Particles Detection - Regression Analysis

Politecnico di Torino
Aurona Gashi
Student ID: s322791
Riccardo Racca
Student ID: s315163

***Abstract*—In this paper, an analysis is proposed to predict the position at which a particle is detected when passing through an RSD (Resistive Silicon Detector). The RSD is characterized by a two-dimensional surface composed of 12 metallic pads in the shape of a snowflake.**

I. Problem Overview

The competition presents a regression problem on a dataset containing various features recorded by an RSD sensor, representing the passage of particles through the sensor's surface. The data are collected from 12 pads on the surface. For the purposes of the competition, the dataset has been divided into two parts:

- A *development set* , consisting of 385,500 events (i.e., particle passages), characterized by 90 features and two additional columns that represent the target;
- An *evaluation set*, consisting of 128,500 events with the same features, excluding the target, which needs to be predicted, and also having an additional column id (which will be removed).

The features in question are as follows:

- *pmax[0], pmax[1], ..., pmax[17]:* representing the measured magnitude in mV of the positive peak of the signal;
- *negpmax[0], negpmax[1], ..., negpmax[17]:* representing the measured magnitude in mV of the negative peak of the signal;
- *tmax[0], tmax[1], ..., tmax[17]:* representing the delay in ns of the signal peak compared to a reference time;
- *area[0], area[1], ..., area[17]:* representing the area under the curve of the signal;
- *rms[0], rms[1], ..., rms[17]:* representing the root mean square of the signal.

The target is:

- *(x, y):* representing the coordinates of the particle passage through the sensor.

In the assignment, it is stated that there are six fictitious pads depicting noise due to hardware constraints during data acquisition.
The first phase will be to detect these 6 noisy pads, which will be executed by calculating a measure for feature importance, the most important one will be used to perform a statistical test to determine which distributions are the most anomalous among those present.
Moreover, it is highlighted that there is a complete absence of missing values.

While the problem is stated as a regression problem, the target variables are stored discretely, as can be clearly seen in Fig.1, where is plotted the position of the development targets.

It is noted that for each discrete value of the target variables, there are 100 instances, with 3855 classes.
To assess the goodness of the solution, the chosen measure in the task was the sample mean of the Euclidean distance between bi-dimensional vectors.

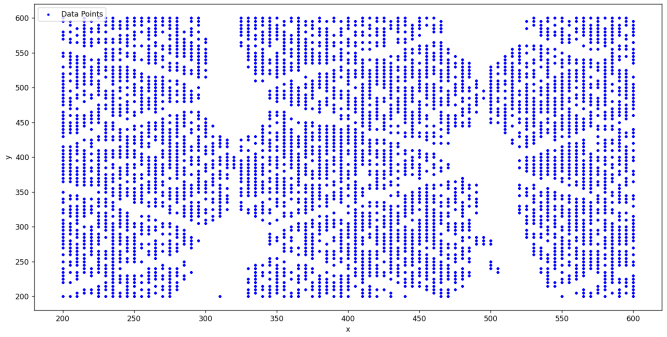

Fig. 1. Target distribution

II. Proposed Approach

A. Preprocessing

The first step of the preprocessing was to determine the feature importance. For clarity, the process is presented using

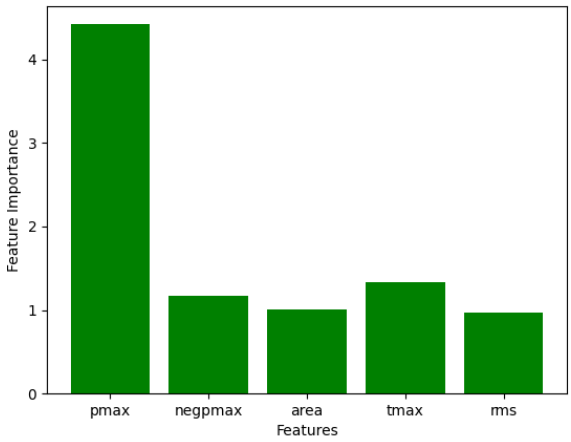

Fig. 2. Feature importance

the *pmax* feature, but the same applies to the others. It was assumed that, during the passage of a particle, the nearest pads were the ones most triggered. Therefore, the six pads that recorded the highest pmax values for each event were selected. Subsequently, only the columns related to these six main pads were retained, including six more columns containing the pads' indices. it resulted in 36 more features. At this point, for each target-class, the "mode" of each index-pad-column was calculated. In other words, for the first column, the mode of the most solicited pad was determined, and the same process was applied to the other five columns.

Consequently, a measure of deviation was computed, in the following way: for each row of a fixed class, a value of 1 was assigned if the index registered in a pad-column was different from the mode of the class in the same column, and 0 otherwise. Obtaining for each row a list of 6 binary entries. Then, the mean was calculated column-wise for the rows within the same class, resulting in a list of six entries with values between 0 and 1, indicating on average how much each column tends to deviate from the mode for a fixed class. So now there are 3855 lists of six entries each. After this, an aggregate column wise-mean was computed among all of the 3855 classes, obtaining a single list of six entries having values between 0 and 1. In the end, the mean of these six entries was calculated.

The lower the value, the greater the importance of the feature in determining the position. Therefore, the importance of each feature was assigned as the inverse of this mean. It occurs that pmax is the most important feature, i.e., it better explains the position for a fixed class. Fig. 2 shows the features' importance found.

Now, it will be drawn the distribution pmax recorded by the various pads, and the Kolmogorov-Smirnov statistic will be calculated to determine how different two distributions are from each other. The six pads that show the greatest differences will be discarded. At the end of this process, the noisy pads were detected and corresponded to the indices: 0, 7, 12, 15, 16, and 17. Fig.3 shows the comparison between the distribution of pmax[1] and pmax[0], the first one occurred to be an actual sensor while the second is categorized as fictitious.

Since dimensionality reduction might improve the processing time and the performance of data mining algorithms further feature exclusions were made. Up to now, six pads have been excluded which translates into thirty features dropped.

The strategy chosen involved the analysis of the results returned by three regression methods. In other words, while keeping their parameters fixed, the models have been trained and evaluated each time with a different subset of features, among: *pmax, negpmax, area, tmax, rms* . The methods in question are Random Forest, SVR, and ML-NN, which unanimously determined that *pmax* and *negpmax* formed the best-performing subset in terms of both performance and time among $2^5 - 1$ possible choices. Where, e.g. *pmax*, referred to the 12 features involving pmax, so the preprocessing concludes with a total of 24 features plus two target columns.

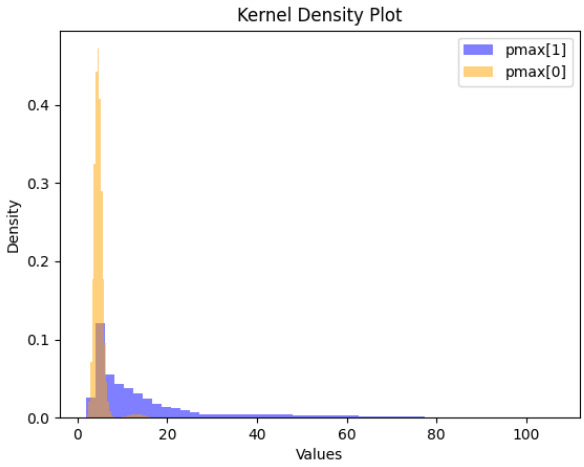

Fig. 3. Density difference between true pad and noisy pad

This approach tries to find irrelevant features i.e. those attributes that do not carry relevant information to be exploited. It is acknowledged that this strategy may have been too simplistic, but in this case, it seems effective in both time and performance.

B. Model selection

Different models have been evaluated:

- **Linear Regression:** chosen for its simplicity, useful in explaining the importance of features;
- **Ridge and Lasso Regression:** variations of linear regression with regularization terms to limit overfitting;
- **SVR (Support Vector Regression):** a regression method based on solving an optimization problem that fits the data by regulating an allowed error;
- **Random Forest Regression:** an ensemble method that uses multiple regression trees to fit the data through voting and limit overfitting;
- **Multi-Layer Neural Network:** a regression method that utilizes a structure similar to that of the human brain, less interpretable compared to others.

C. Hyperparameters tuning

It has been observed that the dimensions of the provided dataset prevent a parameter search within acceptable time frames. Therefore, a representative sampling technique has been proposed as follows: since the response variables are divided into categories, even though the problem and treatment have been framed as regression, this can be exploited for a correct representation of the data through a sample of them. While keeping in mind this observation, it is possible to randomly sample for each category, i.e., for each target-couple (x, y), an integer number ofrows, with m s.t $1 \leq m \leq 100$. These samples will constitute a sub-dataset on which to perform all future parameter selection (where 100 was chosen because the cardinality of each category is 100).

Furthermore, due to the fact that different methods are characterized by different computational complexities and execution times, the chosen value of m might differ between them.

The next step after this sampling was to propose a classical analysis, through the division of the development dataset into a training set (80%) and a test set (20%), where the former is used for the training of the methods and for the selection of parameters, while the latter for evaluation.
For training and setting the correct parameters, only the training set was used, and the process chosen to find the correct parameters was through a cross-validation technique for each combination of parameters.

The idea is to divide the training set into k-folds, where each of them, in turn, becomes an evaluation set while the others form the training set.

Fixing the parameters of the model, it is first trained and then evaluated k-times, one for each fold. After this, the evaluations are combined and this represents a single estimation of the generalization error of the model with the fixed parameters.

After all this, the estimate is compared with the ones obtained from other models with different parameters, and the model with the lowest generalization error estimate is chosen. What has just been described constitutes the underlying idea of the Grid-Search.
For the purpose of conducting a comprehensive analysis, it was necessary to make a trade-off between completeness in searching the parameter space and the computational time required. Therefore, even though a complete grid search might be more precise, a similar technique called Halving Grid-Search was opted for.
The idea behind this technique is very intuitive: it conducts a stratified grid-search, trying to prune the combinations of parameters that perform worse as early as possible, avoiding processing models that, in the initial evaluations, have led to worse results. This process is carried out in a stratified manner because, in the initial stages, to save computational complexity, the models are evaluated on a reduced subset of the dataset. Those with worse evaluations indicate the parameters to prune. With each iteration, the dataset is doubled until a model is obtained.
This technique allowed for the selection of a pseudo-optimal configuration in less time and across a larger parameter space. However, it is believed that the complete grid-search provides more certainty in the parameter selection phase. Therefore, the Halving Grid-Search was applied only in the parameter selection for the Multi-Layer Neural Network, where a deep search was needed.
The parameter search space for the various models is shown in the table I.

TABLE I
HYPERPARAMETERS TUNING

Model	**Parameters and Values**
Linear Regression	intercept: {True, False}
Ridge Regression	intercept: {True, False}, α:{1, 5, 10}
Lasso Regression	intercept {True, False}, α:{1, 5, 10}
SVR	C :{1, 10, 100, 200}, kernel: {'rbf'}, gamma = {'scale', 'auto}, epsilon:{0.1, 0.5, 1},
Random Forest Regression	n_estimators: {50, 100, 150}, criterion:{'squared_error','friedman_mse'}, max_features: {'sqrt'}, min_samples_leaf: {1, 3, 5}
ML-NN (1 hidden layer)	hidden layer sizes: {50, 100, 150, 200}, activation: {'logistic', 'tanh', 'relu'}, learning rate: {'constant', 'invscaling', 'adaptive'} alpha: {1e-5, 1e-4, 1e-3}
ML-NN (at most 2 hidden layers)	hidden layer sizes: {(150,), (200,), (50, 50), (100, 50), (50, 100), (100, 100)}, activation: {'logistic', 'tanh', 'relu'}, learning rate: {'constant', 'invscaling', 'adaptive'} alpha: {1e-5, 1e-4, 1e-3}
ML-NN (at most 3 hidden layers)	hidden layer sizes: {(200,), (50, 100), (100, 50), (100, 100), (50, 50, 50), (50, 50, 100), (50, 100, 50), (100, 50, 50)}, activation: {'logistic', 'tanh', 'relu'}, learning rate: {'constant', 'invscaling', 'adaptive'} alpha: {1e-5, 1e-4, 1e-3}

The best parameters obtained from the grid search will be used to construct the models finally evaluated with the following formula:

$$d = \frac{1}{n} \sum_{i} \sqrt{((x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2)}$$

measured in μm.

Once the optimal parameters have been obtained, a final phase of aggregating the results from various methods was proposed. The technique in question is called "ensemble", which demonstrates significant performance improvements compared to individual models. The final result is obtained by calculating a weighted average of the different results proposed by individual methods[1]. In detail, the method is constructed as follows: first, the development dataset is divided into a training set and a test set; the former is further divided into a training set and an evaluation set. In the training set, a series of models to be combined are trained. Using the evaluation set, the individual performance of the methods is assessed, which provides the weights for the convex combination. Indeed, the lower the error of a model, the higher its importance in the combination. Therefore, the coefficient for each model has

been chosen as $\frac{1}{error^k}$. As k increases, more accurate models become increasingly influential in the combination.
At this point, this series of methods can be applied individually, one by one, to the test sets, saving the results they provide and convexly combining them with the weights just found to obtain the final result, which will be then evaluated on the true label of the test set.

III. Results

In Table II, the best parameters suggested by the grid search for various models are reported, together with their results once applied to the test set. The parameter for execution time feasibility m is also specified.

TABLE II
BEST HYPERPARAMETERS AND RESULTS

Model	**Parameters**	**Results**
Linear Regression	intercept: True	17.023 (m=100)
Ridge Regression	intercept: True, α: 1	17.672 (m=100)
Lasso Regression	intercept: True, α: 1	17.049 (m=100)
SVR	C:200, kernel: 'rbf', gamma: 'auto', epsilon: 0.5	4.843 (m = 30)
Random Forest Regression	n_estimators: 150, criterion: 'squared_error', max_features: 'sqrt', min_samples_leaf: 1	4.952 (m = 100)
ML-NN (at most 3 hidden layers)	hidden layer sizes: 200, activation: 'logistic', learning rate: 'constant'	4.337 (m=100)

Given the clear inferiority of the Linear, Ridge, and Lasso Regression methods, they will not be part of further evaluations in the aggregation phase. The results obtained with the aggregation are reported in Table III.

TABLE III
ENSEMBLE RESULTS AND TIME EXECUTION

Ensemble	**Results**	**Results with approx.**	**Time (s)**
RF	4.940	4.971	2487
RF + ML-NN	4.298	4.286	6343
RF + ML-NN + ML-NN	4.287	4.241	9824
RF + ML-NN + ML-NN + SVR	4.276	4.204	13236

IV. Discussion

The various applied methods yielded good results, but the decision to aggregate them, as seen in the graphs and tables above, improved the performance. The example provided in Table III is a simplified set to illustrate the method. For better performance, other models have been added or removed. Specifically, the best result uploaded was obtained by aggregating: three Random Forests, five Multi-Layer Neural-Network and two SVR, fixing $k = 200$, obviously changing each time the random seed. The dataset's peculiarity on the target variables almost reduces the task to a classification problem. Therefore, it is valuable to note that if discretization is applied to the obtained results, they yield a better outcome, as shown in Table III. However, it should be noted that this makes the model less applicable if certain data detection conditions change. Despite achieving about a tenth of benefit on the leaderboard, its usage is not recommended unless there are clear similar conditions.
To perform as better as possible in the challenge, two results are submitted: the non-approximated and the approximated target predictions, both obtained by the above mentioned ensemble.
This analysis is stable in terms of error evaluation, as evidenced by the proximity of the model evaluation and the error reported on the leaderboard. Therefore, the proposed approach is considered a reliable method, and this concludes the analysis.

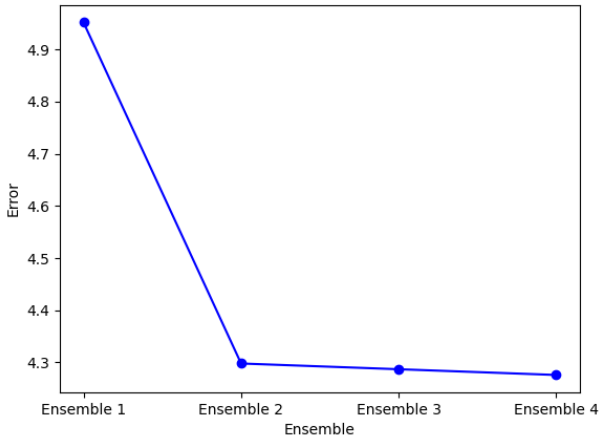

Fig. 4. Ensemble errors

V. References

[1] P. N. Tan, M. Steinbach, A. Karpatne, V. Kumar. *Introduction to Data Mining (2nd edition)*. Pearson Education, 2019.

[2] M. Mohri, A.Rostamizadeh, A. Talwalkar. *Foundations of Machine Learning (2nd edition)*. The MIT Press, 2018.

[3] D.P.Kroese, Z.I.Botev, T. Taimre, R.Vaisman. *Data Science and Machine Learning*.

[4] G.James, D. Witten, T. Hastie, R. Tibshirani *An Introduction to Statistical Learning (2nd edition)*.