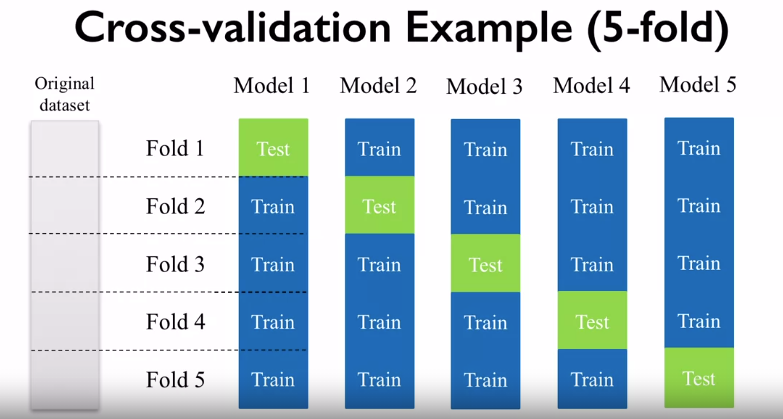
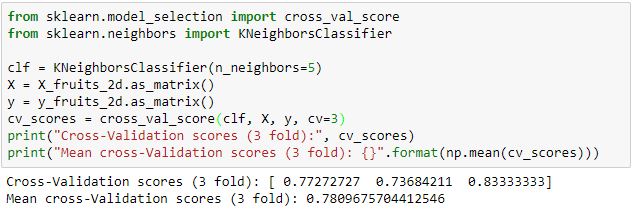
**Cross Validation:**

Cross validation goes beyond evaluating a model using one set of train\_test\_splits and uses multiple versions of this on the same model to get a better understanding of the true behaviour of the model once it has been deployed.

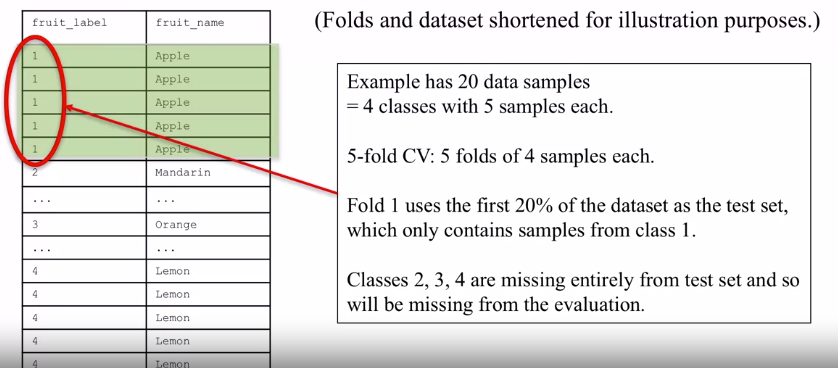
This is similar to having multiple random seeds in the train\_test\_split function. A different seed would result in different training and test results when we call the score functions.



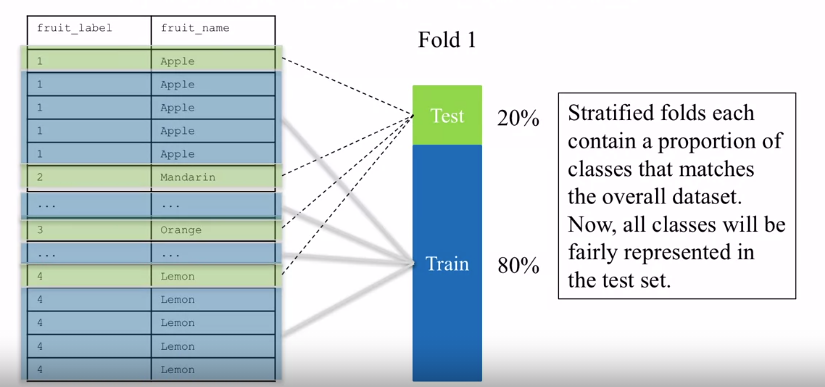


The as\_matrix is used here to convert the series into a matrix so the cross\_val\_score can be used. “cv” is used here to control the number of folds.

Cross validation also shows us how sensitive the model is to the training data; we can plot a distribution of the training scores to see what the chance of the model performing very well or poorly on new data instances. This gives us a worst case or best-case range for our model. However, this does come at a cost as the CV takes more time to run.



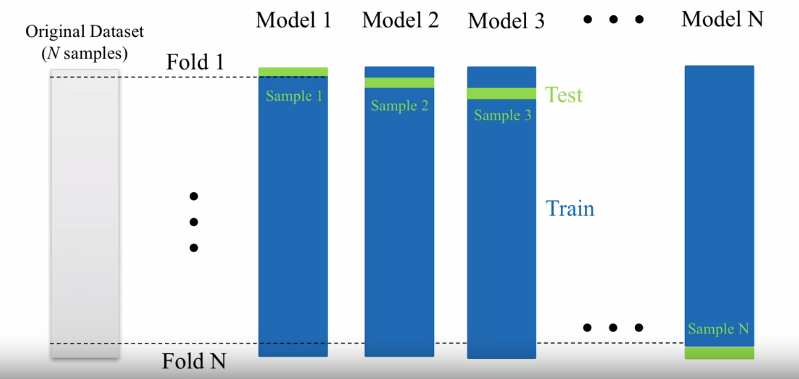
In the default set up of CV is that the first fold uses the first 20% (for a 5-fold cv), this first 20% might not include all of the possible target labels. This would result in strong class bias when testing the model’s performance. Instead we do the below:



When we ask Sklearn to do a CV for us what it actually does is a **Stratified** **Cross-Validation** which can be seen above. This ensures that a balanced proportion of the target labels end up in all the test folds. This allows for fair representation of data.

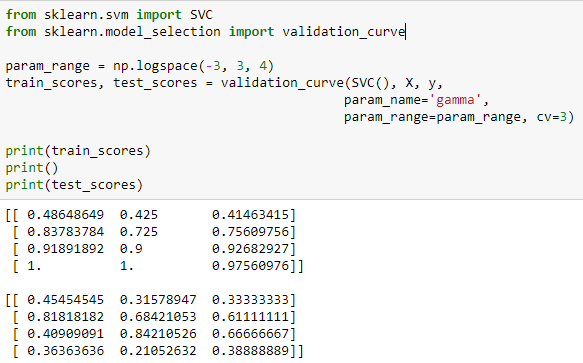
For **Regressions** Sklearn doesn’t do this **Stratified CV** because there is technically only one target label category (numbers).

**Leave-one-out cross-validation**

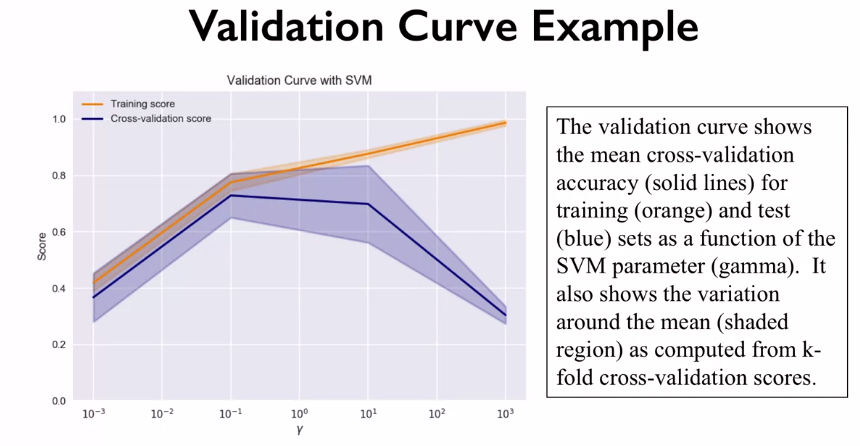
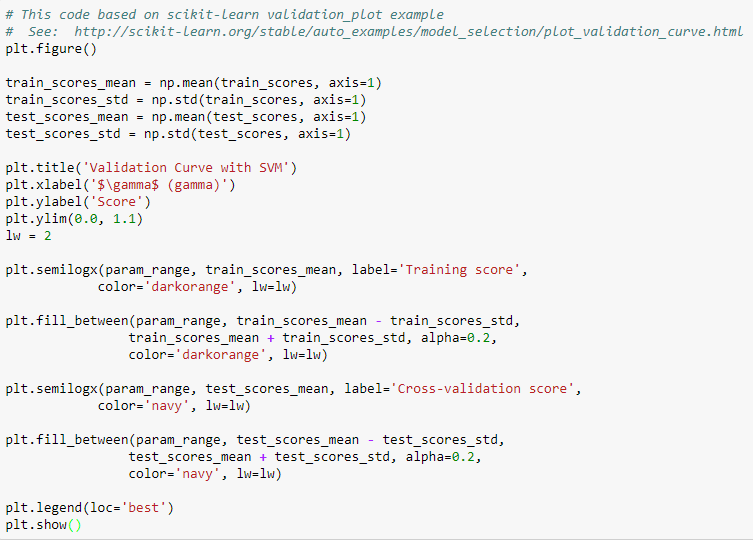


This is CV but with the number of folds in the CV set to N-1 of the data length. E.g. each fold consists of just a single sample of the test data, and the rest as training data. This can be very valuable for small data sets as you can have a very good understanding of how the model will behave when deployed.

Below looks at how to evaluate a model using CV with varying parameter (gamma SVM).



One row per parameter, and one column per CV fold. You would then plot out these values as: Training score and cross-validation scores and then see which values have the highest scores and are also the closest to one another.



Above the cross-validation scores are from the test data.

CV is only used to validate a model and is not used to tune a model.