

Model Development Phase Template

Date	15 March 2024
Team ID	740139
Project Title	Acoustic Fire Extinguishing Prediction
Maximum Marks	4 Marks

Initial Model Training Code, Model Validation and Evaluation Report

The initial model training code will be showcased in the future through a screenshot. The model validation and evaluation report will include classification reports, accuracy, and confusion matrices for multiple models, presented through respective screenshots.

Initial Model Training Code

```
[ ] def model_evaluation(classifier):  
    cm = confusion_matrix(y_test, classifier.predict(x_test))  
    counts = [value for value in cm.flatten()]  
    labels = [f'{v1}' for v1 in counts]  
    labels = np.asarray(labels).reshape(2,2)  
    sns.heatmap(cm, annot = labels, cmap = 'Greens', fmt = '')  
    y_pred = classifier.predict(x_test)  
    yt_pred = classifier.predict(x_train)  
    print('The Training Accuracy of the algorithm is', accuracy_score(y_train, yt_pred))  
    print('The Testing Accuracy of the algorithm is', accuracy_score(y_test, y_pred))  
    return [(accuracy_score(y_train * 100, yt_pred * 100) * 100), (accuracy_score(y_test * 100, y_pred * 100) * 100), precision_score(y_test, y_pred, average='macro')]
```

Training the model in multiple Algorithms (K Nearest Neighbors Model)

```
[ ] knn = KNeighborsClassifier()  
    knn.fit(x_train,y_train)
```



▼ KNeighborsClassifier
KNeighborsClassifier()

(SVM Model)

```
[ ] Svm = SVC()  
    Svm.fit(x_train,y_train)
```



▼ SVC
SVC()

(Naive Bayes)

```
[ ] gnb = GaussianNB()  
    gnb.fit(x_train, y_train)
```



▼ GaussianNB
GaussianNB()

(Logistic Regression)

```
▶ lr = LogisticRegression()  
  lr.fit(x_train, y_train)
```

```
⇒ ▾ LogisticRegression  
   LogisticRegression()
```

Decision Tree Model

```
[ ] dt = DecisionTreeClassifier(max_depth= 11)  
     dt.fit(x_train, y_train)
```

```
⇒ ▾ DecisionTreeClassifier  
   DecisionTreeClassifier(max_depth=11)
```

(Random Forest Model)

```
[ ] rf = RandomForestClassifier(max_depth=11)  
     rf.fit(x_train, y_train)
```

```
⇒ ▾ RandomForestClassifier  
   RandomForestClassifier(max_depth=11)
```

(Gradient Boosting Model)

```
[ ] gb = GradientBoostingClassifier()
    gb.fit(x_train,y_train)
```



```
▸ GradientBoostingClassifier
GradientBoostingClassifier()
```

Model Validation and Evaluation Report:

Model	Classification Report	Accuracy	Confusion Matrix
KNN	<p>(K Nearest Neighbors Model)</p> <pre>knn_r = model_evaluation(knn)</pre> <p>The Training Accuracy of the algorithm is 0.946821472884563 The Testing Accuracy of the algorithm is 0.91923727231614</p>	94&91	-
SVM	<p>(SVM Model)</p> <pre>svm_r = model_evaluation(svm)</pre> <p>The Training Accuracy of the algorithm is 0.8899161478651473 The Testing Accuracy of the algorithm is 0.892419486116209</p>	88&89	-
Naïve bayes	<p>(Naive Bayes)</p> <pre>gb_r = model_evaluation(gnb)</pre> <p>The Training Accuracy of the algorithm is 0.868220561885332 The Testing Accuracy of the algorithm is 0.87908283179505</p>	86&87	-
Logistic regression	<p>(Logistic regression)</p> <pre>lr_r = model_evaluation(lr)</pre> <p>The Training Accuracy of the algorithm is 0.87523658397764 The Testing Accuracy of the algorithm is 0.8701516256893</p>	87&87	-
Decision Tree	<p>(Decision Tree Model)</p> <pre>dt_r = model_evaluation(dt)</pre> <p>The Training Accuracy of the algorithm is 0.9845194583838464 The Testing Accuracy of the algorithm is 0.9892663115063199</p>	98&94	-
Random Forest Model	<p>Random Forest Model</p> <pre>rf_r = model_evaluation(rf)</pre> <p>The Training Accuracy of the algorithm is 0.8863978642285834 The Testing Accuracy of the algorithm is 0.952880481333227</p>	98&95	-
Gradient boosting model	<p>Gradient Boosting Model</p> <pre>gb_r = model_evaluation(gb)</pre> <p>The Training Accuracy of the algorithm is 0.9532716978427578 The Testing Accuracy of the algorithm is 0.947548451100095</p>	95&94	-