Particles Positions in an RSD

Nicola Bavaro Lorenzo Gargasole
Politecnico di Torino
Student id: s328790 s331409
s328790@studenti.polito.it s331409@studenti.polito.it

***Abstract*—This report presents a method to obtain the coordinates of the positions of different particles passing through a RSD (Resistive Silicon Detector), analyzing a data set of different type of measurements. To achieve that, we will do some preprocessing on our data, trying to understand which features are useful for our predictions. Lastly we will build a regression model in order to predict the coordinates of particles based on their measurements**

I. Problem overview

The dataset includes informations about 18 readings of 5 macrofeatures obtained from each of the 12 sensor pads. In particular, the features of each record specify information about the wine's:

• pmax: the magnitude of the positive peak (mV)

• negpmax: the magnitude of the negative peak of the signal (mV)

• tmax: the delay from a reference time when the positive peak of the signal occurs (ns)

• area: the area under the signal

• rms: the root mean square (RMS) value of the signal

The dataset is divided into two parts:

• A development set, which contains 385500 records. Each record is labelled with the coordinates

• An evaluation set, which contains 128500 records. This portion of the data doesn't include the information about the position of particles

We intend to construct a model only utilizing data from the development set, and our evaluation will be based on the separate evaluation set. This approach ensures that we avoid the risk of developing an overfitting model. All features in our dataset are numerical; therefore, encoding is not necessary. Upon examining the data, we observe that there aren't features with missing data. It is important to note that we have 18 measurements whereas only 12 pads are present in the sensor, this happens due to hardware constraints during the data acquisition phase. Consequently, a subset of the 18 features does not consist of actual readings but rather noise. In preprocessing, we will take care of analyzing the data to eliminate the less significant features and the noises. We performed a thorough analysis by visualizing the x and y coordinates, revealing spatial patterns. Upon closer examination, specific points were identified where the presence of particles appears less distinct. Importantly, these points correlate with the positions of measurement pads.

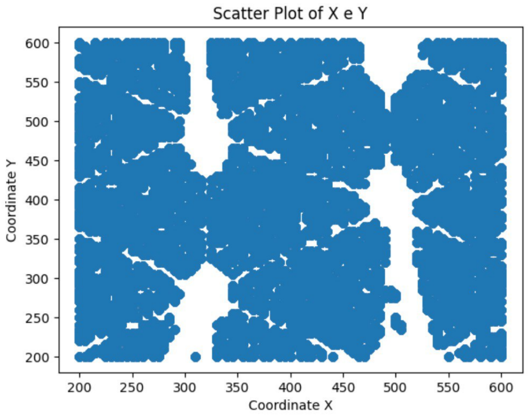


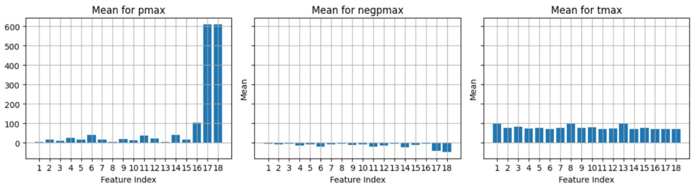
Fig. 1. Scatter plot of positions

II. Proposed approach

A. Preprocessing

Our regression analysis involves studying 90 different aspects derived from 18 measurements each for 5 main features (pmax, negpmax, tmax, area, rms). To make our analysis more accurate, we need to carefully prepare the data. This report focuses on the early steps we took to make our dataset cleaner and more robust. We mainly looked at the average values and how much the data varies, and we used this information to decide which features are important for our analysis.

a) Average Analysis: At first, we looked closely at the average (mean) values of each feature. The average helps us to understand where most of the data is centered. Some features, like tmax[0] to tmax[17], and pmax[15], pmax[16], pmax[17], area[15], area[16], area[17], had much higher averages than the rest. We thought these might affect our analysis, so we decided to leave them out and focus on the others. This step helps us keep our dataset in good shape for further analysis.



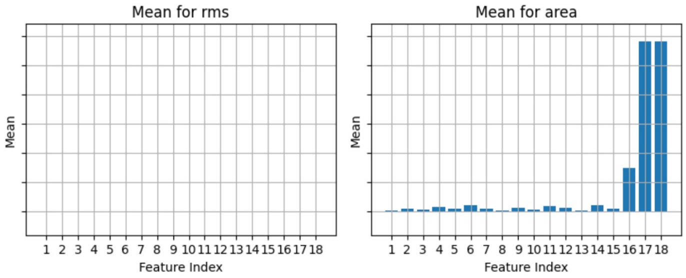


Fig. 2. Means of all features

b) Standard Deviation Analysis: After looking at the averages, we also considered how much the data varies, looking for Standard Deviation. This helps us understand how spread out the data is. The root mean square (rms) features (rms[0] to rms[17]) turned out to not vary a lot compared to the other features. Since rms is not directly related to the signal we're studying, we decided to remove these features from our dataset. This step refines our selection, making sure we focus on the most important features for our regression task.

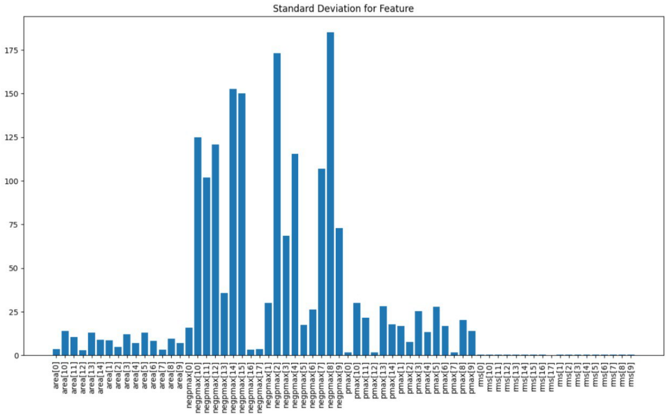


Fig. 3. Standard Deviation of remaing features

c) Boxplot Analysis: We made the decision to exclude pmax[0], pmax[7], and pmax[12] from our dataset after a careful examination of boxplots. The diminutive size of the interquartile range and the small box sizes suggested minimal variability, leading us to categorize these instances as noise values. This rigorous selection process aims to refine the dataset by eliminating features with limited informative content and enhancing the overall robustness of our analysis.

d) Correlation Matrix Analyisis: In the course of the correlation matrix analysis, we conducted a careful evaluation of the relationships between different features. Based on this analysis, we made the decision to eliminate some specific features, namely 'area[0]', 'area[1]', 'area[6]', 'area[7]', 'area[9]', 'area[12]', and 'area[14]'. This choice was motivated by the observation that these particular features show significant correlation with other features already present, suggesting possible overlapping information

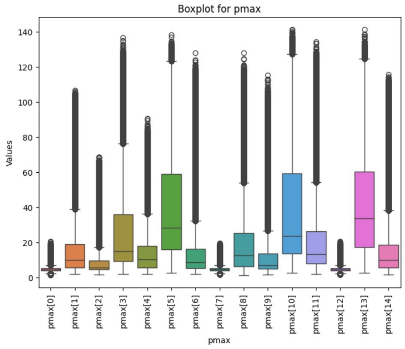


Fig. 4. Boxplot for pmax

B. Model selection

The regression models that have been tested are:

• *Linear regression*: stands out as the simplest and fastest method in the set. However, it is susceptible to the influence of outliers, which can affect its performance.

• *Random Forest*: This technique employs multiple decision trees for regression tasks. It exhibits robustness in the presence of outliers and is well-suited for handling sparse data. Yet, it tends to display bias toward features with high cardinality.

• *Ridge*: Utilizing an L2 regularization technique, this model optimizes by reducing the overall size of weight values. This regularization aids in curbing overfitting, enhancing the model's generalization.

• *Lasso*: Employing an L1 regularization technique, the Lasso model sets some feature weights to zero, effectively excluding those features from the model. This feature selection property helps in simplifying the model and reducing overfitting.

C. Hyperparameters tuning

Firstly, we choose the hyperparameters related to Linear Regression, then for Ridge and Lasso. Lastly, after observing that none of these types of regression models provided the best results, we selected the hyperparameters for Random Forest. All the hyperparameters were determined through a Grid Search. This process is useful for finding the optimal combination of parameters that yields the lowest average Euclidean distance. We obtained different distances for each possible parameter combination. Subsequently, using the best parameters that produced the optimal result, we conducted tests for each model listed in Table 2.

TABLE I

Model	Parameters	Values
LinearRegression	*fit_intercept*	{*True*, *False*}

Ridge	*fit_intercept*
	alpha
		{*True*, *False*}
		{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1}

Lasso	*fit_intercept*
	alpha
		{*True*, *False*}
		{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1}

Random Forest	*n_estimators*
	max_features
	criterion
		{100, 200, 250}
		{*sqrt*, *log2*}
		{*squared_error*}

III. Results

The Random Forest algorithm emerged as the frontrunner, delivering the best outcomes in both public and local evaluations. On the other hand, Ridge, Lasso, and Linear Regressor algorithms demonstrated similar average Euclidean scores during validation, but these were notably higher than the scores achieved by Random Forest. The optimal configuration for Random Forest was identified with hyperparameters set to $\{$*n_estimators*=250, *max features*=*sqrt*, criterion=*squared_error*$\}$, resulting in a public score of 4.85. This achievement surpassed significantly the naive baseline. The superiority of Random Forest over other regression models underscores the critical importance of meticulous algorithm selection and parameter fine-tuning to achieve optimal results in the given context.

IV. Discussion

Despite the excellent performance achieved using the Random Forest algorithm, it is important to note that a considerable number of features remain in our dataset, and therefore, there is still room to explore and further improve the performance of the model.

The Random Forest, while a powerful algorithm, may have reached its limit in terms of significant improvements. Exploring alternatives, such as using other types of regressors, might be a direction to consider. However, it should be noted that SVM (Support Vector Machine) may not be the most suitable choice given the complex nature of the problem.

One option to explore could be the use of neural networks. Their complex learning capability could prove advantageous in a context where the relationship between features and particle coordinates is highly intricate. At the same time, the issue of the large number of features in the dataset must be addressed. The use of PCA (Principal Component Analysis) could be a strategy to reduce dimensionality and simplify the model. However, it is critical to carefully evaluate the impact of this reduction on the accuracy of the predictions, as PCA could result in the loss of crucial information. Careful design of the network architecture and a careful training phase could lead to significant results.

References

[1] Scikit-learn library: Principal Component Analysis (PCA). https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html

[2] Scikit-learn library: Multi-layer Perceptron Regressor (MLPRegressor). https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPRegressor.html