Report title

First and last name
Politecnico di Torino
Student id: s123456
email@studenti.polito.it

***Abstract*—RSD is a sensor that tracks the position of passing particles. Throughout this report, we propose a data science pipeline that uses as input several features extracted by the RSD signals, removes the outliers and the noise, and extracts the maximum positive peak and the relative values with respect to it. The preprocessed data is provided to three regression models. The results exceed a naive baseline and the performance of the same models on the original data.**

I. Problem overview

The group project consists in predicting the positi
a particle when it traverses an RSD sensor. The sen
flat, hence the position is represented by (x, y) value
task has to be performed through a multi-output regr
pipeline, one per coordinate.
The Resistive Silicon Detector(RSD) sensor is compo
12 metallic *pads*. They have an asterisk-shape and th
be clearly distinguished in Figure **??**.
Each pad records a signal that is transferred and stor
the system. Due to hardware constraints, not all the me
transferred are meaningful, but 6 of the 18 readings are

We define as *event* the transit of a particle throu
sensor. The dataset is composed of 514,000 events d
into: - 385,500 in the *development* set - 128,500
evaluation set

All the fields have a not null valid value. Each
contains some features of an event extracted from
signal provided by the RSD sensor. They are: - pma
value of the positive peak of the signal in mV - neg
the value of the negative peak of the signal in mV - are
value of the area under the signal - tmax: the time between a reference moment and the positive peak in ns - rms: the root mean square of the signal The signal is represented as a number inside the square brackets at the end of the column name. Therefore, "pmax[0]" means the feature pmax obtained by the signal 0. We are going to use the same notation throughout the report.
The position of each event is enforced during the experiments and it is also provided in the development set. On the other hand, the evaluation set contains only the identifier of the events.

The algorithms have to exploit the information provided by the development set to predict the position of the events present in the evaluation set. Then, the results are submitted to an online platform, where they are evaluated according to the average Euclidean distance (1).

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

The development set can be used to make some analysis, in particular, according to the position.
A first observation is that for every (x, y) present in the

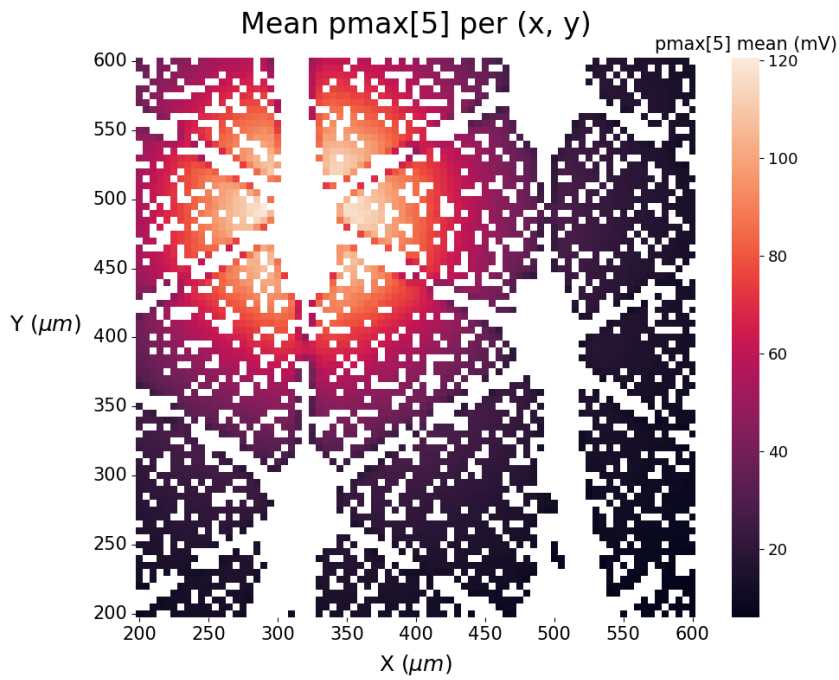

Fig. 1. Heatmap pmax[5]

records, there are 100 events. The x and y values present are all the integers in the range $(200, 600)$ with step 5. Furthermore, not all the possible (x, y) combinations are present as reported in Figure 1. In particular, the positions of the pads can be identified by the complete absence of events. Only the shape of a few of them is complete because the positions collected are only of the central zone of the sensor. This is because when a particle hits the sensors, it does not pass through it and the event is not valid.

Figure 1 represents the mean pmax aggregating by (x, y). Comparing the heat maps of the average pmax of different signals was useful for discerning the noise. An example is the

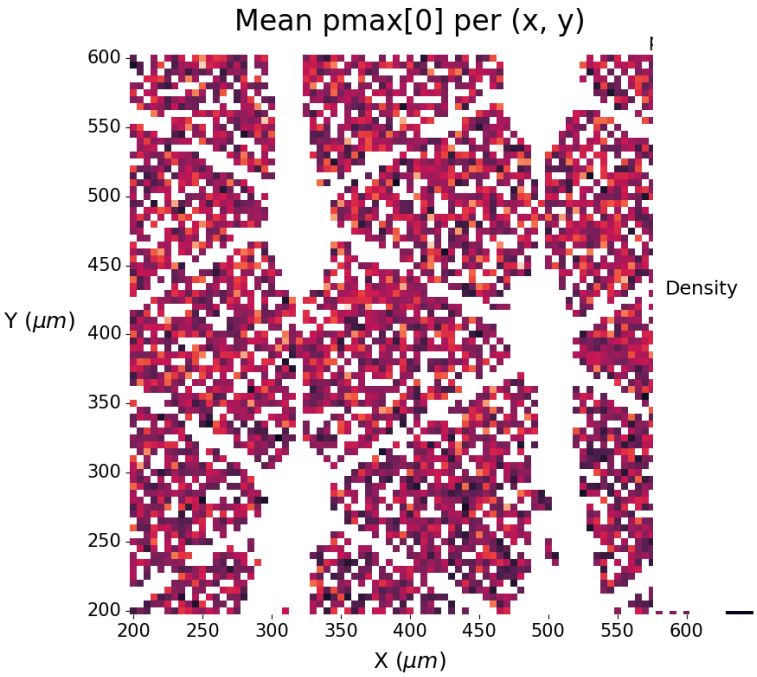

Fig. 2. Heatmap pmax[0]

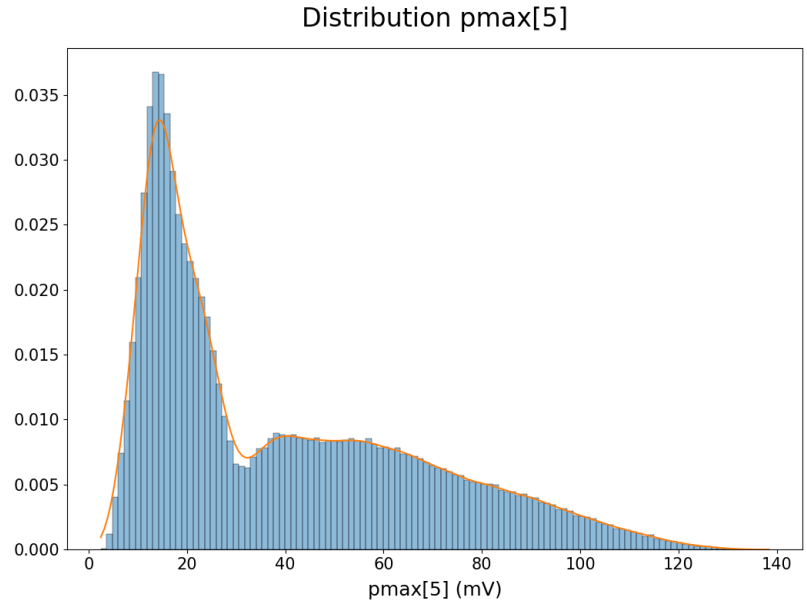

Fig. 3. Distribution pmax[5]

difference between the figures 1 and 2. The latter is
noise.
The same heat maps were used to identify th
corresponding to each signal. The mean value of a s
pmax for a given (x, y) is high only if the passing p
of the particle is close to the pad. For instance, tl
corresponding to signal 5 is the one surrounded by
encoded with brighter colors in Figure 1. The pad a
corresponding signal number are represented in Figure

The difference between the noise and the signal
also observed in the distributions of the values. A
pmax probability function of a pad is represented in
3. The noise signals have completely different shap
range of values. Most of the values have a low mag
This occurs because most of the time the pad is distan
the passing position.

The same analyses were performed on the other types presented in the listing I. Similar results were obtained for negpmax, area, and tmax. This confirmed that the features that share the same trailing "[n]" derive from the same signal. The tmax fields give less clear visualizations than the other three. On the other hand, the heat maps of the rms values do not present any visual pattern and the values are in the order of a few mV. The reason is that only a small portion of the signal can be characterized by peaks. Most of the time, the signal has a small magnitude of random noise. As a consequence, the latter part prevails in the computation of the rms.

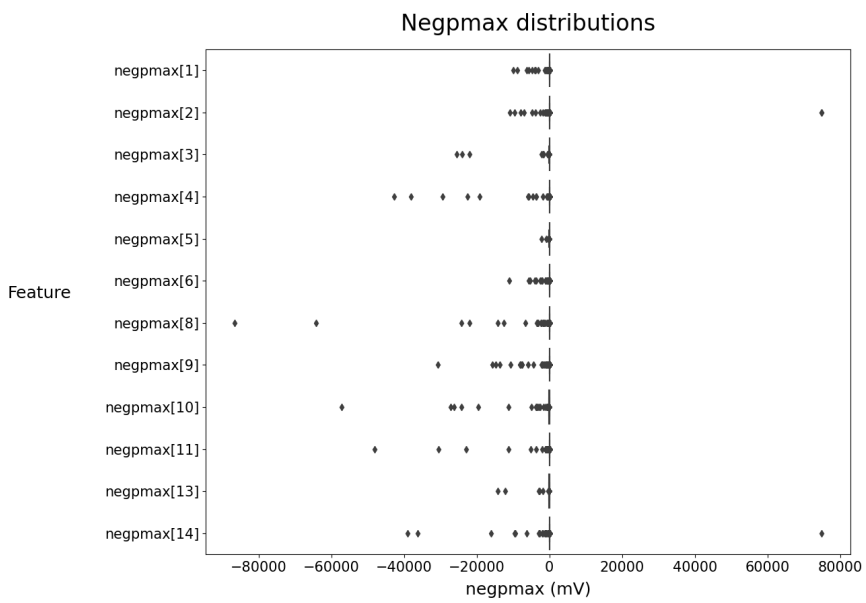

Fig. 4. Negpmax boxplots

Another consideration that can be made is on the distributions of negpmax. Its density functions have an unusual number of outliers compared to the other features(see Figure 4).

II. Proposed approach

A. Preprocessing

We have seen from a visual perspective, which are the features that contain most of the information. These

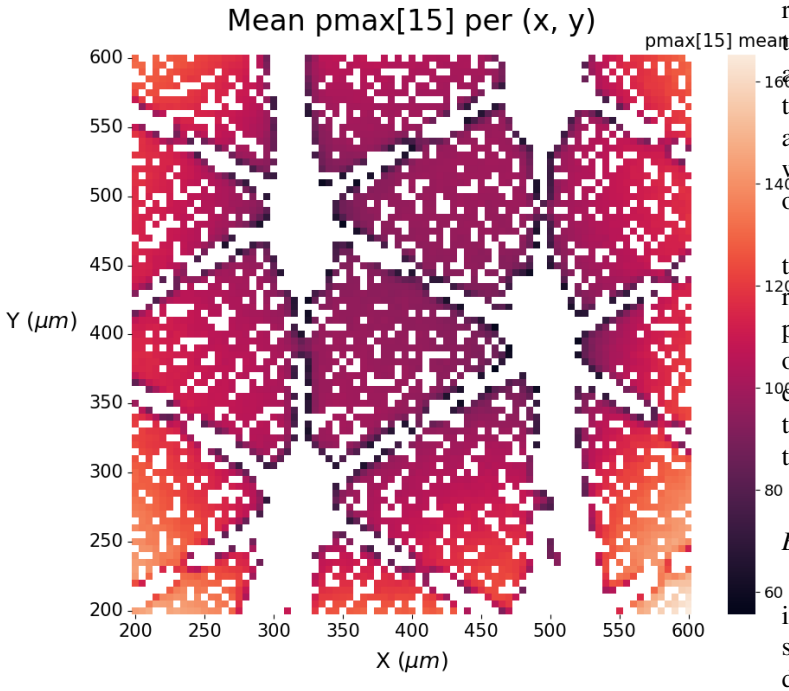

Fig. 5. Heatmap pmax[15]

intuitive considerations were confirmed by an analysis of the importances obtained by a Random Forest regressor with default parameters. Only the pmax, negpmax, and the area of the signal derived from the pads were significantly used by the algorithm.
An interesting exception is the signal 15. Even if it should be only random noise, the $pmax[15]$ is taken into consideration. Observing Figure 5, we see that the values are on average higher on the borders of the sensor and lower near the metal of the pads.
For these reasons, we kept only pmax, negpmax, and the area of the signal corresponding to a pad and the feature $pmax[15]$.

Another important aspect to take into consideration in this phase is the presence of outliers in the training set, from the distributions analysis we could notice that only in the columns of "negpmax" there were the presence of values completely out of scale respect to the other data in the same distribution, considering the importance of the features we decide to proceed with the removal of the outliers in this case. We did not do the same thing for other columns because from the distributions we could not notice evident outliers, so there wea the risk to remove some values that were not common but were not even noise, values that could help in the training of the regressor. Seeing that the value of negpmax could space in a broad range, close to zero if the paritcle hit was very far from the sensor, or viceversa in the opposite case, we had to be carefull on labeling records as outliers. To avoid the removal data that could be usefull, we used the following method, first we detected and saved the minimum "negpmax" for each row, then we analysed this values using the Tukey's Fences technique for outlier detection, this method detects otuliers and extrime outliers, since the number of outliers, in relation to records in the training set, was susbstantial, and removing all of them was not improving the quality of the predictions, we decided to remove only the outliers detected as extreme outliers.

In Section I, we have seen how the closer the position to a pad, the higher the corresponding pmax value. For this reason, we introduced a new feature that is the maximum pmax for each event. Then, we added the normalized value of every pmax by the maximum pmax of that event. This is done because this information provides the algorithm a way to infer how near is the position to a pad in comparison to the closest.

B. Model selection

We have tested the following models: - Random forest(RT): it is an ensemble machine learning technique used for classification and regression. During the training phase, several decision trees are created on different random datasets, sampled with replacement from the original data. For a regression problem, the output of the random forest is the mean of the predictions of the single decision trees. Every split is learned during training considering a criterion, e.g. mean squared error, and possibly on a random subset of the features [**?**]] [**?**].
The usage of multiple decision trees leads to a model more robust to noise [**?**]. The performance and the training times depend on the number of decision trees. The improvement provided by an increase in the number of decision trees is significant up to a certain number [**?**].
Even if the model is not interpretable as a decision tree, the overall importance of the features can still be obtained.

- Extra-trees regressor(ET): it is an ensemble machine learning technique. There are only two main differences with the random forest method. The first is that every decision tree is built using the whole learning sample. The second is that the split is randomly selected from a uniform distribution inside the candidate feature's empirical range. Then among all the random splits, the best one is chosen and used to grow the tree.
The computational efficiency of the algorithm is an important advantage of this model [**?**].

- Voting regressor: it is an ensemble machine learning technique. It is based on the simple idea of using the average of different regression models. In this way, the advantages of multiple models can be combined. In this case, we used the mean of the random forest and the extra tree regressors.

C. Hyperparameters tuning

The tuning was performed on the hyperparameters of the RT and the ET. As a consequence, the voting regressor used the best performing configurations of the other two models.
The RT and the ET algorithms share all the parameters we considered for the tuning. They are: - the number of estimators

n_estimators - the number of features considered at each split *max_features* - the maximum depth of each decision tree *max_depth* - the splitting criterion *criterion*

Over a certain threshold, the increase in the number of decision trees does not improve significantly the performance, while extending considerably the computational time [**?**].// We defined that for this problem it is not worth using a number of estimators over 130.
For this reason, we tested using a grid search values of n$_estimatorsnear100$.

The maximum depth obtained by the RT and the ET with default parameters is respectively 38 and 41. We tested the configurations with max_depth:None, 60%, and 80% of the previous values.

We divided the development dataset 80/20. On the 80The 20

III. Results

The tuning showed that for RF and ET the best configuration of hyper-parameters is very similar, the only difference was in "max features", 1.0 for ET and "sqrt" for RF. The best configuration for RF was:

- n_estimators: 100
- criterion: "squared_error"
- max_features: "sqrt"
- max_depth:None

The best configuration for ET was:

- n_estimators: 100
- criterion: "squared_error"
- max_features: 1.0
- max_depth:None

max_depth: None means that the growth of the trees in the regressors are not limited. We did the tuning on the training test using cross-validation, then we tested the results of the two regressors, using the best parameters configuration, on the test set, the distance obtained for the RT was 4.261,3.907 for ET , 3.990 for VR With the configuration obtained, we trained the VR on the full development set and used it to label the data contained in the evaluation set, the score obtained on the public scoreboard was 6.637.

Dring the discussion every time we refer to local tests, they are test done splitting the development set and doing the training on 80% of it and the testing on the 20% of it. Instead to label the evaluation set and upload our submission on the public scoreboard we trained the regressor on 100% of the developement set The feature selection process brought an important improvement to our solution, after the removal of the features that we decided to discard during the features analysis, our solution went from 5.039 to valore. Adding then the maximum pmax and the normalized pmax for each event improves our solution locally and on the online score board, the results we obtain without them are while addigng them we obtained With the development set that we obtain after pre-processing, and with the best set of hyper-parameters, the ET regressor alone, obtains lower average distance on the test set respect to the RF alone and the VT , the solution for the three of them are 3.90 for ET, 4.26 for RF and 3.99 for VR, but when we do our tests online we can see that the best score is obtained with VR, and it is valore. We may assume that this happens because VR take the advantages of the regressors, that are very similar in this case, and merge them, in this way it works better when in the evaluation set it finds data that were not in the training set. We also defined a naive regressor to compare our solution with it. The naive solution just predicts the average between xmax and xmin, where xmax is the maximum value of x in the development dataset and xmin is the minimum, and it does the same reasoning for the y. The solution of the our regressor and the naive one are respectively valore, valore. Finally we can notice that our proposed solution it is considerably better than the baseline that we can find in the scoreboard, the difference between the two is 1.992. If we compare our result with the ones obtained by other grups on the scoreaboard, we can see that we are on the first third of the table, for sure there is room for improvement considering that the solutions in the first positions are significantly superior.

IV. Discussion

From the results we obtained we can see that the predictions done with the VR are closer to the solution than the ones we obtained using the algorithms separetly, this is what we expected from the voting regressor, that with similar algorithms is supposed to exploit the advantages of them and obtain better results. Figure A permit us to show also that the position near the metal bar of the pads were the ones that were harder to predict, in those position we can see that the error is bigger if compared to others not so close to the pads. To seek a solution that improves the proposed one it is possibile to expand the grid search testing more configuration for the attributes on which we focused, it could be also interesting to add more attributes to the grid search. Different and improved results could be found trying various technique, methods that exploits other Machine Learning regressor that we did not explore. Also additional domain knowledge, like pyhisical consideration on the particle or on the sensonr, could help during the feature extraction. In the end, taking as reference the baseline, we are satisfied by the solution obtained using our approach.

References