Python Task

Anand Sahu[1]

[1] *Supramolecular and Material Chemistry Lab,*
School of Physical Sciences, Jawaharlal Nehru University,
New Delhi, India, 110067

I. OVERVIEW

In this work, I have utilized data.traj(provided) to preparation a dataset which contain structural features describing the atomic environment of the Al atoms in the structures stored as ASE atoms objects in 'atoms_list' and try to build a model which can correlate the structural features with the isotropic shieldings stored in 'atoms_data'. I have utilized python package ase, pandas, numpy, matplotlib for data manipulation and visualisation. Sklearn, keras, tensorflow for Machine learning and deep learning algorithm implementation.

II. COMPUTATIONAL SECTION

a. Dataset Preparation to generate structural features describing the atomic environment of the Al atoms in the structures stored as ASE atoms objects in 'atoms_list'

for the feature calculation, I have utilised "atoms_list" data, and "idx" from "atoms_data" from data.traj. I have then generate second order features using function of ASE library like "get_positions()", "get_atomic_numbers()", "get_masses()", "cell". Combining these data results into a pandas DataFrame as features, required for systematic data feeding to algorithms. While the target isotropic shieldings was derived from "atoms_data".

iso	0	1	2	3	4	5	6
486.4486135	0.011689386	0.371407133	0.851440615	0.347117103	0.032079841	0.864434132	0.486150199
488.8196301	0.037014262	0.35025086	0.845077981	0.363599369	0.020265513	0.862299157	0.481890326
493.7503488	0.02799598	0.375446498	0.810379083	0.372496224	0.01994833	0.851894915	0.477159089
489.5651908	0.01318509	0.367093376	0.853374455	0.352147068	0.038374414	0.848617278	0.49700301
487.2563184	0.0204824	0.369769403	0.8317415	0.341819168	0.046585261	0.855683743	0.484957077
490.7713656	0.051919084	0.336778699	0.830248536	0.356531332	0.030720222	0.845680435	0.473673064
486.6437625	0.039835413	0.356735894	0.824046221	0.361938832	0.019442634	0.864855112	0.472193605
485.1714612	0.003460705	0.388639024	0.847743273	0.335358983	0.046070441	0.853207975	0.503202745
492.5281861	0.003960432	0.37084186	0.870548555	0.360013301	0.023489013	0.858911263	0.48462314
490.8323469	0.024369541	0.365211715	0.844997822	0.363812059	0.02494231	0.851463911	0.485588288
493.2751856	0.011100813	0.375478108	0.846039891	0.364736918	0.021554676	0.861366985	0.480027942

	iso	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
0	486.448614	0.011689	0.371407	0.851441	0.347117	0.032080	0.864434	0.486150	0.076047	0.593066	0.528933	0.395165	0.099335	0.301574	0.608082	0.086289	0.040677	0.534872	0.600978	0.007334	0.460148	0.349266
1	488.819630	0.037014	0.350251	0.845078	0.363599	0.020266	0.862299	0.481890	0.063309	0.598033	0.513646	0.398735	0.104579	0.301504	0.620638	0.091639	0.039705	0.534184	0.587502	0.009038	0.464921	0.351089
2	493.750349	0.027996	0.375446	0.810379	0.372496	0.019948	0.851895	0.477159	0.071741	0.614877	0.507593	0.398489	0.117243	0.295745	0.606754	0.095276	0.049553	0.527804	0.592422	0.000204	0.476694	0.340802
3	489.565191	0.013185	0.367093	0.853374	0.352147	0.038374	0.848617	0.497003	0.070585	0.586206	0.533701	0.389934	0.093329	0.299179	0.597844	0.098573	0.030148	0.524594	0.611758	0.556955	0.459093	0.357720
4	487.256318	0.020482	0.369769	0.831742	0.341819	0.046585	0.855684	0.484957	0.071078	0.598764	0.537566	0.405023	0.078871	0.295604	0.587390	0.114153	0.035109	0.526862	0.595387	0.553851	0.468915	0.336986
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1552	493.972531	0.917768	0.105013	0.270539	0.907720	0.238011	0.259076	0.513679	0.735670	0.734674	0.435318	0.904540	0.806789	0.723980	0.929692	0.760457	0.518540	0.311844	0.351185	0.613884	0.167912	0.363594
1553	493.112637	0.889703	0.088643	0.245073	0.913017	0.221530	0.294436	0.523726	0.729009	0.766162	0.443937	0.900334	0.003277	0.717621	0.923591	0.767278	0.512521	0.317216	0.348709	0.592245	0.167655	0.364819
1554	492.038928	0.885342	0.078820	0.288345	0.898647	0.213862	0.237824	0.520878	0.745011	0.759633	0.430388	0.910216	0.024899	0.717345	0.923808	0.760128	0.523921	0.313977	0.354411	0.611839	0.162852	0.369919
1555	497.101677	0.075237	0.111669	0.209906	0.908348	0.213735	0.316182	0.513050	0.736844	0.746714	0.426552	0.912716	0.822154	0.708394	0.906059	0.770824	0.521779	0.298088	0.352206	0.615469	0.166655	0.355303
1556	495.378204	0.890704	0.195693	0.274269	0.869219	0.264813	0.129563	0.522214	0.729560	0.755241	0.444723	0.907746	0.000972	0.708174	0.922892	0.751283	0.526180	0.301979	0.339272	0.616924	0.161242	0.356849

1557 rows × 165 columns

Fig. 1: Sample of dataset prepared. The Column "iso" is the target while rest columns are features.

The resultant features of the dataset was normalized. Then, Data was separated into two parts. First, Training dataset and second, External testing dataset, which will be used to check the validity of final model. Then, training dataset was again divided into three part viz Training set, Validation set, and Testing set. Furthermore, Using, Training set featurization of important features was done using pearson's correlation coefficient. After, selecting the important features, training of algorithms begin, their performance was tested using validation set, while, testing set was used to check model prediction accuracy for unseen data.The complete python code for data preparation step discussed above in given in a ipynb file named "Step-1-Dataset-Preparation.ipynb". The sample of overall dataset is given in Fig.1 .

b. Modelling for function to correlates the structural features with the isotropic shieldings stored in 'atoms_data'

In order to correlate structural features with the isotropic shieldings, I first prefered to use Machine learning model due to their less computation extensivity in comparison to deep learning model. Firstly, I preliminarily compared several machine learning models from simple linear models to tree based models, then ensemble, bagging and boosting models. Then, I selected top 5 best performing models, and again performed hyperparameter optimization and tuning, also applied cross validation to improve generalisation and avoid overfitting or underfitting. Thereafter, these best models were tested against external unseen data, and the model which performed best among other was finalised.

Also, in order to further check if deep learning can outperform best performing machine learing models, I built a deep learning model using keras and optimized it using Bayesian optimization.Similar to Machine learning process, deep learning model was also tested against external unseen dataset. At last, deep learning model was compared with machine learning models, and the model that performed best, was finalized.

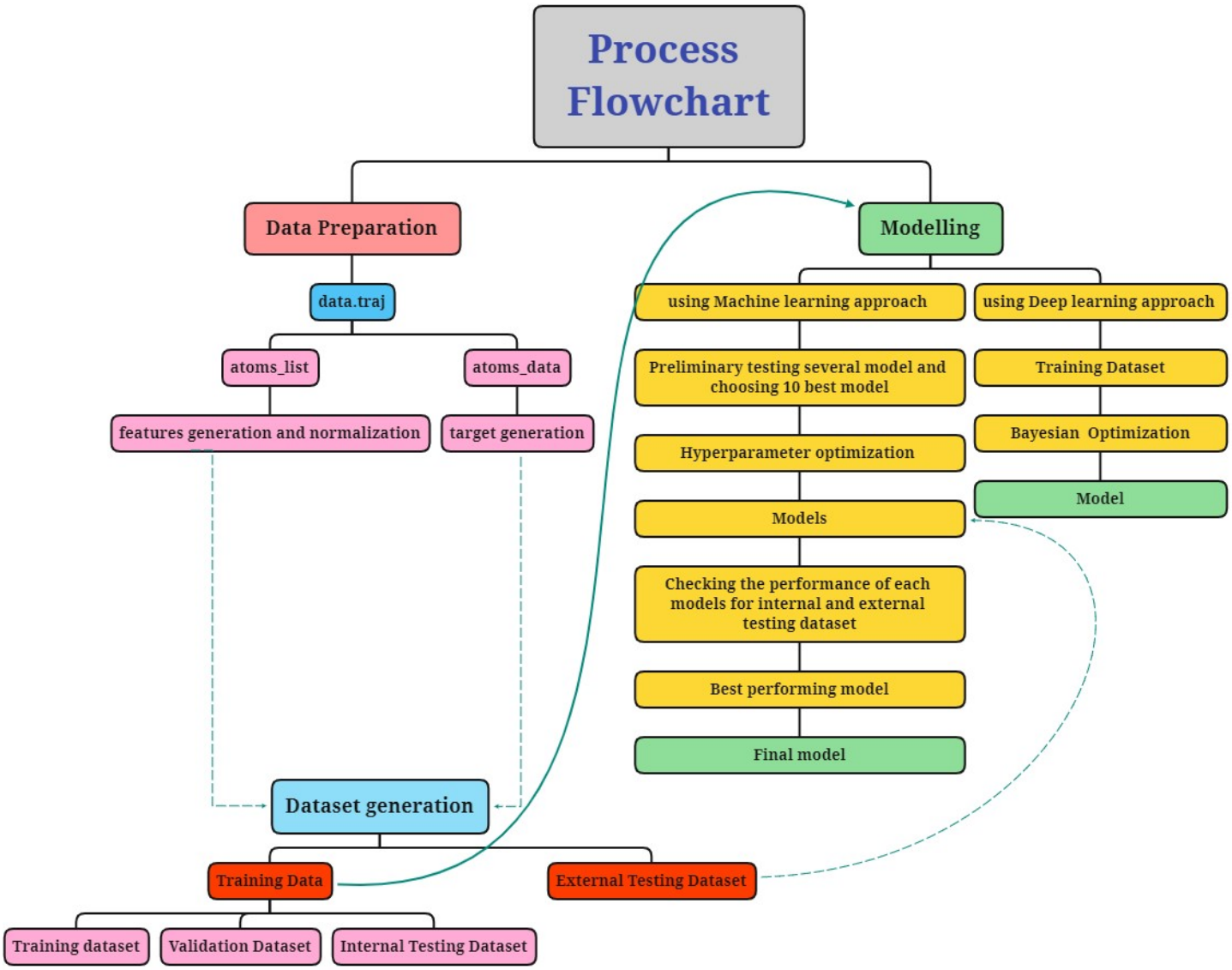


Fig. 2: a) Task workflow

III. Results and Discussion

a. Machine learning approach

Firstly, a quick preliminary comparison of several different machine learning models is summarised in Table **??**. Taking the comparison, I selected 5 models viz Random Forest Regressor(RF), NuSVR, Gradient Boosting Regressor(GBR), Decision Tree Regressor(DTREE), and Adaboost with Decision Tree (ADA_DTREE) models to train. After training the models, the performance of model was again checked for internal data as well as external data. and the result is summarised in (Table:1a). As Random Forest performed best among other model, I used Random Forest to hyperparameterize it further to improve it prediction efficieny. The result of the prediction is summarised in Table (Table:1b)

Table:1

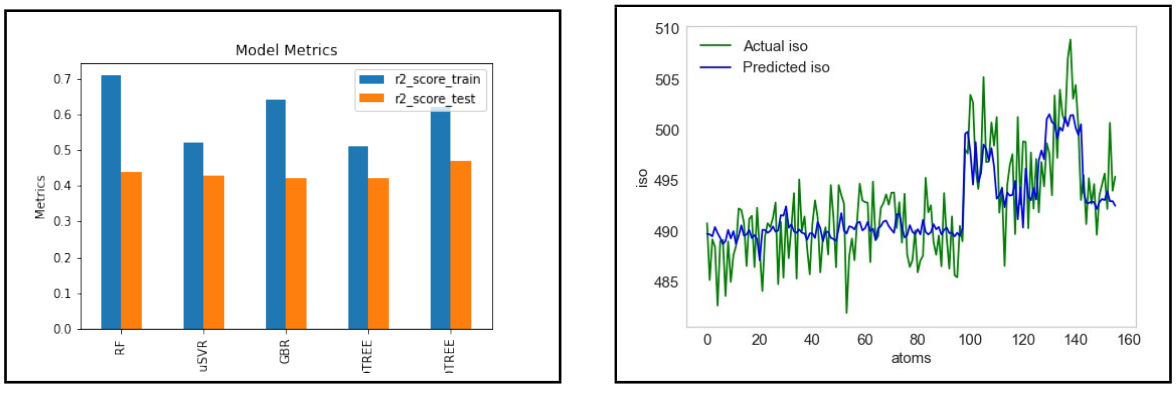
Model	R-Squared
NuSVR	0.44
GradientBoostingRegressor	0.44
SVR	0.43
AdaBoostRegressor	0.42
KNeighborsRegressor	0.41
ExtraTreesRegressor	0.41
XGBRegressor	0.41
Ridge	0.41
LinearRegression	0.41
TransformedTargetRegressor	0.41
PoissonRegressor	0.41
RidgeCV	0.41
HuberRegressor	0.41
BayesianRidge	0.41
HistGradientBoostingRegressor	0.4
SGDRegressor	0.4
ElasticNetCV	0.4
RandomForestRegressor	0.4

a) Comparison of top performing models

Actual_iso	Predicted_iso	percent error
490.77	489.72	0.21
485.17	489.67	-0.93
489.18	489.48	-0.06
488.46	490.4	-0.4
482.65	489.8	-1.48
489	489.31	-0.06
489.13	488.72	0.08
483.6	489.01	-1.12
488.98	490.12	-0.23
484.98	489.27	-0.89
487.62	489.99	-0.49
488.53	488.74	-0.04
492.23	489.51	0.55
492.08	490.56	0.31
490.83	489.55	0.26
486.55	489.66	-0.64
491.19	490.11	0.22
491.5	489.31	0.45

b) Comparison of predicted vs actual iso value by Trained Random Forest models

Table:2



(a) Comparison of performance of tops ML models **(b)** Predicted iso is ploted against actual iso using Random Forest

Fig. 3: Comparative analysis of algorithms prediction ability and correspondingly predictive values are plotted against actual values.

As we can see in the Table:2a, Random Forest best performed against Decision Tree, Gradient Boosting and NuSVR. And the predicted iso values in Table:2b correlates well with the actual values using Random Forest machine learning algorithm. It can be seen that Random Forest perfectly generalise the data. There is not case of over-fitting or under-fitting present which proves the supremacy of the Random forest. It is also significant to highlight the fact that the percentage error between prediction values and actual values is less than 1.5% for all the predictions, which is quite amazing. Thus, I can say that the model is working well enough is acceptable as a predictive model.

b. Deep learning approach

Furthermore, After applying machine learning model, I was curious to check the prediction given by Deep learning algorithm to see if there is any improvement in prediction in comparison to machine learning models. I used keras python libary to implement a sequential model which was optimized using bayesian optimization to develop of a suitable architecture of appropriate layers and neurons in each layers.Prediction of iso values using DNN learning gives the percentage error values less than 1% for most of the cases. Deep learning plot for validation loss and mean absolute error against epoch. In Fig.5, the predicted values are well correlated with actual values. Thus, this deep learning model is performing parallel to Machine learning model. This, can also be Final model for predictive task

Actual	Predicted	percent
487.5	487.19	0.06
495.53	495.72	-0.04
483.36	487.21	-0.8
496.15	487.17	1.81
505.19	498.47	1.33
491.15	487.17	0.81
496.91	490.29	1.33
489.04	487.18	0.38
483.75	487.21	-0.72
486.79	487.21	-0.09
487.72	492.3	-0.94
493.25	496.09	-0.58
494.7	487.19	1.52
492.24	487.2	1.02
498.33	497.57	0.15
499.71	498.31	0.28
497.27	498.39	-0.23
496.71	491.28	1.09
494.6	491.1	0.71
492.95	492.42	0.11
500.77	491.72	1.81

TABLE 1: (A)

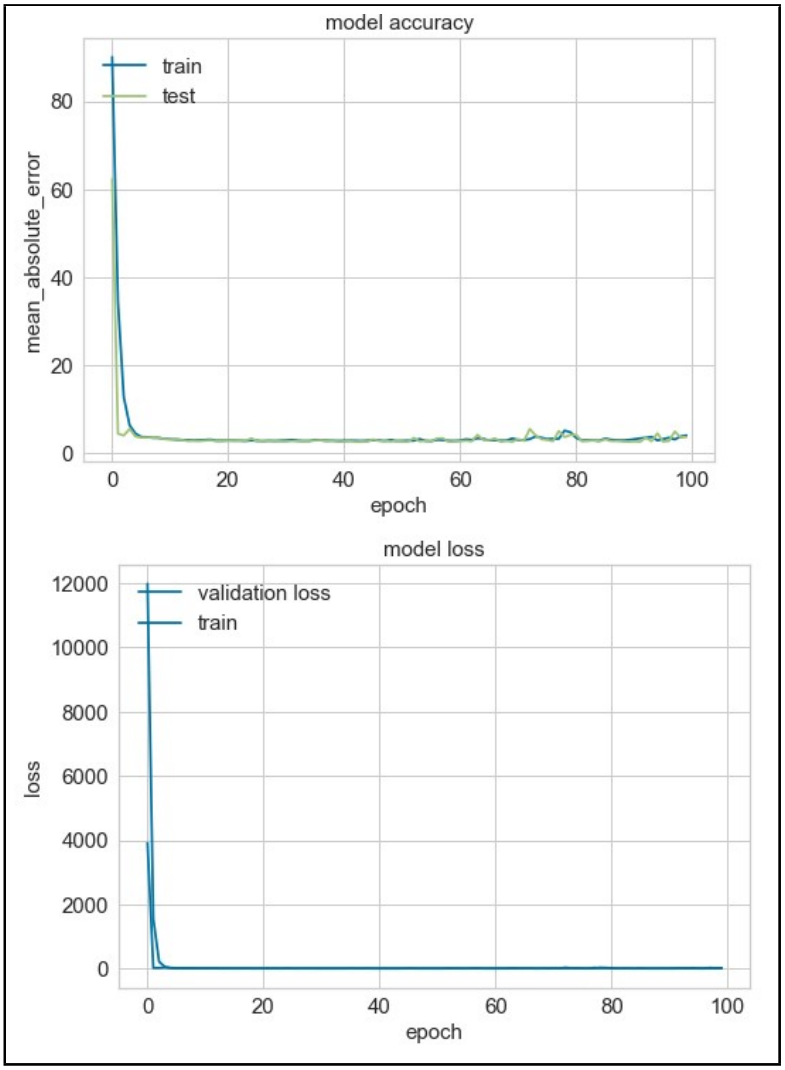
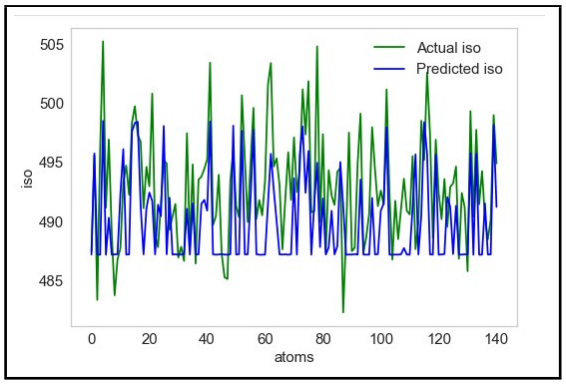


Fig. 4: b

Table shows the percentage error between predicted and actual values, while second chart shows the learning process of DNN model



(a) Correlation plot between predicted and actual values

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (1134, 1120)              28000
dense_1 (Dense)              (1134, 1248)              1399008
dense_2 (Dense)              (1134, 1536)              1918464
dense_3 (Dense)              (1134, 3008)              4623296
dense_4 (Dense)              (1134, 96)                288864
dense_5 (Dense)              (1134, 1888)              183136
dense_6 (Dense)              (1134, 2560)              4835840
```

(b) Predicted iso is ploted against actual iso using DNN

IV. Conclusions

In this article, I prepared a dataset using trajectory data, generated structural features describing the atomic environment of the Al atoms in the structures stored as ASE atoms objects in 'atoms_list'. Futhermore, build a functional predictive model that correlates the structural features with the isotropic shieldings stored in 'atoms_data' by using Machine learning and Deep learning algorithms. Also, quantified the performance of the regressors.

V. Source code of task

The source code(python Jupyter files) for the task is available at github link *"https://github.com/Anandsahujnu11/python-task-prague.git"*

In order to see the Data Preparation step, please open file "Step-1-Dataset-Preparation.ipynb" and to see Modelling step, please open "Step-2-Modelling.ipynb" in the above given git repository.