After exploring the methodology section in the earlier part of this thesis, we will now shift our attention to the results obtained through the proposed methodology.

4.1 Suspicious activity detection using mobile sensor data with modified subspace KNN (msK)

Using msK technique, we could detect any normal activity versus suspicious activity with 99.7 % accuracy. The figure 4.1 shows a correlation between capture rate and error. As the packet capture rate increases (More packets per seconds), the percentage error, in guessing the correct event, decreases. This is because as more packets are received, the system has more information to work with, which leads to better accuracy. The figure 4.1 also shows that the error rate is not zero even when the capture rate is high. This is because there is always some uncertainty associated with event identification, even when a large number of packets are captured. However, the error rate does decrease as the capture rate increases, which suggests that the system becomes more accurate as it receives more information. The figure 4.2 depicts the relationship between the time intervals between consecutive computations and the capture rate. As the capture rate rises, the number of received packets per second increases, enabling the processing of larger data volumes. With a faster computing machine and shorter time intervals between

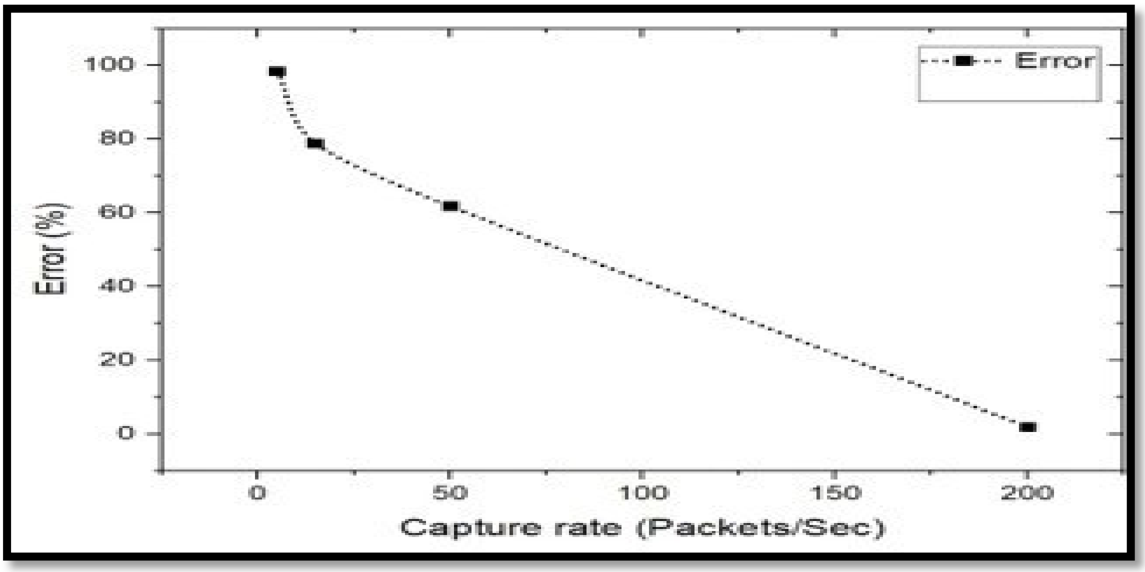


Fig. 4.1: Capture rate versus error. Capture rate increases in packets per second percentage error in guessing the correct event decreases.

As more packets are received gives more information and hence results in better accuracy.

computations due to the higher capture rate, it can be inferred that the computing machine experiences periods of waiting, leading to reduced computational activity. As the number of incoming packets per second continues to grow, the time between successive computations decreases until it reaches the computational limits of the system. The figure 4.3 presents the Confusion matrix representing the accuracy of a classifier based on a test dataset consisting of 5994 events. Among 2988 normal events, 2988 were correctly classified as normal, while 9 of them were incorrectly classified as suspicious events. Out of 2997 suspicious events, 2989 were accurately classified as suspicious, but 8 of them were erroneously classified as normal events. The modified Subspace KNN system demonstrates a high level of accuracy, accurately classifying suspicious events with a 99.7 % accuracy rate and an average error of only 0.28 %.

The scatter plot as shown in the figure 4.4, shows the relationship between two variables in the original data set, column 2 and column 3. Each data point is represented by a circle, and the position of the circle on the x and y axes corresponds to the values of the two variables for that data point.

The overall pattern of the data points suggests that there is a positive correlation

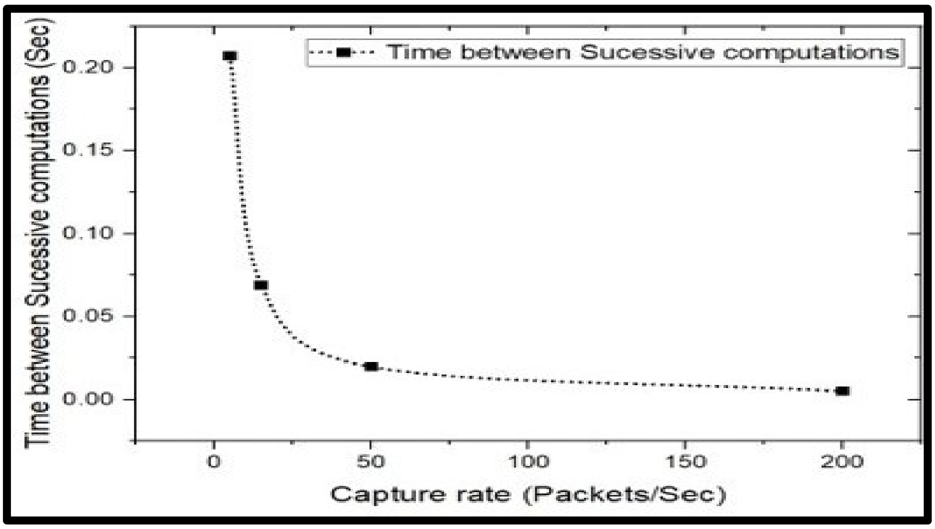


Figure 4.2: The time between successive computations versus capture rate.

As the capture rate increases, so does the number of packets received per second, allowing for more data to be processed. As the computing machine becomes faster and the time between successive computations decreases with the capture rate, it is possible to conclude that the computing machine is in a wait state, causing computations to become low. As more packets are received per second, successive computing time decreases until it reaches computational limits.

Between column 2 and column 3. This means that as the values of column 2 increase, the values of column 3 also tend to increase. However, there is also some scatter in the data, which means that there are some data points that do not follow this general pattern.

One possible explanation for the scatter in the data is that there are other factors, in addition to column 2, that influence the values of column 3. For example, the data set may contain data from multiple different groups, and each group may have its own unique relationship between column 2 and column 3.

Overall, the scatter plot provides a useful overview of the relationship between column 2 and column 3 in the original data set. However, it is important to keep in mind that the scatter plot is just a single data visualization, and it should not be used to make any definitive conclusions about the relationship between the two variables.

The Receiver Operating Characteristic (ROC) curve and the Area Under the Curve (AUC) are essential tools for evaluating binary classification models. The ROC curve shown in figure 4.5, depicts the tradeoff between True Positive Rate (TPR) and False Positive Rate (FPR) at varying threshold values. TPR measures the proportion of correctly identified positive cases, while FPR gauges the proportion of falsely identified negative cases as positive. The AUC quantifies the entire performance of a binary classifier across all possible threshold values, with a score of 1.0 signifying perfect classification and 0.5 indicating random guessing. In the context of Model 1's ROC curve, the model demonstrates excellent dis criminatory power, as evidenced by an AUC of 1.00, implying flawless ranking of positive and negative cases. It's important to note that a classifier's performance can differ depending on the dataset, necessitating evaluation on a separate test set before deployment. In depth definitions of TPR, FPR, ROC curve, and AUC are provided for clarity, highlighting their significance in classifier assessment.

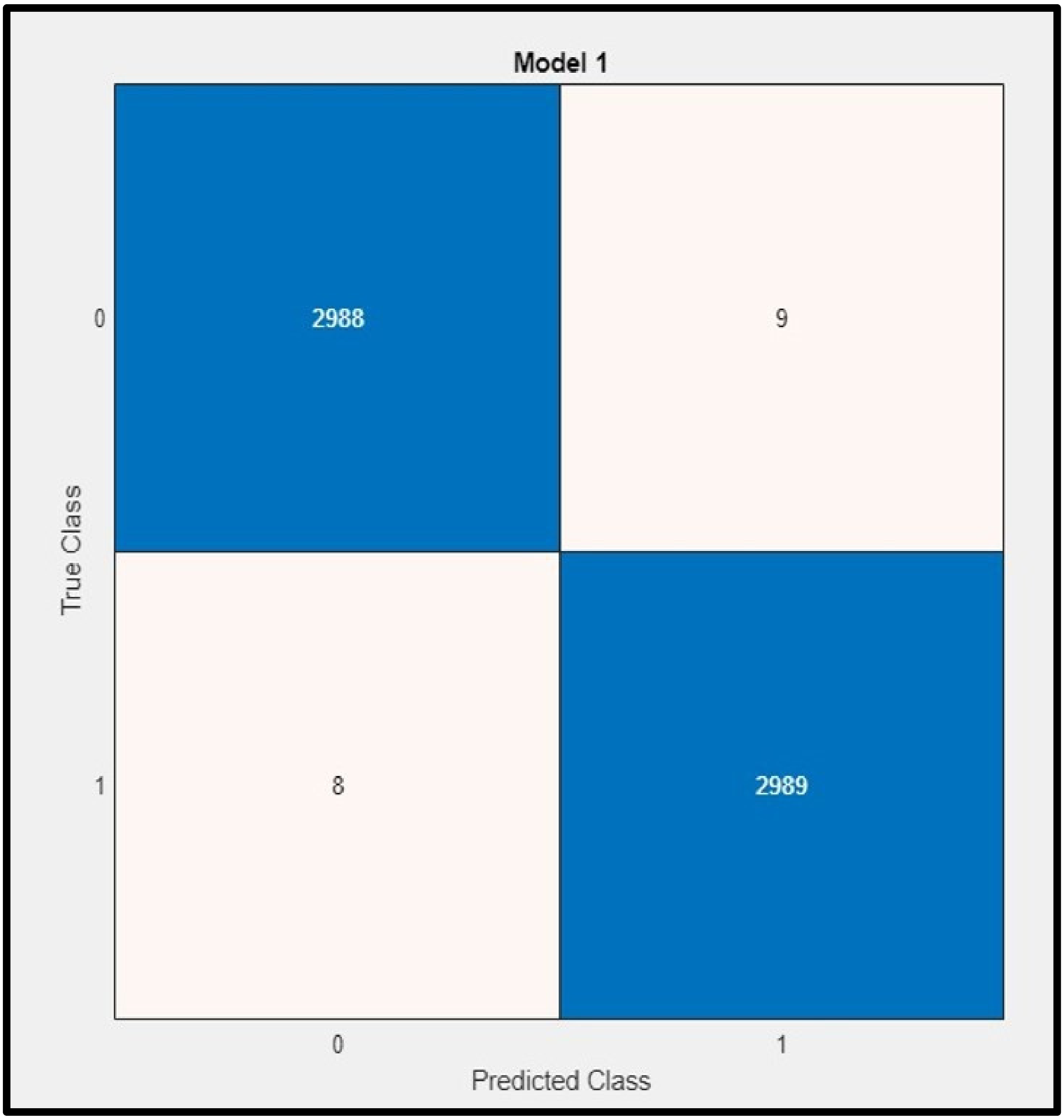


Figure 4.3: The Confusion matrix for classifier accuracy out of 5994 test events 2988 normal events were classified as normal, whereas 9 of the normal events were classified as suspicious events.

Out of 2997 suspicious events 2989 were perfectly classified as suspicious events, while 8 such samples were misclassified as normal events. The designed system of modified Subspace KNN can accurately classify the suspicious event with 99.7 % accuracy and an average error of 0.28 %.

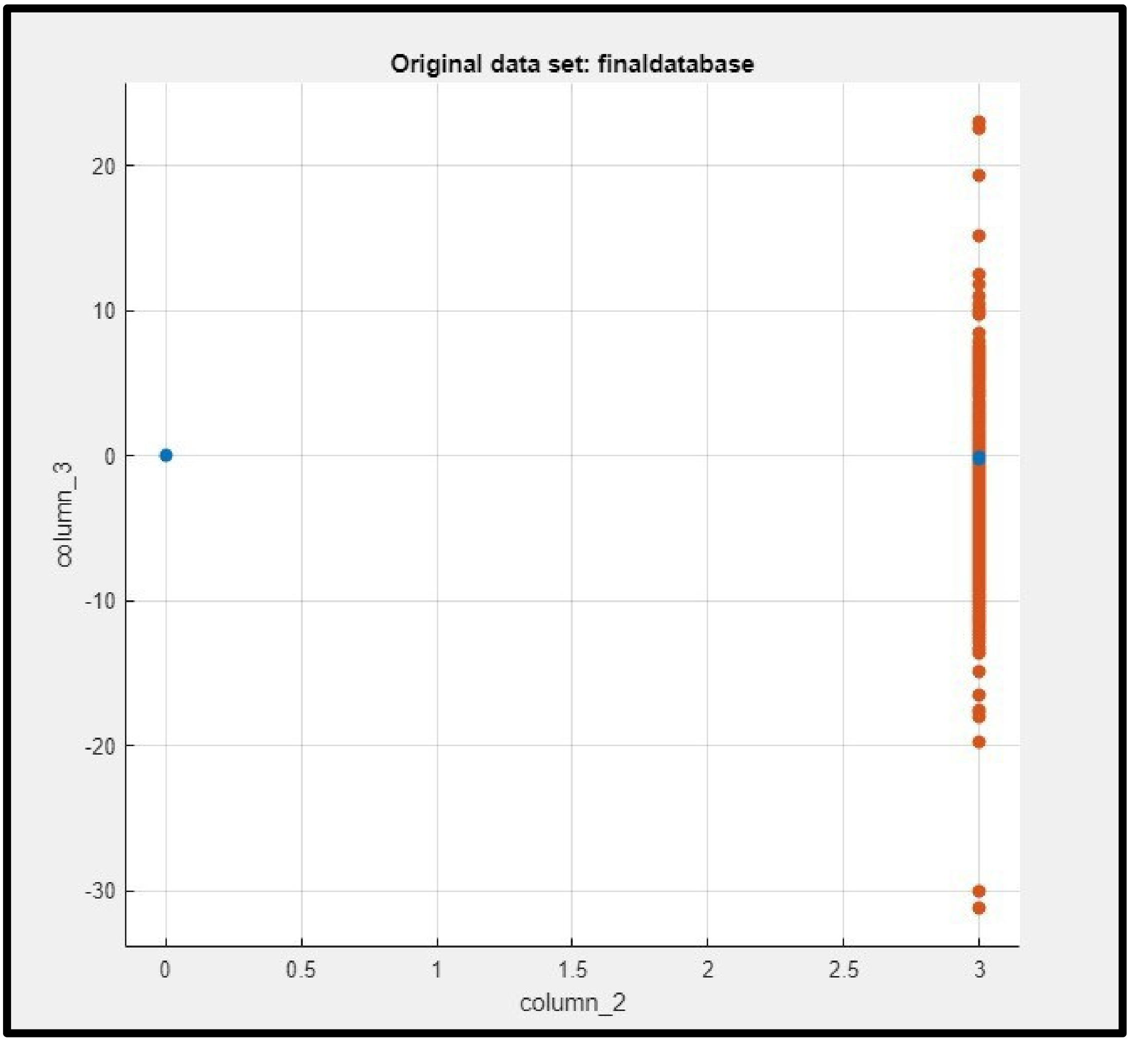


Fig. 4.4: Scatter plot of the original data set

The diagram 4.6 shows the standard deviation of predictions for a model with 11 features. Each column represents a different feature, and the height of the bar shows the standard deviation of the predictions for that feature.

The standard deviation is a measure of how spread out the predictions are. A higher standard deviation means that the predictions are more spread out, while a lower standard deviation means that the predictions are more bunched together. In this diagram, we can see that the standard deviation of the predictions is higher

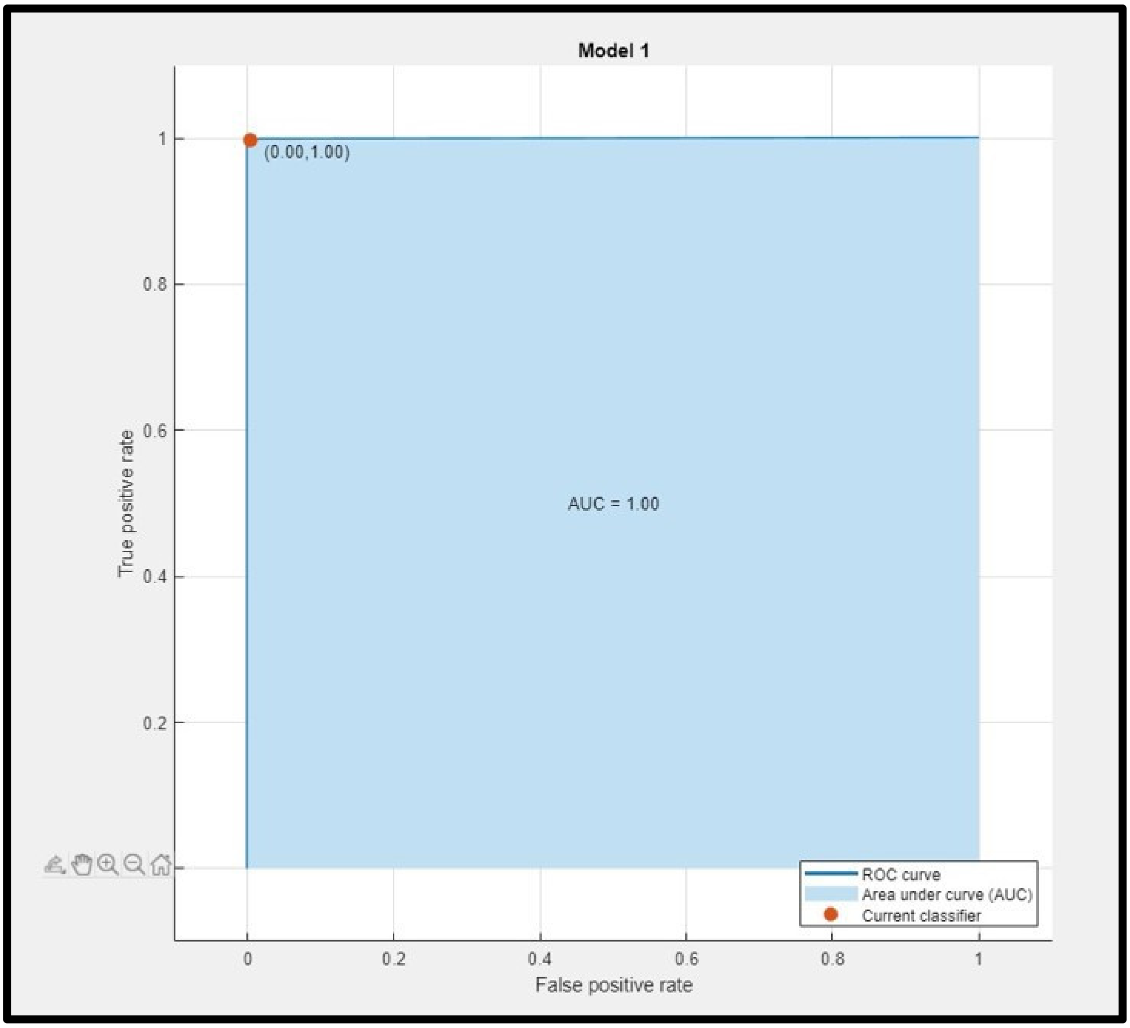


Fig. 4.5: Receiver Operating Characteristic (ROC) curve and Area Under the Curve (AUC) for Model 1

for some features than for others. For example, the standard deviation for feature 10, is much higher than, the standard deviation for feature 1. This means that, the model is less confident in its predictions for feature 10 compared to feature 1. There are a few possible explanations for why the standard deviation of the predictions might be higher for some features than for others. One possibility is that the data for those features is more noisy or less informative. Another possibility is that the model is not able to capture the relationship between those features and the target variable as well as the relationship between other features and the target variable. Overall, the diagram provides insights into the performance of the proposed model on different features. It can be used to identify features where the model is less confident in its predictions, and it can also be used to identify features that are more important to the model's predictions. Here are some additional

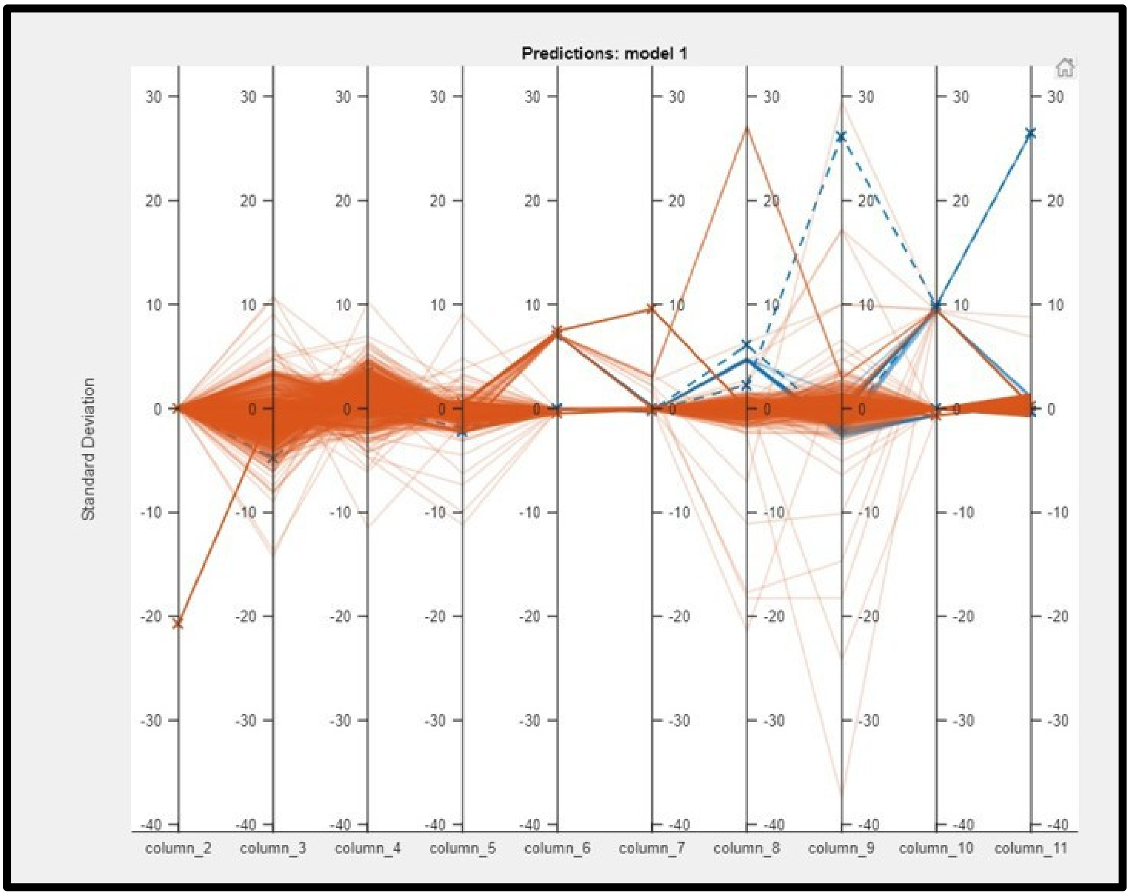
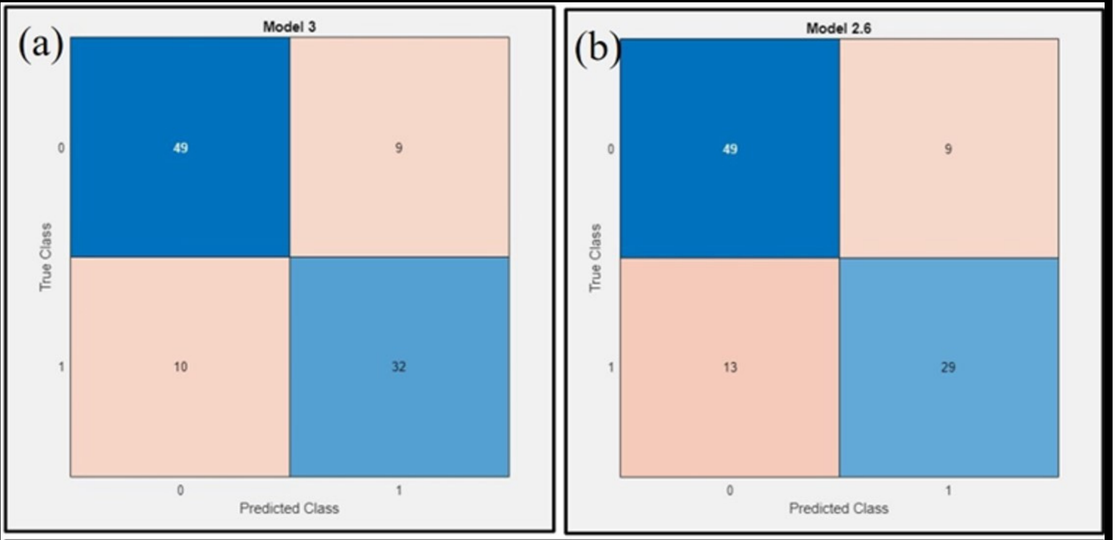


Fig. 4.6: Standard deviation of predictions

insights that can be gained from the diagram: The overall standard deviation of the predictions is relatively low, which suggests that the model is generally making good predictions. The standard deviation of the predictions is slightly higher for the first few features than for the last few features. This could suggest that the model is learning more from the last few features than from the first few features. There is one feature, feature 10, where the standard deviation of the predictions is much higher than for the other features. This suggests that the model is less confident in its predictions for this feature. The figure 4.7 shows the results of two classification methods, KNN and Gaussian SVM, on a dataset. The results are presented in the form of confusion matrices and evaluation metrics. Confusion matrix: A confusion matrix is a table that shows the number of correct and incorrect predictions made by a classification model. It is a useful tool for evaluating the performance of a classification model. The confusion matrix for KNN is shown in Figure 4.7(a). The rows of the confusion matrix represent the actual classes, and the columns represent the predicted classes. The diagonal



(c)

Measure	Value	Derivations
Sensitivity	0.8305	TPR = TP / (TP + FN)
Specificity	0.7805	SPC = TN / (FP + TN)
Precision	0.8448	PPV = TP / (TP + FP)
Negative Predictive Value	0.7619	NPV = TN / (TN + FN)
False Positive Rate	0.2195	FPR = FP / (FP + TN)
False Discovery Rate	0.1552	FDR = FP / (FP + TP)
False Negative Rate	0.1695	FNR = FN / (FN + TP)
Accuracy	0.8100	ACC = (TP + TN) / (P + N)
F1 Score	0.8376	F1 = 2TP / (2TP + FP + FN)
Matthews Correlation Coefficient	0.6089	TP*TN - FP*FN / sqrt((TP+FP)*(TP+FN)*(TN+FP)*(TN+FN))

(d)

Measure	Value	Derivations
Sensitivity	0.7903	TPR = TP / (TP + FN)
Specificity	0.7632	SPC = TN / (FP + TN)
Precision	0.8448	PPV = TP / (TP + FP)
Negative Predictive Value	0.6905	NPV = TN / (TN + FN)
False Positive Rate	0.2368	FPR = FP / (FP + TN)
False Discovery Rate	0.1552	FDR = FP / (FP + TP)
False Negative Rate	0.2097	FNR = FN / (FN + TP)
Accuracy	0.7800	ACC = (TP + TN) / (P + N)
F1 Score	0.8167	F1 = 2TP / (2TP + FP + FN)
Matthews Correlation Coefficient	0.5443	TP*TN - FP*FN / sqrt((TP+FP)*(TP+FN)*(TN+FP)*(TN+FN))

Fig. 4.7: Results with two classification methods

(a) Confusion matrix using KNN (b) Confusion matrix using Gaussian SVM. (c) Results obtained, after the KNN classification was implemented. (d) Results obtained, after the Gaussian SVM classification was implemented. TP: True positive, FP: False positive, TN: True negative and FP: False positive were used for analysis purpose.

elements of the confusion matrix represent the number of correct predictions, and the off diagonal elements represent the number of incorrect predictions. The confusion matrix for Gaussian SVM is shown in Figure 4.7(b). It is similar to the

Classification Method	Accuracy (%)
Sub Space KNN	98
RUS Boosted Tree	99.5
Sub space disriminat	83.4
Neural Network	99.4
Modifed Sub space KNN(Proposed algorithm)	99.7

```
mobiledev with properties:

                   Connected: 1
           Available Cameras: {'back' 'front'}
                     Logging: 1

   AccelerationSensorEnabled: 1
AngularVelocitySensorEnabled: 1
       MagneticSensorEnabled: 1
    OrientationSensorEnabled: 1
       PositionSensorEnabled: 1

Current Sensor Values:
                Acceleration: [0.27 0.23 -10.19]  (m/s^2)
             AngularVelocity: [-0.22 0.07 0.06]   (rad/s)
               MagneticField: [3.56 1.56 -48.19]  (microtesla)
                 Orientation: [85.91 -27.1 0.35]  (degrees)

       Position Data:
                    Latitude: 41.29  (degrees)
                   Longitude: -72.35  (degrees)
                       Speed: 25  (m/s)
                      Course: 83.6  (degrees)
                    Altitude: 200.1  (m)
          HorizontalAccuracy: 9.0  (m)
```

Fig. 4.8: Sample of sensor data points collected during a fight sequence

confusion matrix for KNN, except that the Gaussian SVM model has a slightly higher number of correct predictions. Evaluation metrics: Evaluation metrics are used to quantify the performance of a classification model. Some common evaluation metrics include accuracy, precision, recall, and F1 score. The evaluation metrics for KNN and Gaussian SVM are shown in Figures 4.7(c) and 4.7(d), respectively. The Gaussian SVM model has a higher accuracy, precision, and recall than the KNN model. However, the F1 score is similar for both models. Overall, the Gaussian SVM model outperforms the KNN model on this dataset.

Figure 4.8 shows a table of sensor values and position data that were collected during a fight sequence. The data includes acceleration, magnetic field, angular velocity, orientation, latitude, longitude, altitude, speed, course, and horizontal accuracy. This data could be used to analyze the movements of the fighters and the environment in which the fight took place. The acceleration data shows that the fighters were moving at high speeds. The magnetic field data shows that the two fighters were in close proximity to each other during the fight. The angular velocity data shows that the fighters were spinning and turning rapidly. The orientation data shows that the fighters were facing each other during the fight. The position data shows that the fight took place outdoors at a location with corresponding coordinates.

This data could be used to study when fights between two individuals have happened, to develop training data one creates simulations of fight sequences through trained actors. Other sensors, such as gyroscopes and barometers, could also be used to collect additional data. Data collected during a fight sequence can provide valuable insights into the movements of the people involved and the environment in which the fight took place and can be used as evidence in court.

In this part of the thesis, we discuss how KNN and Gaussian SVM we explored for binary classification purposes. The work was carried out at the preliminary stage of the thesis, when we were unsure will our own proposed model would work or not. To train a model that will be able to classify the normal and abnormal sequence, the k-nearest neighbors (KNN) and the Gaussian SVM algorithms were used. Here, a training dataset is generated through a mobile application. It is connected to various sensors which are deployed in various locations to record real time data. The dataset is stored in an object called MobileDEV. The KNN algorithm is a classification technique that forms a group or cluster of similar looking data. The group or cluster is treated as a class. On the other hand, gaussian SVM is a classification technique that differentiates the dataset based on features that are extracted from the data. Figure 4.9 displays the flowchart of the proposed system. Data is collected with the help of various mobile sensors and an object named MobileDev is used to store the data. A machine learning model is trained using KNN and Gaussian SVM. The results are stored in a database and sent for further analysis. The system undergoes classification of sequence. In case the sequence is termed as ”normal”, the sensors are instructed to continue

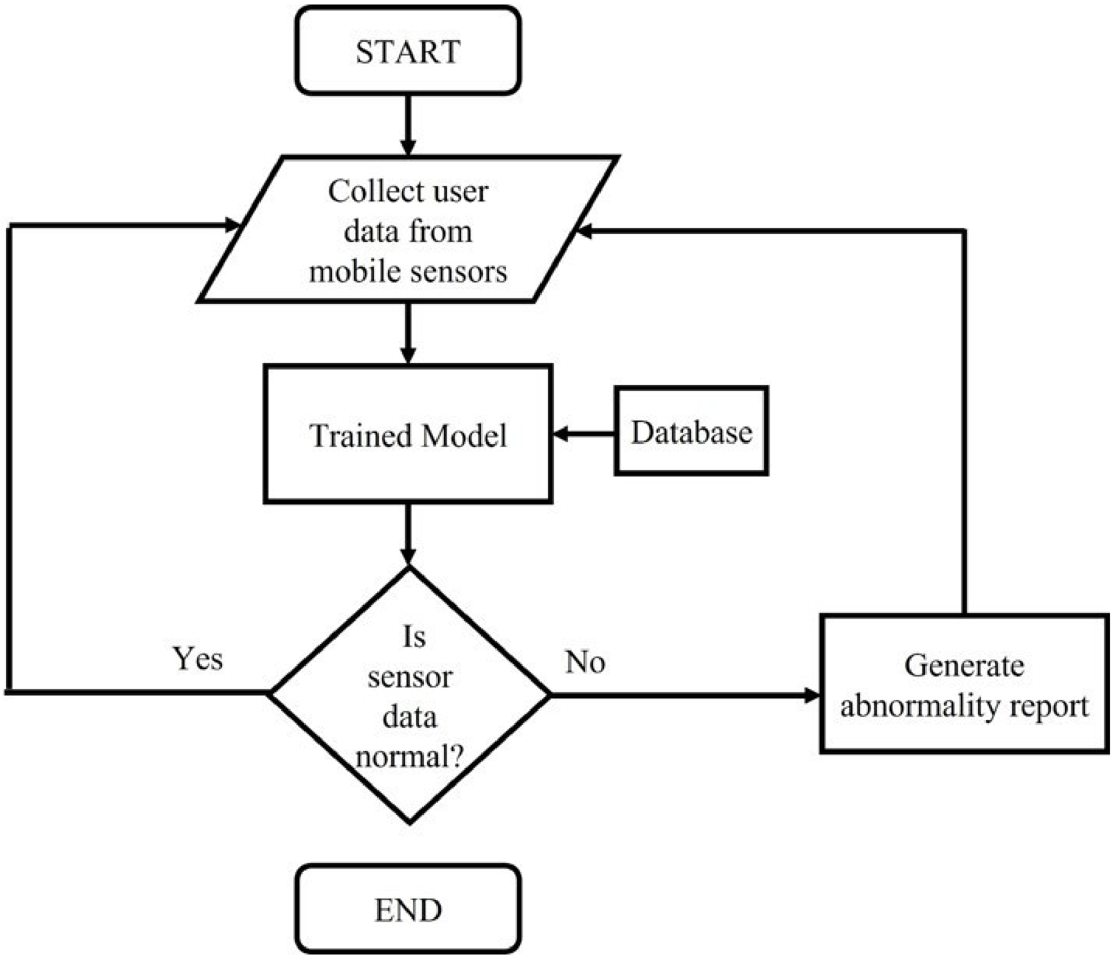


Fig. 4.9: Proposed system flowchart. Data is collected using various mobile sensors, and the data is stored in an object called MobileDev.

KNN and Gaussian SVM machine learning models are used during model training for classification. Features such as principle component analysis are used on time series data. The results are saved in a database and sent to be analyzed further. Once trained the system goes through real time sequence classification based on the trained model. If the data sequence is classified as “normal,” then the sensors are instructed to continue collecting data. In the other case, i.e. the “abnormal” case, an abnormality report is generated.

Truth data

Classifier results

	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7	Class 8	Class 9	Class 10	Classification overall	User's accuracy (Precision)
Class 1	891	1	5	1	1	0	0	1	3	0	903	98.671%
Class 2	2	892	9	5	1	2	0	1	0	1	913	97.7%
Class 3	3	2	902	1	0	3	0	1	11	0	923	97.725%
Class 4	4	3	20	887	3	1	0	1	1	11	931	95.274%
Class 5	5	6	0	2	892	2	11	0	1	0	919	97.062%
Class 6	1	12	22	6	1	890	1	1	1	11	946	94.08%
Class 7	7	1	0	2	3	2	894	2	6	1	918	97.386%
Class 8	3	8	6	0	5	4	1	893	1	11	932	95.815%
Class 9	2	9	10	1	11	3	0	3	870	2	911	95.499%
Class 10	10	1	1	0	34	1	1	1	1	854	904	94.469%
Truth overall	928	935	975	905	951	908	908	904	895	891	9200	
Producer's accuracy (Recall)	96.013%	95.401%	92.513%	98.011%	93.796%	98.018%	98.458%	98.783%	97.207%	95.847%		

Fig. 4.10: Confusion matrix illustrating the performance of the proposed IFDenseNet-138 model for 10-class classification.

collecting data. However, in the reverse case, that is the ”abnormal” case, an abnormality report is generated.

This image in 4.10 shows a confusion matrix that summarizes the performance of the IFDenseNet-138 model on a 10-class classification task. The rows represent the true labels of the data points, while the columns represent the predicted labels. The diagonal entries of the matrix show the number of correctly classified data points for each class. The off-diagonal entries represent the errors made by the model, where a data point of one class is incorrectly classified as another class.

These are some specific observations that can be made from the confusion matrix:

- The model achieved an overall accuracy of 96.32%, which means that it correctly classified 96.32% of the data points.
- The average precision of the model is 96.36%, which means that for each class, the model was able to correctly identify a high proportion of true positives.
- The average recall of the model is 96.4%, which means that for each class, the model was able to correctly classify a high proportion of the data points that actually belong to that class.

- The confusion matrix shows that the model made relatively few errors, and the errors that were made were mostly between classes that are visually similar. For example, there were a few cases where Class 6 was mistaken for Class 5, and vice versa.

Overall, the confusion matrix shows that the IFDenseNet-138 model is a highly accurate and effective classifier for this 10-class classification task.

4.2 Testing with trained actors

Upon finalizing the theoretical groundwork for our classification algorithm, a pivotal step in its practical implementation involved enlisting the assistance of trained actors to simulate various crime scenes. The objective was to gather real world data through the mobile application we had developed. The scenarios enacted included both attacker victim interactions and instances where an individual posed harm to themselves, simulating suicide attempts.

To comprehensively capture the nuances of each scenario, the entire enactment process was meticulously recorded through video footage. This served a dual purpose: firstly, it provided a visual representation of the acted out scenes, and secondly, it established a direct correlation between these scenarios and the data collected through our mobile app.

The collected data, encompassing both attacker and victim perspectives, became the foundational dataset for training the neural network outlined in our proposed approach. A crucial aspect of this training involved the extraction of images from the recorded video frames. Each image encapsulated a specific moment within the enacted scenario, contributing to the establishment of event durations and providing novel temporal dimension to the dataset.

This extracted image data played a pivotal role in both the validation and specific training phases of the proposed neural networks. By utilizing real-world scenarios acted out by trained individuals, our neural networks were exposed to diverse and realistic inputs, enhancing their ability to discern and classify different activities accurately.

The depicted figures in fig 4.11 illustrate a staged sequence portraying an accident scenario wherein a person riding a bike collides with a pedestrian. The

Fig. 4.11: Event accident

presence of a helmet on the rider becomes pivotal, enabling them to escape unidentified, thereby transforming the incident into a critical hit and run case. The mobile application gathers data from both the victim of the accident and the driver of the hit and run scooter. The figure 4.12 illustrates a staged scenario representing a murder, with frames captured just before the actors simulated the act. In forensic situations resembling events like murder or rape, criminals often flee the scene. The black box application gathered data from both the female victim and the perpetrator. To establish correlation, the events were synchronized with the actual frames of the extracted video using a clapper board. The data from both mobile devices underwent distinct training processes, leading to the recommendation that for each type of crime, two separately trained AI networks should be employed.

Figure 4.13 shows two people fighting. In such senario it become essential first to stop fight even before camera is taken out. The picture frame extracted from video, one can see image of 2 people are fighting with each other. It becomes very essential to collect data from both fighters as one of them might loose life after loosing in fight. Even for some forensic both fighter might runaway Hence always it becomes easier to catch them.

The figure 4.14 illustrates an individual attempting self harm by cutting the veins on the forearm, with the potential consequence of fatal blood loss. The resulting pattern in forensic reports would differ from that of a murder. In cases of suicide, collecting data is particularly crucial. There are instances in legal proceedings where the victim is deceased, and determining whether it was a suicide or a staged murder is challenging.

In essence, our approach involved a synergy between theoretical algorithmic frameworks and practical real world data collection. The utilization of trained actors and the subsequent extraction of images from recorded enactments added a layer of authenticity and complexity to the training process, contributing to the robustness and effectiveness of the proposed neural networks in classifying activities within a crime scene context.

All 14 classification cases were recorded, encompassing instances acted out by various actors. The four newly introduced cases, involving a fight between two

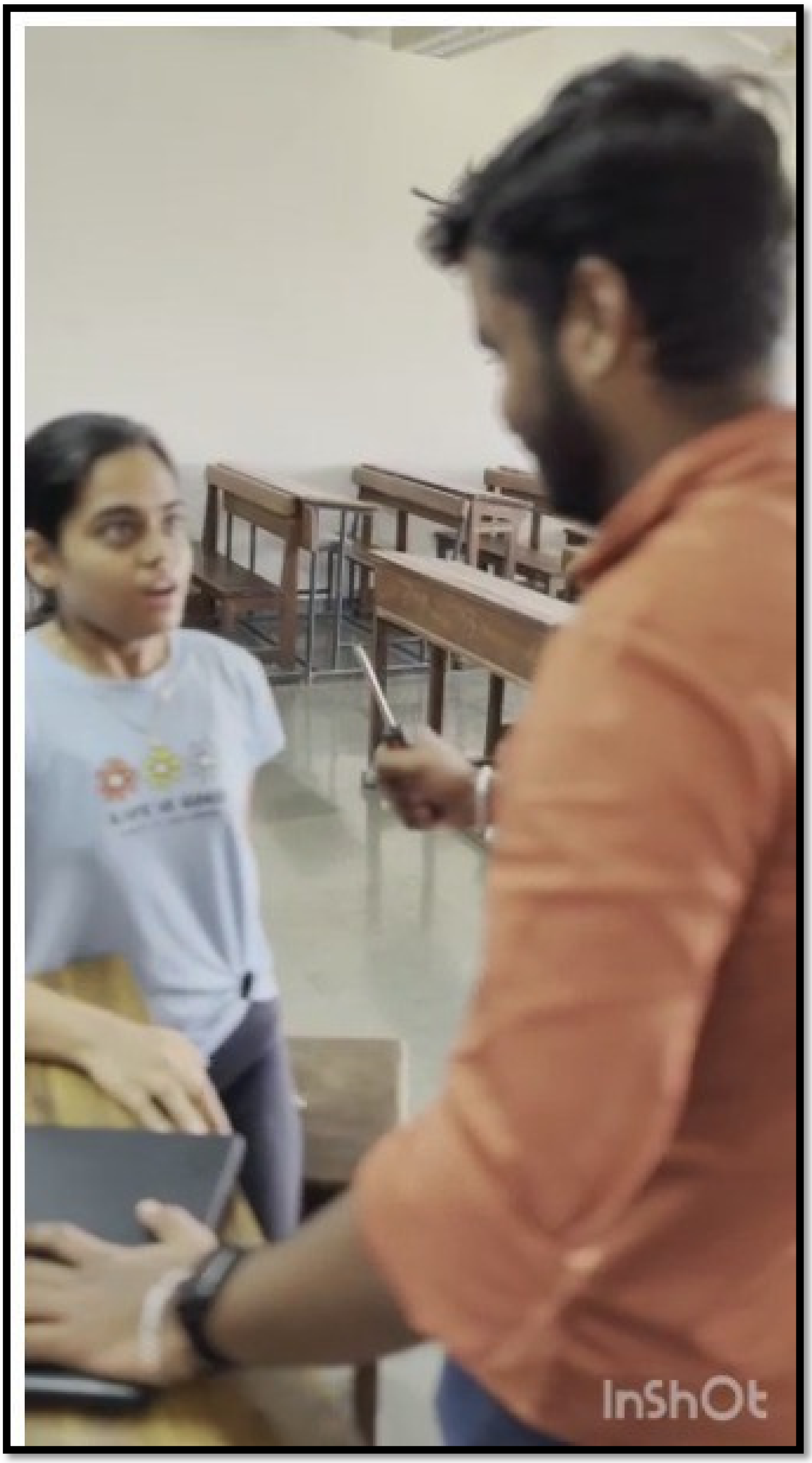


Figure 4.12: Image from acted session where Attacker is about the attack victim

Fig. 4.13: Two people recreating fighting scene

Fig. 4.14: Person acting for scenario like suicide

people, murder, suicide, and an accident, were seamlessly integrated with the initial 10 cases. The application we developed ran on all phones during the enactment of scenes. Each scene was meticulously recorded, capturing both routine activities and subsequent criminal events. This approach allowed for the comprehensive collection of ground truth data for normal activities, enhancing the robustness of the dataset.