

Project ID: TMED031

Using Automated Infant Posture Recognition to Reduce SIDS Risk

Isha Narang
Ardrey Kell High School
Charlotte, NC

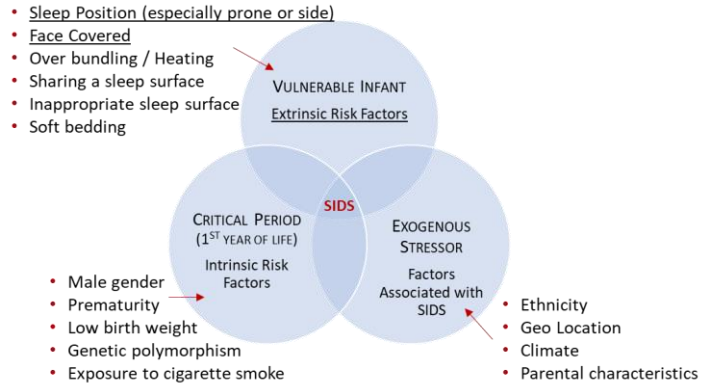
Table of Contents

1. Introduction.....	3
1.1. Motivation.....	3
1.2. Related Work	3
2. Key Terms and Descriptions	4
2.1. PoseNet – Key points and scores	4
2.2. Random Forests	4
2.3. Feature Importance	5
2.4. Performance of Machine Learning model	5
2.5. Python Libraries.....	5
3. Softwares, Methodology and Procedure.....	6
3.1. Softwares	6
3.2. Data Collection	6
3.3. Building an algorithm and a model	8
3.3.1. Geometric Algorithm	8
3.3.2. Machine Learning Model	10
4. Results and Interpretation	12
5. Conclusion and Discussion	15
6. References and Acknowledgements.....	16
6.1. References.....	16
6.2. Acknowledgements.....	16
7. Appendix A: Infant Pictures and Source Links	17
8. Appendix B: Python Code for Machine Learning Model.....	19

1. Introduction

1.1. Motivation

According to the CDC, approximately 3,500 infants die annually in the United States from sleep-related infant deaths, including Sudden Infant Death Syndrome (SIDS). In order to reduce the risk of all sleep-related infant deaths, the American Academy of Pediatrics (AAP) recommends a safe sleep environment. These recommendations include supine positioning: sleeping on the back. In addition to the sleep position not being prone (on the stomach) or on the side, the Triple risk model (as displayed by the Venn diagram) identifies a covered face as an extrinsic risk factor.



My research questions are:

Can a Pose Estimation technique be used to identify infant body parts? What automated method would effectively recognize infant lying posture and initiate an alert for postures that are high-risk for SIDS?

My hypothesis is

- a. that PoseNet can be used to identify infant body parts.
- b. a machine learning (ML) model could effectively recognize infant lying posture.

1.2. Related Work

I found a *SIDS Info* app developed by Rutgers University that guides caregivers on how to follow AAP safety recommendations. Rutgers University, along with Microsoft, has also developed a *Baby be Well* app for Android that has more features.

However, I could not find an automated approach to help with identifying an infant's posture.

2. Key Terms and Descriptions

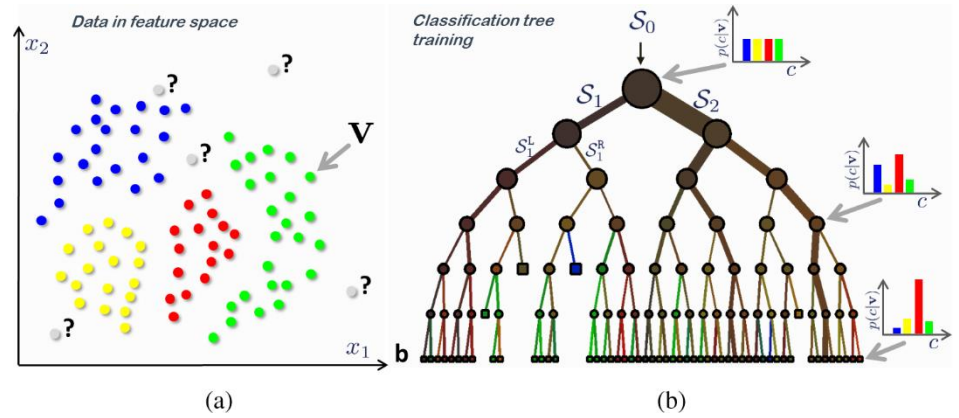
2.1. PoseNet – Key points and scores

Google researchers released PoseNet, a Pose Estimation technique which uses computer vision methods to develop a detailed visual understanding of people in unconstrained environments. This technique uses a two-stage model in which convolutional neural networks process an image: in the first stage a bounding box is produced around each candidate person, and in the second stage a person tree-structured graph consisting of 17 key points (body parts, see table) is drawn. For each person in the image, every key point has (x, y) coordinates and a confidence score. PoseNet uses TensorFlow.js and can run on both mobile phones and desktops.

Id	Body Part	Id	Body Part
0	nose	9	leftWrist
1	leftEye	10	rightWrist
2	rightEye	11	leftHip
3	leftEar	12	rightHip
4	rightEar	13	leftKnee
5	leftShoulder	14	rightKnee
6	rightShoulder	15	leftAnkle
7	leftElbow	16	rightAnkle
8	rightElbow		

2.2. Random Forests

Random Forests (RF) is an ensemble learning method in which multiple decision trees are constructed, and then the class is estimated by averaging all the trees. When constructing a single decision tree (as shown



in the diagram), the algorithm decides which variable to use and what threshold value to use at every node in the tree. The goal is to minimize the objective function (T). The decision tree is constructed node-by-node:

$h(v, X_j) : F \times T \rightarrow \{0, 1\}$. F : Feature space, T : Objective Function

$\{0, 1\}$: False and True. v : a data point denoted by $v = (x_1, x_2, \dots, x_d) \in F$

x_i : represents some attributes of the data point – features

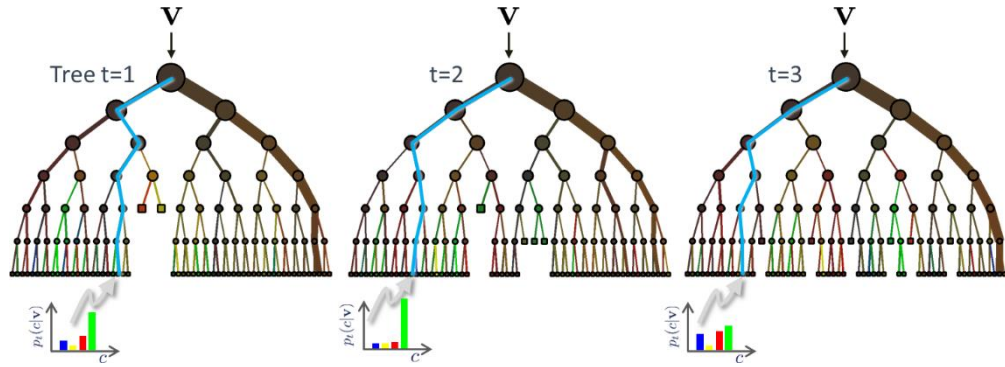
d : dimensionality of feature space

If not all features are needed, only a small set is used:

$\emptyset(v) = (x_{\emptyset 1}, x_{\emptyset 2}, \dots, x_{\emptyset d'}) \in F_{d'}, \in F$

$X \in T$: parameters of objective function at the j th split node.

As an example, in the diagram three different trees are constructed for the incoming data point v , and when the average across all trees is taken, it is classified as green.



I chose two RF hyperparameters to optimize – the number of trees and the objective function. While the number of trees is straightforward, the objective functions I used were ‘gini impurity’ and ‘entropy’. Gini impurity is a measure of how often a randomly chosen element from the set would be incorrectly labeled if it were randomly labeled (according to the distribution of labels in the dataset). Entropy is the measurement of disorder or uncertainty that is reduced as more information is gained.

2.3. Feature Importance

After an RF algorithm has been run on a dataset, the average of how each feature minimized the objective function across all the decision trees is taken. This gives the relative importance of every feature. This importance can also be identified using multiple attribute evaluator algorithms such as Principal Component Analysis, Information Gain, and Correlation.

2.4. Performance of Machine Learning model

Classification Machine Learning (ML) models can be evaluated by how accurately they predict the class variable in a test dataset and how much they reduce the error. I have used Accuracy and Root Mean Square Error (RMSE) values to measure the performance of my models.

A further evaluation of classification is done using a Confusion Matrix which provides the number and percentage of True Positive, True Negative, False Positive and False Negative classifications.

2.5. Python Libraries

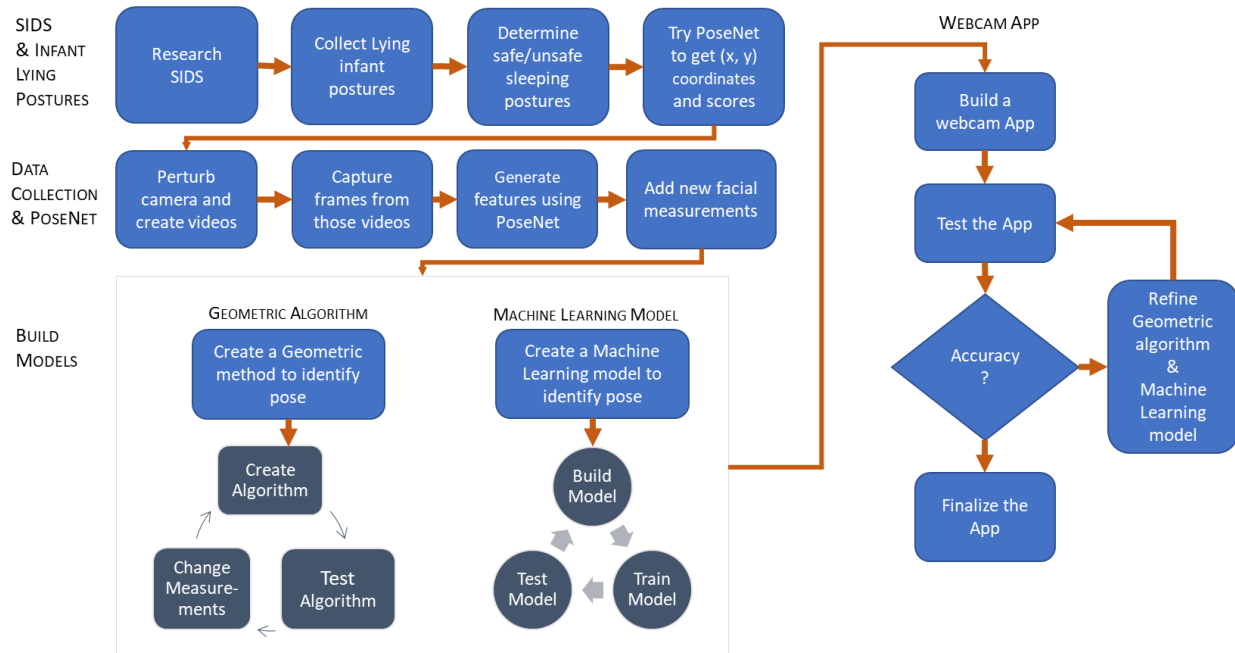
I used many Python libraries for my geometric algorithm, ML modeling and visualization. These are: numpy, pandas, scikit-learn, matplotlib, and seaborn. For processing images and video frames, and generating coordinates of body parts, I used OpenCV and PoseNet.

3. Softwares, Methodology and Procedure

The methods I used are described in the flowchart representation below. This process consisted of researching SIDS and infant lying postures, collecting data, building a geometric algorithm and a machine learning model, and finally, building a webcam app.

3.1. Softwares

I used Excel, WEKA (a data mining and machine learning software), PyCharm (a development environment for Python), pictures of infants (downloaded from the Internet) and Microsoft Paint (a software for adjusting zoom, changing contrast, and altering exposure of a picture). For capturing infant images and videos, I used a webcam and a webcam stand.



3.2. Data Collection

I started by researching SIDS to understand which infant sleeping postures would be safe (considered to not be at risk as per AAP) or unsafe (high-risk as per AAP). There were many variations in the infants' sleeping postures (ex: an infant could be lying on its side, back, or stomach and its face could be visible or covered, etc.) Since AAP recommends that an infant should not be on its side or its stomach, I decided to label all such postures as unsafe. There were cases where an infant was on its side and its face was visible, but to be cautious, I labelled such pictures to be unsafe as well. Using this approach, I labelled 53 pictures (which included infants of different races). The labels consisted of body position, face position and whether the lying posture was high-risk (unsafe) for SIDS. See the Appendix for the pictures and their source links.

When I ran the PoseNet model to identify the infants' body parts, I made several interesting observations:

- The angle from which the picture was taken, and the angle of the webcam pointed towards the infant picture were both important in determining the relationship between the coordinates of body parts. So, I decided that the webcam must be placed directly on top of and parallel to the sleeping infant.
- When I was searching for images, my webcam inadvertently was facing the computer monitor, and PoseNet recognized multiple infants in different pictures on the screen. Since this would not be a realistic scenario in real-life when monitoring only one infant, I modified the PoseNet code so that only one infant would be identified in every picture.
- The webcam did not recognize body parts if it was not the right distance from the infant's picture on the screen. So, for some smaller-sized pictures, I had to adjust the zoom level after opening the picture in Microsoft Paint.
- Once the joints of an infant were identified, I could move the webcam in any angle or distance, and it continued to identify the joints with circles. This gave me the idea to perturb the webcam to generate various distances and angles. Since the frame rate was between 10 and 12 fps, PoseNet captured different coordinate values for body parts from the same pictures (because I had around 1000 frames from each picture). This variation in my dataset allowed me to simplify my app, since I did not have to use geometric transformations on frames from the webcam video to generate various coordinate combinations.
- In some cases, I found that body parts were clearly identified during the afternoon and not during the morning when sunlight came into my room. This led me to experiment with different light exposures and contrasts (in Microsoft Paint), which was especially useful in determining the light conditions under which my application functioned properly. So, I changed my app to give an audio alert (a beeping sound) when an infant's pose was not recognized. This is the same audio alert that occurs when an infant's pose is unsafe. This is important because when a caregiver is putting their infant to sleep and turning on the app, an audio alert would prompt the caregiver to adjust the lighting if necessary.
- For a few sleeping positions when the infant was on its side, the PoseNet model classified body parts incorrectly (by putting the dots for joints in incorrect places). I realized that the model was making incorrect assumptions about angles and distances between body parts, which were not possible for such a posture. So, I labelled these pictures as unsafe.
- I found that regardless of an infant's race, PoseNet identified its body parts correctly. This was useful because even for a general search string, a Google search provides more pictures of certain races over others. So, selecting PoseNet rather than another object recognition model is especially important because other ML techniques could potentially rely on the colors in pictures (and thus develop a racial bias).

Then, I started the webcam and collected about 1,000 frames of video for each picture. As the data from each frame was getting recorded, I moved the webcam stand around so that different distances and angles were captured. For every picture of an infant lying on its side or lying on its stomach, I categorized that picture as unsafe. At the end, I had a comma separated values (CSV) file with 53,000 rows of data (one per video frame) with coordinates of body parts from PoseNet.

As I analyzed the infant pictures, I came up with 6 categories/classes (see table). I did not create a category for postures such as body-on-side and face-down or body-on-back and face-down because they were not realistic. (Also, in both cases an alert would be generated because no face features are visible). I added a category of “None” when no body parts were visible or if they could not be recognized. The Alert column in the table indicates ‘No’ for safe posture and ‘Yes’ for unsafe posture.

Body Pose	Face Pose	Alert
Stomach	Side	Yes
Stomach	Down	Yes
Side	Up	Yes
Side	Side	Yes
Back	Up	No
Back	Side	No

3.3. Building an algorithm and a model

I developed a geometric algorithm and an ML model by using WEKA and Python, and I went back and forth between the data analysis and the code while capturing various nuances in the images of infants. My goal was to improve the accuracy with which the algorithm and model identified the infants’ pose.

3.3.1. Geometric Algorithm

With the data I had collected, I built an algorithm to identify each body part with a Boolean True or False value (whether a body part was visible or not). I found that my algorithm could make decisions about the safety of an infant’s lying posture by using PoseNet coordinates of an infant’s nose, eyes, ears, shoulders, elbows and wrists. In the majority of my pictures, hips, knees and ankles were either not visible or covered. Other observations and considerations that I used in the algorithm were –

- I found that when an infant’s face was on its side, the distance from its nose to one eye was different from its nose to the other eye. So, I added two measures that were the distances between nose and each of the eyes.
- In some pictures, PoseNet showed that there was an ear with a low confidence score when the ear was not clearly visible. I decided to mark partially visible ears as ‘False’ when the angle between the axis of eyes and the ear was less than 90 degrees because such an angle is not possible. This led me to add two measures which were the angles for each ear.
- In the pictures where the infant’s face was on its side, the infant’s arms could hinder the clarity, but PoseNet was able to identify the infant’s elbow and wrist. In order to make this clear, I decided to add two more measures: the angles between shoulder, elbow, and wrist for each side (both left and right).

I modified my CSV file by adding these additional features to my data set. See table with a list of the additional measurements.

When I ran my geometric algorithm, I used all of these measurements to decide if I should mark a part of the face as ‘True’ (visible) or ‘False’ (not visible).

Added Measurements
Angle_leftShoulder_leftElbow_leftWrist
Angle_rightShoulder_rightElbow_rightWrist
Angle_rightEar_rightEye_leftEye
Angle_leftEar_leftEye_rightEye
Angle_leftEye_rightEye_Nose
Distance_Nose_leftEye
Distance_Nose_rightEye

```
# Check if angle of left ear with axis of eyes is less than threshold
angleLeftEar = abs(angle_LEr_LEy_REy)
if (angleLeftEar < 90):
    dictParts['leftEar']= False

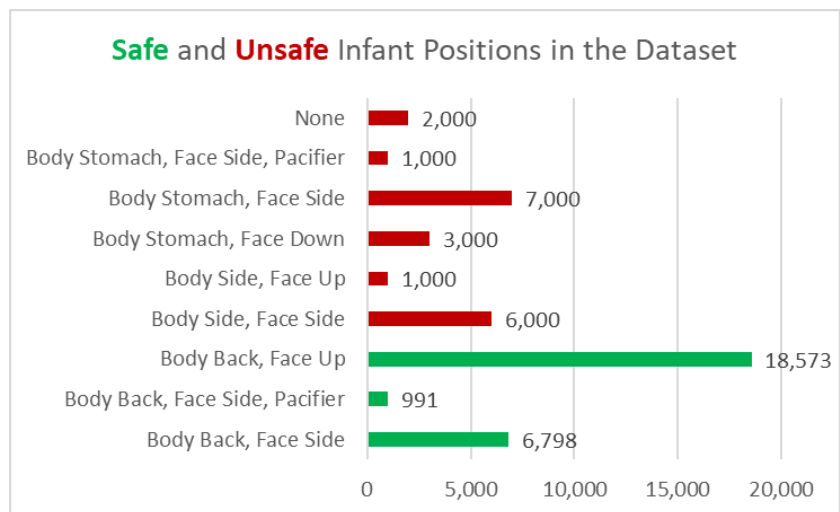
# Check if angle of right ear with axis of eyes is less than threshold
angleRightEar = abs(angle_REr_REy_LEy)
if (angleRightEar < 90):
    dictParts['rightEar']= False

# Check if angle of nose with eyes is greater than threshold
angleNose = abs(angle_LEy_Nose_REy)
if (angle_LEy_Nose_REy > 150):
    print("Angle of Nose with Eyes is too wide . Set Nose to False.")
    dictParts['nose'] = False
```

This Python code snippet is from my geometric algorithm function. I took the distance and angle measurements of the infant’s face to make a decision about marking the left ear or the right ear as ‘False’ (not visible).

Interestingly, I found that relative to an adult’s face, the angle between an infant’s nose and eyes can be large.

By the time I completed my geometric algorithm, the categories in my dataset were distributed as in the horizontal bar chart. This reduced the number of rows in the dataset (46,362). It was the result of deleting rows that had errored data (ex: rows with several zero values), and deleting rows with a category of images where the body was on the stomach, but the face was up. Such a pose is more likely a studio picture than an infant’s lying position.



While I was able to recognize parts of the face, it was difficult to identify body parts by using threshold values of angles or distances. So, rather than programmatically write a decision function, I decided to build a machine learning model which could process all of the features in the entire dataset.

3.3.2. Machine Learning Model

I loaded all of the CSV data into WEKA. Since each row of data was labelled, I marked the category variable as the **dependent variable** or the nominal class that had to be predicted for every row. All columns or attributes (ex: coordinates, distances, and angles) were the **independent variables**. I split the data set into training and testing, and used two different splitting methods – percentage split of 66% training and 34% testing and 10-fold cross validation. In cross-validation testing, only one-fold of data is taken at a time and the final results are the average of all the folds.

Algorithm	Accuracy
Decision Table	64.78 %
Neural Network (NN, DL4JMLP in WEKA)	78.40 %
Support Vector Machine (SMO in WEKA)	82.86 %
k-Nearest Neighbors (IBK in WEKA)	90.96 %
Decision Trees	88.96 %
Random Forest	91.07 %

I first ran different classification ML algorithms and found that Random Forest had the highest accuracy of all the algorithms. I was surprised to find that the RF model performed better than the deep learning neural network algorithm. Upon further reading, I found that the randomness of the splitting criteria at every node of every decision tree in the RF algorithm results in a more accurate prediction for tabular datasets such as mine. I also found that RF algorithms took less time in WEKA than neural networks – which apparently is an advantage of this algorithm.

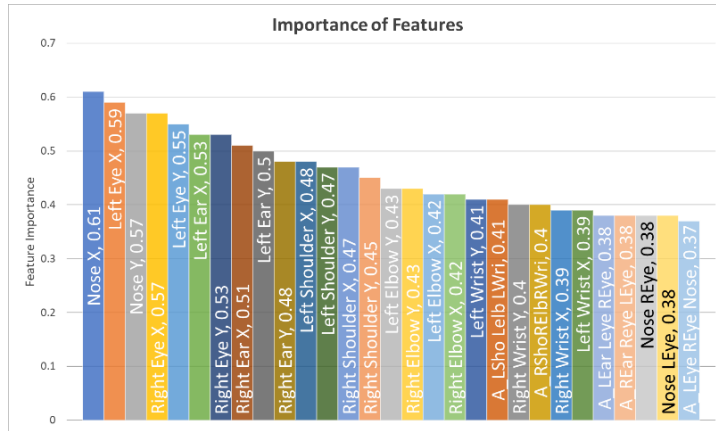
Since I had used the default values of hyperparameters, I decided to modify them and run the RF algorithm again.

	Rows	Description	Trees	Testing Method	Model Accuracy	RMSE
1	53,000	All columns	100	10-fold CV	91.07%	0.1047
2	49,363	Delete errors	100	10-fold CV	96.61%	0.0723
3	49,363	No change	200	10-fold CV	96.61%	0.0721
4	49,363	No change	500	10-fold CV	96.62%	0.072
5	49,363	No change	1,000	66% Train, 34% Test	96.67%	0.0719
6	49,363	No change	2,000	10-fold CV	96.62%	0.0719

Although it took time to build the model when the number of trees increased to 1,000 or 2,000, there was no significant increase in accuracy or decrease in error. So, I decided to keep the number of Trees at 500. The RMSE value of 0.072 was not much different from the RMSE value from RF with a higher number of trees.

Later, when I took the RF model and incorporated it into my

webcam application, I changed the objective function from ‘gini impurity’ to ‘entropy’, but there was no significant difference in the accuracy or the error. So, I kept the default ‘gini’ impurity objective function in scikit-learn.



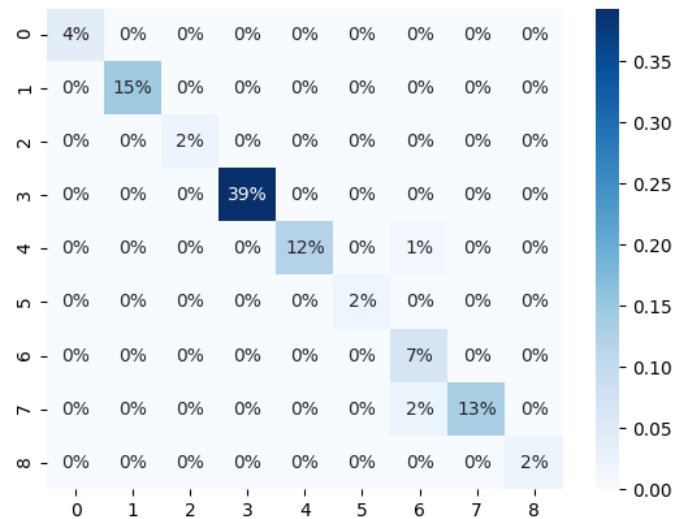
To simplify the RF model, I looked at Feature Importance and found that the values varied from 0.61 to 0.37 as generated by the RF algorithm. (See the color-coded vertical bar chart on the left.) When I removed the least important feature or other lower valued features, the model's accuracy changed by half a percent point. I also tried Principal Component Analysis, Information Gain, and Correlation attribute evaluator methods in WEKA.

I found that they generated a list of under 10 features instead of the total of 29 that I was trying to reduce. Since recognizing the SIDS related posture is critical, I decided to not reduce this number of features or simplify the model. An example of reducing the number of features was removing the x coordinate value while keeping the y coordinate value of the same body part. While this may make sense algorithmically for highly correlated numerical features, it does not make sense for a body part in a real-life infant image. So, I kept both x and y coordinate values.

Note that identifying the impact of individual features is akin to isolating the effect of that variable. This is similar to testing in a **controlled** environment where one variable is varied while others are held constant.

To identify False Positives or False Negatives, I checked the Confusion Matrix. Since the RF model predicts posture to be in one of 9 classes (numbered 0-8 on the axes), the True Positive (TP) values along the diagonal from top-left to bottom-right identify correct classification of an image. Reading row-wise, the matrix has False Negatives for the feature in the diagonal and reading column-wise, the matrix has False Positives for that feature.

Given the high accuracy of 96% for the overall RF model, there are very few False Negatives (FN) and False Positives (FP). The three categories with FN or FP values are 4:Side_Side, 6:Stomach_Down and 7:Stomach_Side.



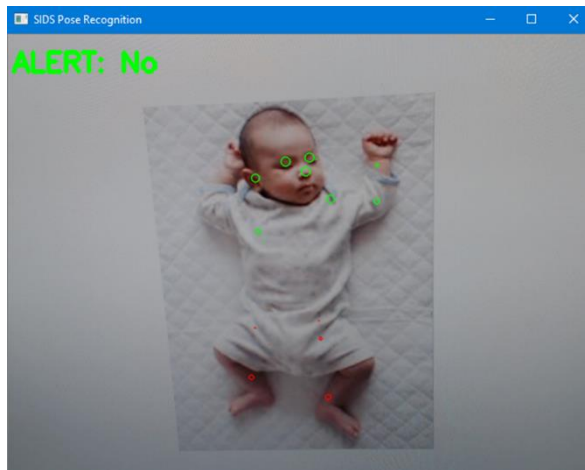
Confusion Matrix: BodyPose_FacePose
 0:None, 1:Back_Side, 2:Back_SideP, 3:Back_Up,
 4:Side_Side, 5:Side_Up, 6:Stomach_Down
 7:Stomach_Side, 8:Stomach_SideP

With these results, I incorporated this RF model with its hyperparameter values into my "SIDS Pose Recognition" application that I wrote in Python. It runs on any computer with a webcam.

4. Results and Interpretation

When I run my “SIDS Pose Recognition” app with the webcam pointing at an infant’s picture, it displays an alert on the screen and prints the following result. Whenever the posture is unsafe, it displays the visual alert and gives an audio alert (a beeping sound). See the pictures with alerts in Green text (safe posture) and Red text (unsafe posture).

The printout below shows the measurements and results as displayed on screen for the geometric algorithm as well as the machine learning (ML) model. The overall alert only uses the results from the ML model. It is the selected automated method because it recognizes postures more accurately even across a large dataset.



PoseNet Parts of Body

Nose 99.76%, Left Eye 99.32%, Right Eye 99.83%, Left Ear 5.30%, Right Ear 96.24%, LeftShoulder 52.27%, Right Shoulder: 47.92%, Left Elbow 66.89%, Right Elbow 9.45%, Left Wrist 41.45%, Right Wrist: 1.62%

Calculated Measurements of Face

Left Ear angle is: 160.41
 Right Ear angle is: 186.67
 Angle the Nose forms with Eyes is: 89.93
 Left Eye - Nose distance is: 14.31
 Right Eye - Nose distance is: 20.53

More Right side of face is visible

Calculated Measurements of Shoulders, Elbows and Wrists

leftShoulder_leftElbow_leftWrist angle is: 265.02
 rightShoulder_rightElbow_rightWrist angle is: 58.52

GEOMETRIC ALGORITHM

Visible parts of Face: Nose, Eyes, Right Ear
 Face Pose: Face Up

ML MODEL

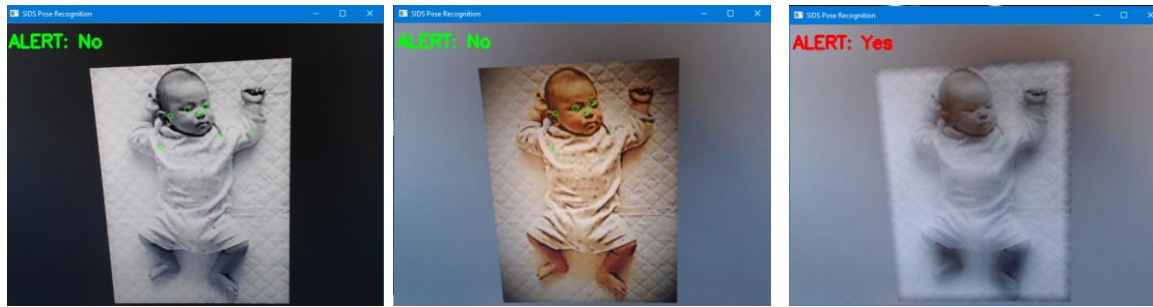
Model: Random Forest Classifier
 Predicted Pose: Body Back, Face Up

Geometric Algorithm Alert: No

ML model Alert: No

OVERALL ALERT: NO

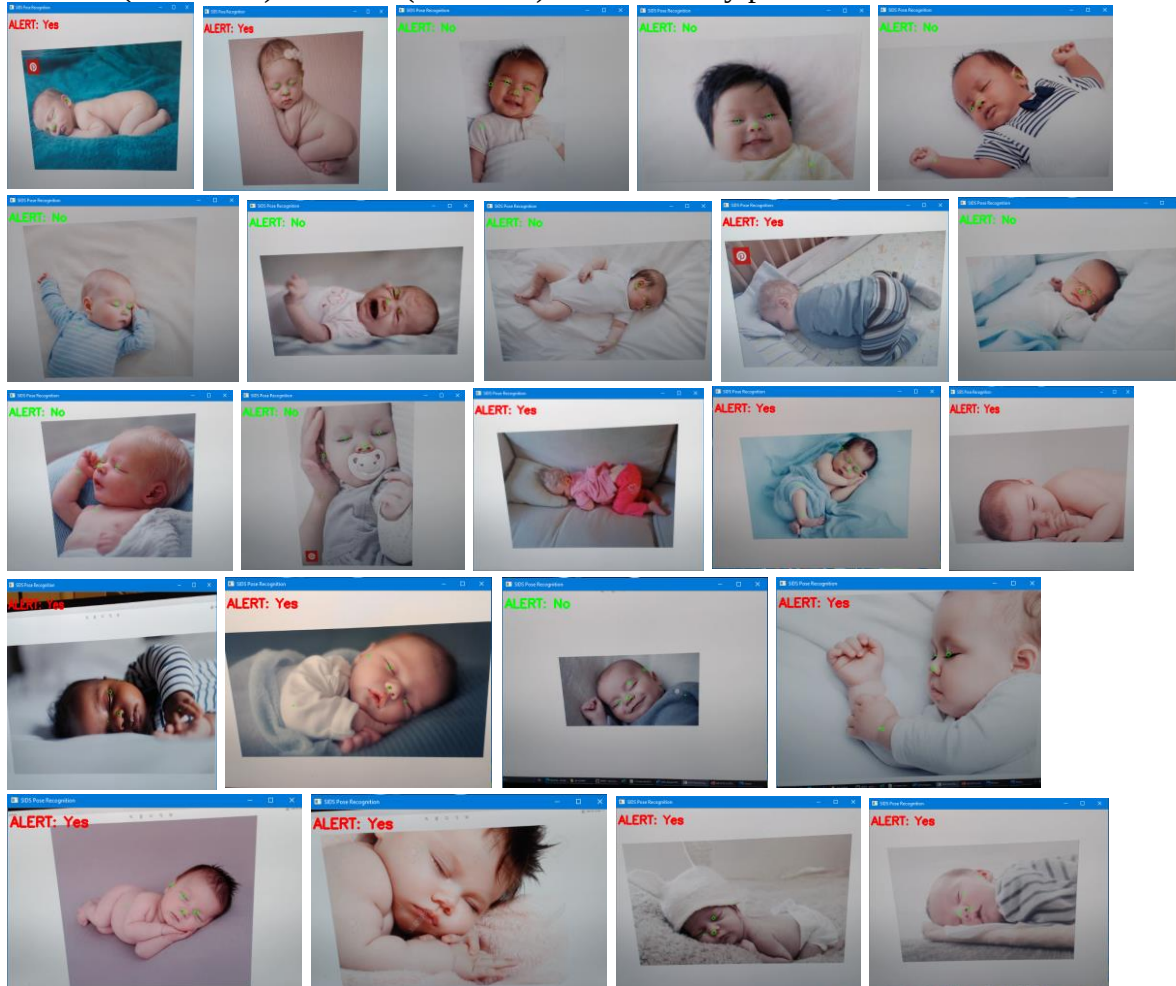
Under simulated low light conditions, the app gives:



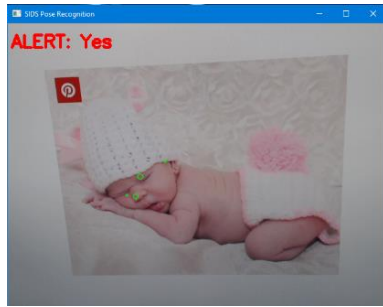
This indicates that the app can be used in low light conditions.

Regardless of the state of infant's eyes (open or shut), the model correctly recognizes the infant's posture.

See Red (Alert: Yes) or Green (Alert: No) Alert on every picture below:



The ML model categorized the following picture intermittently as safe or unsafe, and this



continued to happen in spite of perturbing the webcam. In the frame-by-frame classification, it was going back and forth

between BodyBack_FaceUp (Class 3, No Alert)

BodyStomach_FaceSide (Class 7, Alert). The predicted class probability was close to 50% in each case. This problem occurred frequently with this picture and at certain webcam angles. To take a cautious approach, I categorized this posture as unsafe.

5. Conclusion and Discussion

My conclusion is two-fold:

- a. My first hypothesis that PoseNet can be used to identify infant body parts was correct.
- b. My second hypothesis that a machine learning model could effectively recognize infant lying posture was also correct.

Since the geometric algorithm is a deterministic method, it relies on threshold values that are difficult to modify for each new image and also achieve a high accuracy. In contrast, the ML model, although probabilistic, is more effective and accurate in recognizing an infant's lying posture across a larger set of images. I was able to successfully build a webcam application that can recognize an infant's posture and alert the caregiver about unsafe postures that are high-risk for SIDS.

In terms of limitations,

- PoseNet does not recognize ears when they are the only visible part of the infant's face. An improvement such as using a haarcascade model could be made to identify infant ears.
- The PoseNet model I used to capture localization data was built using pictures which primarily consist of adult faces and bodies. Between parts of the adult body, there are clearly defined and well-measured proportional distances and angles. For infants, these measurements would be different. It is likely that this was the underlying reason for the ML model classifying a very few pictures, at certain webcam angles, intermittently into two different classes. Therefore, a PoseNet model customized for infants is needed where the underlying mathematical spring model of human body parts is tuned for infants.

For future research,

- The app could be improved by using object recognition techniques to identify pacifiers (since PoseNet does not recognize pacifiers or an infant's mouth). I tried to use Dlib, which identifies more than 65 key points on a human face, but it did not work for infant faces.
- Monitoring an infant under low light conditions (such as nighttime) is difficult. This could be improved by using infrared cameras and image matrix transformations. Also, the underlying machine learning model in PoseNet could be modified to better address this issue.
- With access to a larger number of infant pictures and videos, my dataset would be more representative of the entire infant population. With this, I could conduct a thorough error analysis to build a more statistically sound model.
- Since my app currently utilizes a live webcam feed, I could use this to target other risk factors of SIDS as well. By using the intensity variations of reflected light around an infant's collar bone and chest, I could extract data about the breathing pattern. And, by addressing posture as well as breathing patterns, I would be able to build a more precise and conclusive SIDS app.
- If I recorded the sequence of infant postures over time, I could identify specific movements. This data could be used for numerous movement disorders, such as dystonia and tics.

6. References and Acknowledgements

6.1. References

Computer Vision: Algorithms and Applications, 2nd Ed., szeliski.org/Book/.

“Data and Statistics for SIDS and SUID.” *Centers for Disease Control and Prevention*, Centers for Disease Control and Prevention, 10 Nov. 2020, www.cdc.gov/sids/data.htm.

Duncan, Jhodie R. “Figure 2.1: Triple Risk Model for SIDS Proposed by Filiano and Colleagues in 1994, Highlighting the Intrinsic, Extrinsic and Additional Risk Factors for SIDS. (Adapted from (32).) - SIDS Sudden Infant and Early Childhood Death - NCBI Bookshelf.” *National Center for Biotechnology Information*, U.S. National Library of Medicine, www.ncbi.nlm.nih.gov/books/NBK513399/figure/fig2_1/?report=objectonly.

“Just Had a Baby? A New App Helps Keep Them Safe.” *Rutgers University*, www.rutgers.edu/news/just-had-baby-new-app-helps-keep-them-safe.

“Machine learning - Random forests.” *YouTube*, uploaded by Nando de Freitas, 21 Feb. 2013, www.youtube.com/watch?v=3kYujfDgmNk

“New Rutgers App Seeks to Reduce Infant Mortality.” *Rutgers University*, www.rutgers.edu/news/new-rutgers-app-seeks-reduce-infant-mortality.

“Now We Know Why Babies Shouldn't Sleep Face Down.” *ScienceDaily*, ScienceDaily, 25 Oct. 2017, www.sciencedaily.com/releases/2017/10/171025103142.htm.

Papandreou, George, et al. “PersonLab: Person Pose Estimation and Instance Segmentation with a Bottom-Up, Part-Based, Geometric Embedding Model.” *ArXiv.org*, 22 Mar. 2018, arxiv.org/abs/1803.08225.

Papandreou, George, et al. “Towards Accurate Multi-Person Pose Estimation in the Wild.” *ArXiv.org*, 14 Apr. 2017, arxiv.org/abs/1701.01779.

Rwightman. “Rwightman/Posenet-Python.” *GitHub*, github.com/rwightman/posenet-python.

“SIDS and Other Sleep-Related Infant Deaths: Updated 2016 Recommendations for a Safe Infant Sleeping Environment.” *Pediatrics*, U.S. National Library of Medicine, pubmed.ncbi.nlm.nih.gov/27940804/.

“Sudden Infant Death Syndrome (SIDS): Symptoms & Causes: Boston Children's Hospital.” *Boston Children's Hospital*, www.childrenshospital.org/conditions-and-treatments/conditions/s/sudden-infant-death-syndrome-sids/symptoms-and-causes.

Tensorflow. “Tensorflow/Tfjs-Models.” *GitHub*, github.com/tensorflow/tfjs-models/tree/master/posenet.

6.2. Acknowledgements

I thank my dad for referring me to resources on using machine learning techniques and building applications in Python.

7. Appendix A: Infant Pictures and Source Links

	Image	Link
1		https://hips.hearstapps.com/hmg-prod.s3.amazonaws.com/images/baby-sleeping-covered-with-soft-blanket-royalty-free-image-937849444-1556912403.jpg?crop=0.668xw:1.00xh;0,0&resize=980*
2		https://static.zentrum-der-gesundheit.de/img/4e9038100f6cb5f6a715c5bd8a5f275?width=721&height=403
3		https://cdn.cnn.com/cnnnext/dam/assets/170508111409-05-grandparents-parenting-live-video.jpg
4		https://i.guim.co.uk/img/media/f6e1f91d558339630c08cf388a275a1d78ca24de/0_191_5700_3419/master/5700.jpg?width=1200&height=900&quality=85&auto=format&fit=crop&s=3a580ce705a3ea0027f22e0d5ef47c31
5		https://hips.hearstapps.com/hmg-prod.s3.amazonaws.com/images/baby-sleep-position-1538650093.jpg?crop=0.782xw:0.587xh;0.218xw,0.173xh&resize=1200:*
6		https://images.theconversation.com/files/242494/original/file-20181026-7056-t9kw9c.jpg?ixlib=rb-1.1.0&q=45&auto=format&w=1200&h=675.0&fit=crop
7		https://i0.wp.com/intelligentmother.com/wp-content/uploads/2018/01/baby-sleeping-face-down.jpg?resize=1020%2C510&ssl=1
8		https://findyourmomtribe.com/wp-content/uploads/2019/11/Face-Down-%E2%80%93-Should-My-Baby-Sleep-In-That-Position3.jpg
9		https://images.agoramedia.com/ugcphotoservice/100/2017/9/17/53895848/216c621b-7c47-43c9-bd4e-962b1a841d2d.jpg
10		https://i.ebayimg.com/images/g/IZUAAOSw4UtWTsxz/s-l1600.jpg
11		https://promobox.hu/wp-content/uploads/2015/06/latasproblemak.jpg
12		https://cdn2.momjunction.com/wp-content/uploads/2014/05/100-Modern-And-Stylish-Baby-Boy-Names-With-Meanings1.jpg
13		https://images2.minutemediacdn.com/image/upload/c_fill,g_auto,h_1248,w_2220/v1555275036/shape/mental/floss/55013-istock-523497638.jpg?itok=aaqv4VZG
14		https://images.ctfassets.net/ccu710f53h3h/1mHrgb19OrqviBsGfciH8d/4086a4e8e82d960eee72a85aa66919a4/Unique-Baby-Boy_1536x680.jpeg
15		https://i1.wp.com/www.themomkind.com/wp-content/uploads/2020/11/Untitled-design-2020-11-30T114053.256.png?fit=1600%2C900&ssl=1
16		https://i2.wp.com/intelligentmother.com/wp-content/uploads/2018/01/Tummy-Sleeping-In-Babies.jpg?w=730&ssl=1
17		https://i0.wp.com/intelligentmother.com/wp-content/uploads/2018/01/Baby-Sleep-On-Tummies.jpg?w=730&ssl=1
18		https://i.etsystatic.com/6257626/r/il/5c326d/1027399860/il_794xN.1027399860_59ho.jpg
19		https://img.staticdj.com/d18c9ae9e95e1a494bd62e423a5c1028_1080x_nw.jpeg
20		https://img.staticdj.com/77e64c969025d5e78ff74ce46e4ca6e8_1080x.png
21		https://theparentingsolution.com/wp-content/uploads/2020/03/sleep-mistakes-1280x960.jpg
22		https://i.insider.com/5dfd36ad954bda45e60f81a9?width=1136&format=jpeg
23		https://i0.wp.com/post.medicalnewstoday.com/wp-content/uploads/sites/3/2020/03/GettyImages-a0028-000090_hero-1024x575.jpg?w=1155&h=1528
24		https://imgix.bustle.com/2017/1/25/e6477665-15d8-4c71-9c53-39ecad79ff92.jpg
25		https://i2.wp.com/intelligentmother.com/wp-content/uploads/2018/01/Baby-Sleeping.jpg?w=730&ssl=1
26		https://i1.wp.com/intelligentmother.com/wp-content/uploads/2018/01/Baby-sleeping-face-down.jpg?w=730&ssl=1
27		https://i1.wp.com/intelligentmother.com/wp-content/uploads/2018/01/Back-to-Sleep.jpg?w=730&ssl=1

	Image	Link
28		https://www.missmum.at/wp-content/uploads/2020/05/Mann-trennt-Baby-Nabelschnur-mit-Schutzmaske-ab-scaled.jpg
29		https://cdn2.familie.de/d7/f7/ab/8d47018aa50c459b561a6f3b04_YyAxOTkzeDk5Nis2NCsyMDkCcmUgMTIwMCA4MDADZDc4Nzg1NGIxZjA=_newborn-boy.jpg
30		https://live.mrf.io/statics/i/ps/cdn.primedia.co.za/primedia-broadcasting/image/upload/c_fill,h_437,w_700/nf0ukzu2wflqm2zdoby?width=1200&enable=upscale
31		https://cdn.mos.cms.futurecdn.net/EjPxrGwjHKpsP9EGgjEVx-1200-80.jpg
32		https://thumbs.dreamstime.com/b/asian-baby-smiling-17977176.jpg
33		https://imagesvc.meredithcorp.io/v3/mm/image?url=https%3A%2F%2Fstatic.onecms.io%2Fwp-content%2Fuploads%2Fsites%2F38%2F2015%2F08%2F13013803%2F550_102243782.jpg&q=85
34		https://images.indianexpress.com/2020/01/baby-sleep.jpeg
35		https://cdn.shopify.com/s/files/1/0758/3703/products/image_3d7179d9-85f3-4338-b696-5317edcbec55_580x@2x.jpeg?v=1545215303
36		https://post.healthline.com/wp-content/uploads/2019/10/Baby_Sleeping_Blanket_1200x628-facebook.jpg
37		https://zase.mk/uploads/baby-sleep-position-1538650093.jpg
38		https://www.swissmom.ch/typo3temp/assets/_processed_/9/c/csm_iStock-888422284_301dd2755f.jpg
39		https://img.staticdj.com/ef5172ec47241ba8628b8a3f0e8e61e8_1080x.png
40		https://sleepylilbear.files.wordpress.com/2018/05/baby-sleep.png?w=736
41		https://raisingchildren.net.au/__data/assets/image/0022/49243/cosleeping-with-your-babynarrow.jpg
42		https://post.healthline.com/wp-content/uploads/2019/07/Baby_Sleeping_1200x628-facebook.jpg
43		https://cdn01.dailycaller.com/wp-content/uploads/2019/06/shutterstock_305671484-e1560792548613.jpg
44		https://themilitarywifeandmom.com/wp-content/uploads/2014/05/how-to-get-a-newborn-to-sleep.jpg
45		https://i.pinimg.com/originals/96/56/f5/9656f56417c7aad0a9444fc1151841bb.jpg
46		https://babyexcel.com/wp-content/uploads/2020/04/newborn-sleep-patterns.jpg
47		https://www.asiaone.com/sites/default/files/original_images/Oct2019/20191029_sleeping-baby-yawn-asian_unsplash.jpg
48		https://jnj-content-lab.brightspotcdn.com/67/51/3a0a4a904a69b0f8751386f8b774/baby-sleep-3-shutterstock-144844168-1640.jpg
49		https://nmforum.s3.eu-west-1.amazonaws.com/20170219/1a271f6a11a8a1cb6f4fd49b59020aa8.jpg
50		https://thumbs.dreamstime.com/b/beb%C3%A9-lindo-que-duerme-primer-hermoso-de-la-cara-del-s-ni%C3%B1o-132183530.jpg
51		https://cdn.newsapi.com.au/image/v1/d44ce2031d9b6fc82741be1ea7197221?width=640
52		https://images.pexels.com/photos/2111997/pexels-photo-2111997.jpeg?auto=compress&cs=tinsrrgb&dpr=2&h=650&w=940
53		https://thumbs.dreamstime.com/b/portrait-sleeping-innocent-infant-baby-hat-129237478.jpg

8. Appendix B: Python Code for Machine Learning Model

Function for building Random Forest model:

```
def build_model(X, y):
    # build a Random Forest model and return model

    from sklearn.ensemble import RandomForestClassifier
    # Try different values for criterion (objective function) and n_estimators (trees)
    # in different runs
    model = RandomForestClassifier(criterion='gini', n_estimators=500)
    model.fit(X, y)
    model_name = type(model).__name__
    print(colored("\nML MODEL", attrs=['bold']))
    print("ML model name: ", model_name)

    return model
```

Function for generating classification report and text version of confusion matrix:

```
def generate_confusion_matrix(model, X_input, y_input):
    # generate classification report and confusion matrix

    predicted_pose = model.predict(X_input)
    expected_pose = y_input

    print("Classification Report")
    print(metrics.classification_report(expected_pose, predicted_pose))

    print("Confusion Matrix")
    confusionMatrix = metrics.confusion_matrix(expected_pose, predicted_pose)

    return
```

Function for identifying infant pose in each image:

```
def predict_single_pose(model, arrSinglePose):
    # predict the pose from feature values in the input array

    # stack with zeros because predict expects more than a single row
    new_row = np.zeros(arrSinglePose.size)
    arrTemp = np.vstack([arrSinglePose, new_row])
    arrPredictedSinglePose = model.predict(arrTemp)
    predicted_single_pose = arrPredictedSinglePose[0]

    return predicted_single_pose
```