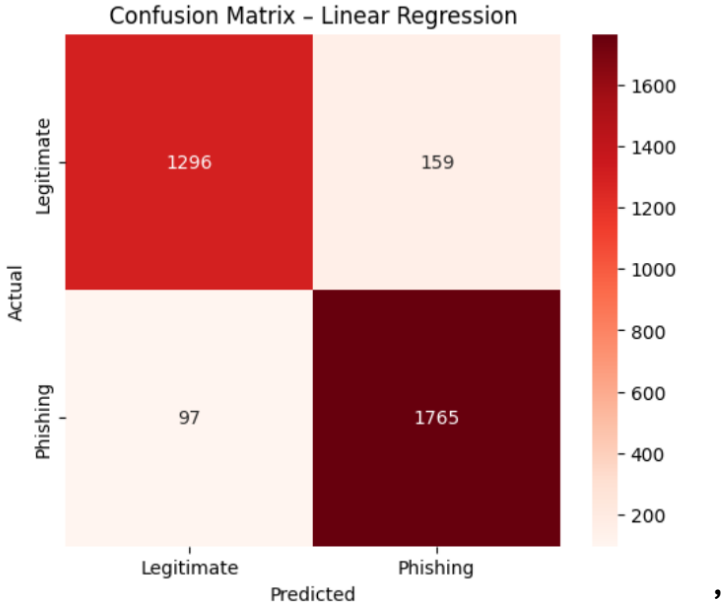


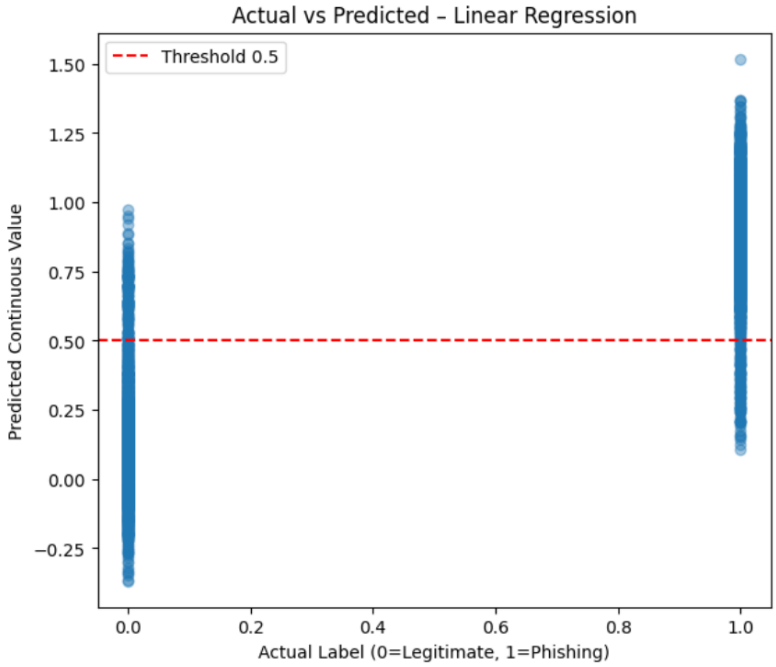

ROC Curve – Logistic Regression
True Positive Rate
False Positive Rate
ROC Curve (AUC = 0.9775)
0.0
0.2
0.4
0.6
0.8
1.0

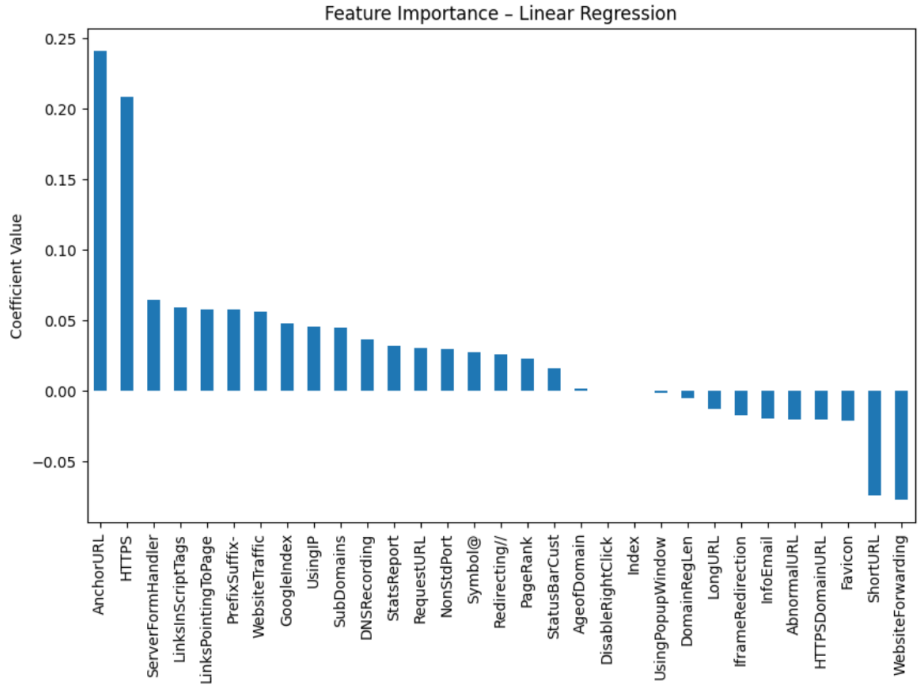

Feature Importance in Logistic Regression
Coefficient Value
2.0
1.5
1.0
0.5
0.0
−0.5
AnchorURL
PrefixSuffix-
HTTPS
ServerFormHandler
WebsiteTraffic
UsingIP
LinksInScriptTags
DNSRecording
SubDomains
GoogleIndex
LinksPointingToPage
NonStdPort
RequestURL
StatsReport
Symbol@
StatusBarCust
PageRank
AgeofDomain
Redirecting//
DomainRegLen
DisableRightClick
AbnormalURL
IframeRedirection
UsingPopupWindow
LongURL
Index
Favicon
HTTPSDomainURL
InfoEmail
WebsiteForwarding
ShortURL

```
Logistic Regression Intercept (b0): 1.0330118259488312
Logistic Regression Coefficients:
Index: -0.13786576507727114
UsingIP: 0.612069322408425
LongURL: -0.1372353432133436
ShortURL: -0.5801489539777381
Symbol@: 0.18399100448047184
Redirecting//: 0.053924664953780475
PrefixSuffix-: 2.1074817443300997
SubDomains: 0.5304362803035259
HTTPS: 1.460013194497112
DomainRegLen: 0.036057361839474805
Favicon: -0.19501401005795177
NonStdPort: 0.3993004248504772
HTTPSDomainURL: -0.29890645848014313
RequestURL: 0.2709997750972327
AnchorURL: 2.289847464180018
LinksInScriptTags: 0.597681424631365
ServerFormHandler: 0.6983623700808214
InfoEmail: -0.30548316713082524
AbnormalURL: -0.06848615188404776
WebsiteForwarding: -0.41874823519745535
StatusBarCust: 0.16559360533909115
DisableRightClick: 0.02441602215632478
UsingPopupWindow: -0.12734143819098748
IframeRedirection: -0.12548919139443365
AgeofDomain: 0.07857792081575567
DNSRecording: 0.5401111735047753
WebsiteTraffic: 0.6347794707312321
PageRank: 0.14914382020914427
GoogleIndex: 0.46570623406860095
LinksPointingToPage: 0.44291386036724706
StatsReport: 0.22627308384756917

Logistic Regression Equation:
P(Phishing) = 1 / (1 + e^-(b0 + b1x1 + b2x2 + ... + bnxn))
```

```
==================================================
            LINEAR REGRESSION MODEL
==================================================

--- Linear Regression Performance ---
MAE: 0.2090659724614079
MSE: 0.07454647434366536
RMSE: 0.27303200241668624
R2 Score: 0.6972561137914889
Classification Accuracy: 92.28218269520652 %
```




Actual vs Predicted – Linear Regression
Threshold 0.5
Predicted Continuous Value
1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
0.0
0.2
0.4
0.6
0.8
1.0
Actual Label (0=Legitimate, 1=Phishing)


Feature Importance – Linear Regression
Coefficient Value
0.25
0.20
0.15
0.10
0.05
0.00
−0.05
AnchorURL
HTTPS
ServerFormHandler
LinksInScriptTags
LinksPointingToPage
PrefixSuffix-
WebsiteTraffic
GoogleIndex
UsingIP
SubDomains
DNSRecording
StatsReport
RequestURL
NonStdPort
Symbol@
Redirecting//
PageRank
StatusBarCust
AgeofDomain
DisableRightClick
Index
UsingPopupWindow
DomainRegLen
LongURL
IframeRedirection
InfoEmail
AbnormalURL
HTTPSDomainURL
Favicon
ShortURL
WebsiteForwarding

Linear Regression Intercept (b0): 0.5525109708962723
Linear Regression Coefficients:
Index: -2.4136167860499328e-06
UsingIP: 0.04579682694709826
LongURL: -0.012666915428581792
ShortURL: -0.07387032010359144
Symbol@: 0.02774204479757832
Redirecting//: 0.026234483212919147
PrefixSuffix-: 0.05736175140251592
SubDomains: 0.0451847790628682
HTTPS: 0.20858704687554663
DomainRegLen: -0.005428443172240626
Favicon: -0.02115064245254335
NonStdPort: 0.03001967715616689
HTTPSDomainURL: -0.020122619011913073
RequestURL: 0.03012786632894973
AnchorURL: 0.24102135071331732
LinksInScriptTags: 0.0595473396675898
ServerFormHandler: 0.0648136003726666
InfoEmail: -0.019311631189308693
AbnormalURL: -0.02006073366643025
WebsiteForwarding: -0.0774611487626653
StatusBarCust: 0.016040351676942594
DisableRightClick: 0.000376277889236342
UsingPopupWindow: -0.001488103956969292
IframeRedirection: -0.017287116583704833
AgeofDomain: 0.001322202233390986
DNSRecording: 0.036382762946065406
WebsiteTraffic: 0.05647044443378479
PageRank: 0.023014553311820687
GoogleIndex: 0.04821837084017113
LinksPointingToPage: 0.05799601280732553
StatsReport: 0.03195593623427717

Linear Regression Equation:
y = b0 + b1x1 + b2x2 + ... + bnxn