




?

NEW AND  
REVISED  
EDITION

ALEXANDER  
KIRA

# THE BATH ROOM

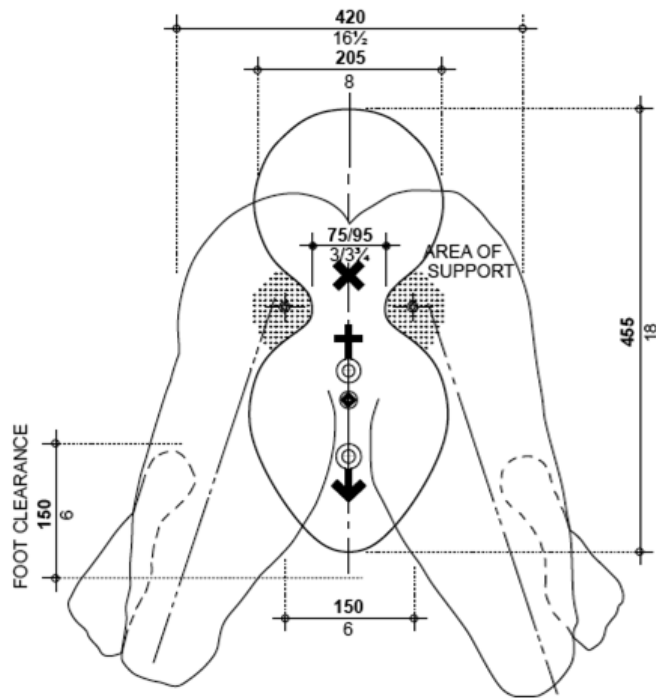
 A VIKING COMPASS BOOK

\$7.95

Alexander Kira, *Łazienka*  
Alexander Kira, *The Bathroom*

## Analiza pozycji ludzkiego ciała na toalecie kucanej. Analysis of the position of the human body on the semi-squat toilet.

Widok z góry: niezbędne wymiary i odstępy  
dla toalety kucanej przy wysokości 255 mm (10 cali).  
Plan view of necessary dimensions and clearances  
for a semi-squat water closet at 255 mm (10 inch) height.





## VIRGINIA POLYTECHNIC INSTITUTE AND STATE UNIVERSITY

Public Restroom Study

Did you have to wait in line to use the restroom?  
(check one) ☐ YES ☐ NO

If yes, please estimate the time you waited in line  
(check one)

☐ More than 5 minutes

☐ 1 to 5 minutes

☐ Less than 1 minute

Please check all of the activities you performed while  
you were in the restroom.

<input type="checkbox"/> Wash hands	<input type="checkbox"/> Put in/take out contacts
<input type="checkbox"/> Check appearance	<input type="checkbox"/> Clean glasses
<input type="checkbox"/> Comb/brush hair	<input type="checkbox"/> Take medicine
<input type="checkbox"/> Straighten clothes	<input type="checkbox"/> Smoke
<input type="checkbox"/> Straighten tie	<input type="checkbox"/> Talk
<input type="checkbox"/> Change clothes	<input type="checkbox"/> Wait on other person
<input type="checkbox"/> Wash face	<input type="checkbox"/> Change diaper
<input type="checkbox"/> Brush/floss teeth	<input type="checkbox"/> Assist child/children
<input type="checkbox"/> Urinate	<input type="checkbox"/> Other (please specify, _____)
<input type="checkbox"/> Defecate (bowel movement)	<input type="checkbox"/> Other (please specify, _____)

Which activity do you believe took the most time?

Did you use (place a check by your answer):

the regular stall?	<input type="checkbox"/> YES	<input type="checkbox"/> NO
the handicapped stall?	<input type="checkbox"/> YES	<input type="checkbox"/> NO
the urinal?	<input type="checkbox"/> YES	<input type="checkbox"/> NO

What did you like most about the restroom?

What do you think would have improved the restroom?

Year of birth: \_\_\_\_\_

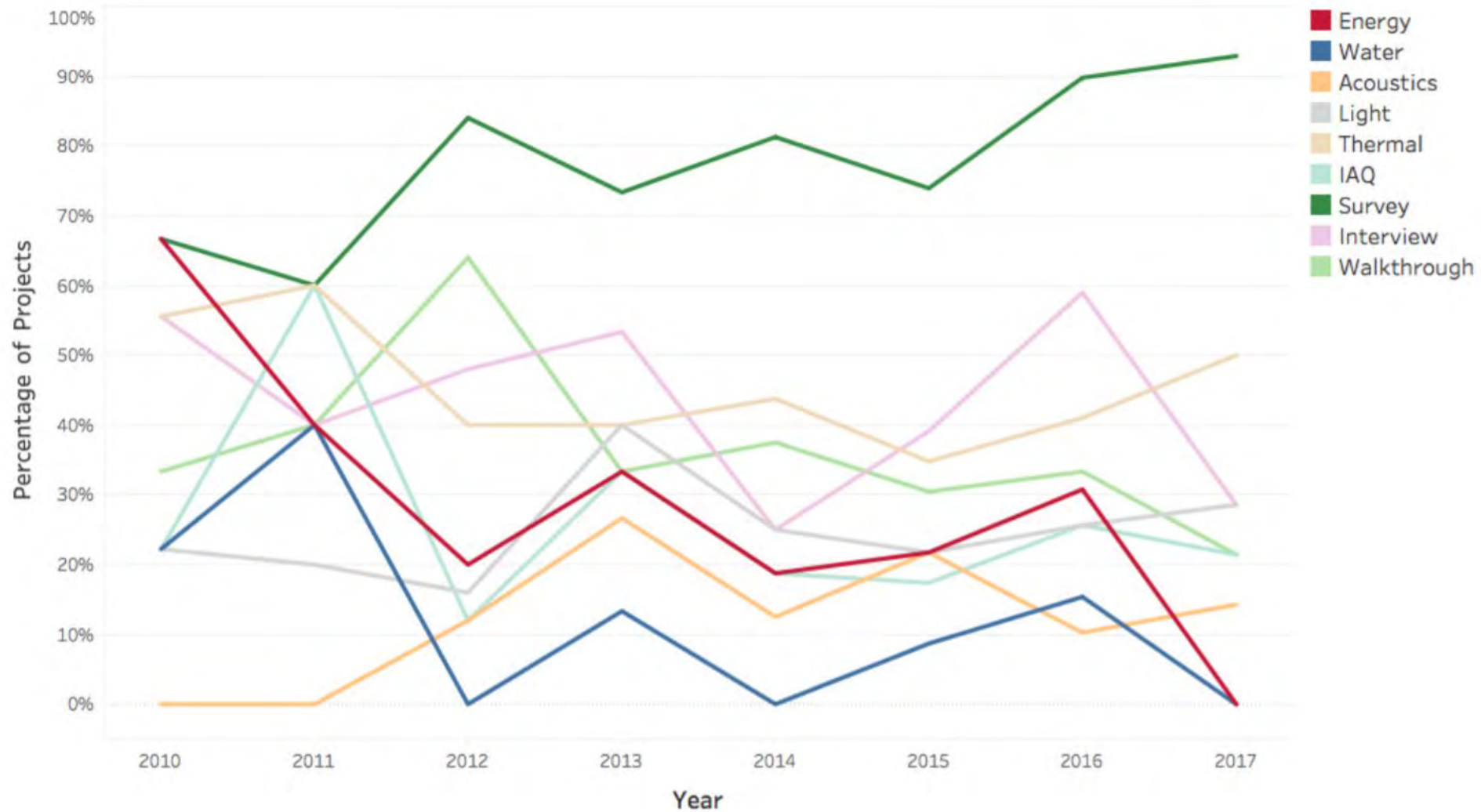
Thank you for your time and cooperation.

Activities Which Took the Most Time for Male and  
Female Respondents at the Airport Restrooms

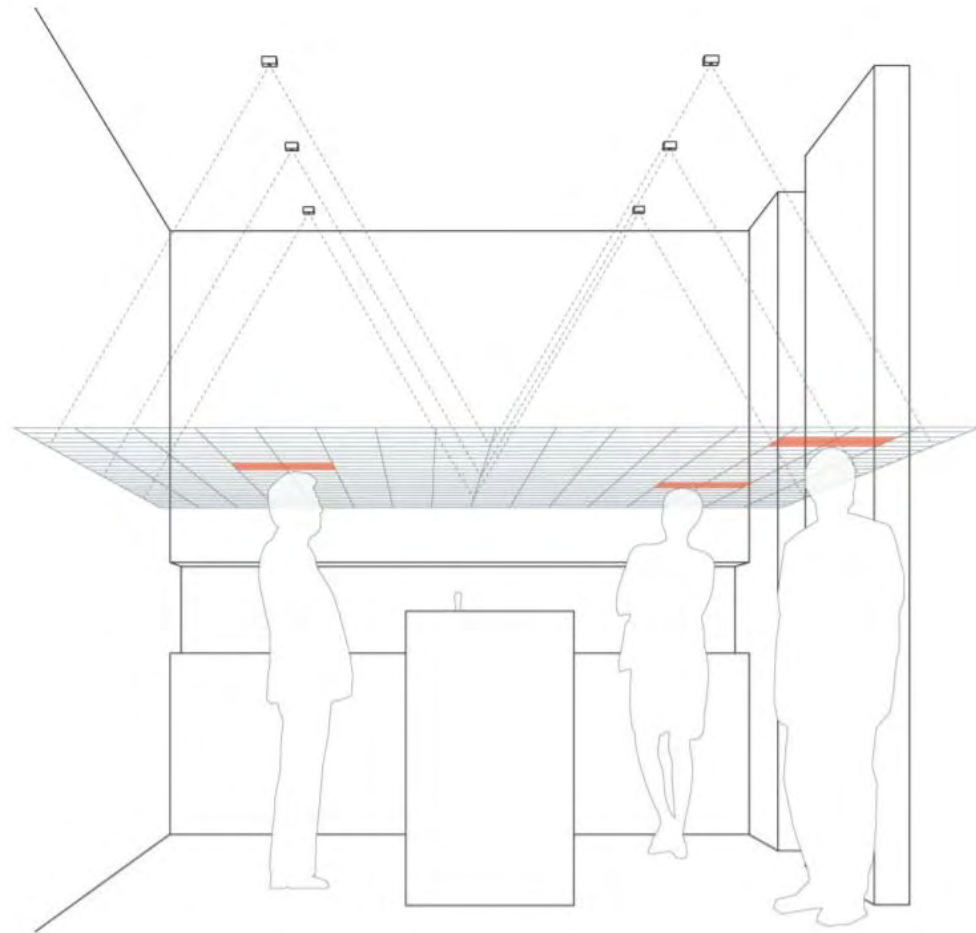
Activities	Males		Females	
	n	%	n	%
	(N = 34)		(N = 40)	
Urinate	19	55.9	16	40.0
Wash hands	4	11.8	6	15.0
Check appearance	3	8.8	2	5.0
Straighten tie	2	5.9	NA	NA
Comb/brush hair	1	2.9	2	5.0
Change clothes	1	2.9	1	2.5
Defecate	1	2.9	0	0.0
Straighten clothes	0	0.0	3	7.5
Wash face	0	0.0	1	2.5
Wait on other person	0	0.0	1	2.5
Assist child/children	0	0.0	1	2.5
Apply make-up	NA	NA	1	2.5
Other	3	8.8	6	15.0
Total	34	100.0	40	100.0

1. Reid, E., and Novak, P.. 1975. "Personal Space: An Unobtrusive Measures Study."  
Bulletin of the Psychonomic Society. 5 (3): 265-266  
***(Undergrad thesis, peering through a crack)***
1. Rawls, S. K. 1988. "Restroom Usage in Selected Public Buildings and Facilities :  
A Comparison of Females and Males."  
<https://vtechworks.lib.vt.edu/handle/10919/53598>.  
***(Inside the bathroom where people could see him counting)***
1. Personal communication from Kira (1994)  
***(Unpublished)***











What degree of fidelity can be reached when using a combination of sensor technology and machine learning for a privacy-preserving data gathering system to measure human behaviour in buildings?

More specifically, how can workplace sanitary facilities usage be monitored using such a system?



What behaviours *can* be  
detected in bathrooms?

More specifically, how can workplace sanitary facilities  
usage be monitored using such a system?



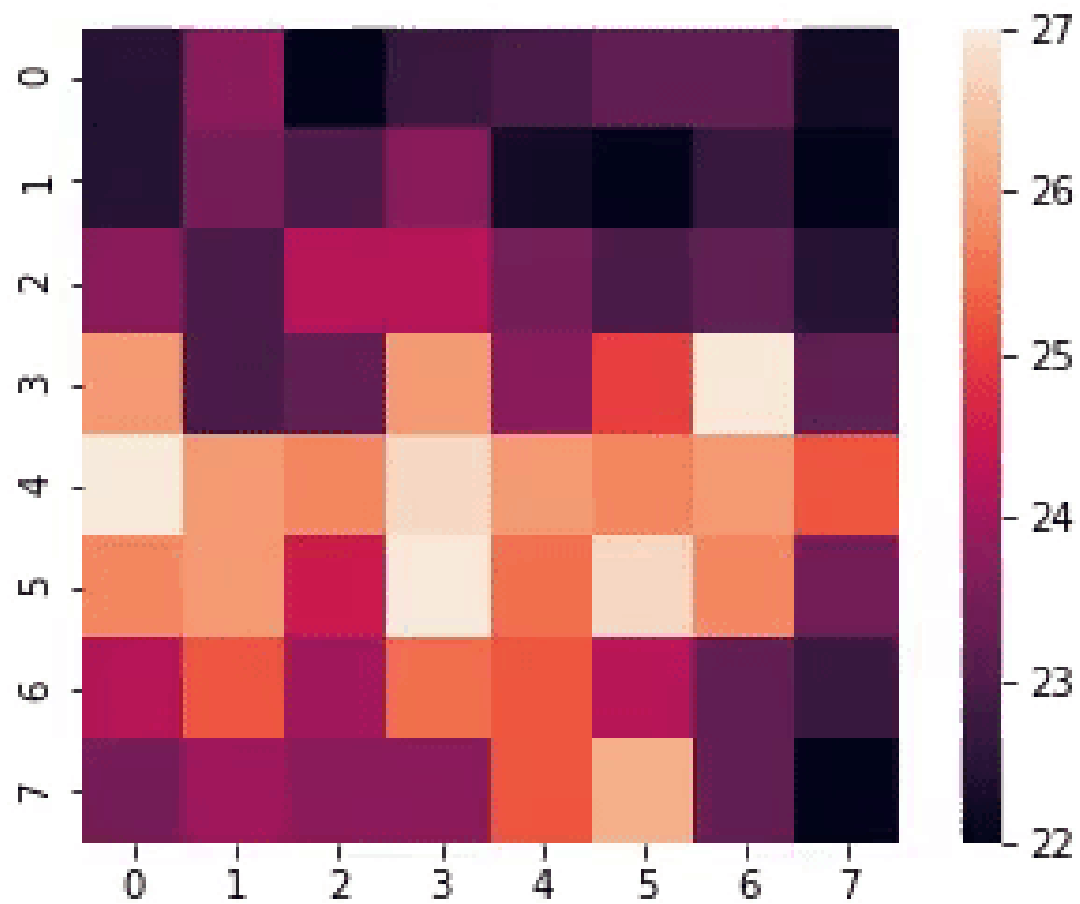
















```
def findCelsius(low, high, address):
    # Return the celsius result from the high and low byte data

    # Find the relevant 12bit signed data
    binaryString = ("0000" + "{0:b}".format(high))[-4:] + (
        "00000000" + "{0:b}".format(low))[-8:]

    #print(("00000000" + "{0:b}".format(low))[-8:] + ("777{0:b}".format(low))[-8:], end="")

    # Determining if the value is negative and setting the signed bit to -1 or 1 respectively
    signedBit = -((int(binaryString[0]) * 2) - 1)

    # Convert the binary string to a number, multiplying by the signed bit
    value = int(binaryString[1:], 2) * signedBit

    # Return the value in celsius (data has a granularity of 0.25 degrees celsius)
    return value / 4

def updateStateFile(text):
    with open(testFileName, 'w+') as f:
        f.write(str(time.time()) + " " + str(text))

def fakeTempsNow():
    return [24, 25, 26, 23.5, 24.25, 29.75, 26, 23.5, 24, 25, 26, 23.5, 24.25, 29.75, 26,

def tempsNow():
    """
    Collect and organise temperature values at a given time.
    Return a list of length 64 of the current temperatures.
    """
    # Initialising the I2C stuff
    bus = smbus.SMBus(1)
    address = 0x69
```

```

32.25,31.5,31.0,26.5,26.0,27.5,28.5,28.5,32.0,
30.75,23.75,22.75,24.5,26.5,31.25,30.5,29.0,25.25,23.
26.75,28.25,31.5,29.25,26.0,26.5,28.75,22.75,29.5,24.
27.25,25.0,30.75,30.75,31.75,28.75,31.0,30.5,31.75,27.
26.75,27.25,29.75,23.25,24.0,22.75,26.0,23.0,25.25,28.
29.75,27.0,26.25,30.75,28.5,26.5,25.75,26.0,24.0,25.7
30.0,25.5,27.75,32.25,27.5,23.25,25.25,23.5,27.5,30.0
30.5,32.5,25.25,28.0,29.75,26.75,24.25,22.5,31.75,25.
22.75,28.5,31.75,23.75,27.5,25.75,27.0,22.75,32.25,26.
31.25,22.5,24.25,26.5,30.0,28.75,23.75,25.25,24.75,27.
25.0,26.0,22.5,23.25,32.25,31.75,26.5,23.0,30.5,24.0,
28.0,28.0,28.5,23.75,26.75,31.75,25.25,28.25,25.75,32.
30.5,31.5,27.5,23.5,27.0,32.25,27.25,23.0,24.25,29.0,
26.75,30.5,22.5,32.0,30.75,28.5,31.75,31.0,32.5,30.0,
25.5,31.5,32.0,32.25,28.5,28.5,30.75,23.75,29.25,31.0
28.5,32.25,22.5,29.25,31.0,28.25,28.25,22.5,24.0,26.7
24.75,28.0,30.25,30.25,27.25,31.5,29.25,26.0,28.0,26.0
30.0,28.25,28.25,30.75,31.75,29.0,26.75,28.25,22.5,23.
27.25,30.5,29.5,29.5,27.0,27.25,22.5,23.25,23.75,30.7
32.0,28.75,32.5,26.5,30.0,30.25,29.0,29.5,32.5,24.0,2
28.25,29.25,23.25,26.0,24.0,26.25,23.5,24.25,31.5,32.0
24.5,30.5,23.0,30.75,31.25,30.0,26.0,32.0,29.5,32.5,2
24.5,25.25,32.5,32.25,22.5,22.5,29.0,24.75,30.5,24.25
30.5,24.5,31.0,25.75,29.25,29.5,22.5,25.5,32.0,26.25,
24.5,28.25,23.0,28.75,31.75,30.0,28.25,23.25,24.5,27.
31.5,24.0,24.25,28.25,23.25,29.0,28.5,25.5,31.0,22.5,
30.25,23.25,28.0,27.5,32.0,29.0,28.5,32.5,31.25,31.5,
27.75,23.5,24.75,28.75,25.75,22.5,31.0,28.75,30.75,32.
28.0,32.5,31.25,24.75,32.0,25.0,27.25,30.5,23.5,27.25
26.25,31.75,32.5,25.0,26.0,23.25,29.0,29.5,27.75,26.2
26.25,23.5,31.25,23.75,28.0,30.0,24.25,24.25,25.28.
23.75,32.0,26.5,25.75,30.0,28.75,25.25,28.0,29.0,23.7
32.0,24.75,27.25,28.75,25.25,26.25,26.25,24.25,24.5,2
28.25,32.5,25.25,27.75,25.25,22.75,31.25,31.0,29.25,2
29.25,25.25,26.25,23.25,29.75,24.5,25.75,29.25,30.75,
31.5,23.5,27.5,31.75,27.5,29.75,30.5,31.75,22.75,24.5

```

```

        frameList.extend(subList)

# Turning the items into strings
frameList = [str(x) for x in frameList]

# Returning everything, the data as a CSV string
return frameData["state"], frameList

def changeDomain(value, startDomain, endDomain):
    """Change a value from its position in an initial domain to its position relative

    value should be a float
    startDomain should be a [float, float]
    endDomain should be a [float, float]
    """
    # Getting a percentage of where it is relative to the first domain
    valuePct = (value - startDomain[0]) / (startDomain[1] - startDomain[0])

    # Scaling up to the end domain, and adding the shift
    valueFinal = valuePct * (endDomain[1] - endDomain[0]) + endDomain[0]

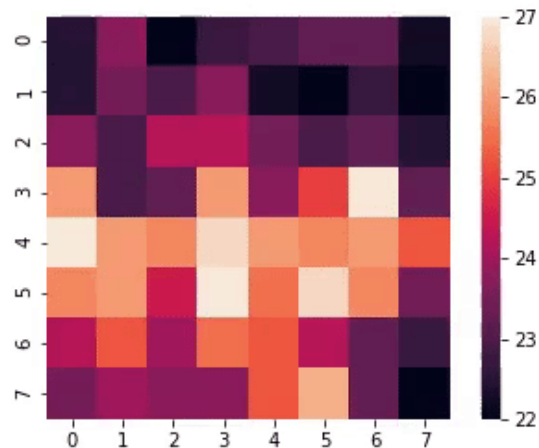
    return valueFinal

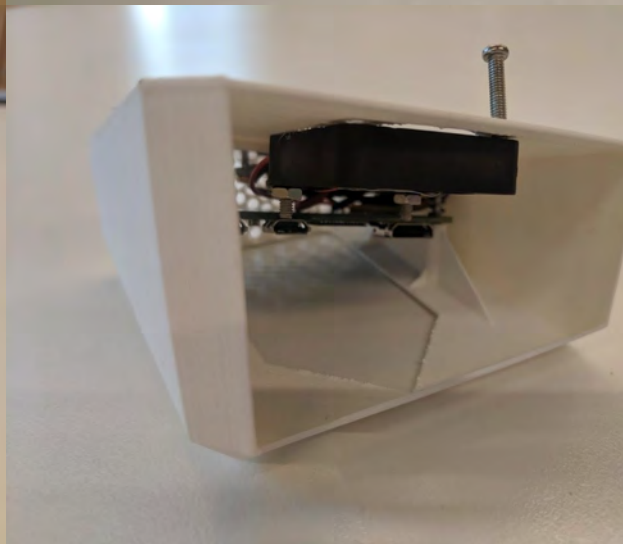
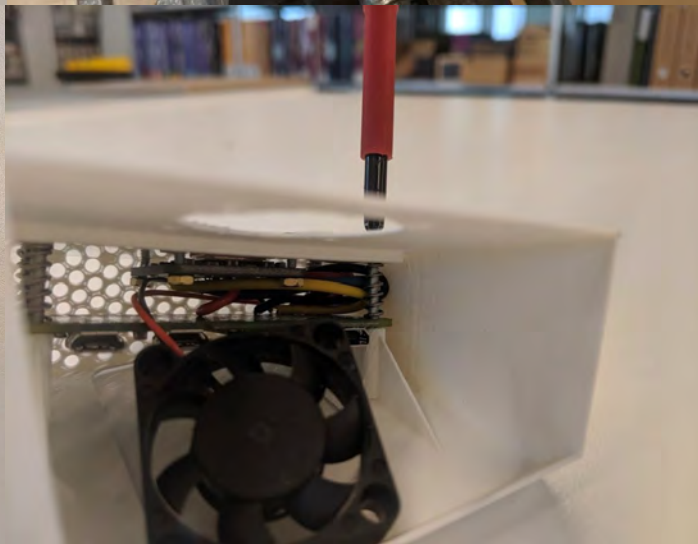
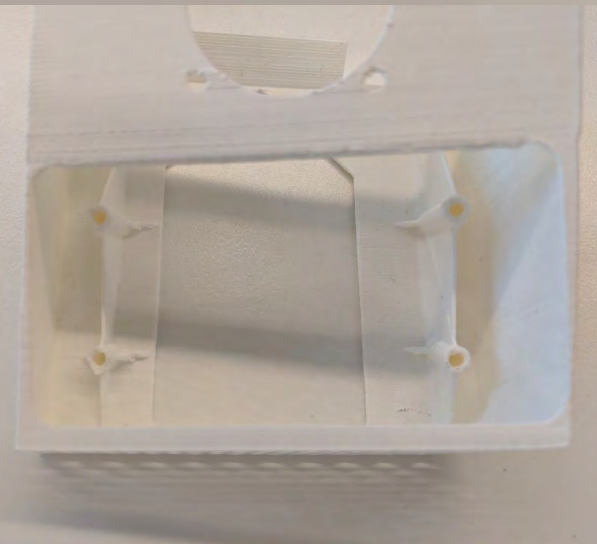
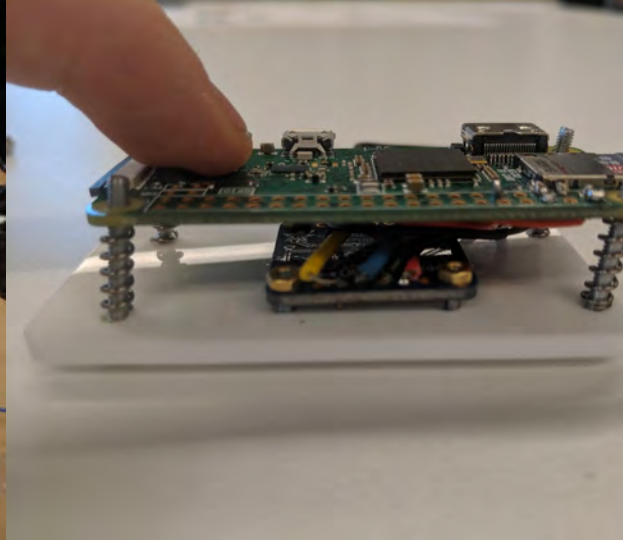
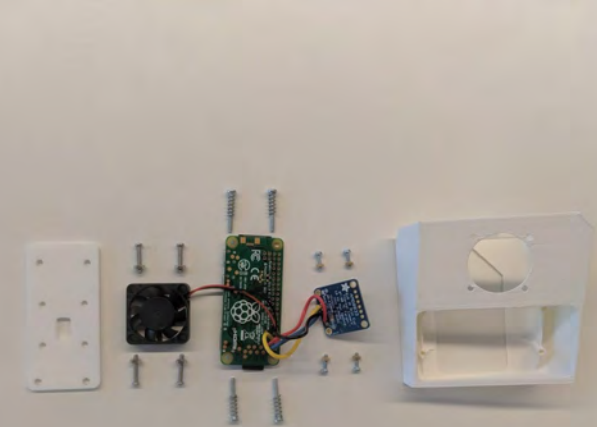
def tempToHexColor(temp):
    """Turn a float temp to a str hex rgb color that represents its temperature

    Input temps will likely be in the range 15-40 (that's being generous)
    Some example outputs of the same format: 'rgb(255, 70, 0)', 'rgb(11, 172, 230)', '
    """
    # Setting the (somewhat arbitrary) temperature range
    tempRange = [15, 40]

    # Keeping the temp within the acceptable range
    if temp < tempRange[0]:
        temp = tempRange[0]

```





Electronics get hot







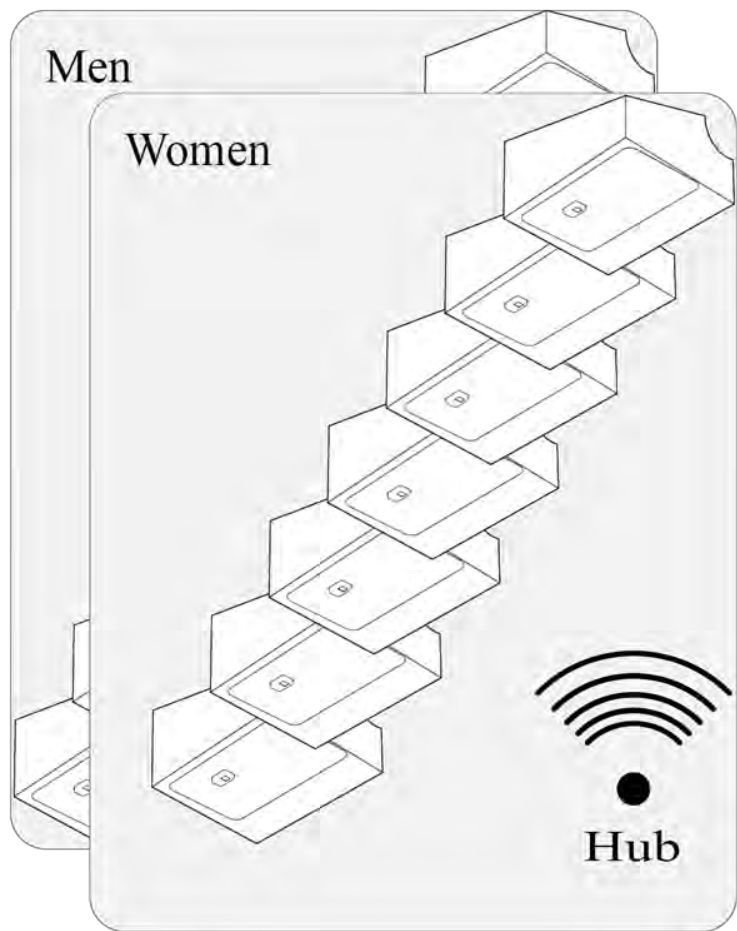
Screens produce heat



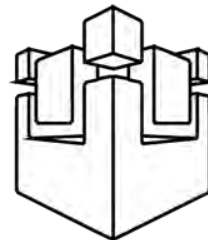


A photograph of a hallway. On the left is a dark, textured wall. In the center is a white wall. On the right is a dark door with a vertical window and a metal handle. The text "Walls affect networking" is overlaid in white on the white wall.

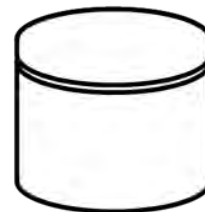
Walls affect  
networking



Organisation  
WiFi



AWS IoT

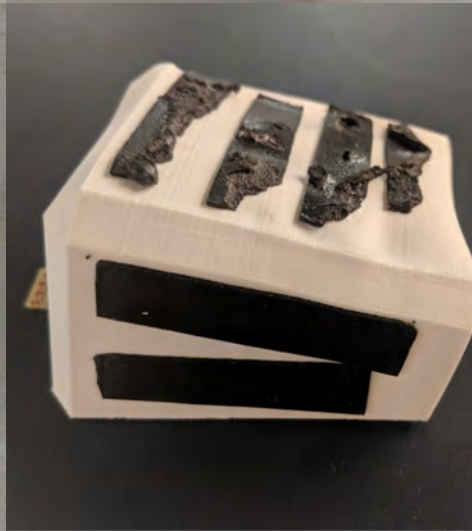
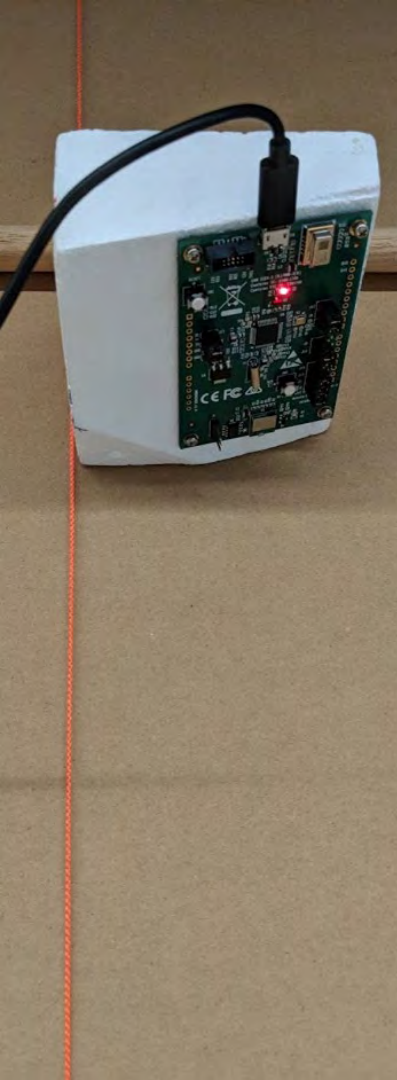


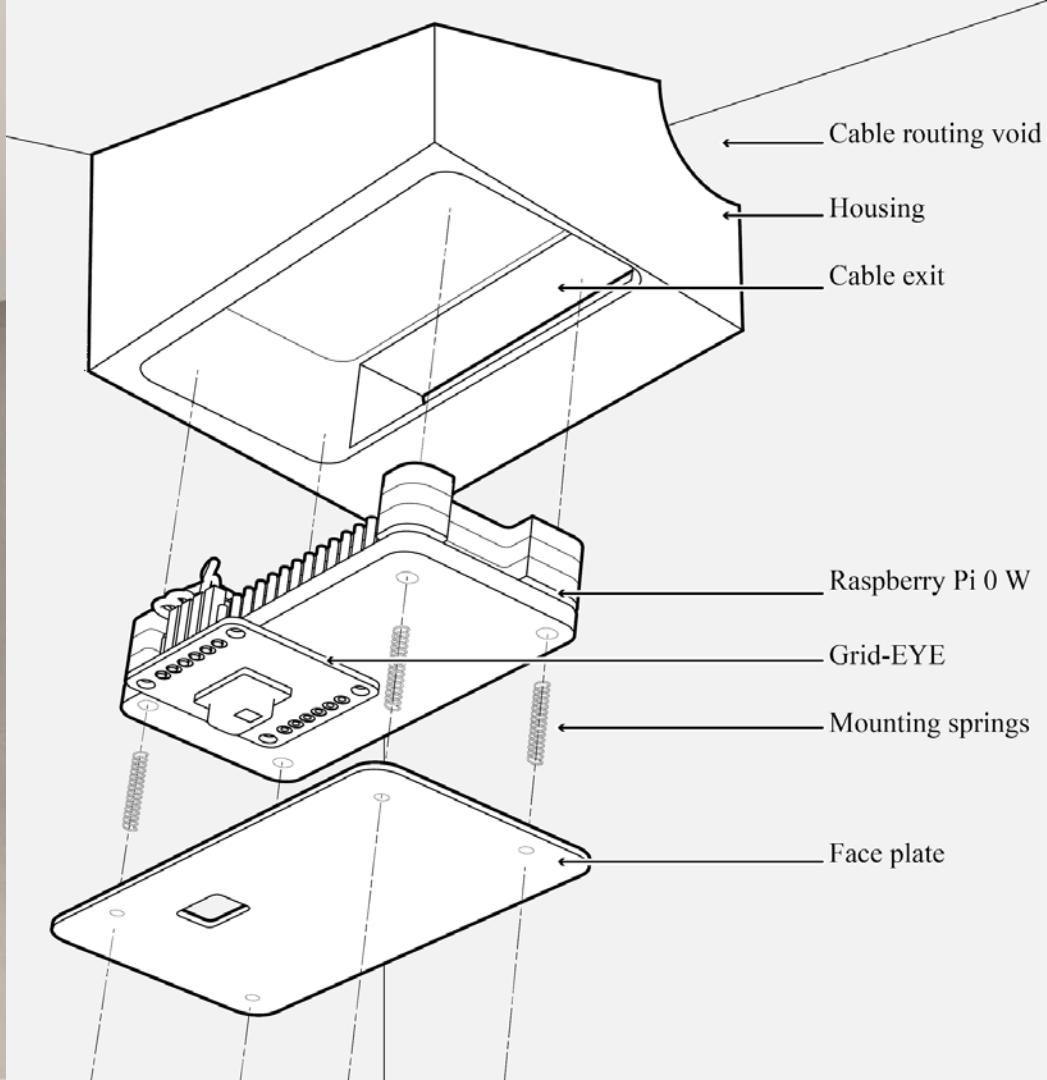
Database

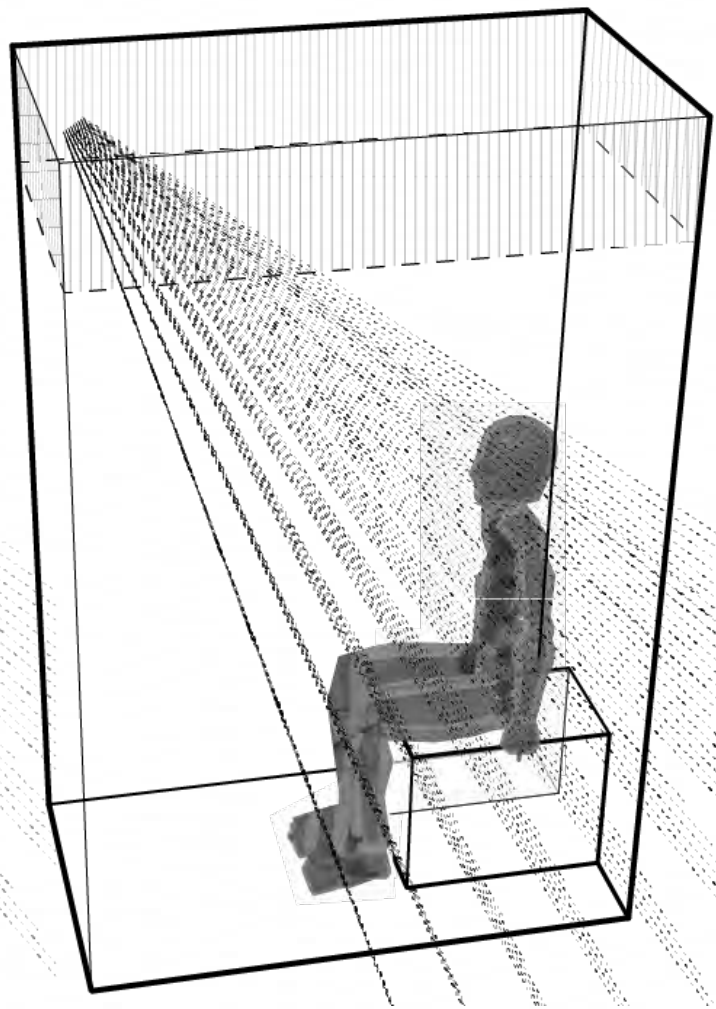
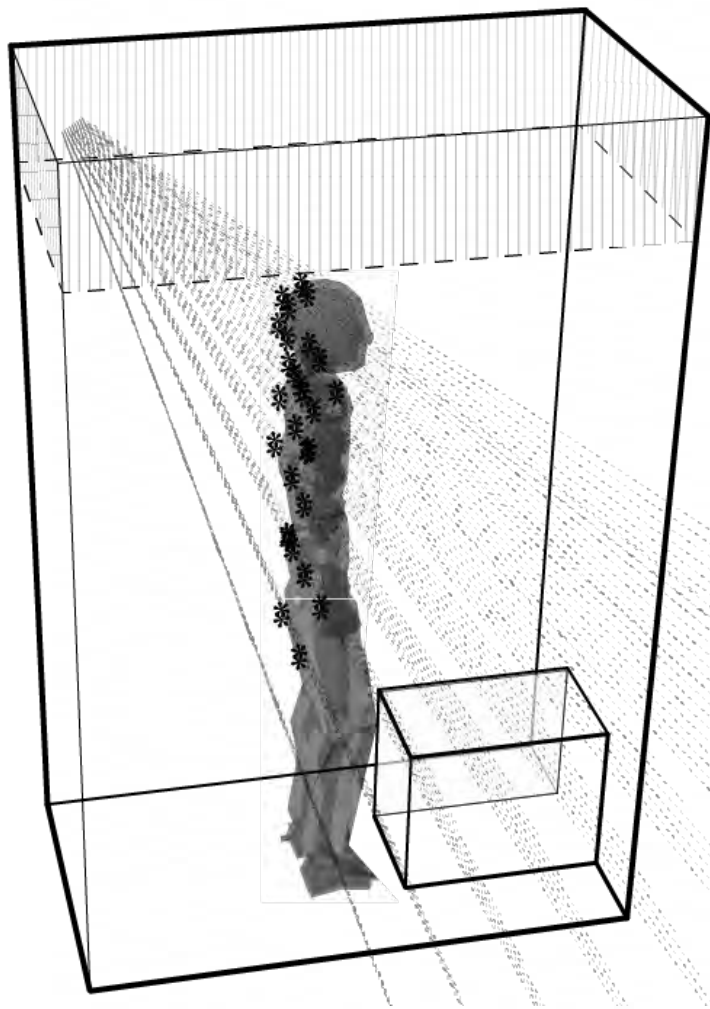


Analysis

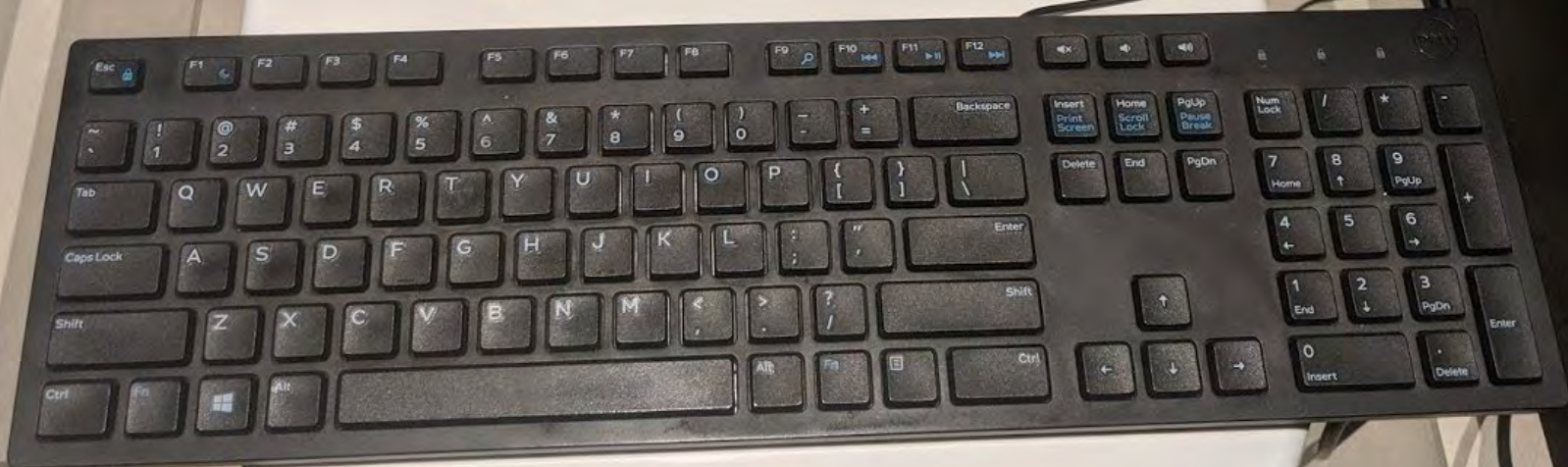
















# Behaviours:

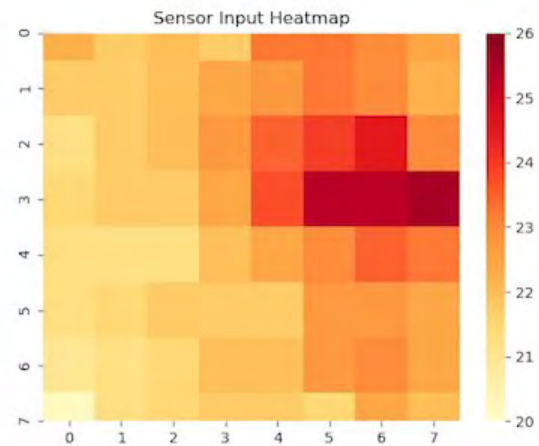
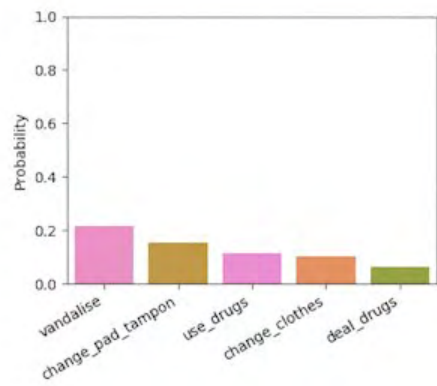
["sitting", "browsing\_phone", "take\_phone\_call", "standing",  
"straighten\_clothes", "comb\_brush\_hair", "leave"]

```
Epoch 19/40
91/91 - 28s - loss: 0.0141 - acc: 0.9639 - val_loss: 0.0130 - val_acc: 0.9609
Epoch 20/40
91/91 - 28s - loss: 0.0120 - acc: 0.9705 - val_loss: 0.0162 - val_acc: 0.9492
Epoch 21/40
91/91 - 29s - loss: 0.0105 - acc: 0.9753 - val_loss: 0.0131 - val_acc: 0.9570
Epoch 22/40
91/91 - 26s - loss: 0.0098 - acc: 0.9773 - val_loss: 0.0109 - val_acc: 0.9688
Epoch 23/40
91/91 - 27s - loss: 0.0089 - acc: 0.9787 - val_loss: 0.0099 - val_acc: 0.9688
Epoch 24/40
91/91 - 29s - loss: 0.0070 - acc: 0.9863 - val_loss: 0.0095 - val_acc: 0.9766
Epoch 25/40
91/91 - 30s - loss: 0.0065 - acc: 0.9894 - val_loss: 0.0073 - val_acc: 0.9844
Epoch 26/40
91/91 - 29s - loss: 0.0061 - acc: 0.9890 - val_loss: 0.0149 - val_acc: 0.9629
Epoch 27/40
91/91 - 29s - loss: 0.0053 - acc: 0.9924 - val_loss: 0.0061 - val_acc: 0.9844
Epoch 28/40
91/91 - 25s - loss: 0.0048 - acc: 0.9924 - val_loss: 0.0083 - val_acc: 0.9824
Epoch 29/40
91/91 - 30s - loss: 0.0040 - acc: 0.9952 - val_loss: 0.0059 - val_acc: 0.9941
Epoch 30/40
91/91 - 29s - loss: 0.0033 - acc: 0.9962 - val_loss: 0.0044 - val_acc: 0.9941
Epoch 31/40
91/91 - 27s - loss: 0.0027 - acc: 0.9983 - val_loss: 0.0041 - val_acc: 0.9980
Epoch 32/40
91/91 - 27s - loss: 0.0023 - acc: 0.9990 - val_loss: 0.0037 - val_acc: 0.9961
Epoch 33/40
91/91 - 31s - loss: 0.0019 - acc: 0.9993 - val_loss: 0.0038 - val_acc: 0.9961
Epoch 34/40
91/91 - 28s - loss: 0.0019 - acc: 0.9986 - val_loss: 0.0038 - val_acc: 0.9941
Epoch 35/40
91/91 - 29s - loss: 0.0015 - acc: 0.9993 - val_loss: 0.0032 - val_acc: 0.9961
Epoch 36/40
91/91 - 24s - loss: 0.0014 - acc: 0.9993 - val_loss: 0.0030 - val_acc: 0.9961
Epoch 37/40
91/91 - 26s - loss: 0.0012 - acc: 0.9993 - val_loss: 0.0028 - val_acc: 0.9980
Epoch 38/40
91/91 - 26s - loss: 0.0013 - acc: 0.9993 - val_loss: 0.0029 - val_acc: 0.9961
Epoch 39/40
91/91 - 26s - loss: 0.0011 - acc: 0.9993 - val_loss: 0.0028 - val_acc: 0.9980
Epoch 40/40
91/91 - 29s - loss: 0.0017 - acc: 0.9990 - val_loss: 0.0030 - val_acc: 0.9980
```

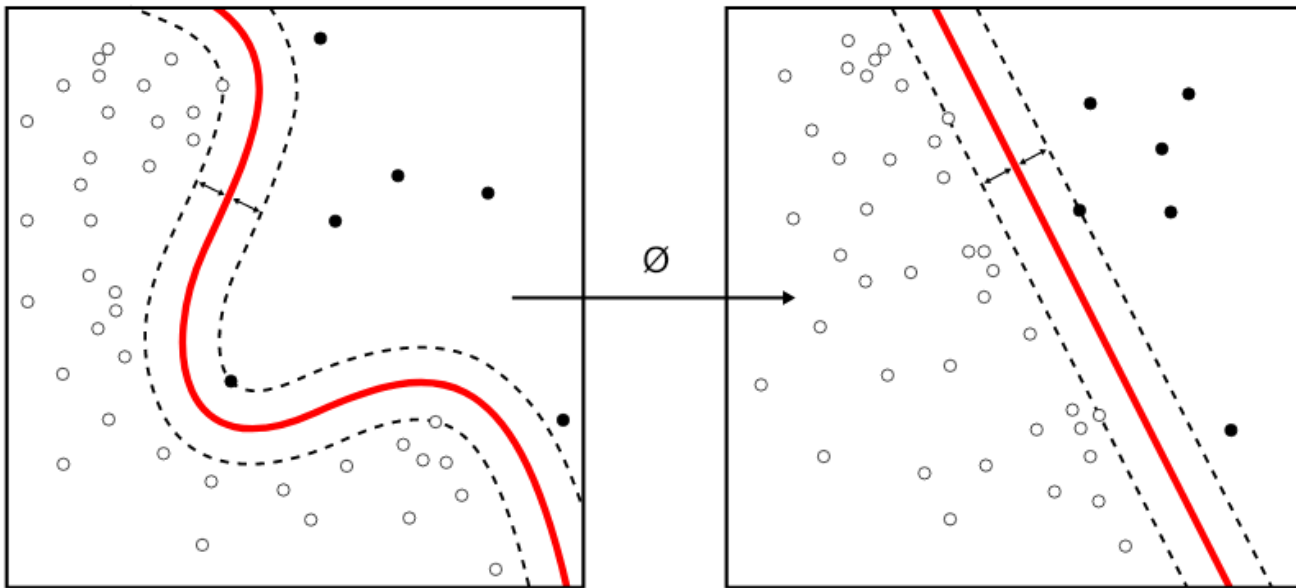
99.8%!

# Behaviours:

['adjust\_jewelry\_scarf', 'breast\_feeding', 'change\_clothes',  
'change\_diaper', 'change\_pad\_tampon', 'clean\_glasses',  
'cover\_seat\_with\_toilet\_paper', 'deal\_drugs', 'defecate',  
'drink\_alcohol', 'eat\_food', 'exercise', 'have\_solace', 'hide', 'nap',  
'put\_in\_take\_out\_contacts', 'read', 'smoke', 'spy',  
'squat\_on\_toilet', 'take\_medicine', 'take\_phone\_call', 'talk',  
'urinate', 'use\_drugs', 'vandalise', 'write\_notes']

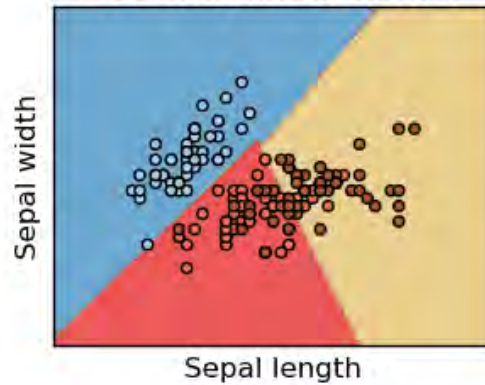


# SVM (Support Vector Machine)

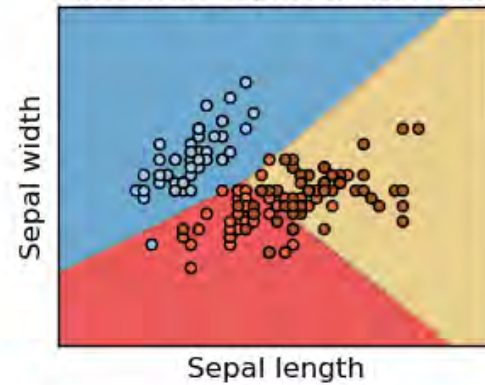




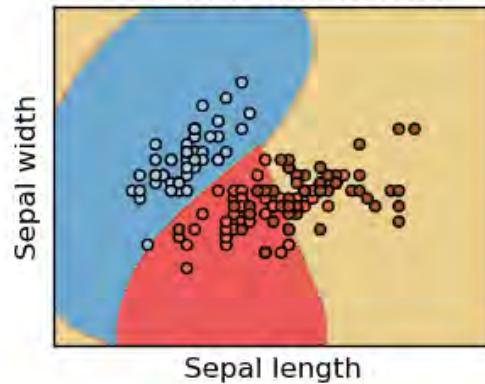
SVC with linear kernel



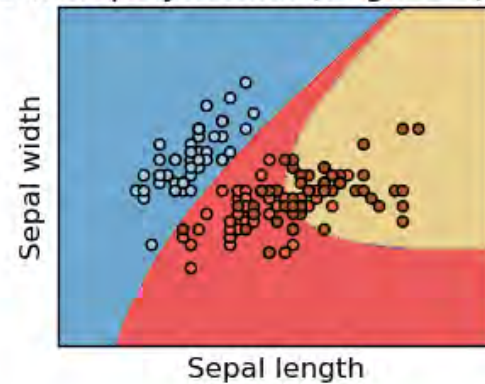
LinearSVC (linear kernel)



SVC with RBF kernel



SVC with polynomial (degree 3) kernel



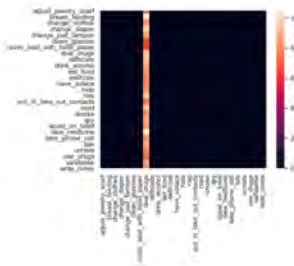
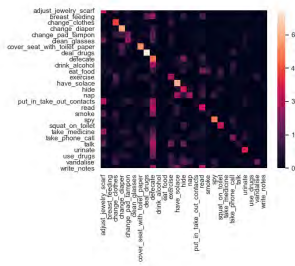
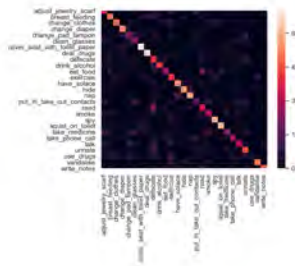
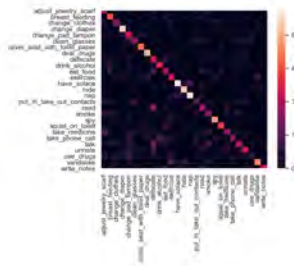
K: Linear

K: Poly

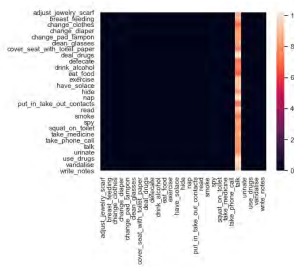
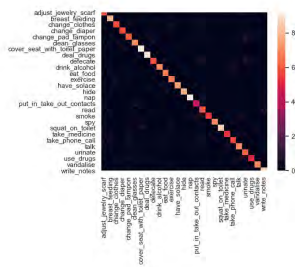
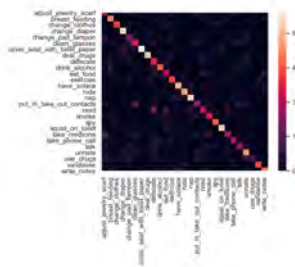
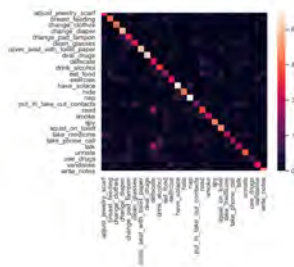
K: RBF

K: Sigmoid

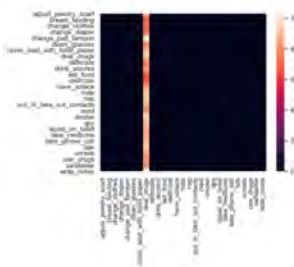
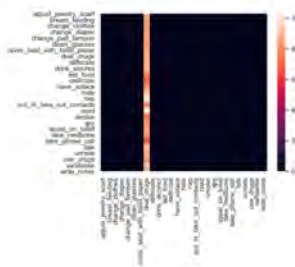
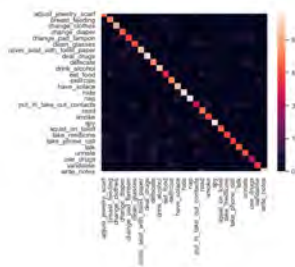
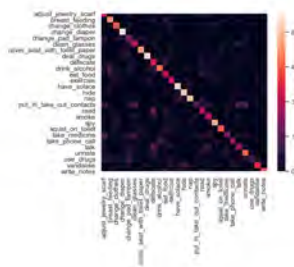
$\gamma$ : Low



$\gamma$ : Medium



$\gamma$ : High

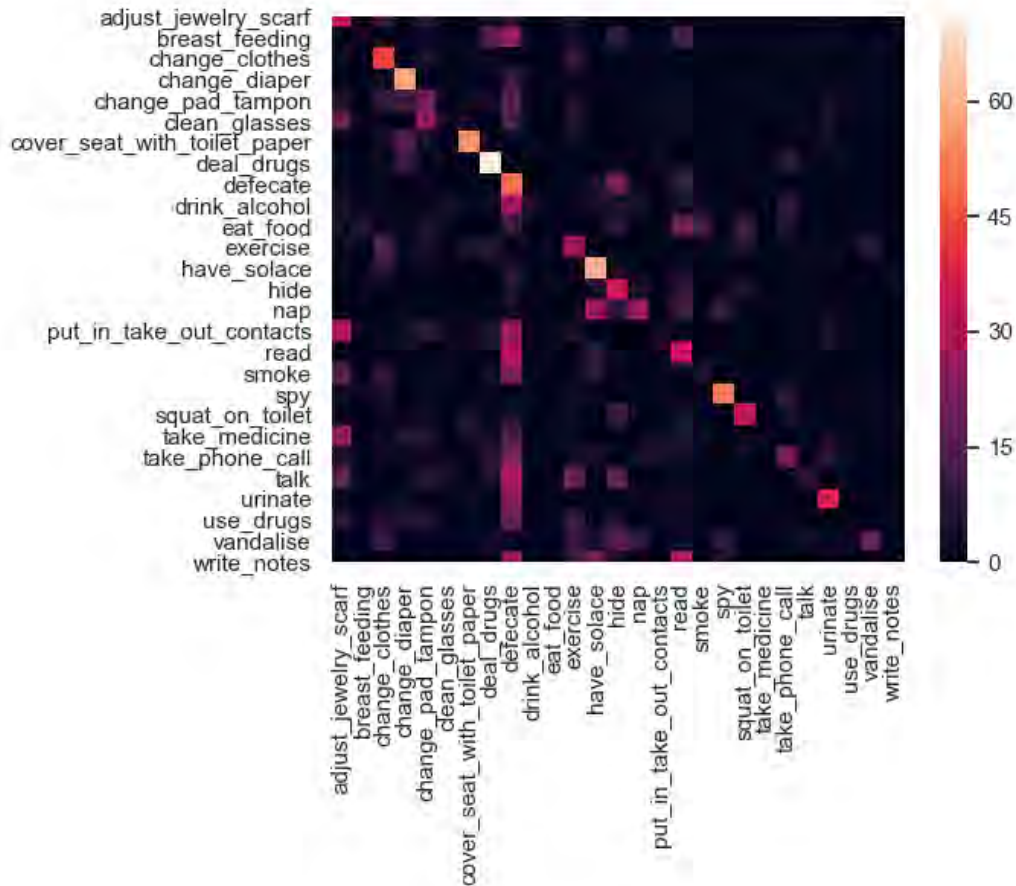


# Confusion Matrix

**Model:** SVM (Support Vector Machine)

**Kernel:** RBF (Radial Basis Function)

**Gamma:** 0.001

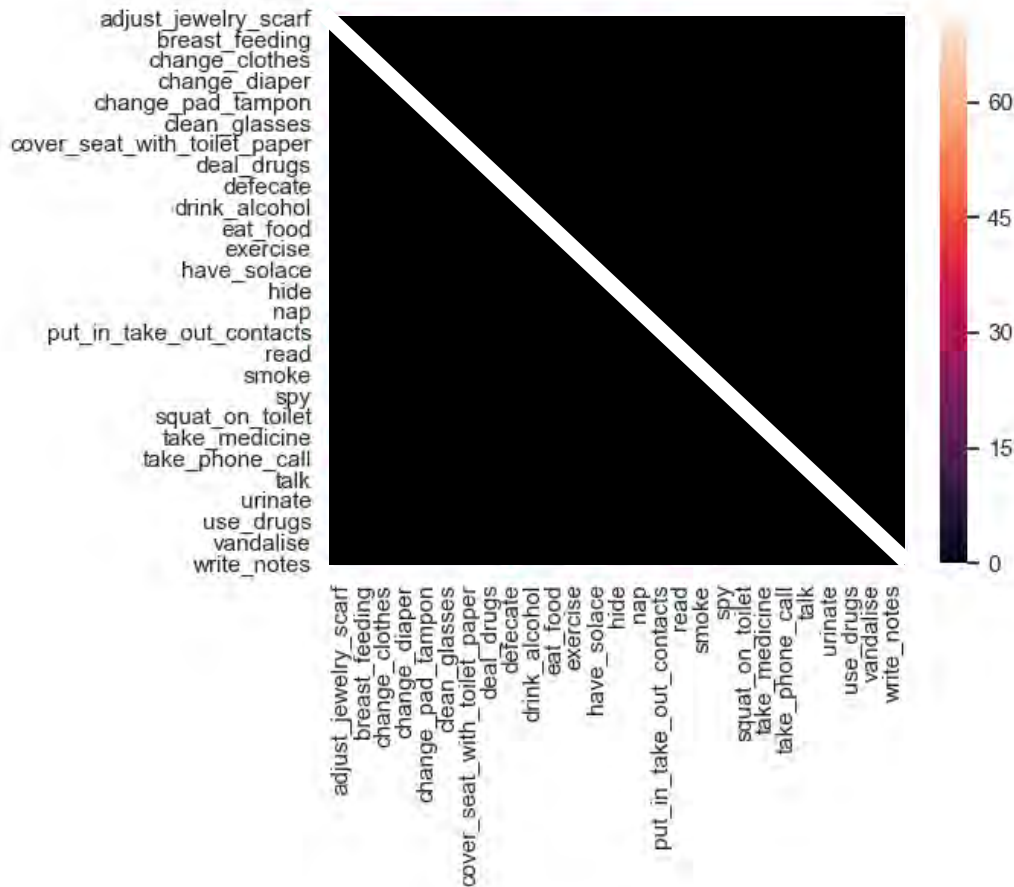


# Confusion Matrix

**Model:** SVM (Support Vector Machine)

**Kernel:** RBF (Radial Basis Function)

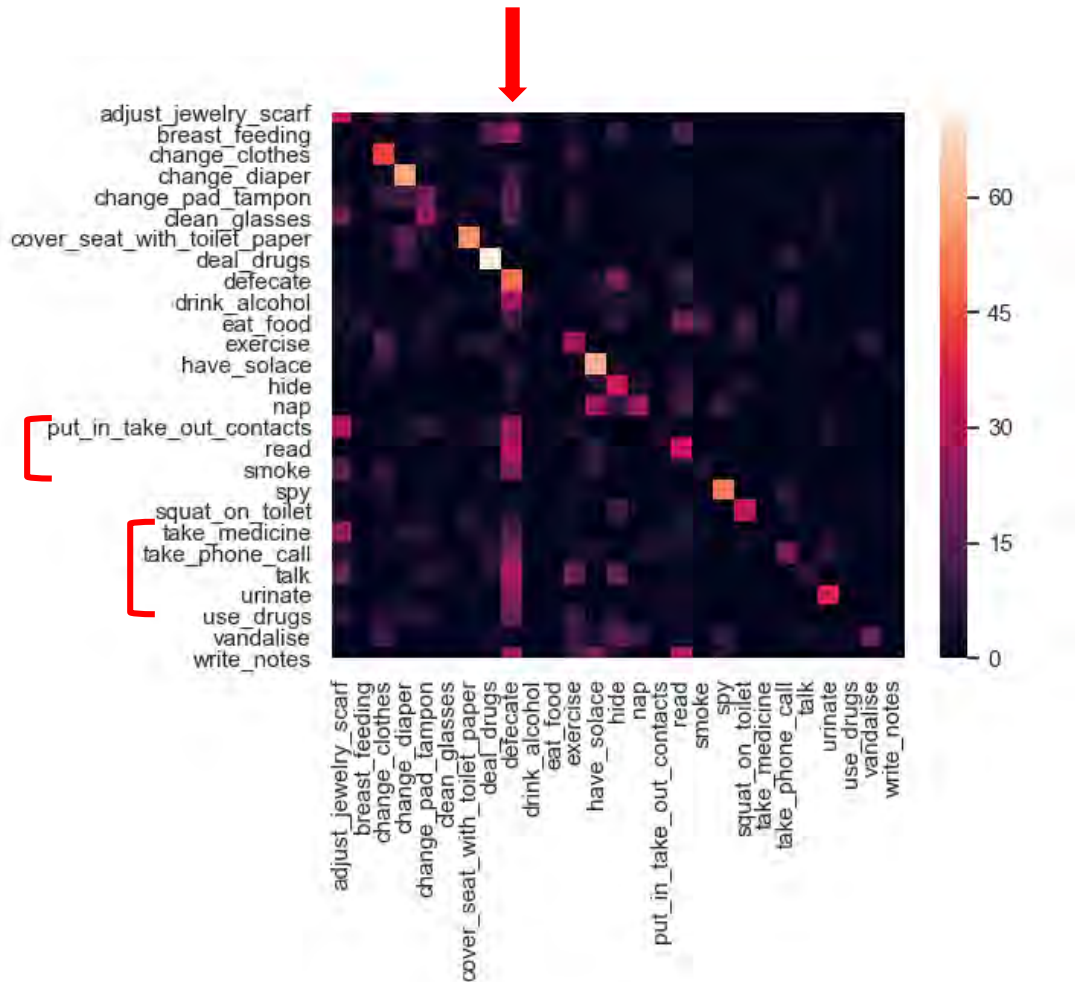
**Gamma:** 0.001



**Model:** SVM (Support Vector Machine)  
**Kernel:** RBF (Radial Basis Function)  
**Gamma:** 0.001

**Kernel:** RBF (Radial Basis Function)

**Gamma:** 0.001

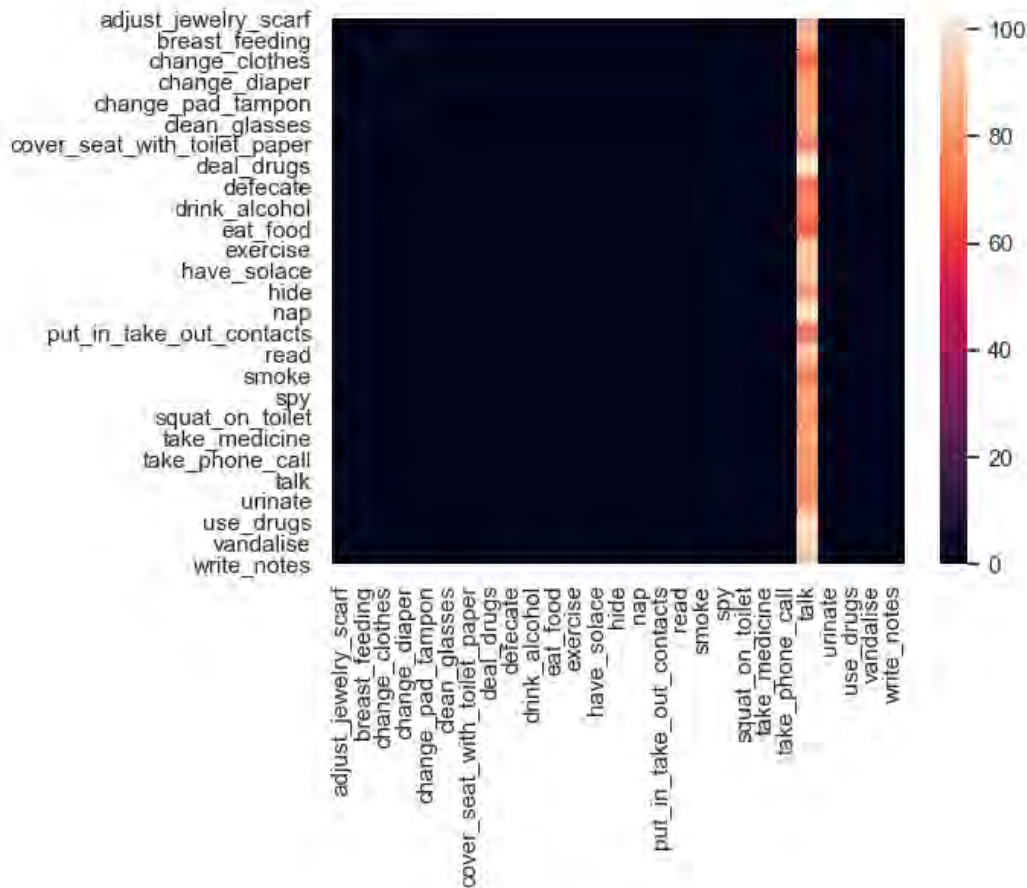




**Model:** SVM (Support Vector Machine)  
**Kernel:** Sigmoid  
**Gamma:** 0.1

**Kernel:** Sigmoid

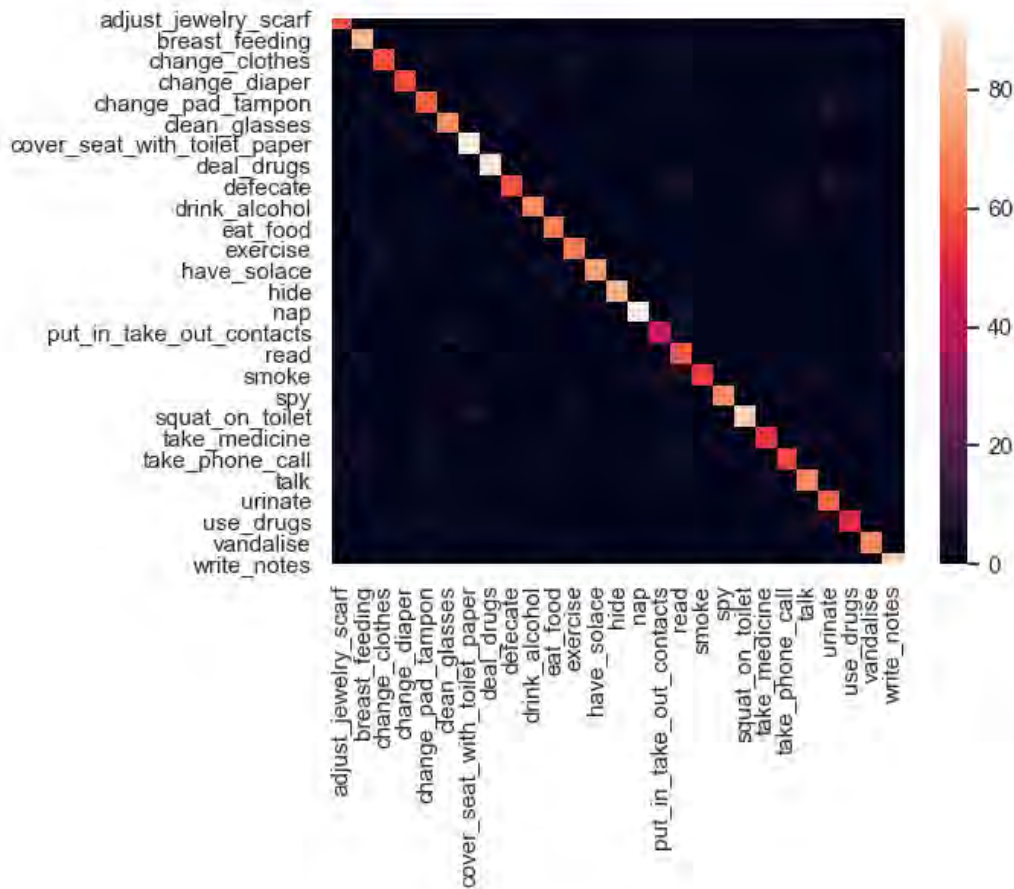
**Gamma:** 0.1



**Model:** SVM (Support Vector Machine)  
**Kernel:** RBF (Radial Basis Function)  
**Gamma:** 0.1

**Kernel:** RBF (Radial Basis Function)

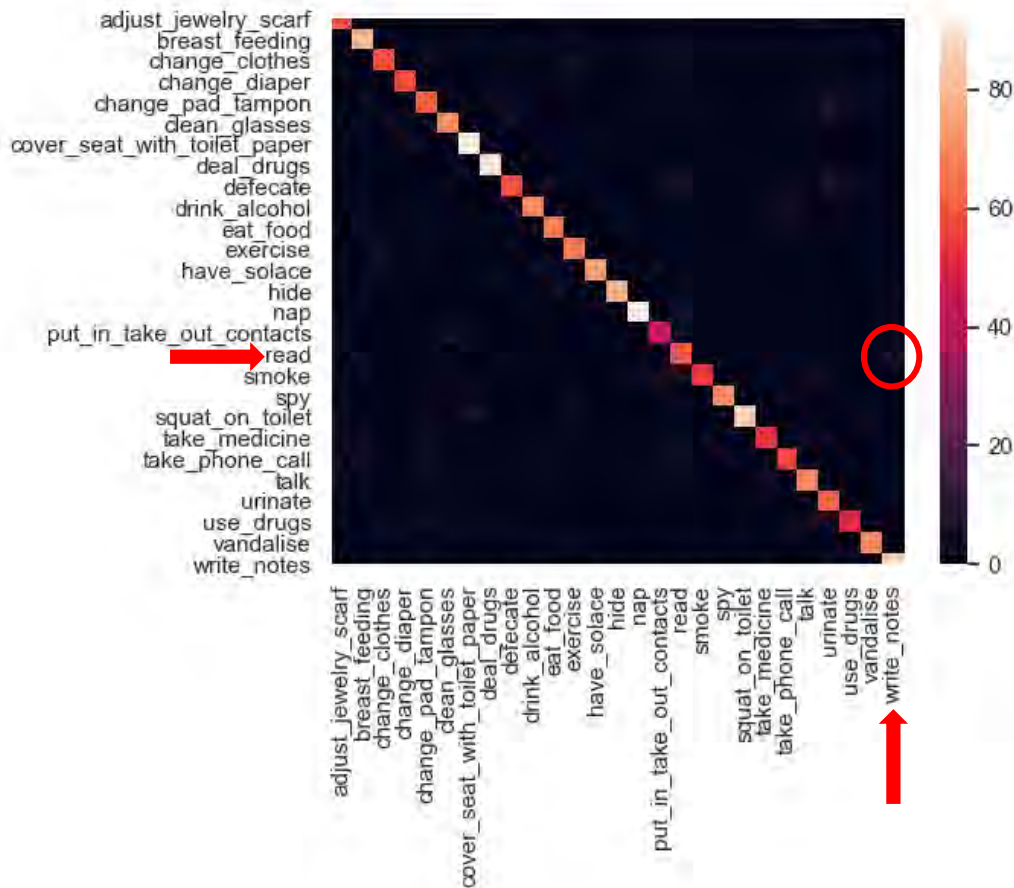
**Gamma:** 0.1



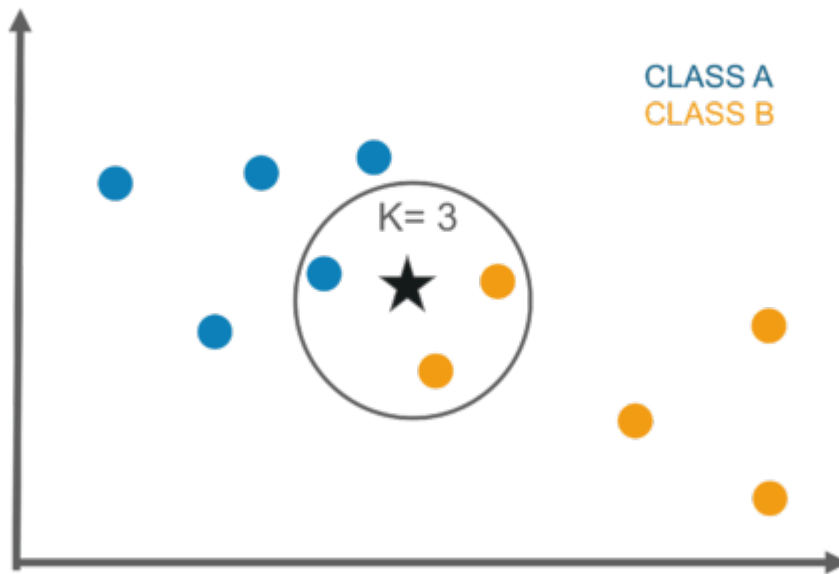
**Model:** SVM (Support Vector Machine)  
**Kernel:** RBF (Radial Basis Function)  
**Gamma:** 0.1

**Kernel:** RBF (Radial Basis Function)

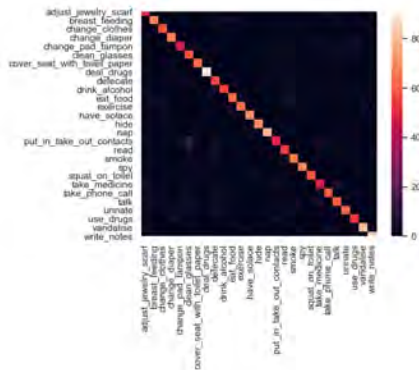
**Gamma:** 0.1



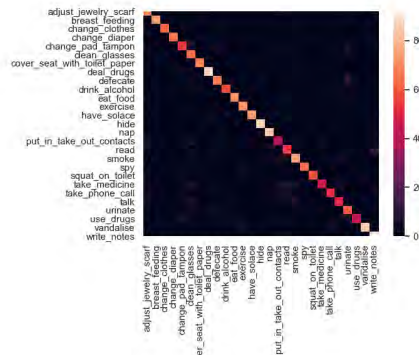
# KNN (K-Nearest Neighbours)



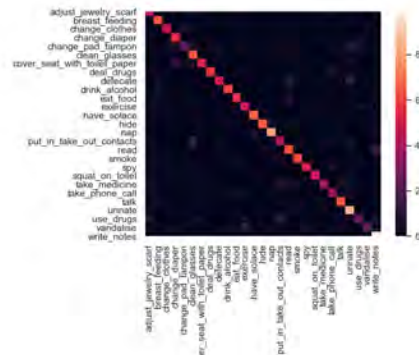
$K = 1$



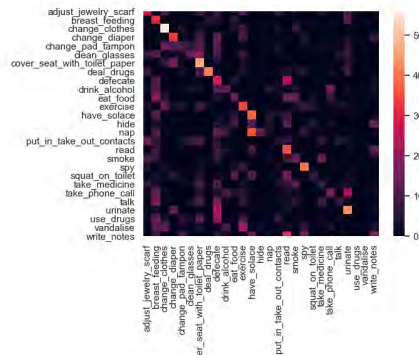
$K = 10$



$K = 100$



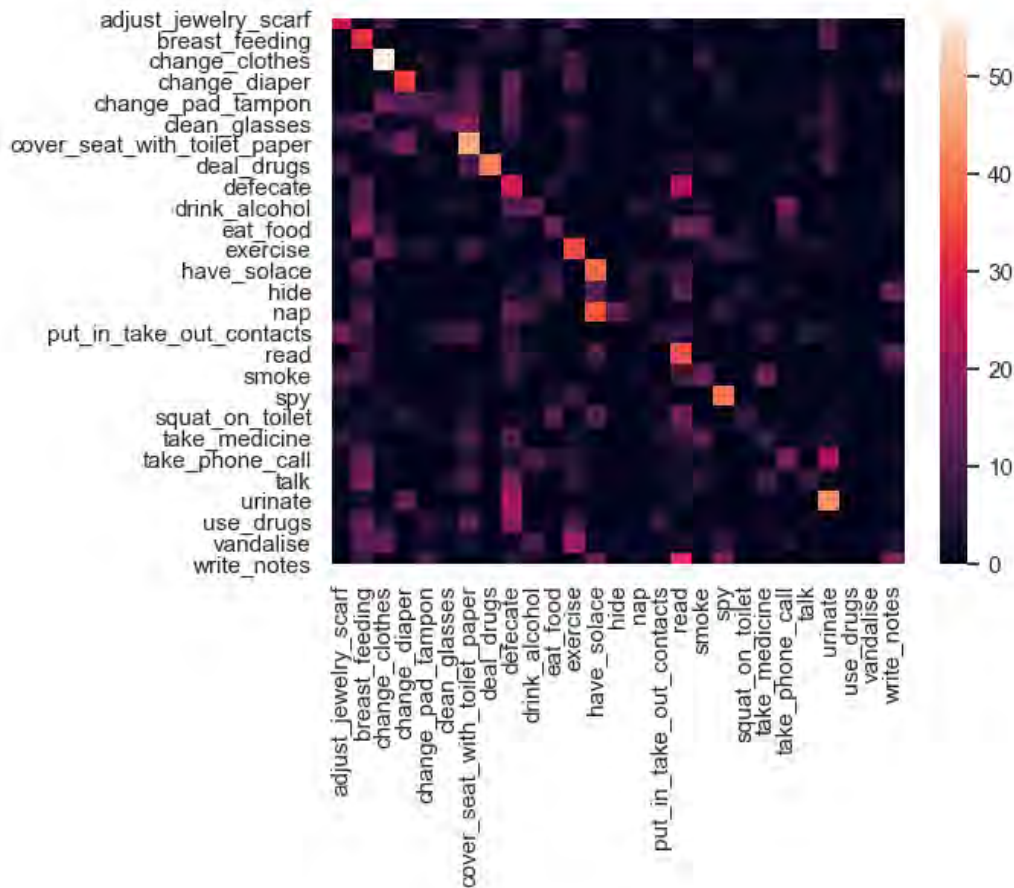
$K = 1000$





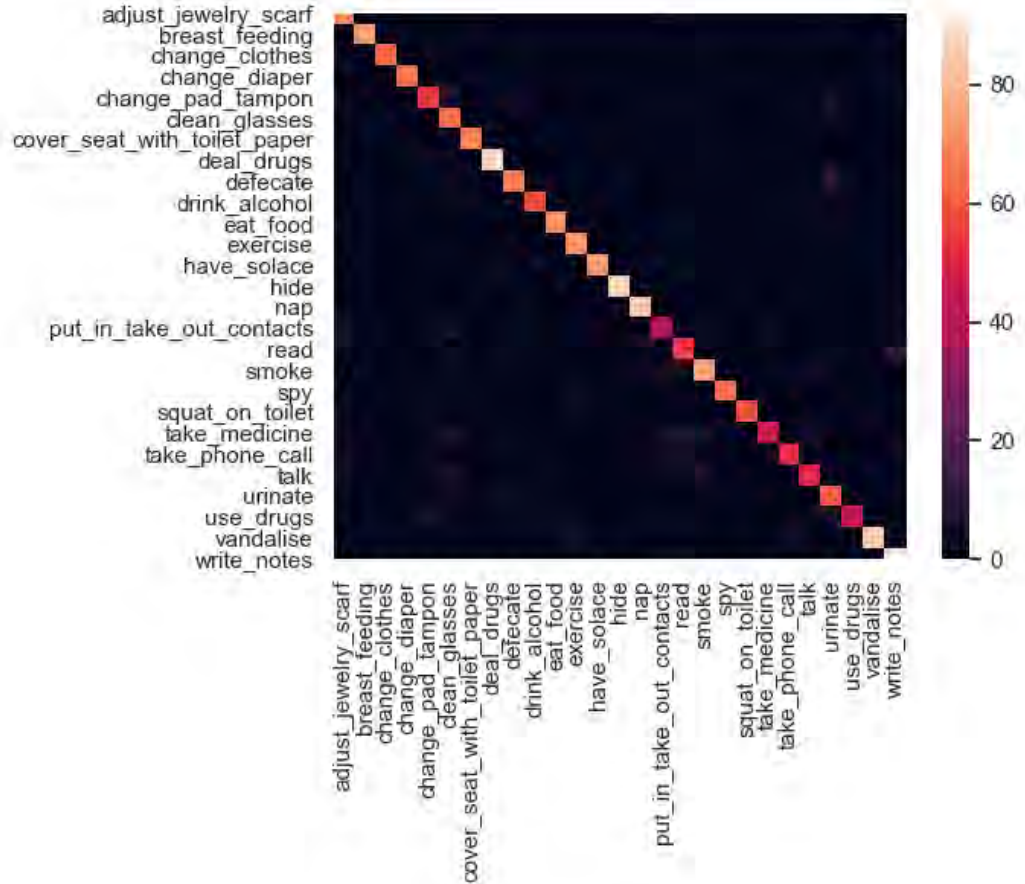
**Model:** KNN (K-Nearest Neighbours)  
**K:** 1000

**K:** 1000

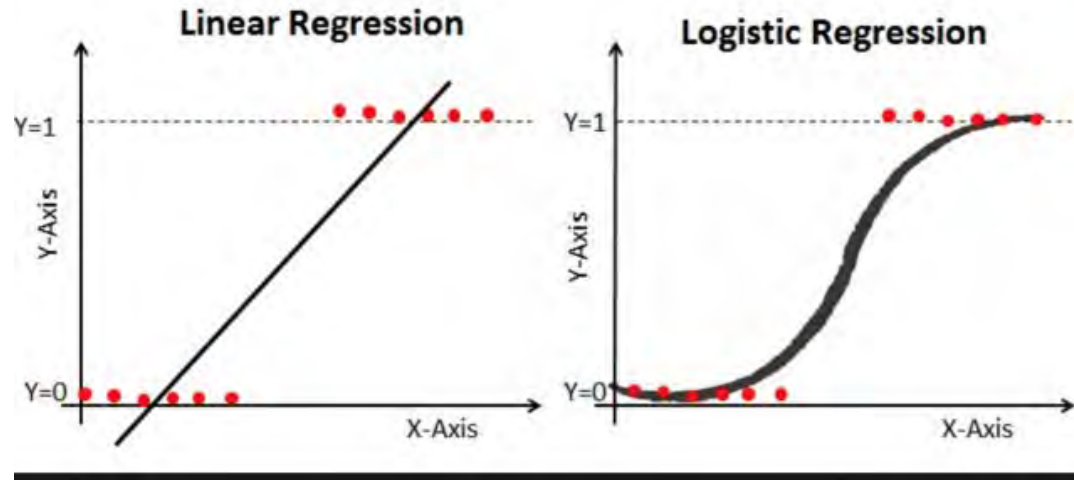


# Confusion Matrix

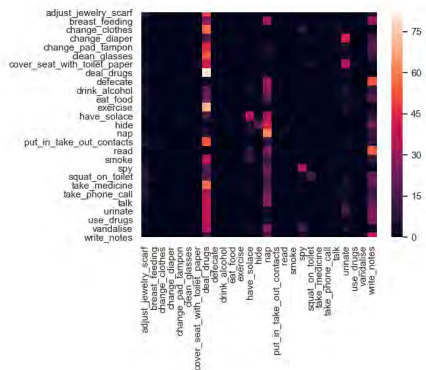
**Model:** KNN (K-Nearest Neighbours)  
**K:** 10



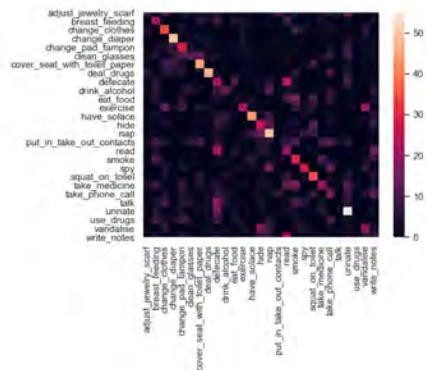
# Logistic Regression



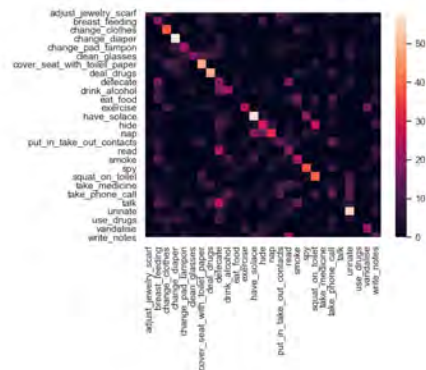
$K = 1$



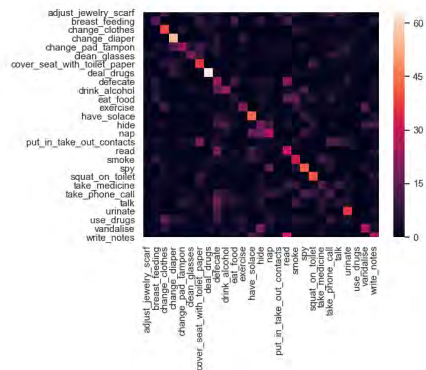
$K = 10$



$K = 100$



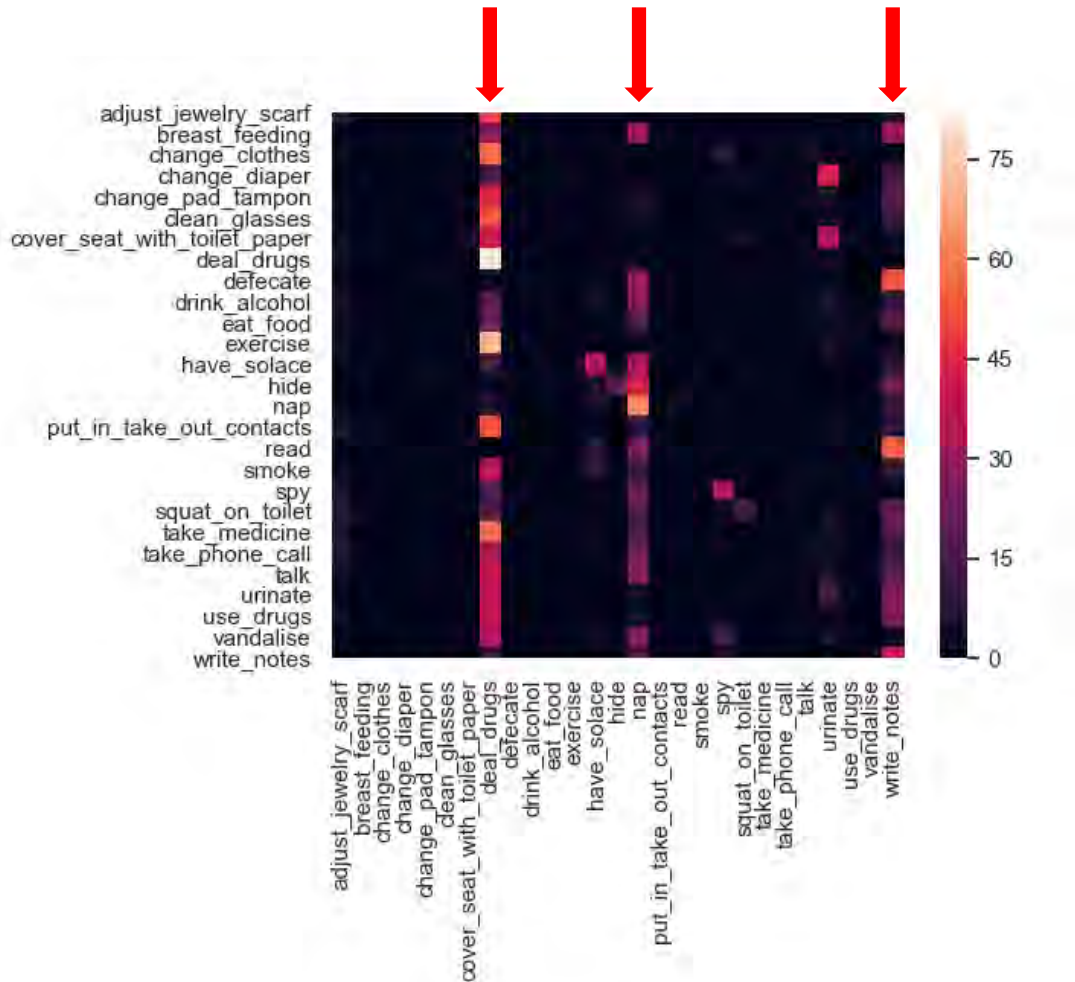
$K = 1000$



**Model:** Logistic Regression  
**Solver:** LBFGS  
**Regularisation Strength:** 0.001

**Solver:** LBFGS

**Regularisation Strength: 0.001**

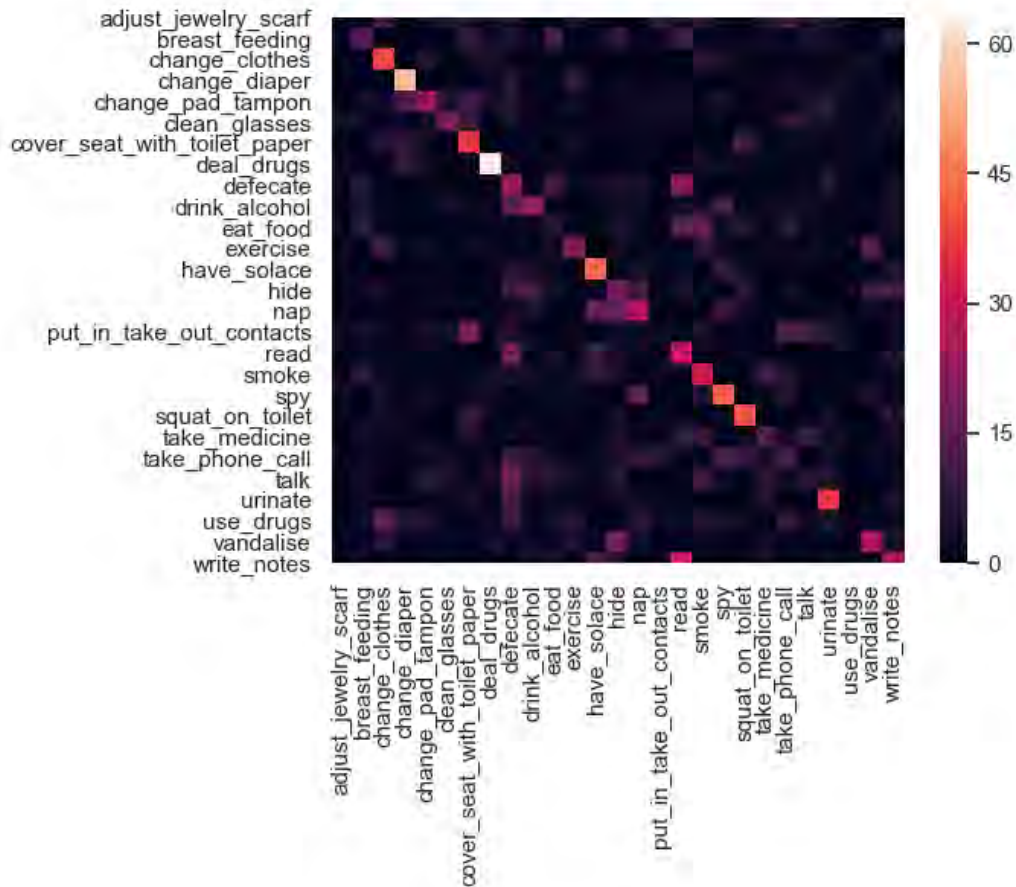




**Model:** Logistic Regression  
**Solver:** LBFGS  
**Regularisation Strength:** 100

**Solver:** LBFGS

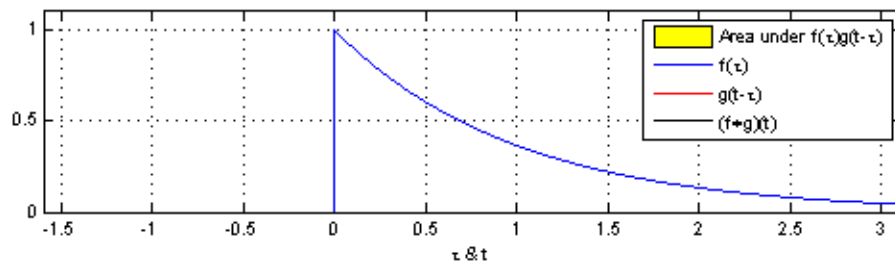
**Regularisation Strength: 100**



# **ConvLSTM**

**(Convolutional Long Short-Term Memory)**

# Conv? (Convolutional)

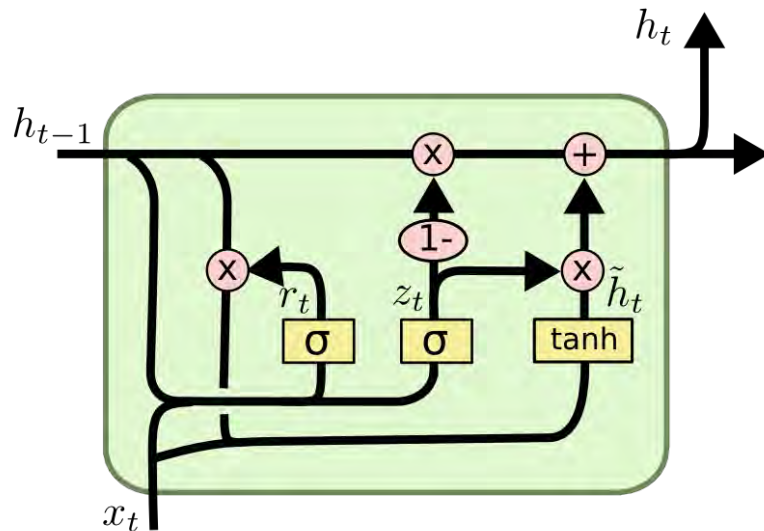
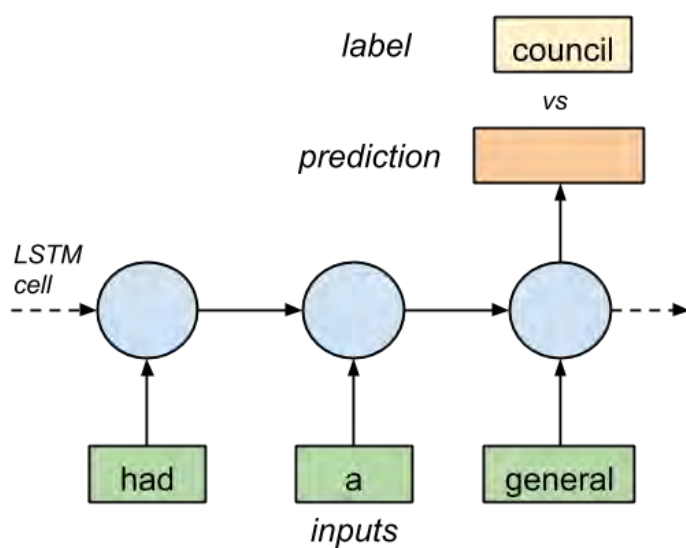


1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	32	33	34	35
36	37	38	39	40	41	42
43	44	45	46	47	48	49

0.1	0.2	0.3
0.4	0.5	0.6
0.7	0.8	0.9

$$\begin{aligned}
 &= 0.1 \times 10 + 0.2 \times 11 + 0.3 \times 12 \\
 &+ 0.4 \times 17 + 0.5 \times 18 + 0.6 \times 19 \\
 &+ 0.7 \times 24 + 0.8 \times 25 + 0.9 \times 26 \\
 &= 94.2
 \end{aligned}$$

# LSTM? (Long Short-Term Memory)

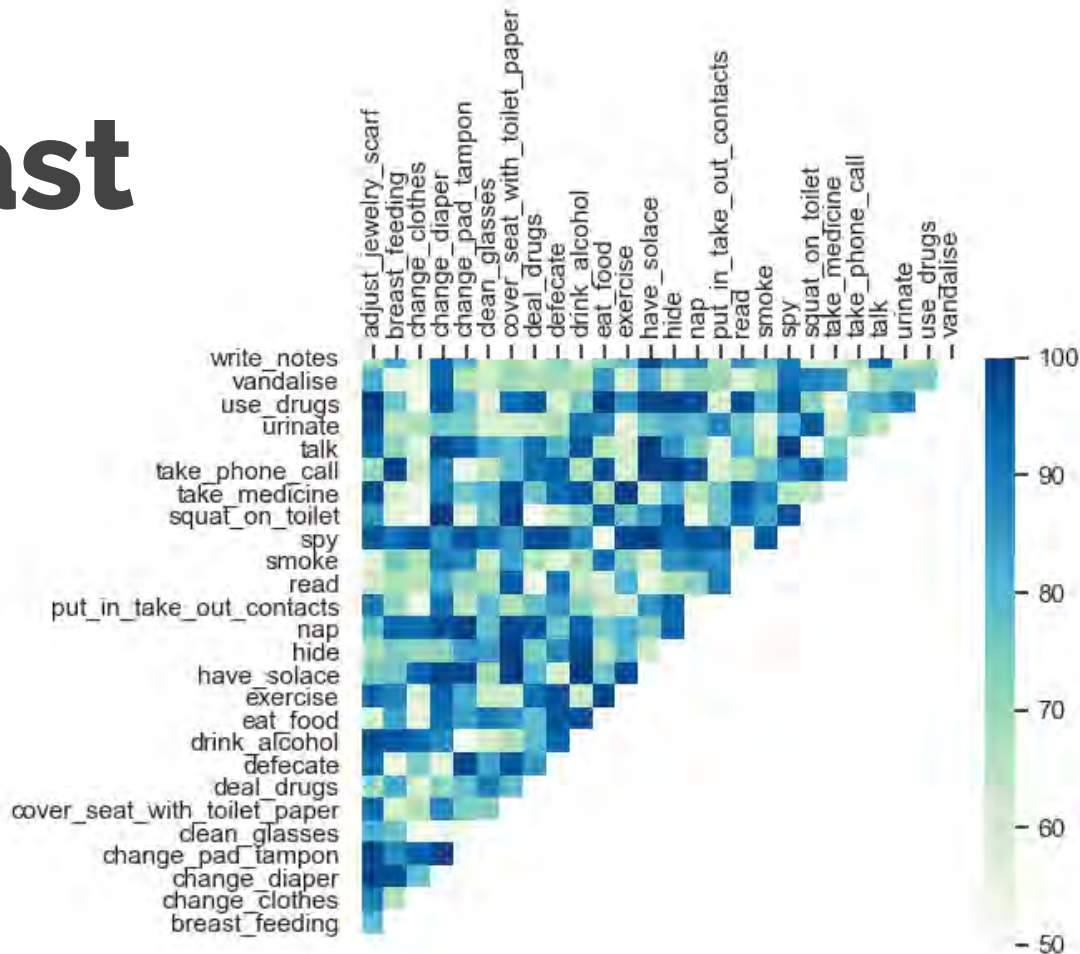


# Pair Contrast

**Classification:** Individual Pairs

**Epochs:** 20 (x496)

**Sequence Length:** 10



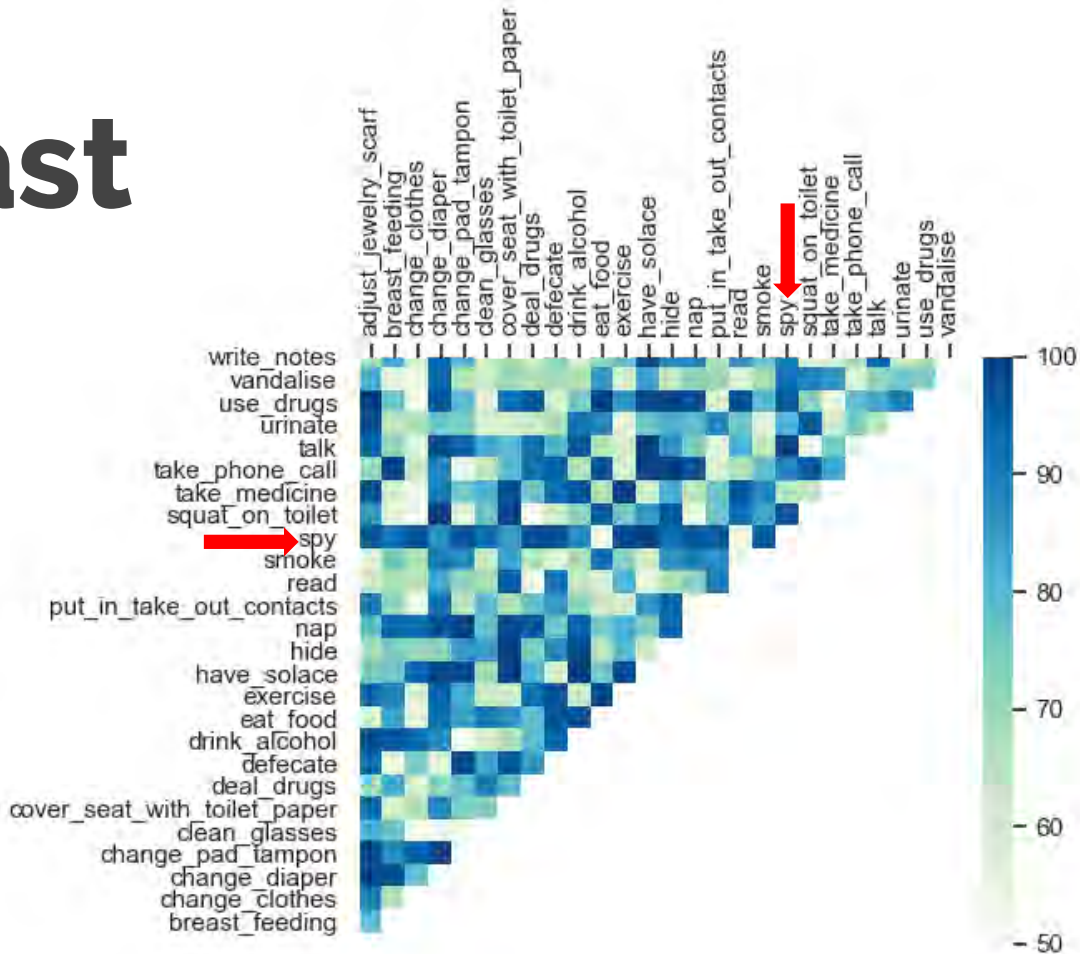


# Pair Contrast

**Classification:** Individual Pairs

**Epochs:** 20 (x496)

**Sequence Length:** 10

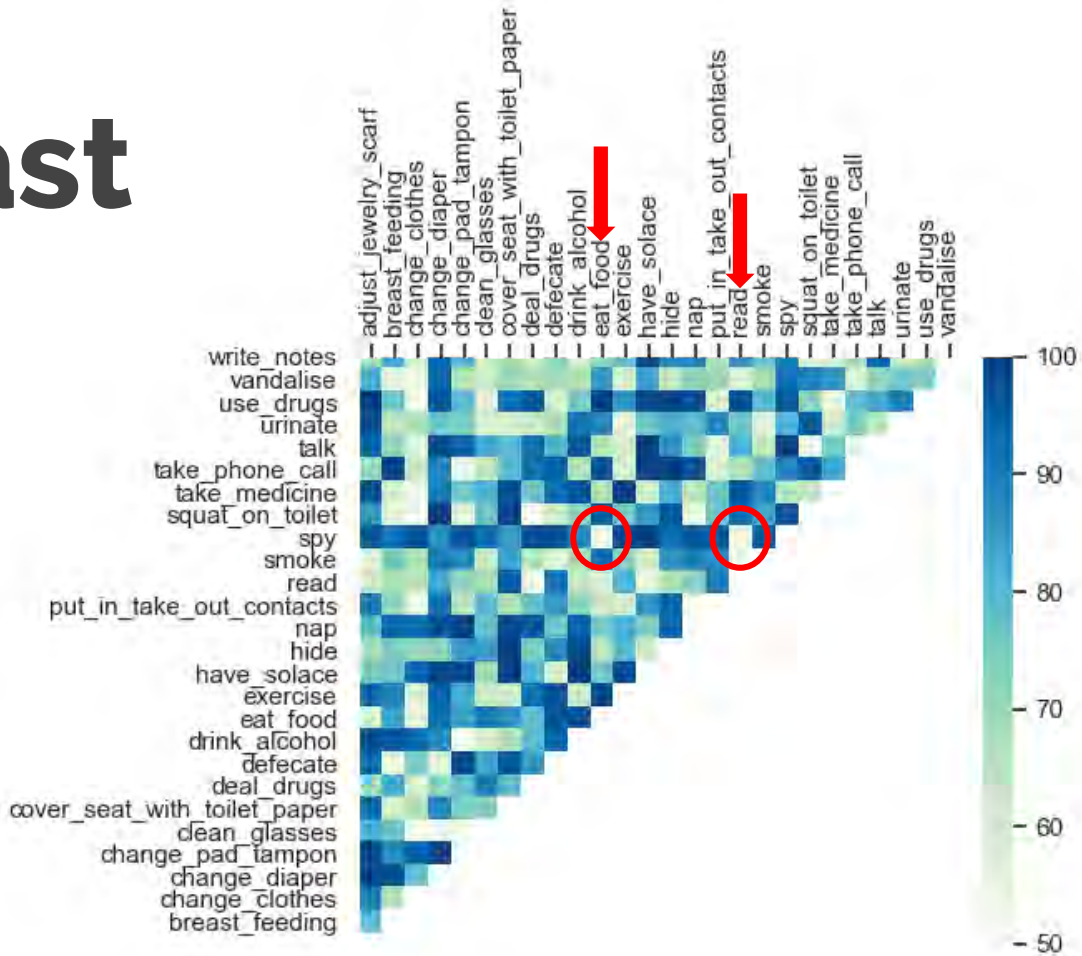


# Pair Contrast

**Classification:** Individual Pairs

**Epochs:** 20 (x496)

**Sequence Length:** 10

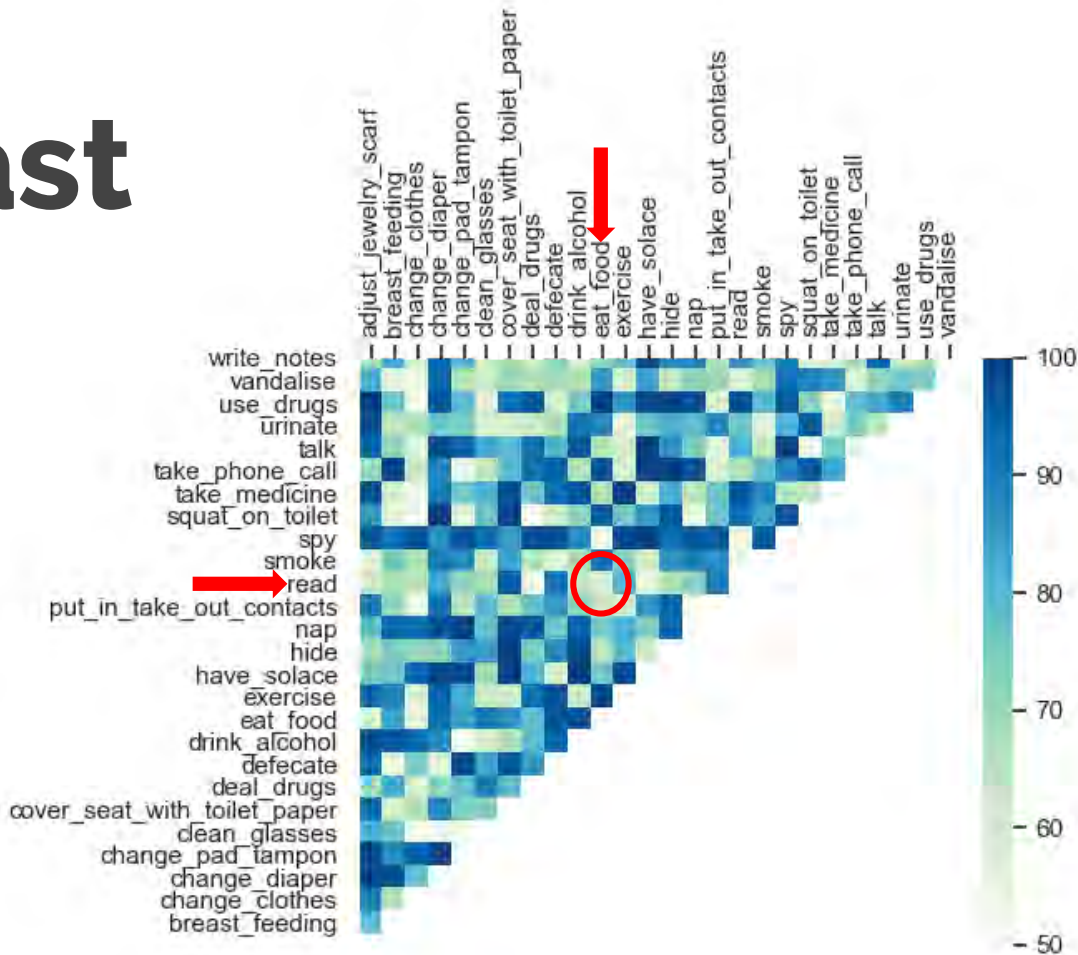


# Pair Contrast

**Classification:** Individual Pairs

**Epochs:** 20 (x496)

**Sequence Length:** 10

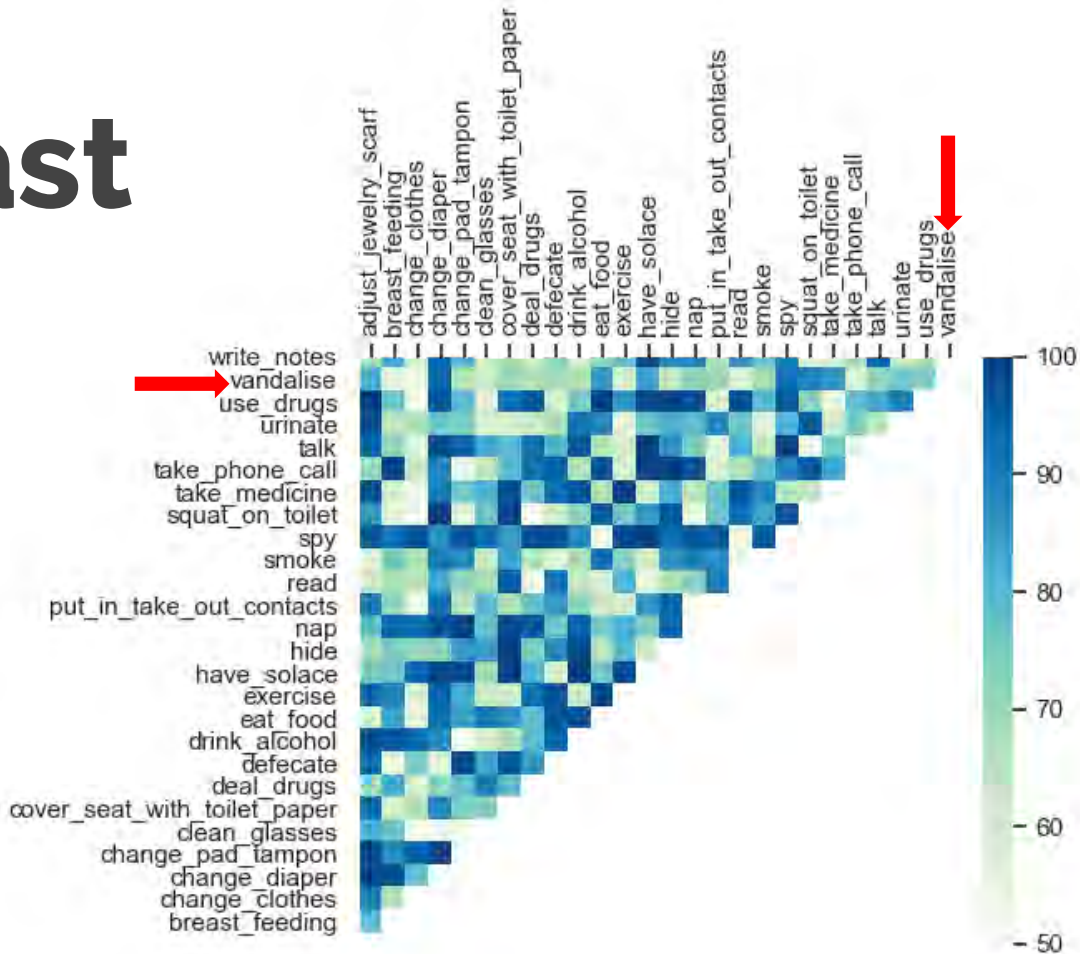


# Pair Contrast

**Classification:** Individual Pairs

**Epochs:** 20 (x496)

**Sequence Length:** 10



**So... ?**

What behaviours *can* be  
detected in bathrooms?



**Model:** SVM (Support Vector Machine)  
**Kernel:** RBF (Radial Basis Function)  
**Gamma:** 0.1

**Kernel:** RBF (Radial Basis Function)

adjust\_jewelry\_scarf  
breast\_feeding  
change\_clothes  
change\_diaper  
change\_pad\_tampon  
clean\_glasses  
cover\_seat\_with\_toilet\_paper  
deal\_drugs  
defecate  
drink\_alcohol  
eat\_food  
exercise  
have\_solace  
hide  
nap  
put\_in\_take\_out\_contacts  
read  
smoke  
spy  
squat\_on\_toilet  
take\_medicine  
take\_phone\_call  
talk  
urinate  
use\_drugs  
vandalise  
write\_notes

adjust\_jewelry\_scarf  
breast\_feeding  
change\_clothes  
change\_diaper  
change\_pad\_tampon  
clean\_glasses  
cover\_seat\_with\_toilet\_paper  
deal\_drugs  
defecate  
drink\_alcohol  
eat\_food  
exercise  
have\_solace  
hide  
nap  
put\_in\_take\_out\_contacts  
read  
smoke  
spy  
squat\_on\_toilet  
take\_medicine  
take\_phone\_call  
talk  
urinate  
use\_drugs  
vandalise  
write\_notes

0 20 40 60 80

# Main takeaways:

- We *can* differentiate between a lot of behaviours

# Main takeaways:

- We *can* differentiate between a lot of behaviours
- SVMs with RBFs seem to be the best

# Main takeaways:

- We *can* differentiate between a lot of behaviours
- SVMs with RBFs seem to be the best
- Vague behaviours are the hardest to classify (e.g. vandalism)

# Main takeaways:

- We *can* differentiate between a lot of behaviours
- SVMs with RBFs seem to be the best
- Vague behaviours are the hardest to classify (e.g. vandalism)
- 80% accuracy!



# What's left?

(a *LOT*)

- Collect some data!

# What's left? *(a LOT)*

- Collect some data!
- Collect some more data! (Sinks? Doorways?)

# What's left? *(a LOT)*

- Collect some data!
- Collect some more data! (Sinks? Doorways?)
- What actually *is* a camera? What *is* privacy infringing?

# What's left? *(a LOT)*

- Collect some data!
- Collect some more data! (Sinks? Doorways?)
- What actually *is* a camera? What *is* privacy infringing?
- Where could this be used? Does it work well?

# What's left? *(a LOT)*

- Collect some data!
- Collect some more data! (Sinks? Doorways?)
- What actually *is* a camera? What *is* privacy infringing?
- Where could this be used? Does it work well?
- How can this data be used for truly *evidence*based design?



**What now?**

**Baptiste.higg.gs**