HUB IQ

Inteligencia Artificial & Machine Learning

Applicaciones en movilidad

Dr. Iván S. Razo Zapata

EngineeringX
Founded by the Royal Academy of Engineering and Lloyd's Register Foundation

Ejercicio de Aprendizaje Supervisado

Idea general

- Empleo de datos reales
  - Etiquetas / Valores objetivo
- EDA - Exploratory Data Analysis
- Preprocesamiento
  - Limpieza, normalización, ingeniería de atributos, manejo de datos faltantes
- Implementación de modelos
  - Regresión lineal/logística, arboles de decisión, k-vecinos más cercanos, redes neuronales
- Evaluación de modelos
- Ajuste de hyperparametros

Road Traffic Accident Dataset of Addis Ababa City

Published: 2 November 2020 | Version 1 | DOI: 10.17632/xytv86278f.1
Contributor: Tarikwa Tesfa Bedane

Description

This data set is collected from Addis Ababa Sub city police departments for Masters research work. The data set has been prepared from manual records of road traffic accident of the year 2017-20. All the sensitive information have been excluded during data encoding and finally it has 32 features and 12316 instances of the accident. Then it is preprocessed and for identification of major causes of the accident by analyzing it using different machine learning classification algorithms algorithms. RTA Dataset.csv is the dataset before preprocessing and cleaned.csv is the preprocessed dataset.

Bedane, Tarikwa Tesfa (2020), "Road Traffic Accident Dataset of Addis Ababa City", Mendeley Data, V1, doi: 10.17632/xytv86278f.1

https://data.mendeley.com/datasets/xytv86278f/1

kaggle

Search

Create

Home

Competitions

Datasets

Models

Code

Discussions

Learn

More

Your Work

RECENTLY VIEWED

Road Traffic Accidents

Road Traffic Accident ...

Road Traffic Accidents...

Road Traffic Accidents

Road Traffic Accident Dataset of Addis Ababa City

Data Card | Code (13) | Discussion (0)

About Dataset

Context

This data set is collected from Addis Ababa Sub city police departments for Masters research work.

Content

The data set has been prepared from manual records of road traffic accident of the year 2017-20. All the sensitive information have been excluded during data encoding and finally it has 32 features and 12316 instances of the accident. Then it is preprocessed and for identification of major causes of the accident by analyzing it using different machine learning classification algorithms algorithms. RTA Dataset.csv is the dataset before preprocessing and cleaned.csv is the preprocessed dataset.

Acknowledgements

Bedane, Tarikwa Tesfa (2020), "Road Traffic Accident Dataset of Addis Ababa City", Mendeley Data, V1, doi: 10.17632/xytv86278f.1

Usability
9.41

License
Attribution 4.0 International (CC ...

Update frequency
Unspecified

Tags

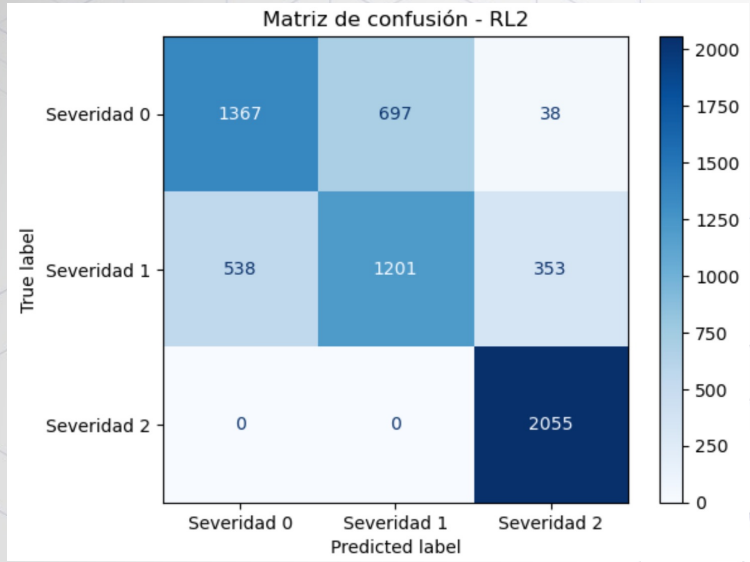
Tabular | Transportation | Beginner | Intermediate | Advanced

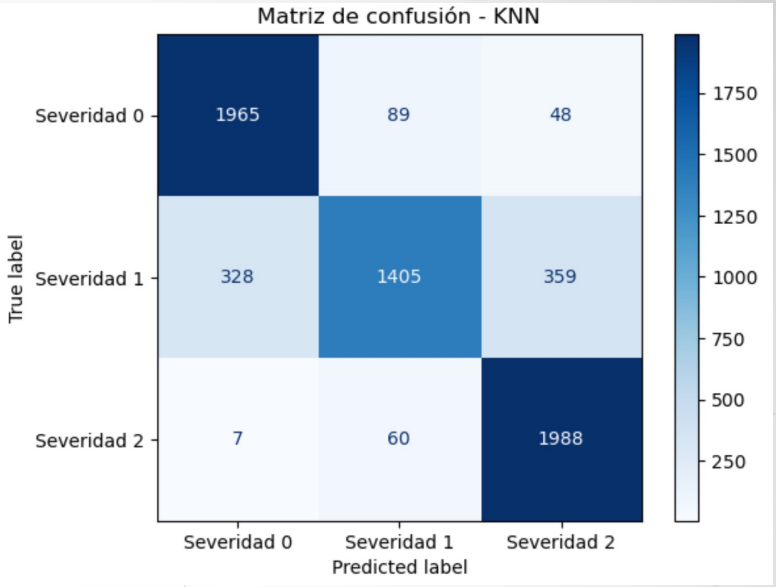
https://www.kaggle.com/datasets/saurabhshahane/road-traffic-accidents

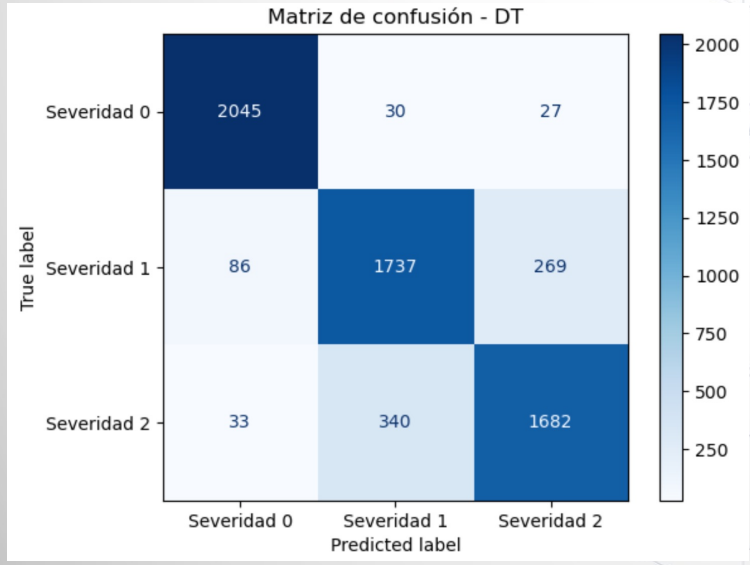
Implementación de modelos

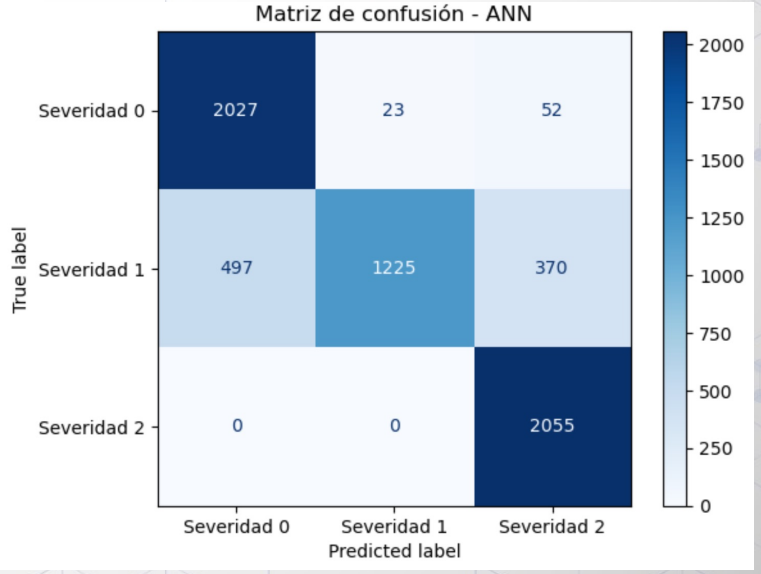
- Regresión logística
- k-vecinos más cercanos
- Árboles de decisión
- Red neuronal

Evaluación de modelos









Ajuste de hyperparametros

3.2. Tuning the hyper-parameters of an estimator

Hyper-parameters are parameters that are not directly learnt within estimators. In scikit-learn they are passed as arguments to the constructor of the estimator classes. Typical examples include `C`, `kernel` and `gamma` for Support Vector Classifier, `alpha` for Lasso, etc.

It is possible and recommended to search the hyper-parameter space for the best cross validation score.

Any parameter provided when constructing an estimator may be optimized in this manner. Specifically, to find the names and current values for all parameters for a given estimator, use:

```
estimator.get_params()
```

A search consists of:

- an estimator (regressor or classifier such as `sklearn.svm.SVC()`);
- a parameter space;
- a method for searching or sampling candidates;
- a cross-validation scheme; and
- a score function.

Two generic approaches to parameter search are provided in scikit-learn: for given values, **`GridSearchCV`** exhaustively considers all parameter combinations, while **`RandomizedSearchCV`** can sample a given number of candidates from a parameter space with a specified distribution. Both these tools have successive halving counterparts **`HalvingGridSearchCV`** and **`HalvingRandomSearchCV`**, which can be much faster at finding a good parameter combination.

After describing these tools we detail best practices applicable to these approaches. Some models allow for specialized, efficient parameter search strategies, outlined in Alternatives to brute force parameter search.

Note that it is common that a small subset of those parameters can have a large impact on the predictive or computation performance of the model while others can be left to their default values. It is recommended to read the docstring of the estimator class to get a finer understanding of their expected behavior, possibly by reading the enclosed reference to the literature.

Ajuste de hyperparametros

3.2.1. Exhaustive Grid Search

The grid search provided by **GridSearchCV** exhaustively generates candidates from a grid of parameter values specified with the `param_grid` parameter. For instance, the following `param_grid`:

```
param_grid = [
  {'C': [1, 10, 100, 1000], 'kernel': ['linear']},
  {'C': [1, 10, 100, 1000], 'gamma': [0.001, 0.0001], 'kernel': ['rbf']},
 ]
```

specifies that two grids should be explored: one with a linear kernel and C values in [1, 10, 100, 1000], and the second one with an RBF kernel, and the cross-product of C values ranging in [1, 10, 100, 1000] and gamma values in [0.001, 0.0001].

The **GridSearchCV** instance implements the usual estimator API: when "fitting" it on a dataset all the possible combinations of parameter values are evaluated and the best combination is retained.

Examples:

- See Custom refit strategy of a grid search with cross-validation for an example of Grid Search computation on the digits dataset.
- See Sample pipeline for text feature extraction and evaluation for an example of Grid Search coupling parameters from a text documents feature extractor (n-gram count vectorizer and TF-IDF transformer) with a classifier (here a linear SVM trained with SGD with either elastic net or L2 penalty) using a `pipeline.Pipeline` instance.
- See Nested versus non-nested cross-validation for an example of Grid Search within a cross validation loop on the iris dataset. This is the best practice for evaluating the performance of a model with grid search.
- See Demonstration of multi-metric evaluation on cross_val_score and GridSearchCV for an example of **GridSearchCV** being used to evaluate multiple metrics simultaneously.
- See Balance model complexity and cross-validated score for an example of using `refit=callable` interface in **GridSearchCV**. The example shows how this interface adds certain amount of flexibility in identifying the "best" estimator. This interface can also be used in multiple metrics evaluation.
- See Statistical comparison of models using grid search for an example of how to do a statistical comparison on the outputs of **GridSearchCV**.

Código

Experimentar

- Ingeniería de atributos
- Otros modelos de aprendizaje supervisado
  - Random Forest
- Diferentes parámetros para Grid Search

- Otros conjuntos de datos

Consideraciones Finales

- Inteligencia Artificial
  - Symbolic AI, Semantic Web/Technologies, Computer Vision
- Aprendizaje Supervisado
  - Naïve Bayes, SVM
- Aprendizaje No Supervisado
- Aprendizaje Por Refuerzo
- Aprendizaje Profundo
  - Transformadores
- Large Language Models
  - GPT-4, LLaMa
- Aspectos éticos
  - Privacidad, interpretabilidad, accountability (responsabilidad)

EngineeringX

Founded by the Royal Academy of Engineering
and Lloyd's Register Foundation

GRACIAS

https://hubiq.mx/

HUBIQRO HUBIQ HUBIQRO