SAPIENZA
UNIVERSITÀ DI ROMA

Machine Learning a.y. 22-23

Homework 1: Report

Davide Gabrielli - gabrielli.1883616@studenti.uniroma1.it

December 9, 2022

1 Introduction

In this report I will describe the implementation for the given two tasks about classification and regression about collisions of drones in a 2D environment.

1.1 Dataset

The dataset is the one provided by the course, it contains 1000 samples of drone collisions in a 2D environment. The dataset contains the following features:

- **For 5 UAVs** (where i is the UAV number)):
  - **UAV_i_track**: clockwise angle from north between the ith UAV and its target (0, 2*pi)
  - **UAV_i_x**: x coordinate of the ith UAV in meters
  - **UAV_i_y**: y coordinate of the ith UAV in meters
  - **UAV_i_vx**: x velocity of the ith UAV in m/s
  - **UAV_i_vy**: y velocity of the ith UAV in m/s
  - **UAV_i_target_x**: x coordinate of the ith UAV target in meters
  - **UAV_i_target_y**: y coordinate of the ith UAV target in meters
- **num_collisions**: number of collisions in the sample
- **min_CPA**: minimum CPA in the sample (An estimated point in which the distance between two objects, of which at least one is in motion, will reach its minimum value)

The dataset is unbalanced, as it contains 1000 samples divided into 5 classes (0, 1, 2, 3, 4) that represent the number of collisions in the sample, as shown in (Fig. 1). This is also due to the fact that the distribution of the number of collisions is not uniform.

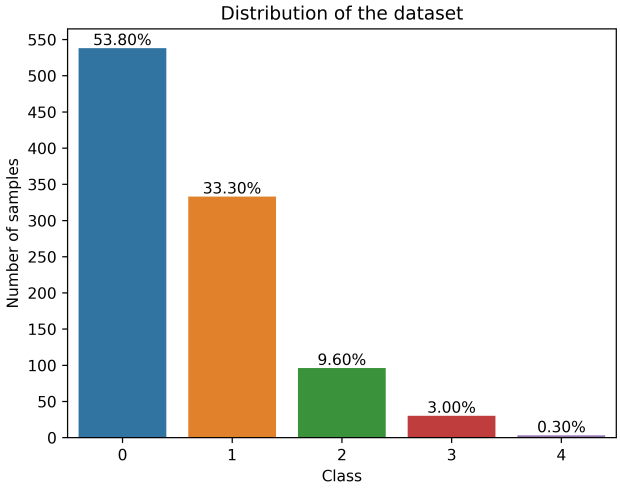

Figure 1: The dataset is composed of 1000 samples divided into 5 classes that represent the number of collisions.

1.2 Classification

The classification task is to predict the number of collisions in a sample. I will use several classification algorithms to compare their performance. The algorithms I will use are:

- **Logistic Regression**
- **Random Forest**
- **Support Vector Machine**
- **Gaussian Naive Bayes**

Since the dataset is unbalanced, I expect better results from Random Forest and Support Vector Machine, as they are able to handle unbalanced datasets. The following metrics will be used to evaluate the performance of the algorithms:

- **Accuracy**: the number of correct predictions divided by the total number of predictions
- **Precision**: the number of true positives divided by the number of true positives plus the number of false positives
- **Recall**: the number of true positives divided by the number of true positives plus the number of false negatives
- **F1-Score**: the harmonic mean of precision and recall

Since the dataset is unbalanced, I will use the F1-Score as the main metric to evaluate the performance of the algorithms. I will also use the confusion matrix to visualize the performance of the algorithms.

1.3 Regression

The regression task is to predict the minimum CPA in a sample. I will use the following regression algorithms to compare their performance:

- **Linear Regression**
- **Random Forest**
- **Support Vector Machine**

WThe following metrics will be used to evaluate the performance of the algorithms:

- **Mean Squared Error**: the average of the squared differences between the predictions and the actual values
- **R2 Score**: the coefficient of determination, which is a measure of how well the predictions fit the actual values

2 Classification Task

In this section I will describe the classification task that I have implemented, the way the dataset has been preprocessed and the results obtained with the different models.

2.1 Data Preprocessing

2.1.1 Feature selection

The first step is to select the features that will be used for the classification task. The dataset for each drone contains the following 7 features: the position, the velocity and the target position of the drone and the angle between the drone and the target (relative to the north). Since the angle is redundant since it can be computed from the position and the velocity (or also from the position and the target position), I have decided to remove it from the dataset. The dataset is then composed by $6 \times 5 = 30$ features for the 5 drones in the environment and the label, which is the number of collisions.

2.1.2 Normalization

Since the dataset contains features with different ranges, it is necessary to normalize the dataset. Since each row represents an environment, the normalization is performed for each row separately. The dataset also have data with different semantic meaning, so it is not possible to use a global normalization. In fact the position of the drone is an absolute value (refer to the point (0,0) in space), while the velocity is a relative value (it's the difference between the current position and the previous one) it's not good to use the same normalization for both of them. So this is the approach that I have decided to use:

- **Positions**: since position and target position have an absolute meaning, I have used the min-max normalization for each row with the following formula:

$$\text{norm}(x_{D^i}) = \frac{x_{D^i} - minCoord}{maxCoord - minCoord} \qquad \text{norm}(y_{D^i}) = \frac{y_{D^i} - minCoord}{maxCoord - minCoord} \tag{1}$$

  Where:

$$\begin{aligned} maxX &= \max_{1 \le i \le 5} x_{D^i} \\ maxY &= \max_{1 \le i \le 5} y_{D^i} \\ minX &= \min_{1 \le i \le 5} x_{D^i} \\ minY &= \min_{1 \le i \le 5} y_{D^i} \\ maxCoord &= \max(maxX, maxY) \\ minCoord &= \min(minX, minY) \end{aligned}$$

  In this way, after normalization, the position of the drones (and their target positions) will be in the range [0, 1]. The sample now represents a new "environment" where the bottom-left

corner corresponds at the position of the drone (or the target position) with the smallest x and y coordinates and it will also preserve the aspect ratio of the original one. The chose about preserving the aspect ratio is because it showed better results in the experiments.

- **Velocity**: since it has a relative meaning, have been used the predefined $maxCoord$ and $minCoord$ values to normalize the data with the following formula:

$$\text{norm}(vx_{D^i}) = \frac{x_{D^i}}{maxCoord - minCoord} \qquad \text{norm}(vy_{D^i}) = \frac{y_{D^i}}{maxCoord - minCoord} \quad (2)$$

2.1.3 Splitting

The dataset has been splitted in training and test set with a ratio of 80/20 in a stratified way, so that the distribution of the classes is the same in the training and test set.

2.1.4 Balancing

The dataset is unbalanced, so it is necessary to balance it. I have tried different methods for oversampling but since there are < 3 samples of the minority class after splitting, the standard oversampling (such as SMOTE or Random Over Sampling) methods are not really effective. Therefore, I have decided to implement my own oversampling method.

The idea is to use the semantic meaning of the dataset to generate new samples. The environment is a 2D plane with drones having a position, a velocity and a target. So I have decided to generate new samples by changing the position of the drones, keeping the velocity and the angle between the drone and the target constant. In fact it is possible to move the drones backwards (using the inverse of velocity) and in this way new samples having the same number of collisions can be generated.

Since the distribution of the classes is not uniform (e.g. the probability of having 4 collisons is lower than the probability of having 3 collisions or 2 collisions, etc.), I have decided to generate new samples for each class separately. The number of new samples generated for each class is proportional to the number of samples of the minority class. In details, the number of new samples generated for each class is as in Table 1.

	Class 0	**Class 1**	**Class 2**	**Class 3**	**Class 4**
Number of samples	50	55	65	70	75

Table 1: Number of samples for each class

2.2 Training

2.2.1 Classification models

I have compared different classifiers for this task.

- **Logistic Regression**
- **Random Forest**
- **Support Vector Machine**
- **Gaussian Naive Bayes**

2.2.2 Hyperparameter tuning

I have used the GridSearchCV method to tune the hyperparameters of the classifiers, using the F1 score as the metric to optimize. The hyperparameters that I have tuned are:

- **Random Forest**: n_estimators, max_features, criterion
- **Support Vector Machine**: C, gamma, kernel
- **Logistic Regression**: C, penalty, solver
- **Gaussian Naive Bayes**: var_smoothing

2.3 Evaluation

The evaluation of the models is performed using the F1 score, the accuracy and the confusion matrix.

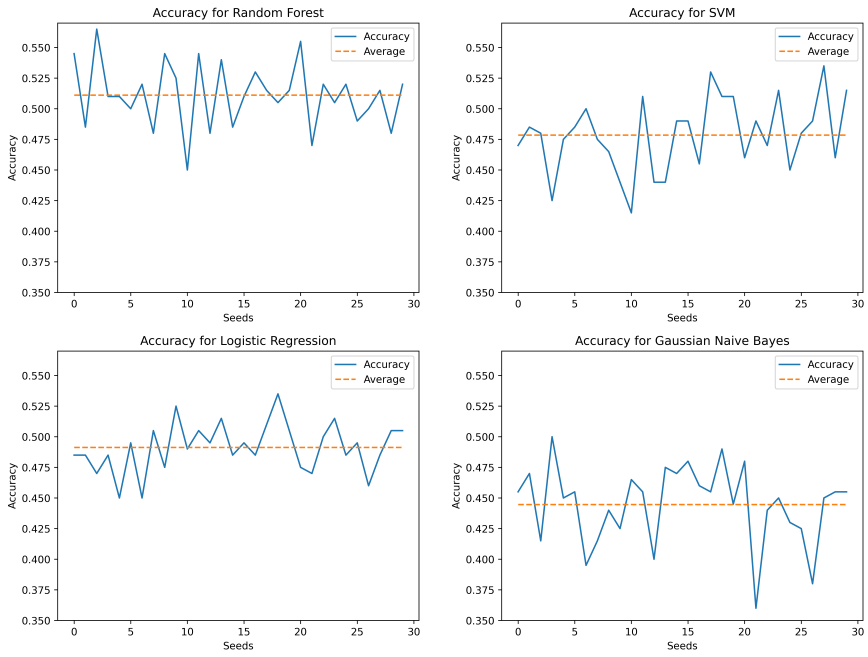

Figure 2: Accuracy of the different classifiers on the test set

Since the dataset is unbalanced it is better to use the F1 score instead of the accuracy, which is the harmonic mean of the precision and the recall. As shown in Figure 3, the Random Forest classifier has the best F1 score, followed by the Logistic Regression classifier.

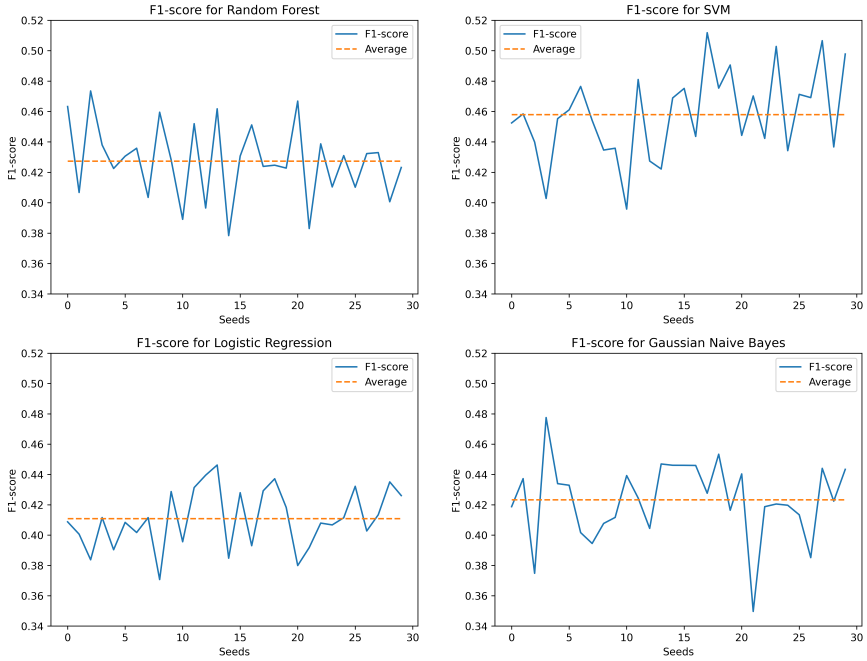

Figure 3: F1 score of the different classifiers on the test set

For further analysis, the confusion matrix of the 4 classifiers is shown in Figure 4.

Also a plot with the prediction against the ground truth is shown in Figure 5 to have a better visualization of the performance of the classifiers and to check if they also predict the minority class.

Figure 4: Confusion matrix of the different classifiers on the test set

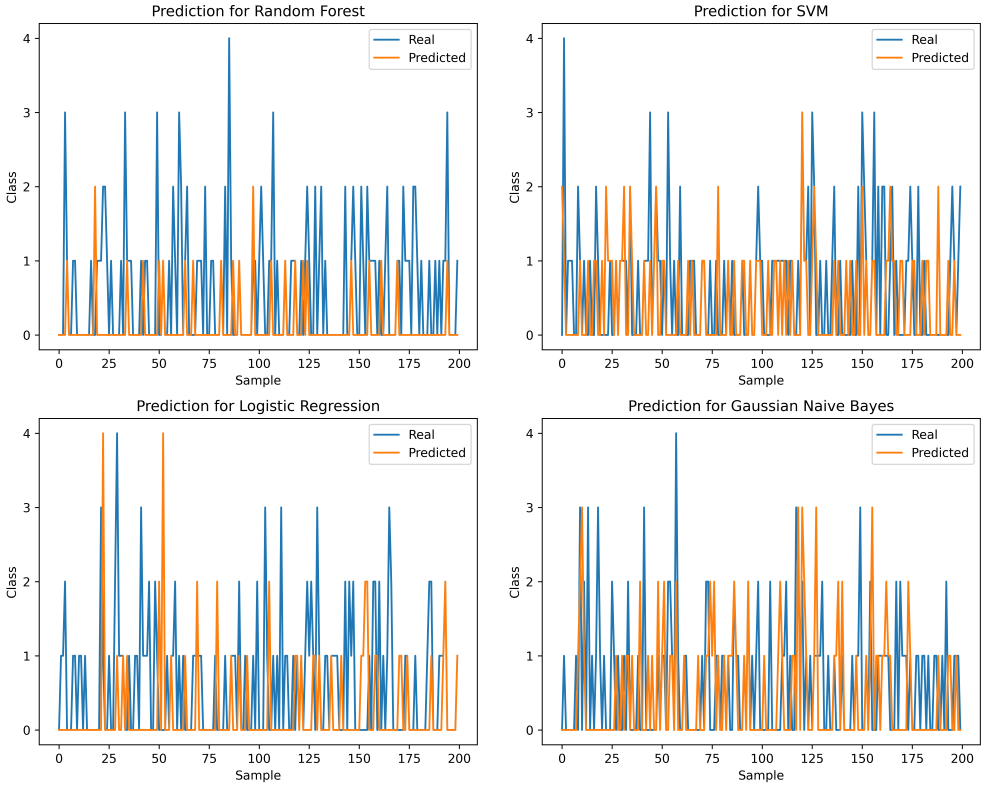

Figure 5: Prediction of the different classifiers on the test set

3 Regression Task

In this section I will describe the regression task that I have implemented, the way the dataset has been preprocessed and the results obtained with the different models.

3.1 Data Preprocessing

3.1.1 Feature selection

The first step is to select the features that will be used for the regression task. The dataset for each drone contains the following 7 features: the position, the velocity and the target position of the drone and the angle between the drone and the target (relative to the north). Also in this case, I have removed the angle from the dataset since it can be computed from the position and the velocity. The dataset is then composed by $6 \times 5 = 30$ features for the 5 drones in the environment and the label, which is the minimum CPA (Closest Point of Approach) between the drones.

3.1.2 Normalization

Since the dataset contains features with different ranges, it is necessary to normalize the dataset. Also in this case, the normalization is performed for each row separately respecting the same approach described in Section 3.1.2. In this case, the normalization is performed also for the label. The minimum CPA is a relative value, so it can used the same normalization applied to velocity.

$$\text{norm}(minCPA_{D^i}) = \frac{minCPA_{D^i}}{maxCoord - minCoord} \tag{3}$$

Where:

$$\begin{aligned}
maxX &= \max_{1 \le i \le 5} x_{D^i} \\
maxY &= \max_{1 \le i \le 5} y_{D^i} \\
minX &= \min_{1 \le i \le 5} x_{D^i} \\
minY &= \min_{1 \le i \le 5} y_{D^i} \\
maxCoord &= \max(maxX, maxY) \\
minCoord &= \min(minX, minY)
\end{aligned}$$

3.1.3 Splitting

The dataset has been splitted in training and test set with a ratio of 80/20.

3.2 Training

3.2.1 Regression models

3.2.2 Hyperparameter tuning

3.3 Evaluation

4 Conclusions

Summarize your results and your findings.