VICTIMS OF FALL
REPORT

Marco Lassandro

1 INTRODUCTION

The project aims to analyze the autopsy data of several victims of fatal fall, in order to estimate from which floor (or from which height) such cases fell and eventually died.

For this purpose, different machine learning models have been constructed to accomplish two main types of task: the regression and the classification task; for each of such tasks, different metrics have been used to evaluate the error on the estimations performed by the models.

The dataset gently conceded, by the forensic medicine group of the Università degli studi di Milano, has been preprocessed in different ways, by using techniques like: filtering of features and cases, features extraction, scaling of the data and also acquisition of new features from the ones available.

2 THE DATASET

The dataset is composed by a total of 385 cases of victims of fall, each one have a total of 33 features where the first 6 features are information related to the subject: ID case, sex, height, weight and BMI index; while the other 27 are information related to the fall of the subject, where the first two are essentially the ground truth values that the machine learning models should estimate: the height of the fall, expressed in meters with minimum value of 6m and maximum value of 24m, and the potential energy released by the fall; the other 22 features are a description of the injury pattern, 4 features represent the 4 major anatomical areas (Head, Thorax, Abdomen and Skeleton) and describe the sum of the major organs that had severe damages after the fall, the remaining 18 features represent the single major organs, describing with 0 a "no lesion" or a "minor lesion" and with 1 a "severe lesion". The cases have been selected according to a balanced stratification of the height of fall, so in practice we have 7 classes of heights (6m or less, 9m, 12m, 15m, 18m, 21m and 24m or more) each one consisting of 55 cases.

3 EXPERIMENT SETUP

3.1 DATASET VARIANTS

Different variants of the dataset have been retrieved from the original one in order to perform the experiments. Before the construction of any of that, some pre-processing steps have been applied over the data:

- Drop of cases: any case with one ore more empty values over features have been dropped, this because some machine learning algorithms are unable to deal with such empty values. From a total of 385 cases we got 341 cases;
- Drop of the ID case feature: this feature has been dropped because it doesn't bring information to the models;
- From meters to floors: the height of fall has been stratified according to the well known rule about the standard and mean heights of the floors in civil buildings; we have decided to convert each class of height as a floor so our labels are described with an integer in a range from 1 to 7, the "height of fall" feature has become the "floor of fall" feature;
- Selection of the ground truth feature: as said before there are two feature that could represent the labels for our models, the "floor of fall" and the "potential energy (mgh)" feature. In this project what we want is to estimate the floor of fall, so the potential energy feature has been dropped from the dataset;
- Height and weight feature: in the first experiments we have noticed that most of the models achieve better results when we remove from the dataset the "height" and "weight" feature, leaving just the "BMI" index to represent the state of the subject's body;

After the application of such pre-processing steps, three main variants from the original dataset have been extracted, each one containing all the remaining information about the subjects, while, about the information related to the injury pattern, we have:

- the "complete" variant: that contains all the features related to the injury pattern of the fall;
- the "only totals" variant: that contains just the features representing the major anatomical areas;
- the "only binary" variant: that contains just the features representing the major organs;

3.1.1 COMPOSITION OF NEW FEATURES

About the "complete" and the "only total" variant, in the original dataset the features related to the major anatomical areas describe the sum of the associated major organs that had severe lesions, we have tried to substitute each one with a percentage of the major organs that had severe lesions after the fall; moreover, a new feature has been added representing the percentage of all the major organs with severe damages.

3.1.2 FEATURE EXTRACTION

Feature extraction is a process of dimensionality reduction by which an initial set of raw data is reduced to more manageable groups for the processing. The process of feature extraction is useful when you need to reduce the number of resources needed without losing important or relevant information. Feature extraction can also reduce the amount of redundant data for a given analysis. Also, the reduction of the data and the machine's efforts in building feature combinations facilitate the speed of learning and generalization steps in the machine learning process.

In all the experiments done, two methods related to the feature extraction step have been applied over the features related to the injury pattern of the fall:

- PCA: it is used to decompose a multivariate dataset in a set of successive orthogonal components that explain a maximum amount of the variance;
- TrucatedSVD: this transformer performs linear dimensionality reduction by means of truncated singular value decomposition (SVD). Contrary to PCA, this estimator does not center the data before computing the singular value decomposition. This means it can work better with sparse matrices efficiently.

3.1.3 SCALING

In all the experiments, after the feature extraction step, a scaling of the data has been applied over the resulted dataset. Different methods have been tried to scale the data:

- Standard scaling: it standardize features by removing the mean and scaling to unit variance. Given a sample x, the standard score is calculated as follow:

  $$z = (x - u)/s$$

  Where u is the mean of the training samples, and s is the standard deviation of the training samples;
- Robust scaling: it scales features using statistics that are robust to outliers. It removes the medians scales the data according to the quantile

range (defaults to IQR: Interquartile Range). The IQR is the range between the 1st quartile (25th quantile) and the 3rd quartile (75th quantile);

- MinMax scaling: It scales and translates each feature individually such that it is in the given range on the training set;
- Normalization: each sample (i.e. each row of the dataset) with at least one non zero component is rescaled independently of other samples so that its norm (l1, l2 or inf) equals one.

In most of the experiments it has been noticed that the best scaling method, that give us better results, is the MinMax scaling.

3.2 MODELS

As said in the introduction we have constructed several models to accomplish the regression and the classification task.
In the following table we have the models used for the regression problem:

MODEL	**LIBRARY**
DecisionTreeRegressor	sklearn
Ridge	sklearn
Lasso	sklearn
SVR	sklearn
LogisticRegression	sklearn
RandomForrestRegressor	sklearn
GradientBoostingRegressor	sklearn
XGBRegressor	xgboost
FFNN	tensorflow

Table 1: Models used for the regression task

About the classification problem, two types of classification have been used to predict the labels:

- Binary classification: in this case, in order to let the classifiers make a binary prediction, the floors have been "clustered" into two classes (0 and 1), this has been done by defining an integer threshold t. Floors lower than the threshold, so with the related integer value $f \leq t$, have been clustered into the class 0, that we can consider as the "lower floors" class; instead floors with value $f > t$, have been clustered into the class 1, we can name such class as the "higher floors" class;
- Multiclass classification: in this case the models have been trained in order to predict the classes as they are. Some machine learning algorithms are

unable to fit models for a multiclass classification (but only for binary classifications), so, to deal with this problem two strategies have been used:

- OneVsRest strategy: that consists in fitting one classifier per class. For each classifier, the class is fitted against all the other classes.
- OneVsOne strategy: that consists in fitting one classifier per class pair. At prediction time, the class which received the most votes is selected.

In the following table is reported the models used for the classification task:

MODEL	**LIBRARY**
DecisionTreeClassifier	sklearn
SVC	sklearn
LogisticClassifier	sklearn
RandomForrestClassifier	sklearn
GradientBoostingClassifier	sklearn
XGBClassifier	xgboost
FFNN	tensorflow
ChainedEstimator	My implementation
GranularBinaryClassifier	My implementation
OrdinalRidge	mord
KMeans	sklearn
FuzzyCMeans	skfuzzy

Table 2: Models used for the classification task

3.2.1 ChainedEstimators GranularBinaryClassifier

We have tried to create two ensemble models that uses different strategies to accomplish the classification task.

The ChainedEstimator model is composed by multiple sub models, where one is a sub-model used for the regression task and the others are binary classifiers (one for each two adjacent floors); when we fed a data point, first it is passed to the regressor, then the prediction is rounded and based on the result a sub classifier is selected in order to make a classification between the related adjacent floors, eventually the outcome of such classifiers will be the prediction of the model.

The GranularBinaryClassifier is composed by a series of binary classifiers structured like a tree. In order to compose the tree, it uses a threshold hyperparameter to cluster the floors into two classes and train the main classifier that will be the root of the tree, then the floors associated to each class are

again clusterd, into two sub classes, and new classifiers are trained, they will correspond to the nodes of the tree; this process is repeated until we get two classes with just one floor per cluster, the classifiers trained over the related data points are the leaves of the tree. When the model makes a prediction it passes first the data point to the root, based on the result (0 or 1) a node is chosen; this process is repeated until a leaf is reached, eventually the prediction of the leaf classifier will be the output of the model.

3.2.2 MODEL SELECTION

In order to evaluate the performances of the machine learning algorithms and to estimate the generalization error of the underlying model with its best hyper-parameters, a Nested-Cross validation procedure has been applied to all the experiments. It is called in this way because nests the cross-validation and the hyper-parameters tuning; model selection without nested CV uses the same data to tune model parameters and evaluate model performance, but, in this way information may thus "leak" into the model and overfit the data. The magnitude of this effect is primarily dependent on the size of the dataset and the stability of the model.

Nested CV uses a series of train/test set splits, for each training set a new CV procedure is run, we can call it inner CV, where the score is approximately maximized here by searching the best hyper-parameters to use for the model considered. The generalization error is computed by retraining the models obtained from the inner CVs over the related training sets of the outer CV and eventually averaging the test scores obtained from the outer CV.

In our experiments the number of k-fold (splits) considered for the inner and outer CV are 10.

3.3 RESULTS

In this section we will present the results, obtained from the experiments done, dividing them by regression and classification task. Before that, here a briefly explanation of the metrics considered for the evaluation of the performances of our models.

3.3.1 METRICS

In the case of the regression problem three metrics have been considered:

- Root mean squared error (RMSE):

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=0}^{N}(y_i - \hat{y_i})}$$

  It is calculated as the sum of the square of the prediction error, which is the real output y_i minus the predicted output $\hat{y_i}$, and then divided by the number of data points N; all this is under the square root.

- R Square metric: it measures how much variability in dependent variables can be explained by the model. It is the square of the Correlation Coefficient(R) and that is why it is called R Square.

$$R^2 = 1 - \frac{SS_{regression}}{SS_{total}} = 1 - \frac{\sum_i (y_i - \hat{y_i})^2}{\sum_i (y_i - \overline{y_i})}$$

  R Square is calculated by the sum of squared of prediction error divided by the total sum of the square which replaces the calculated predictions with the mean. R Square value is in range from 0 to 1, bigger values indicates a better fit between prediction and actual value.

- Mean absolute error: Mean Absolute Error(MAE) is similar to Mean Square Error(MSE). However, instead of the sum of square of error as in MSE, MAE is taking the sum of the absolute value of the error.

In the case of a classification experiment just the accuracy will be reported in the following tables, that is calculated as:

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN}$$

where TP and TN are the number of predictions resulted as "true positive" and "true negative" and FP and FN are the number of predictions resulted as "false positive" and "false negative".

3.3.2 REGRESSION RESULTS

In the following table the best results of the top five models for the regression problem:

Dataset variant	Model	RMSE	R squared	MAE
only totals	RandomForestRegressor	1.47 (std. 0.23)	0.43 (std. 0.18)	1.16 (std. 0.17)
complete	GradientBoostingRegressor	1.51 (std. 0.18)	0.41 (std. 0.15)	1.23 (std. 0.14)
complete	SVR	1.54 (std. 0.2)	0.39 (std. 0.16)	1.21 (std. 0.16)
only totals	Ridge	1.54 (std. 0.17)	0.38 (std. 0.14)	1.23 (std. 0.14)
only binary	DecisionTreeRegressor	1.65 (std. 0.18)	0.29 (std. 0.15)	1.35 (std. 0.15)

Table 3: Best results obtained for the regression task

3.3.3 CLASSIFICATION RESULTS

Here the best results obtained by applying a binary classification, as we can see the best value for the threshold parameter is when $t = 2$:

Dataset variant	Model	Accuracy	Modality
only totals	SVC	0.82 (std. 0.06)	Binary (t = 2)
only totals	RidgeClassifier	0.82 (std. 0.06)	Binary (t = 2)
only binary	XGBClassifier	0.81 (std. 0.06)	Binary (t = 2)
only binary	RandomForestClassifier	0.81 (std. 0.08)	Binary (t = 2)
only totals	GradientBoostingClassifier	0.8 (std. 0.06)	Binary (t = 2)
only totals	DecisionTreeClassifier	0.78 (std. 0.06)	Binary (t = 2)

Here the best results obtained by applying the OneVsRest approach to accomplish a multiclass classification task:

Dataset variant	Model	Accuracy
only binary	XGBClassifier	0.36 (std. 0.08)
complete	SVC	0.34 (std. 0.1)
complete	DecisionTreeClassifier	0.31 (std. 0.06)
only totals	RidgeClassifier	0.3 (std. 0.04)

Here the best results obtained by applying the OneVsOne approach to accomplish a multiclass classification task:

Dataset variant	Model	Accuracy
complete	XGBClassifier	0.35 (std. 0.07)
only totals	SVC	0.35 (std. 0.08)
only totals	DecisionTreeClassifier	0.34 (std. 0.11)
complete	RidgeClassifier	0.32 (std. 0.09)

Here the results obtained by using the ChainedEstimator model with different models as regressor and classifier:

Dataset variant	Sub-Regressor Model	Sub-Classifier Model	Accuracy
only totals	GradientBoostingRegressor	SVC	0.26 (std. 0.06)
only totals	RandomForestRegressor	SVC	0.28 (std. 0.06)
only totals	RandomForestRegressor	DecisionTreeClassifier	0.26 (std. 0.05)

Here the results obtained by using the GranularBinaryClassifier model:

Dataset variant	Sub-classifier Model	Accuracy
complete	SVC	0.18 (std. 0.04)
only binary	DecisionTreeClassifier	0.18 (std. 0.05)
only totals	RidgeClassifier	0.17 (std. 0.03)

3.4 CONCLUSIONS

As we can see above, when we try to make a multiclass classification over our problem, the results are not so satisfying as when we make a binary classification, this may due to the fact that the dataset doesn't contain so many cases per floor, so the majority of the models, for this reason, are in a situation of underfitting.

In general we can see that the results obtained for the regression task are quite good, but as for the classification task, if we had more examples for the training of the related models, the results could have a significant improvement.