Brain Decoding: Working with the Haxby dataset

Antoniya Boyanova (ID: 5507136)

Freie Universität Berlin

Statistical Methods II
(127001 SS21)

Dr. Nicolas Schuck

Friday 15th of August

<u>Problem 1</u>

A.

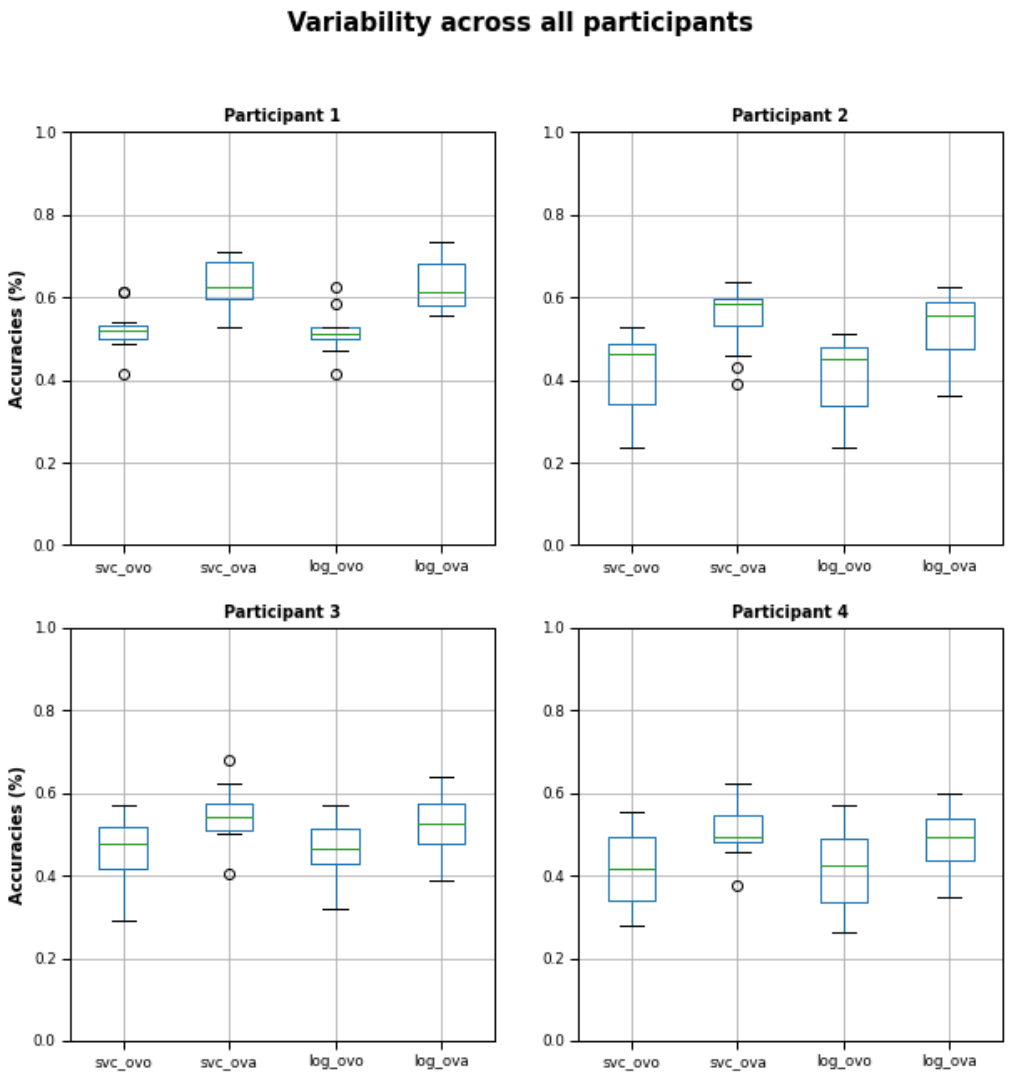


Figure 1. Boxplots representing the accuracy variability of the different classifiers and the different classification strategies for each of the four participants. SVC stands for Support Vector Classifier and Log for Multinomial Logistic Regression. OVO stands for One vs One and OVA approach stands for One vs All (i.e., Rest) approach.

The accuracy variability across the different participants follows a similar pattern with the Support Vector Classifier, which employed the One vs All (i.e., Rest) approach showing highest accuracy across all participants. Importantly, it's Logistic Regression counterpart also exhibited good accuracy. This suggests that the best decoding strategy would be to separate one class, or in the current case one image type, from all the other ones. Notably, this approach is commonly used for combining multiple two-class SVCs in order to build a multiclass classifier (Vapnik, 1998).

B.

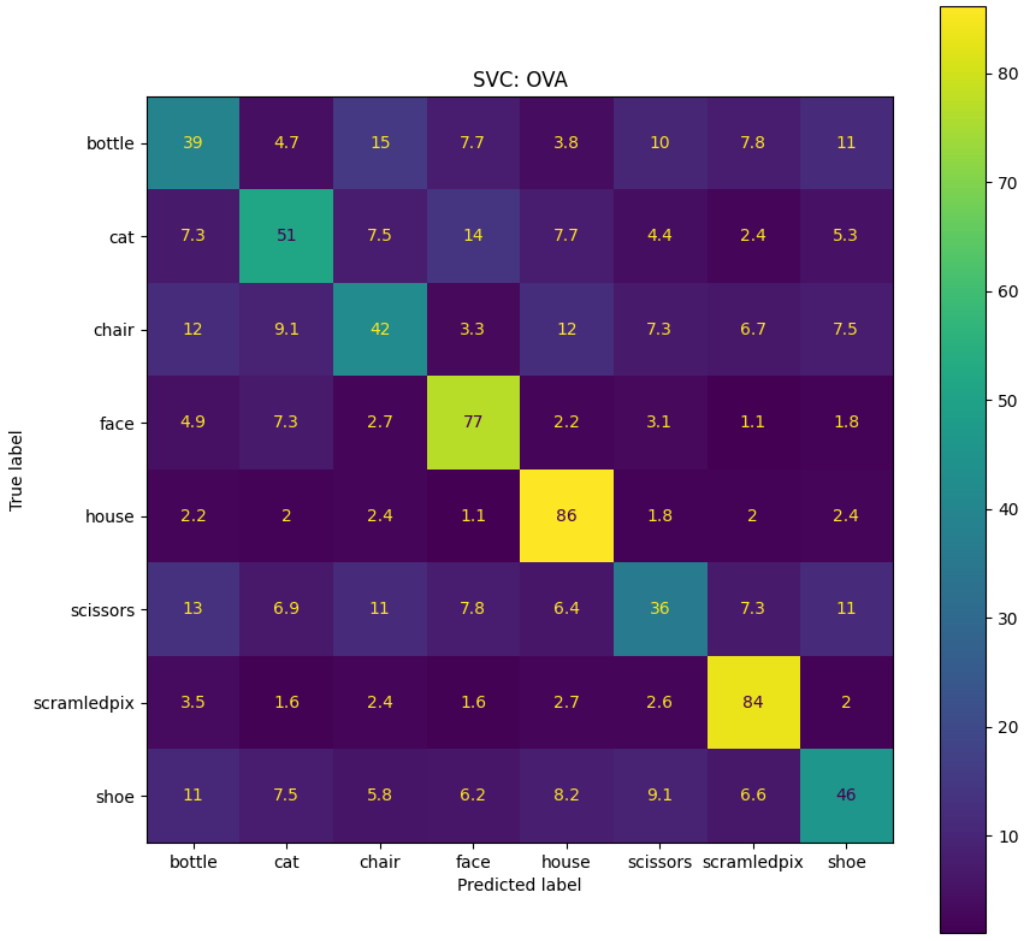
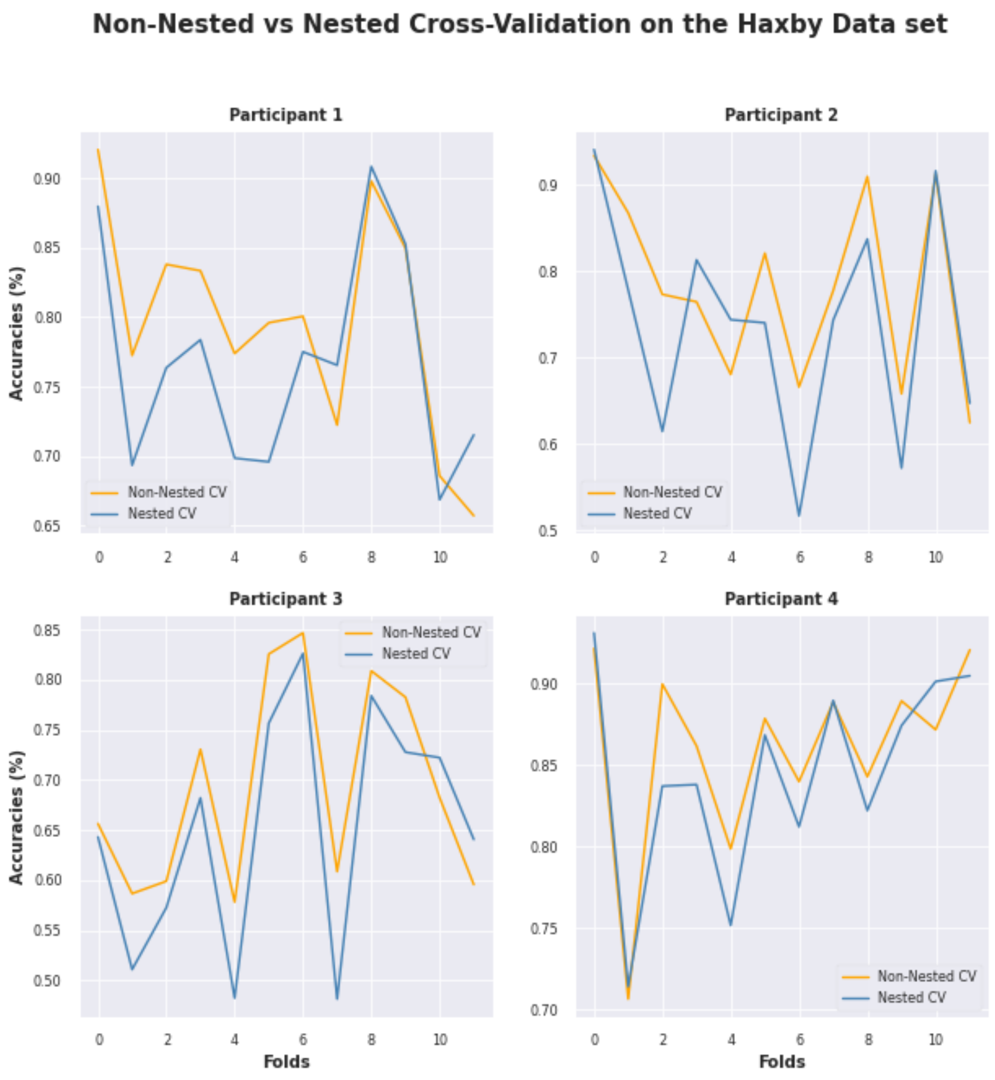


Figure 2. Confusion Matrix for the classifier with the highest performance among participants: Support Vector Classifier with One-vs-Rest Approach. The numbers represented on the Matrix reflect percentage (%).

The mean percentage of recall for the Confusion Matrix of the One-vs-Rest Support Vector Classifier was 57% and the precision was 56%. These results are indicating that out of all the times a label should have been predicted, 57% of the labels were correctly predicted. And for precision, out of the times a label was predicted, 56% of the time the system was in fact correct. However, on a group level these results do not explain which label is associated with the highest decoding accuracy. Based on further investigation of the Confusion Matrix above the three image categories with the highest decoding accuracy are houses, scrambled pictures, and faces. This observation is consistent with the original findings (Haxby, et al., 2001), where the accuracy of identification of the same three categories being viewed based on the patterns of response evoked in ventral temporal cortex were reported without any errors. The poorest decoding accuracy is for the images of bottles, where also the highest false negative values are seen.

C.



1

Figure 4. Non-Nested vs Nested Cross-Validation for each participant.

[1] Please refer to the script called: Brain_Decoding_Haxby, under the section for 1C. to see how these plots were obtained. If you want to check if the code is functional, it is enough to run the cells with the libraries and the data and then execute the cell under the description.

Sometimes selecting the best performing machine learning model with optimal hyperparameters can still end up with a poorer performance once in production**.** Hence, a more rigorous model validation can be key to a successful outcome. Moreover, in high-dimensional settings, such as the current case, when the number of model parameters is much larger than the sample size, some form of model parameter adjustment is needed. Interestingly, an overall accuracy improvement from the nested cross-validation scheme was not observed for any of the participants. The best estimator obtained from the Grid Search differed across folds and participants.

An important observation which can be noted from **Figure 4**. is that the accuracy varies across folds substantially. In the current procedure a fold constitutes training a model on a single run and testing it on all the other runs. One possible explanation for this are potential non-stationarities, e.g., fluctuation of attention of the subject, or sampling noise (Varoquaux, 2018). Moreover, this is an indication that for the cross-validation procedure, the data which is represented under each fold is subject to sampling variability, as it is inherently different in each run. This can result in model instability, which is one symptom of unsuccessful optimization. Another important factor to consider is the overall training sample size. In the current case the training sample size always consisted of just one run. The training sample size has a direct influence on the learning curve of the performance of a given classifier (Beleites et al. 2012). As two categories of prediction errors, the additional systematic deviations (bias) and the random deviations (variance) depend on the training sample size and tend to zero as more cases per class become available. In the current case the training set consisted of in total 72 samples, meaning that roughly there were about 9 cases per class in each training.

Problem 2

In general, the more complex a model, the better it fits the training data, but if it is too complex, its generalizability is reduced, as it memorizes the training data but is less accurate on future test data. To overcome this issue, one needs to regularize a model in order to make the fit complex enough, but not too complex.
One way to think about Regularization is to think in terms of the magnitude of the overall weights of the model. A model with big weights can fit more data perfectly, whereas a model with smaller weights tends to underperform on the train set but can surprisingly do very well

on the test set. However, having the weights too small can also be an issue as it can then underfit the model. In **Figure 4** the weights of Logistic Regression model, fit to simulated data without any regularization is shown. The weights associated with different features vary substantially, with some features having much heavier weight than others.

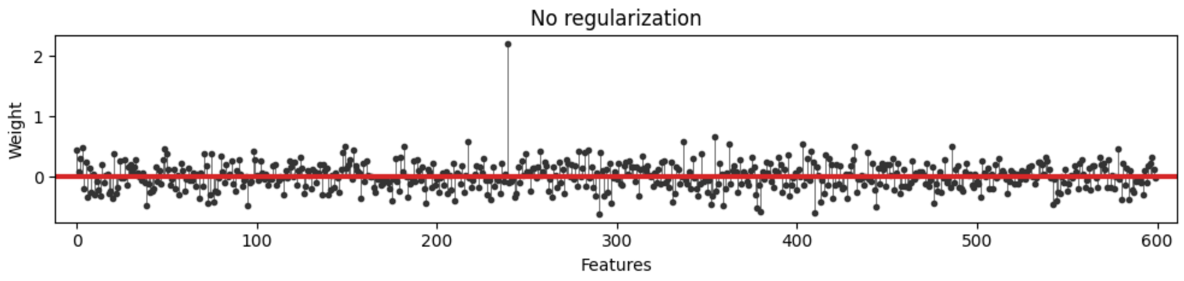


Figure 4. A simulated logistic regression model with 600 features and no regularization. Each dot represents how the model uses each feature to estimate a decision.

There are two main ways in which Regularization can be achieved. A common way is to use an *L*2 or "Ridge" penalty (**Figure 5A**). In this regularization procedure the weights keep the same basic pattern, but the regularized weights are an order-of-magnitude smaller. Another Regularization method is the *L*1, or "Lasso" penalty (**Figure 5B**). Relative to L2 Regularization in this method the weights no longer have the same basic pattern as the non-regularized model. Now most parameters are set to 0.

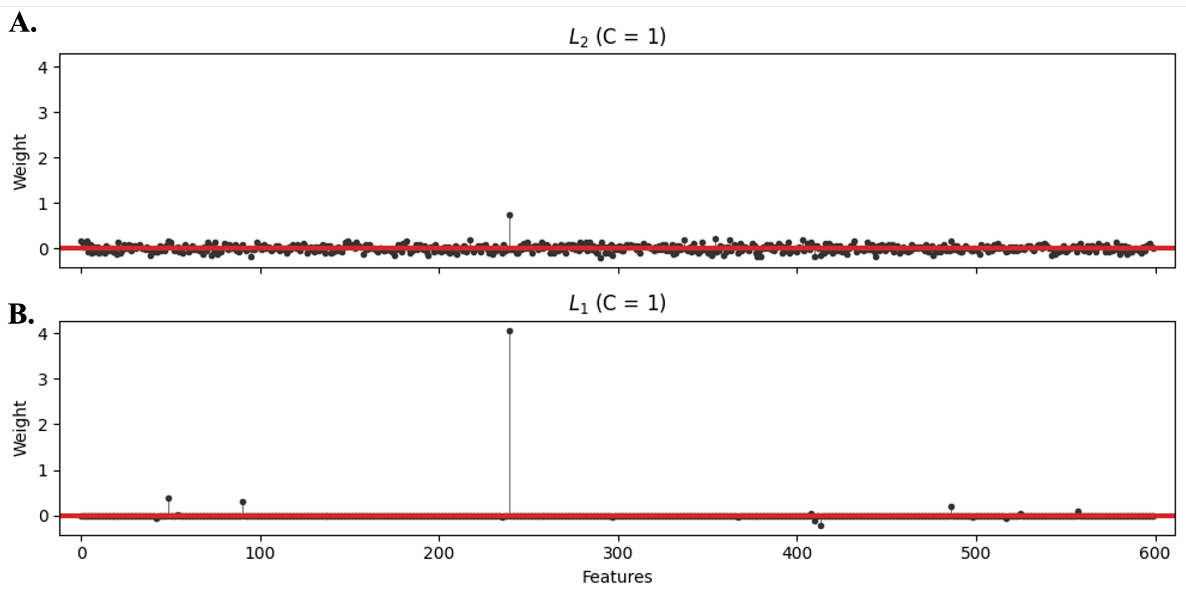


Figure 5. Applying different types of regularization to the simulated data and Logistic Regression. **A.** Depicts L2 ("Ridge") regularisation. **B.** Depicts L1 ("Lasso") regularisation.

The question is how these regularisations are achieved. The general structure of minimizing an error function is as follows:

$$E(w) = E_D(w) + \frac{\lambda}{2}\sum_{j=1}^{M}|w_j|^q$$

The two terms of interest here are λ and q and their effect on regularisation. Both λ and q are known as hyperparameters. There is no unique solution to solve them, it depends on the data and the problem at hand. Both hyperparameters need tuning, as our goal is by adding the additional terms to make $E(w)$ as low as possible. Then there is a trade-off for E(w) and w.

If both λ and q are set to 0, they will no effect on the Error function, hence it is equivalent to no regularization. For $\lambda > 0$ and $q > 0$ we have penalty on the w – the larger they are the bigger the penalization. The error grows linearly (**Figure 6A**) with lambda alone and exponentially with q (**Figure 6B, C**). Additionally, λ sets strength and sparsity of the regularization. To find the best hyperparameters for out models we need to employ the use of cross validation. Generally, we need to be careful with regularization – if the penalization is too weak, we risk overfitting and if it is too strong the model will fail to learn. The effect of q is related to the regularisation method that is being applied. If $q = 1$, this indicates that we are using L1 ("Lasso") regularisation (**Figure 5B**). If $q = 2$, this indicated that we are using L2 ("Ridge") regularisation (**Figure 5A**). If both the terms L1 regularization and L2 regularization are introduced simultaneously in our cost function, then we are using elastic net regularization. The important underlying difference between L1 and L2 regularisation is sparsity. A sparse vector is a such that has mostly zero elements, whereas a dense vector is such, which has mostly non-zero elements. If we think of the weight as a set of vectors, this description is well observed in **Figure 5**.

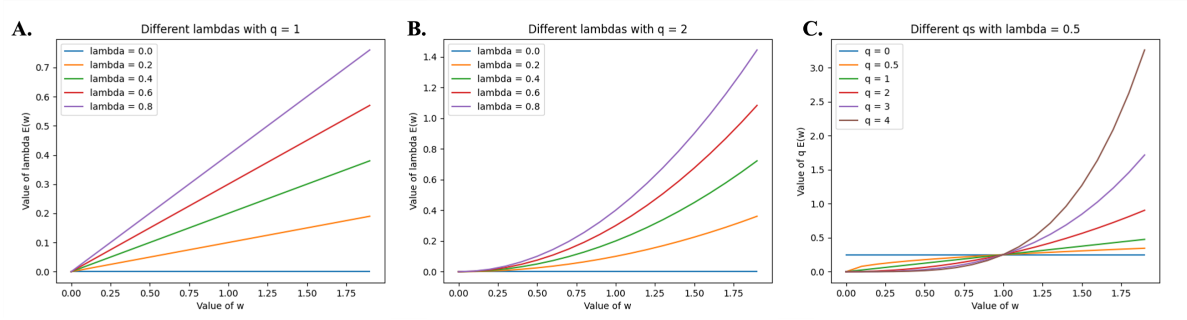


Figure 6. The relationship between the values of lambda and/ or q and the value of the weights (w). **A.** The values of lambda vary, but q remains constant at 1. **B.** The values of lambda vary, but q remains constant at 2. **C.** The values of q vary, but lambda remains constant at 0.5.

Reference:

Beleites, C., Neugebauer, U., Bocklitz, T., Krafft, C., & Popp, J. (2013). Sample size planning for classification models. *Analytica chimica acta*, *760*, 25-33.

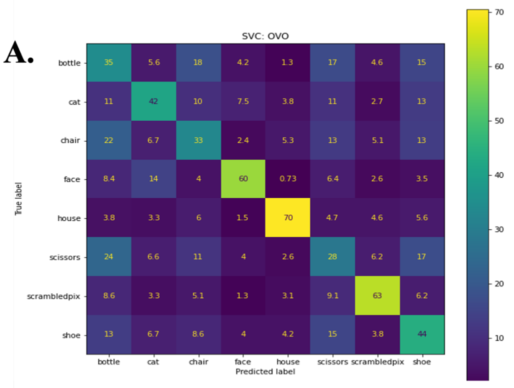
Haxby, J. V., Gobbini, M. I., Furey, M. L., Ishai, A., Schouten, J. L., & Pietrini, P. (2001). Distributed and overlapping representations of faces and objects in ventral temporal cortex. *Science*, *293*(5539), 2425-2430.

Vapnik, V. N. (1998). *Statistical learning theory*. Wiley.

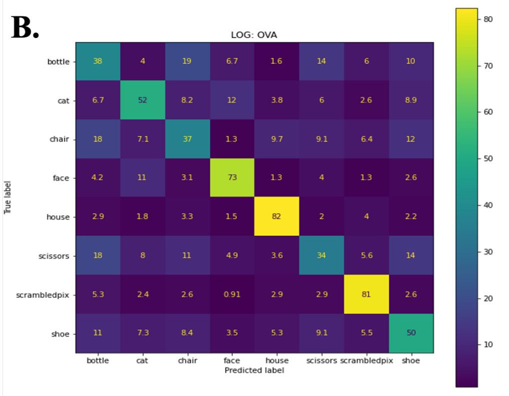
Varoquaux, G. (2018). Cross-validation failure: small sample sizes lead to large error bars. *Neuroimage*, *180*, 68-77.

Appendix: Confusion Matrices from the other Models. **A.** Support Vector Classifier: One-vs-One Classification. **B.** Logistic Regression Classifier: One-vs-Rest. **C.** Logistic Regression Classifier: One-vs-One.

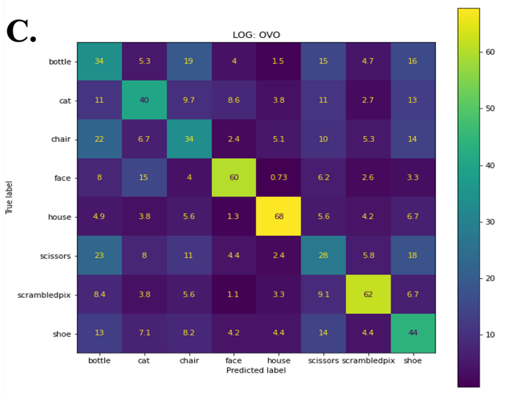
A.



B.



C.


LOG: OVO
True label
Predicted label
bottle 34 5.3 19 4 1.5 15 4.7 16
cat 11 40 9.7 8.6 3.8 11 2.7 13
chair 22 6.7 34 2.4 5.1 10 5.3 14
face 8 15 4 60 0.73 6.2 2.6 3.3
house 4.9 3.8 5.6 1.3 68 5.6 4.2 6.7
scissors 23 8 11 4.4 2.4 28 5.8 18
scrambledpix 8.4 3.8 5.6 1.1 3.3 9.1 62 6.7
shoe 13 7.1 8.2 4.2 4.4 14 4.4 44
bottle cat chair face house scissors scrambledpix shoe