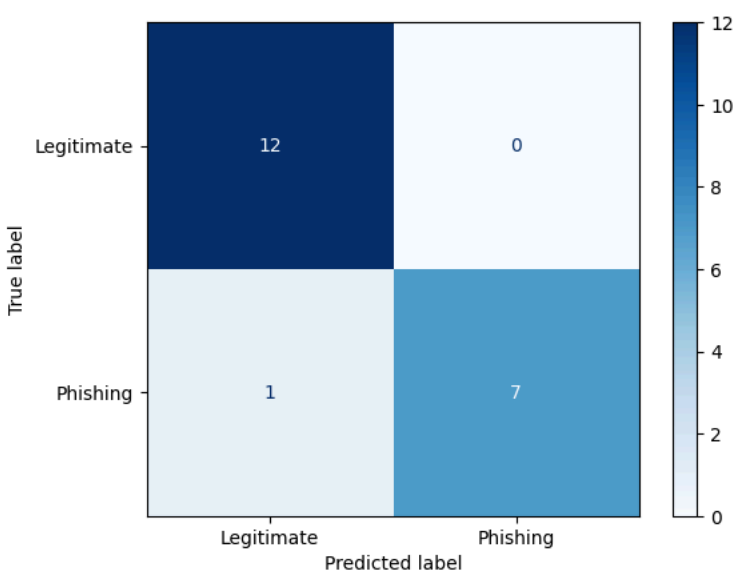
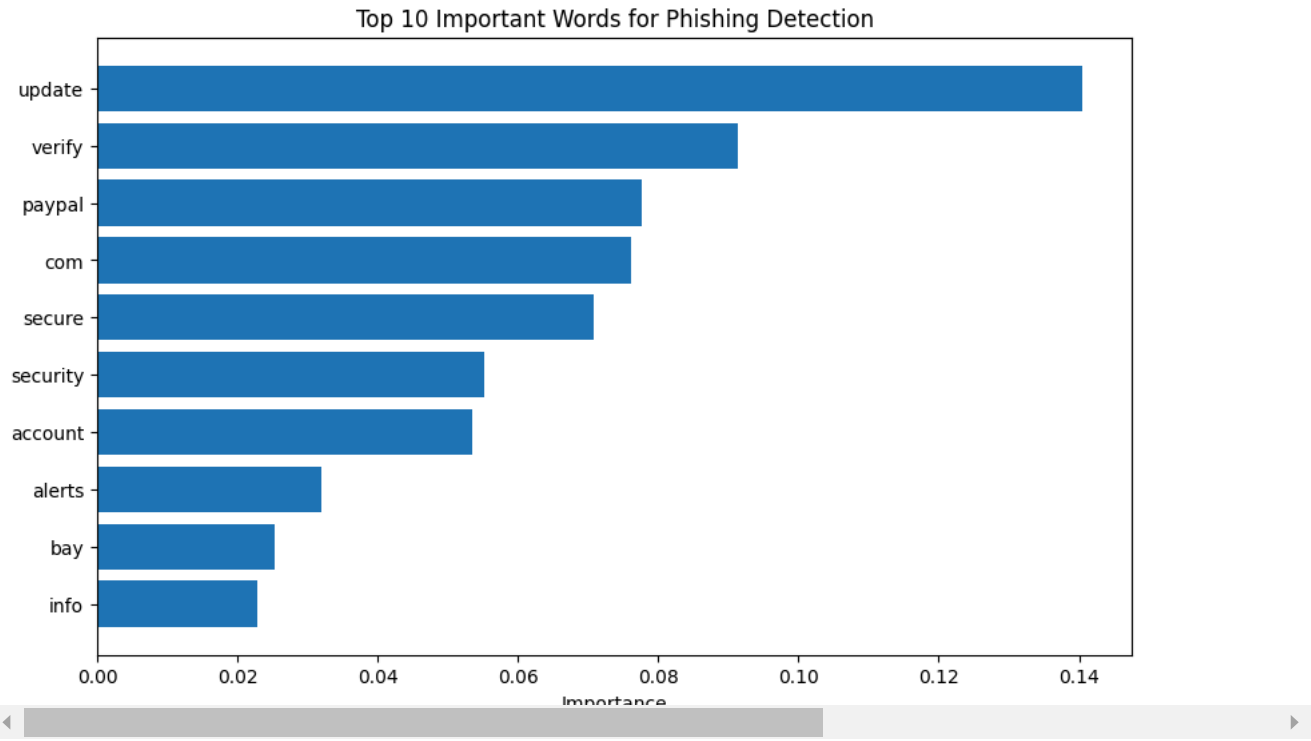
```
# Import required libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report

# 1. Load the Dataset
# Assuming you've combined phishing and legitimate emails into a CSV
# Replace 'emails.csv' with your dataset path.
data = pd.read_csv('/content/Testdataemails.csv')

# 2. Data Preprocessing
# Check for missing values and clean the data
data.dropna(inplace=True)

# Example structure of the dataset:
X = data['text']  # Email content
y = data['label'] # Labels

# 3. Text Preprocessing - Using TF-IDF Vectorization
# Convert the text into numerical form
tfidf = TfidfVectorizer(stop_words='english', max_features=5000)
X_tfidf = tfidf.fit_transform(X)

# 4. Train-Test Split
X_train, X_test, y_train, y_test = train_test_split(X_tfidf, y, test_size=0.2, random_state=42)

# 5. Random Forest Model Training
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)

# 6. Model Prediction
y_pred = rf_model.predict(X_test)

# 7. Evaluation
print("Accuracy:", accuracy_score(y_test, y_pred))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))

# Feature Importance (words most indicative of phishing/legitimate)
import numpy as np
feature_importances = np.argsort(rf_model.feature_importances_)[-10:]
important_words = [tfidf.get_feature_names_out()[i] for i in feature_importances]
print("\nImportant words for classification:\n", important_words)
```

```
Accuracy: 0.95

Confusion Matrix:
 [[12  0]
 [ 1  7]]

Classification Report:
              precision    recall  f1-score   support

  Legitimate       0.92      1.00      0.96        12
    Phishing       1.00      0.88      0.93         8

    accuracy                           0.95        20
   macro avg       0.96      0.94      0.95        20
weighted avg       0.95      0.95      0.95        20


Important words for classification:
 ['info', 'bay', 'alerts', 'account', 'security', 'secure', 'com', 'paypal', 'verify', 'update']
```

```
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import ConfusionMatrixDisplay

# Visualize the Confusion Matrix
cm = confusion_matrix(y_test, y_pred)
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=["Legitimate", "Phishing"])
disp.plot(cmap=plt.cm.Blues)
plt.show()

# Visualize Important Words
importances = rf_model.feature_importances_
indices = np.argsort(importances)[-10:]  # Top 10 important features
plt.figure(figsize=(10, 6))
plt.title('Top 10 Important Words for Phishing Detection')
```

```
plt.barh(range(len(indices)), importances[indices], align='center')
plt.yticks(range(len(indices)), [tfidf.get_feature_names_out()[i] for i in indices])
plt.xlabel('Importance')
plt.show()
```





Start coding or generate with AI.