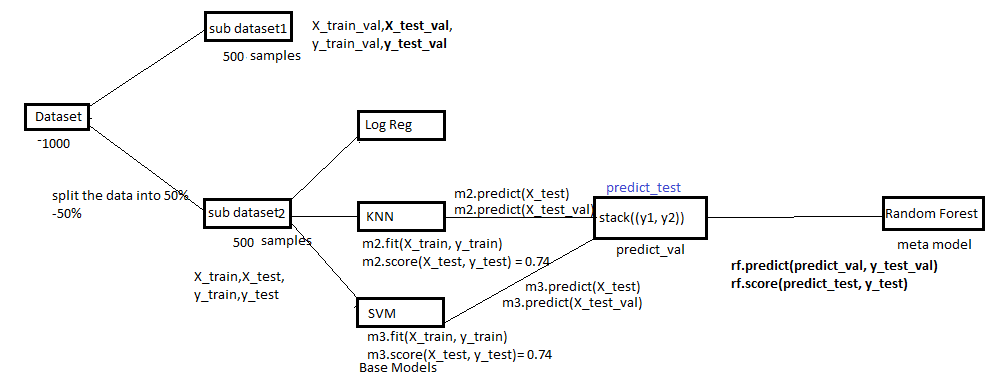
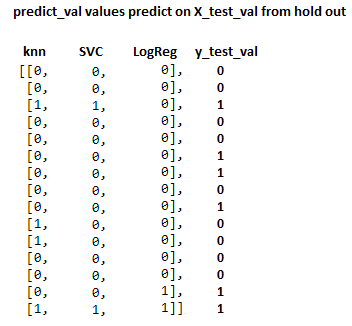
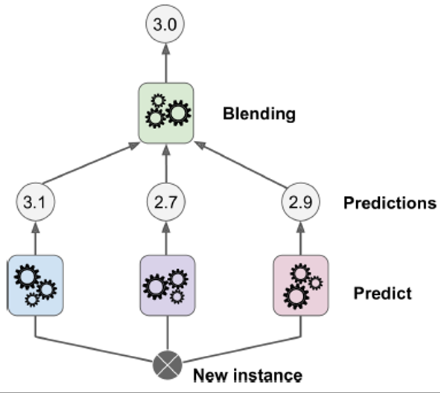
**Stacking (Stacked Generalization)**

Stacking is a type of ensemble technique which combines the predictions of two or more models, also called base models, and use the combination as the input for a new model (meta-model) i.e. a new model is trained on the predictions of the base models.



Suppose you have a classification problem and you can use several models like logistic regression, SVM, KNN, Random forest etc. The idea is to use few models like KNN, SVM as the base model and make predictions using these models. Now the predictions made by these models are used as an input feature for Random forest to train on and give prediction.

Stacking, just like other ensemble techniques, tries to improve the accuracy of a model by using predictions of not so good models and then using those predictions as an input feature for a better model.

Stacking can be multilevel e.g. using base models as level 1 then passing the predictions into another set of sub-base models at level 2 and so on. Then at the end using meta-model/models which take predictions of the last sub base models as input and does prediction.

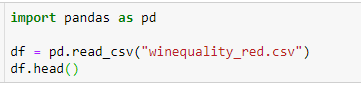
Let's understand more by looking at the steps involved for stacking:

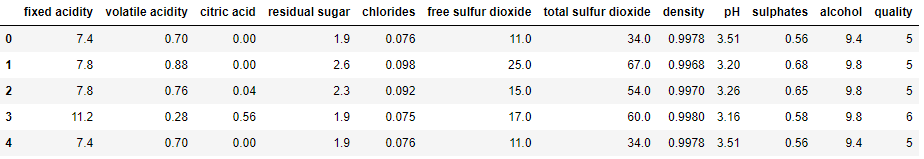
* Split the dataset into a training set and a holdout set. We can use k-fold validation for selecting different set of validation sets.

Generally, we do a 50-50 split of the training set and the hold out set.

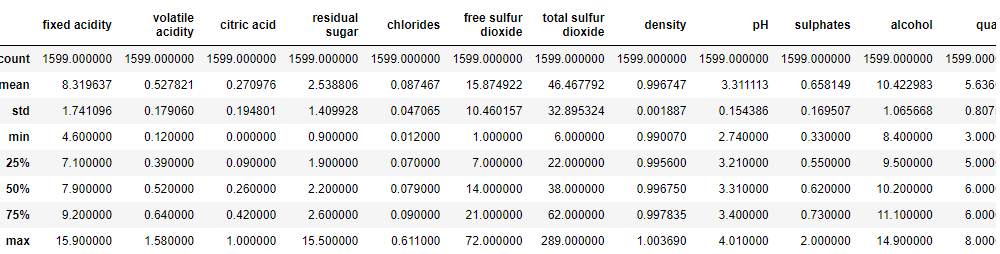
* training set = x1,y1 hold out set = x2, y2
* Split the training set again into training and test dataset e.g. x1\_train, y1\_train, x1\_test, y1\_test
* Train all the base models on training set x1\_train, y1\_train.
* After training is done, get the predictions of all the base models on the validation set x2.
* Stack all these predictions together (you can also take an average of all the predictions or probability prediction) as it will be used as input feature for the meta\_model.
* Again, get the prediction for all the base models on the test set i.e. x1\_test
* Again, stack all these predictions together (you can also take an average of all the predictions or probability prediction) as it will be used as the prediction dataset for the meta\_model.
* Use the stacked data from step 5 as the input feature for meta\_model and validation set y2 as the target variable and train the model on these data.
* Once, the training is done check the accuracy of meta\_model by using data from step 7 for prediction and y1\_test for evaluation.

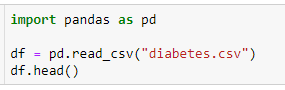
Although, there is no libraries available in Sklearn for stacking, it can still be implemented.

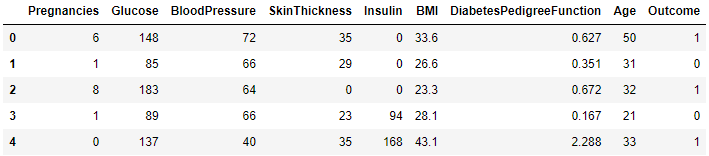




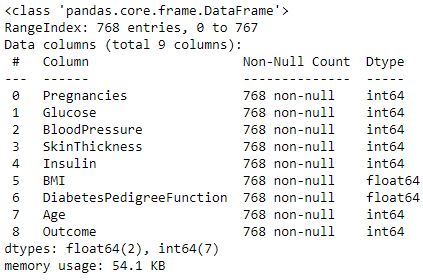




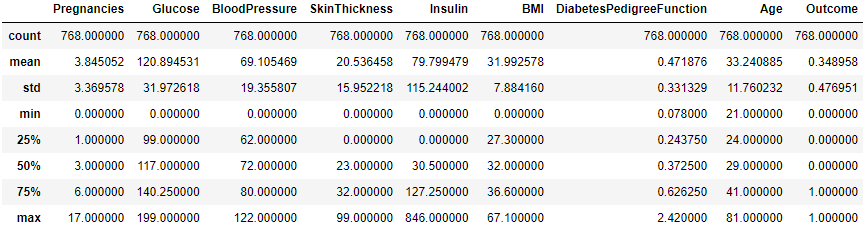




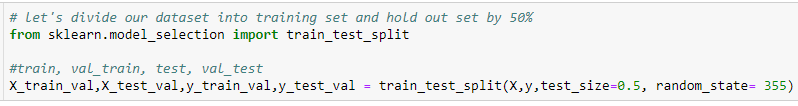




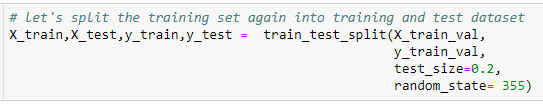








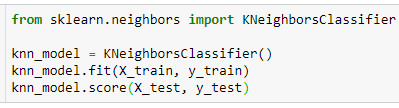




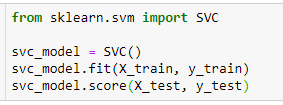


We will use KNN and SVM algorithm as our base models.

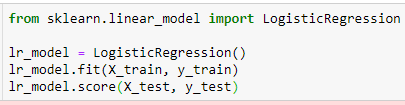
Let's fit both of the models first on the x\_train and y\_train data.



0.7402597402597403

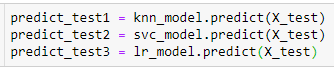


0.7402597402597403

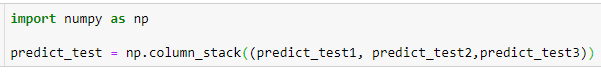


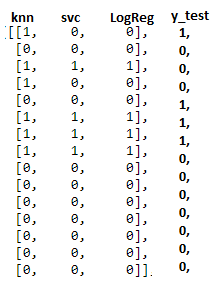
0.7662337662337663

Let's get the predictions of all the base models on the Test set x\_test.

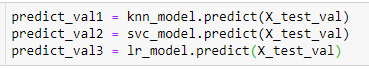


Let's stack the prediciton values for validation set together as "predict\_test"

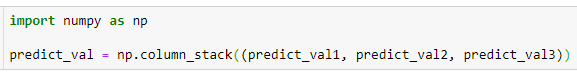




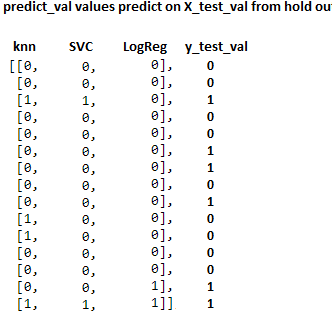
Let's get the predictions of all the base models on the validation set val\_train.

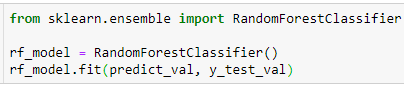


Let's stack the prediciton values for validation set together as "predict\_val"



Let's use the Use the stacked data "predict\_val" and val\_test as the input feature for meta\_model i.e. random forest classifier



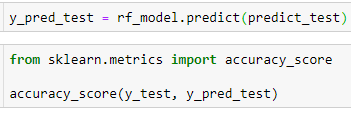


RandomForestClassifier()

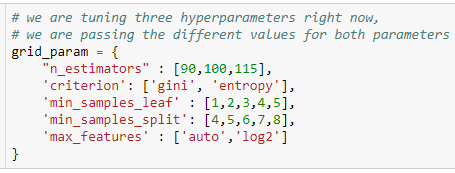
Let's check the accuracy of our meta\_model using predict\_test and y\_test.

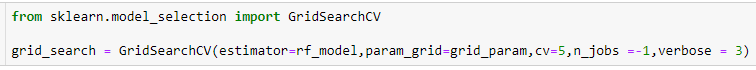


0.7532467532467533

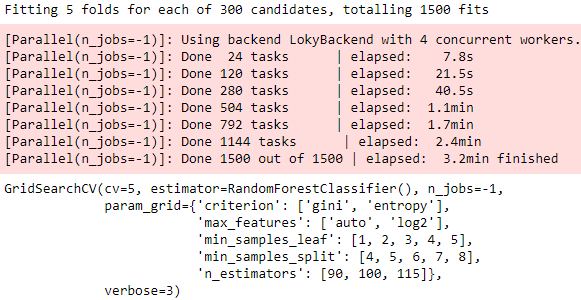


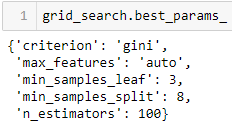
0.7532467532467533

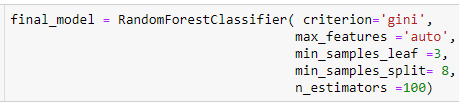










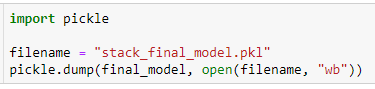




RandomForestClassifier(min\_samples\_leaf=3, min\_samples\_split=8)



0.7662337662337663





array([0], dtype=int64)