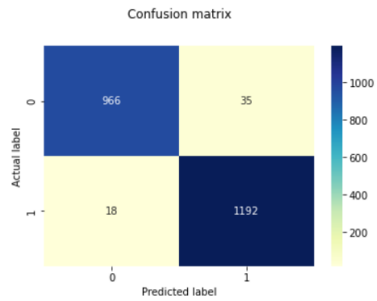
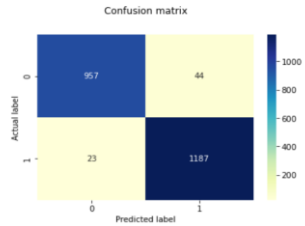
Project Development Phase
Model Performance Test

Date	10 November 2022
Team ID	PNT2022TMID36037
Project Name	Web Phishing Detection
Maximum Marks	10 Marks

Model Performance Testing:

S.No.	Parameter	Values	Screenshot
1.	Metrics	**Regression Model:** MAE - , MSE - , RMSE - , R2 score - **Classification Model:** Confusion Matrix - , Classification Report - Accuracy Score- **Random Forest** Training Accuracy - 98.93% Validation Accuracy - 97.46% **Support Vector Machine** Training Accuracy – 98.42% Validation Accuracy - 96.96% **Logistic Regression** Training Accuracy -92.79 % Validation Accuracy - 93.44% **Decision Tree** Training Accuracy – 98.93% Validation Accuracy - 96.83% **Naïve Bayes** Training Accuracy – 90.79% Validation Accuracy - 91.85%	**Random Forest** Accuracy of train data = 98.93713251922208 % Accuracy of test data = 97.46720940750791 % precision recall f1-score support -1 0.98 0.97 0.97 1001 1 0.97 0.98 0.98 1210 accuracy 0.97 2211 macro avg 0.98 0.97 0.97 2211 weighted avg 0.97 0.97 0.97 2211 t[35]: Text(0.5, 15.0, 'Predicted label')  **Support Vector Machine** Accuracy of train data = 98.4283129805518 % Accuracy of validation data = 96.96969696969697 % precision recall f1-score support -1 0.98 0.96 0.97 1001 1 0.96 0.98 0.97 1210 accuracy 0.97 2211 macro avg 0.97 0.97 0.97 2211 weighted avg 0.97 0.97 0.97 2211

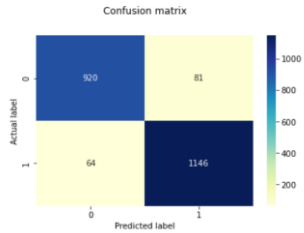
Out[19]: Text(0.5, 15.0, 'Predicted label')



Logistic Regression

Accuracy of train data = 92.79737675260064 %
Accuracy of validation data = 93.441881501583 %

t[14]: Text(0.5, 15.0, 'Predicted label')



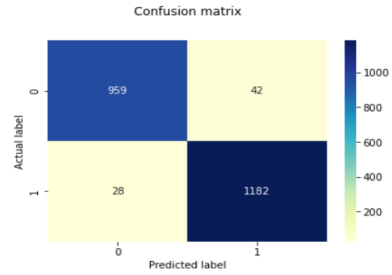
	precision	recall	f1-score	support
-1	0.93	0.92	0.93	1001
1	0.93	0.95	0.94	1210
accuracy			0.93	2211
macro avg	0.93	0.93	0.93	2211
weighted avg	0.93	0.93	0.93	2211

Decision Tree

Accuracy of train data = 98.93713251922208 %
Accuracy of validation data = 96.8340117593849 %

	precision	recall	f1-score	support
-1	0.97	0.96	0.96	1001
1	0.97	0.98	0.97	1210
accuracy			0.97	2211
macro avg	0.97	0.97	0.97	2211
weighted avg	0.97	0.97	0.97	2211

t[24]: Text(0.5, 15.0, 'Predicted label')



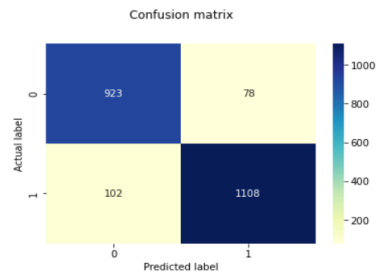
<table>
<tr>
<td></td>
<td></td>
<td></td>
<td>

Naïve Bayes

```
Accuracy of train data =  90.79601990049751 %
Accuracy of validation data =  91.85888738127545 %
```

	precision	recall	f1-score	support
-1	0.90	0.92	0.91	1001
1	0.93	0.92	0.92	1210
accuracy			0.92	2211
macro avg	0.92	0.92	0.92	2211
weighted avg	0.92	0.92	0.92	2211

```
Out[30]:  Text(0.5, 15.0, 'Predicted label')
```



</td>
</tr>
<tr>
<td>2.</td>
<td>Tune the Model</td>
<td>

Hyperparameter Tuning -

Random Forest
Best score is: 0.9706
BEST PARAMETERS: {'bootstrap': True, 'criterion': 'entropy', 'max_depth': 90, 'max_features': 2, 'n_estimators': 1000}

Support Vector Machine
Best Score: 0.9658
BEST PARAMETER: {'C': 10, 'gamma': 0.1, 'kernel': 'rbf'}

Logistic Regression
Best Score: 0.9270
Best Hyperparameters: {'C': 0.1, 'penalty': 'l2', 'solver': 'saga'}

</td>
<td>

Random Forest

```
param_grid_RF = {'bootstrap': [True],
    'max_depth': [80, 90, 100,110],
    'criterion' : ('gini','entropy'),
    'max_features': [2, 3],
    'n_estimators': [ 200, 300, 1000]}
grid_RF = GridSearchCV(RandomForestClassifier(), param_grid_RF, cv=5, n_jobs=-1,scoring = 'accuracy' ).fit(X_train,y_train)


print ("Best score is:",grid_RF.best_score_)
print ("BEST PARAMETERS:",grid_RF.best_params_)
```

```
Best score is: 0.970601216526964
BEST PARAMETERS: {'bootstrap': True, 'criterion': 'entropy', 'max_depth': 90, 'max_features': 2, 'n_estimators': 1000}
```

Support Vector Machine

```
param_grid_SVM = {'C': [0.01,0.1, 1, 10, 100, 1000],
            'gamma': [1, 0.1, 0.01, 0.001, 0.0001],
            'kernel': ['rbf','sigmoid']}
grid_SVM = GridSearchCV(SVC(), param_grid_SVM, refit = True,return_train_score=True, n_jobs=-1,cv=5).fit(X_train, y_train)

# print best parameter after tuning
print('Best Score: %s' % grid_SVM.best_score_)
print("BEST PARAMETER:",grid_SVM.best_params_)
```

```
Best Score: 0.965852515289718
BEST PARAMETER: {'C': 10, 'gamma': 0.1, 'kernel': 'rbf'}
```

Logistic Regression

```
param_grid_LR = {
    'solver' : ['newton-cg', 'lbfgs', 'liblinear','sag','saga'],
    'penalty' : ['l1', 'l2', 'elasticnet'],
    'C' : [100, 10, 1.0, 0.1, 0.01]
}
grid_LR = GridSearchCV(LogisticRegression(),param_grid_LR, scoring='accuracy', n_jobs=-1, cv=5).fit(X_train,y_train)
print('Best Score: %s' % grid_LR.best_score_)
print('Best Hyperparameters: %s' % grid_LR.best_params_)
```

```
Best Score: 0.9270694515141361
Best Hyperparameters: {'C': 0.1, 'penalty': 'l2', 'solver': 'saga'}
```

</td>
</tr>
</table>

		Decision Tree Best score is: 0.96 BEST PARAMETERS: {'criterion': 'entropy', 'max_depth': 35, 'min_samples_split': 2, 'splitter': 'random'} **Naïve Bayes** Best score is: 0.9073 BEST PARAMETERS: {'var_smoothing': 0.3511191734215131}	**Decision Tree** ``` param_grid_DT = {'criterion': ["gini", "entropy"], 'splitter': ["best", "random"], 'max_depth': [10,15,35,30], 'min_samples_split': [2, 3, 4,5,6]} grid_DT = GridSearchCV(DecisionTreeClassifier(), param_grid_DT, cv=5, n_jobs=-1,scoring = 'accuracy').fit(X_train,y_train) print ("Best score is:",grid_DT.best_score_) print ("BEST PARAMETERS:",grid_DT.best_params_) ``` Best score is: 0.9600845634596841 BEST PARAMETERS: {'criterion': 'entropy', 'max_depth': 35, 'min_samples_split': 2, 'splitter': 'random'} **Naïve Bayes** ``` params_NB = {'var_smoothing': np.logspace(0,-9, num=100)} grid_NB = GridSearchCV(GaussianNB(), param_grid=params_NB, cv=5, scoring='accuracy').fit(X_train,y_train) print ("Best score is:",grid_NB.best_score_) print ("BEST PARAMETERS:",grid_NB.best_params_) ``` Best score is: 0.9073950822230 01 BEST PARAMETERS: {'var_smoothing': 0.3511191734215131}