3 Data Pre-processing

In order to create a curate data set, that is complete, robust and yet meaningful, we need to deal with several concerns. Firstly, we fetched all the interactions in `ChEMBL`that concern kinases which resulted in more than 800 thousand activities. A summary of the total number of activities per type of measurement is shown in Fig. 1. We selected those with $IC50$ values and according to the following rules:

1. compounds with SMILES representation so that we can calculate a fingerprint later on
2. `IC50` ≤ 10000 as some values are huge (data errors?)
3. if there are still duplicates between the same pair we keep the one with the lowest value
4. targets with at least 100 interactions to make predictions feasible
5. compounds interacting with at least 2 targets in order to make train/test sets

The last two steps were done iteratively as they depend to each other. At the end or this process, the dataset was reduced to 110 targets, 23361 compounds and a total of 62656 interactions. Among these, 21262 can be considered **active** (i.e. `IC50` $\leq 30nM$). On average, there are 569.60 interactions per target and 2.68 per molecule. We then proceed with partitioning the data to train/test sets. In order to have a balanced 10% of the data for testing, we randomly select $3,132$ from the set of active interactions and the same number from the set of inactive. This partition is then saved to two different files that will be used independently for a meaningful training of each algorithm and a fair evaluation for the selected model.

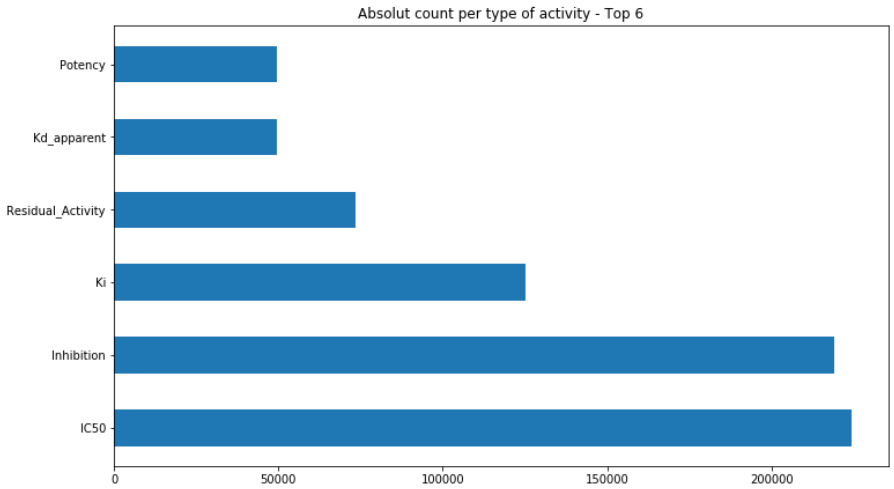
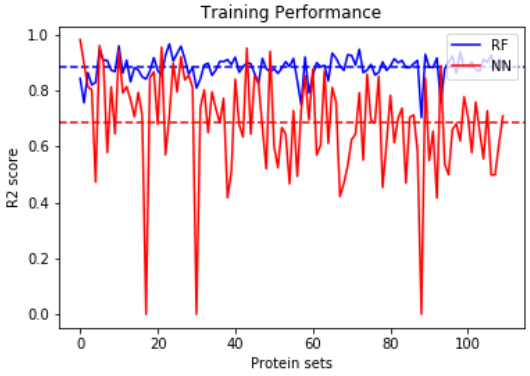


Figure 1: Absolute count of occurrences for each type of activity for the 797 kinases in `ChEMBL`. That's the *initial* state of dataset.

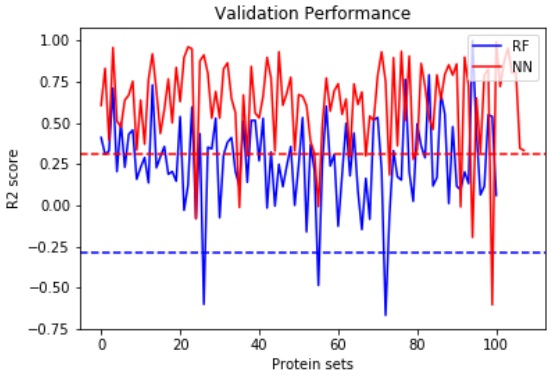
Fingerprints are then calculated with `RDkit`. We used the function `GetMorganFingerprintAsBitVect` with 2048 bits and radius 2 which is similar to ECFP4. Alternatively, we could follow a binary classification approach (active-inactive) by using *pCHEMBL* values instead of focusing on one type of activity values.

4 A first attempt for prediction

We use `Sci-kit learn` and exploit the built-in implementations of Random-Forests (RF) and Neural-Networks (NN) for regression. In order to conduct a meaningful assessment of methods available for matrix completion, we train each method with a cross validation (CV) scheme. In particular, we split the data



(a) R2 score after fitting on the 80% of the training data. Horizontal lines indicate the average performance across the 110 targets, after selecting the best parametrisation with CV. RF achieve a clear advantage over NN.



(b) As in (a) but for the validation set. For the sake of clarity, a couple of outliers (values of -2 or less) have been excluded for both methods but the means are global. NN outperformed RF in almost every case with the latter having a poor performance.

available for training (the 90% of the dataset) in $80-20\%$, use the first for a 5-fold CV and the latter for model validation and selection.

Parameters are selected through a quick grid search; we select those that have the highest average accuracy across the 5 folds, then fit the algorithm to the 80% and evaluate with the validation set. For RF we search for the ideal number of trees among the numbers $[10, 25, 50, 100, 150, 300]$ and the *max depth* from $[10, 100, 200, 500]$. For the case of NN, the unknown parameters are the number of hidden layers and the number of nodes within each. Since training one model for each target is an expensive procedure, we do a quick search among $[(50), (50, 100), (50, 20, 10)]$, which stand for one, two, and three hidden layers. The activation function is set to *logistic* and the selected solver is *Limited-memory BFGS*. More parameters-options could be tested but, as we perform a 5-fold CV for each of the 110 targets, that would take much longer and it's not a priority at this point.

Figure 2a gives some insight on the performance during training for each of the algorithms. RF are really accurate during training, with a robust accuracy of ~ 0.85, outperforming NN, which present higher oscillations. However, things are opposite when it comes for validating with the 20% set, as Figure 2b indicates. Both methods have a quite low average (dashed lines) because of 2-3 cases with super low R2 score; something that needs further investigation. In another perspective, on average across each interaction within the validation set, RF have a R2 score of 0.361086 versus 0.651037 for NN.