Data Science Project

Introduction:

This project objective is for us to critically analyse two datasets. We are challenged to understand the data we are dealing with, process it, create models and then hypothesise what could be done to improve them. This report is divided into two major parts, one for each dataset, **NYC Motor Vehicle collisions to Person** and **Air Quality in China**, respectively. Let's start!

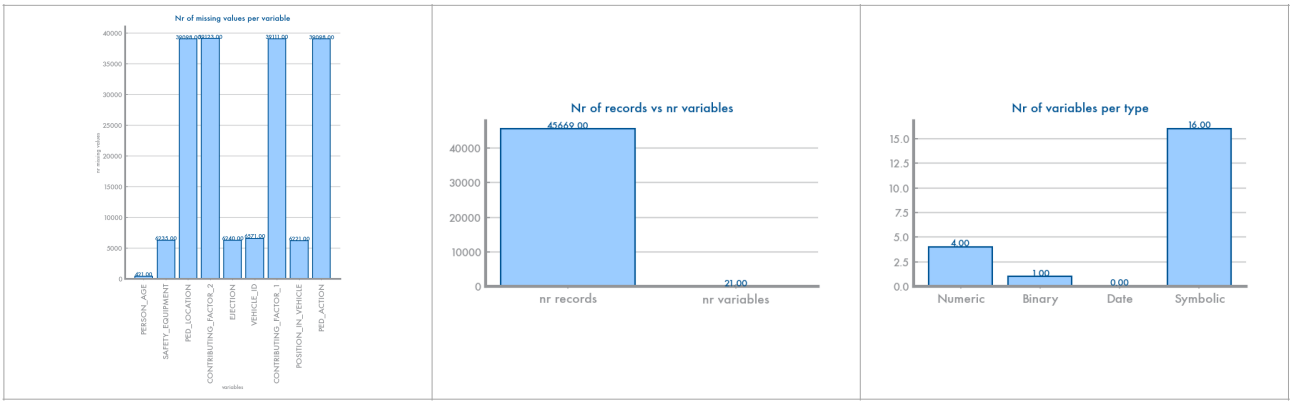
Dataset 1: NYC Motor Vehicle Collisions to Person

1 Data profiling:

Regarding supervised classification methods, we are going to use **PERSON_INJURY** as the target variable, and, for forecasting **NR_COLLISIONS**. We want to be especially alert to these features when analysing our data.

Data dimensionality:

As it is possible to see in [***Figures 1-3 in the appendix***], we have almost 6568 records with missing values for the **VECHICLE_ID** and 421 for the **PERSON_AGE**. We have **21** variables of which **15** are **Symbolic**, **2** are **Binary** and **4** are **Numeric**, and **45650** records. This is already presenting us with some problems: how to treat the high number of records with missing values (can they be simply dropped, or should we don't something else?, regarding the symbolic variables, how are we going to treat them?). Lastly, our target variable has two values: Injured and Killed. On the dataset there are around **45 thousand** records for Injured and around **250** for Killed, making this a very imbalanced dataset.



Data granularity:

Before analysing the data granularity, we need to understand how our data is distributed. The first thing we are able to notice is that there are some **incorrect values** (not noise nor outliers!) on the PERSON_AGE feature, as the maximum value is 9999, and, in fact, we have 29 records with PERSON_AGE > 140 or < 0. For the remaining of the report, these incorrect values **were removed**. Nonetheless, a note must be done that these removed records were all 'Injured' in the target variable. Since it is the majority class, this removal doesn't impact - at all - our work (the features distributions). Moreover, there's little to no interest in

working with all the features that represent an ID: [UNIQUE_ID, VEHICLE_ID, COLLISION_ID, PERSON_ID]. For the remaining of the work, **we won't be working with any of those variables**. It is a fact that they could indeed have some valuable information about the dataset in question, however, and after analysing correlations, distributions and granularity, we find no interest in working with them.

Regard **time hierarchy:**
- CRASH_DATE records are grouped into 'Weekday', 'Weekend' and 'Holiday'. (We discovered two python libraries that could do exactly this for both USA and Chinese holidays, for both datasets).
- CRASH_TIME are also grouped into 'Dawn', 'Morning', 'Lunch time', 'Afternoon', 'Dinner time' and 'Night'.

Regarding granularity, we need to deal with our symbolic features. For them, we present a simple methodology. First, for each feature, we try to create 'major' groups, where, for each group, we can attribute a number (depending on, for example, the severity of an injury, or the degree of protection of a safety equipment) and then, for each 'major' group, if we need more detail (with we usually do), we try to create one more (finer) level, where we also attribute a number depending on some criteria. This way, for each major group, if there are many, we fix a number (10, 20, 30, ...) and then for each finer group, we fix another number (11,12,13... 21,22,23...). This way we can deal with our symbolic features, treating them as ordinal ones. Obviously this has some problems, however, it presents a good trade-off between information, detail, and usability in our models since most can't handle symbolic data, and, to use the one-hot encoding, would also be extremely challenge since we would have a great number of features (as we are going to see). A curious challenge that we found, was, what value to attribute to 'Unknown' or 'Not Applicable'. For each feature, we tried to create, with a given criteria and logic, a 'distance' from these values to the other ones, that sounds good and intuitive.

With this, we proceeded with the following [***All details in the appendix***]:
- POSITION_IN_VEHICLE grouped depending on the risk given the position. Higher the value, higher the risk;
- EMOTION-STATUS, EJECTION, PED_LOCATION, PERSON_TYPE grouped depending on the health risk;
- For the EJECTION feature, EJECTED and PARTIALLY_EJECTED values were grouped into 'EJECTED'.
- BODILY_INJURY grouped with 'major areas'. For example, ['Shoulder - Upper Arm', 'Elbow-Lower-Arm-Hand'] were grouped into 'Arm'. 'Head', 'Face' and 'Eye' were grouped into Head. For each one of these groups, a value was given according to its health risk.
- PED_ACTION grouped into major groups, for example ["Crossing With Signal", "Crossing, No Signal, Marked Crosswalk"] -> "Crossing With Signal / Crosswalk". For each one of these groups, a value was given according to its health risk.
- SAFETY_EQUIPMENT were also grouped and valued as PED_ACTION and BODILY_INJURY. For example, every unique value with the sentence 'Air Bag', was grouped into just 'Air Bag'. In this case, the values were given depending on the protection given (the higher the value, higher the protection).
- CONTRIBUTING_FACTOR_1 we were also able to create major groups, passing from 46 unique values to 23 (similar to CONTRIBUTING_FACTOR_2). In this case, the values are also attributed depending on the risk presented to the population.
- For COMPLAINT, we talked with a 4th year medicine student, from the Faculty of Medicine of the University of Lisbon (thank you João!) and we were able to group each one of the complaints into major groups: ["Green", "Yellow", "Red", "WHEN PATIENT MUST BE TAKEN TO THE SITE OF CARE WITH URGENCY", "CRITICAL WITH ON-SITE EMERGENCY TREATMENT"]. Then, for each group, we gave a value depending on its health impact.

With these granularity changes, we mostly wanted to enter 'common-sense' knowledge and also give to all symbolic features an order. This order, obviously, is limited and has some problems, however, and regarding dummification, is a much better approach than using one-hot encoding.

Feature	POSITION IN VEHICLE	EMOTIONAL STATUS	EJECTION	PED LOCATION	PED ROLE	PERSON TYPE	BODILY INJURY	PED ACTION	SAFETY EQUIPMENT	COMPLAINT	CONTRIBUTING FACTOR 1	CONTRIBUTING FACTOR2
Old unique values	10	8	4	4	5	4	14	16	16	19	46	40
New unique values	5	7	3	4	5	4	10	12	9	7	23	23

Data Distribution:

Regarding data distribution, one important characteristic of this dataset to mention, and something that will be with us during the entire dataset exploration, is that it is highly unbalanced, with 247 Killed records and 44972 Injured ones. This means that we have roughly 180 times more information about Injuries than deaths.

Regarding data distribution, for our only numeric variable, PERSON_AGE, we have a normal distribution with an expected range.

Also, for the symbolic features, theres some informations that we can retrieve. For one, almost every feature has one value that has a much bigger count than all the others. For instance, for the Ejection feature, almost 35 thousands records are just for one value, 'Not Ejected'. There's not much that we can retrieve from the distribution other than the fact that the majority of the accidents are probably quite identical, supported by the fact that so many features have one value with a much higher weight than the others.

Obviously, this is not to say that we can't retrieve more information. For example, almost half of the records (26 thousand) have, as PED_ROLE, the value DRIVER. This is information that is quite trivial and expected. Roughly 13 thousand have PED_ROLE as Passenger. We also observed that the large majority of the records have as 'Ejection', the value 'Not Ejected'. Even though it is quite good and reassuring to know that the majority of the accidents, the majority of the victims are not 'Ejected' or 'Partially Ejected' (ejection we can assume is something bad!), the reality is that it doesn't give us much information in terms of modelling this particular problem of predicting when do we have an accident with death as an outcome. We can only retrieve high level information, as, for example, what is the most common POSITION_IN_VEHICLE (in this particular context).

Data Sparsity:

Regarding data sparsity, we assess that the data is sparse in the sense that we have 'holes' in the space of the data.
Also, from sparsity charts, there's little to no information that we can retrieve. Indeed the majority of the features are symbolic, as such the 'dot' visualisation from the scatter plots is of little help.

As such, and to gather more helpful information, we've used seaborn stripplot with jitter to better understand the features we have, for both Injured and Killed records. Reinforcing that correlation doesn't mean causation, from the stripplot we can obtain information like the fact that there are more injuries starting from 12h to 23h (which probably means that we are talking about accidents in a city, regarding minor ones). More interesting facts is that for Killed, the bodily injured are mainly on the head, entire body and chest, with contrast to the injured records where there's no body part that stands out. In terms of safety equipment, the injured records mostly have Belt (probably because they are on a car), but for the killed records, the most recurrent safety equipment is none, and then an helmet (one can hypothesize that this is because of motorcycles and bicycles). Other relevant information is that for the killed records, there's two values that stand out on the person type feature: occupant and pedestrian, however, in the injured records, only occupant stands out.

Regarding the ejection feature, we can also assess that there is a majority of the killed records that ejected, contrasting with the injured ones, with a majority of non-ejection. For the complaint feature there's also a majority on GREEN for the injured records, while for the Killed records there's a majority for 'Yellow', and GREEN only holds 7% of records.

This type of information might sound quite trivial and of little importation, however it is good to assess how the data is distributed between both records - injured and killed. This way we can better understand with what we are dealing and also better understand our future results (for example, try to better understand why a decision tree uses a certain split or not).

Another area of big difference is the emotional status, where, for the killed records, 65% of the records have the value 'apparent death' and 26% are 'Unconscious'. This finding is of the **upmost importance** because what it is saying is that a majority of the killed records is defined as having an emotional status of 'apparent death'. In a rough generalisation, this is almost what we are trying to predict! This is important because, upfront, we will want to use our final model to predict the target feature but without this feature (to compare the results). For the injured records the majority of the records (92%) have the value 'Conscious', while for the killed ones the conscious value only holds for 6% of the records. Actually, for the injured records only 241 in almost 40 thousand records have EMOTIONAL_STATUS has 'Unconscious' or 'Apparent Death'. For all the features that we didn't mention, either there weren't any findings or they weren't relevant.

Regarding the correlation matrix, for the numeric feature (person age), there's no correlation with the target feature. For the symbolic features there are some correlations that stand out: [PED_ACTION, PED_LOCATION], [CONTRIBUTING_FACTOR_1, CONTRIBUTING_FACTOR_2], [EJECTION, SAFETY_EQUIPMENT], [PED_ROLE, POSITION_IN_VEHICLE]. Most of these correlations, even though they are high, don't give us much information. More important is probably the correlation between the EJECTION and SAFETY_EQUIPMENT, but, even for it, it is because of the use, or not, of a belt.

Regarding correlations with respect to PERSON_INJURY, our target feature, the higher correlation is with the feature COMPLAINT. If we divide the dataset between Injured and Killed, we find a lot of very strong correlations between many of the features for the Killed records. For the injured ones, the correlations are also strong, however not as strong. One important remark is that, as we've seen, the feature EMOTIONAL_STATUS is highly relevant for the prediction of the target feature, however, it has little to no correlation with other features for the Killed records.

2 Data preparation:

Missing Value Imputation, Dummification and other transformations:

To test what works the best we created a pipeline that first imputes the missing values and then encodes them. For the imputation we test three different approaches: a custom missing value imputation; replace the missing values with a constant value; replace the missing values with the most common value. Regarding the encoding we test two approaches: a custom encoding (using the rules that we stated on Data granularity) and one hot encoding.

Our custom missing value imputation works with a simple set of rules:
- The records without person age, are removed;
- The records with PERSON_SEX = "U" are removed;
- For each row of the dataset, the rules ***on the right image*** are applied.
  - Regarding these rules, they were establish considering the knowledge we've gained on the previous sections.
  - Some additional transformations were also made. For example, regardless of the value, if it empty, or, not, if the person type is pedestrian, then the position in vehicle is set as 'Does Not Apply'. If the person type is not pedestrian and is different than 'Occupant', then the ped_location is set as 'Does Not Apply'.

For the other missing value imputations methods, they are quite straightforward. One replaces the missing values with the most common value for each feature, and the other replaces with a constant value (for numeric values replaces with 0, for symbolic replaces with 'Unknown', and for binary replaces with *False*).

Regarding the encoding of the symbolic features, we use two methodologies: one with one-hot encoding and another with a custom encoding method. For this custom encoding, what we do is that we've previously created an excel file, with multiple pages (one for each feature) where we have three columns for each page: "Old Unique Value", "New Value/Group", "Value". This was, and with the values that we've described on the granularity section, each feature is assigned the value of the new respective group. The main idea, just to recap, was to give an implicit order to the values, and, with the concrete numbers that we've attributed, we've tried to create distances between the different groups that were consistent with the criteria used.

For this section and the following ones for this dataset, what we are most interested is in correctly classifying the 'Killed' records. As such, and also since this dataset is highly unbalanced, accuracy won't give us much information, however, precision and recall will. With this said, we will be comparing the different pipelines using only those two metrics.

Imputation Method	Encoding Method	Precision		Recall	
		Naive Bayes	KNN	Naive Bayes	KNN
Custom Imputation	Custom Ordinal Encoding	**0.10**	**0.74**	**0.95**	**0.19**
	One-Hot Encoding	0.011	1.0	0.891	0.081
MV replacement with most common	Custom Ordinal Encoding	0.09	0.61	0.93	0.18
	One-Hot Encoding	0.008	1.0	0.947	0.039
MV replacement with constant	Custom Ordinal Encoding	0.09	0.53	0.96	0.13
	One-Hot Encoding	0.008	1.0	0.947	0.026

Regarding the table, One-Hot Encoding is quickly removed because for KNN the models were just predicting 'Injured', with very low recall. For the Naive Bayes, the true positives were indeed very good (Killed being predicted as Killed) however the false positives were quite bad. For these the accuracy dropped to around 50%. It is true that it is more 'important' to predict an accident as Killed than Injured, however, these are not good results.

If we now focus on the Custom Ordinal Encoding, for the KNN the results were practically the same, however for Naive Bayes there were some differences. On the three imputation methods, the rate of true positives was very good, however, the one with better precision, and that incorrectly classified Injured records as Killed the less, was the Custom Imputation. Therefore, and also because it is the one that holds more detail, we will be proceeding with the Custom Imputation and Custom Ordinal Encoding methodologies.

A final remark about the results, the one-hot encoding increases the number of features so much (to around 300) that it was predictable that the results for KNN (mostly the recall) would be quite bad (as we've observed) as it is more difficult to 'cluster' the neighbours. For Naive Bayes, even though the Recall improved comparing to the Custom Ordinal Encoding, the accuracy was bad (from 0.3 to 0.5).

For the Custom Ordinal Encoding with Naive Bayes as it is a probabilistic classifier, and because of what we've seen on the sparsity section, it is not strange that it behaved so well on all metrics (sure, not in Precision, but if we deep dive, this is in fact not that bad since the data is so unbalanced!). For the KNN and because of the sparsity of the data and its imbalance, the results were also not strange.

Ultimately, what this showed us is that, for KNN - a distance based algorithm -, we aren't able (with our configurations) an high recall (the metric that we've focused the most). We can hypothesise that this might be due to our custom encoding, however here we had to find a trade-off since we needed to transform our symbolic features to numeric ones.

Outliers Imputation:

Regarding outliers for our only numeric variable: PERSON_AGE, with the iqr criteria, 291 records are identified as outliers, and, with the stdev criteria, 2174. Regarding the stdev criteria, 42 of the 2174 found records belong to Killed records. Because of our already deeply unbalanced dataset, these records won't be removed nor be identified by us as outliers, as we don't find the forecast of this trade-off to be of value, this is a pragmatic decision! Regarding the remaining values for the stdev criteria, we are mostly talking about the very young and very old people. If we look at the variable boxplot, it doesn't make sense for us to be loosing information for young individuals, and, regarding old ones, for the stdev criteria starting from 70 years, every record is classified as an outlier. Even if they are 'theoretically' outliers, they present valuable information, and some of these values shouldn't be classified as outliers, as, once again, we can observe in the boxplot. With this, a good trade-off could be to use the iqr criteria, however, 14 of the 291 records are Killed (we still think this is a too high percentage of records from that class to drop), and, to just remove the remaining ones, they are so little that they won't impact any of our models, however, we will proceed with the removal whatsoever.

This way, and for this dataset, our only outlier imputation is for the values that are classified as such with the iqr criteria, and that are Injured and not Killed. ***CAN WE LOOK TO SYMBOLIC UNIQUE VALUES THAT ARE RARE AND DROP THEM?***

For the remaining features, they were Categorical and we then proceeded to encode them, as such, the traditional methods don't work here. Since other methods, as Local Outlier Factor are out of the scope of this work, we won't proceed to analyse outliers for the now transformed numeric features.

Scaling:

Regarding Scaling, our results didn't improve for Naive Bayes, but, after all, that is expected due to its probability and frequency base. For KNN, the results improved:

Previous best configuration	Scaling Method	KNN (% difference comparing with previous best results)	
		Precision	Recall
Custom Imputation + Custom Ordinal Encoding	**z-score**	**0.83 (+12%)**	**0.54 (+284%)**
	min-max	0.82 (+10%)	0.49 (+257%)

To compare both scaling methods is tricky. The best results were obtained with the z-score scaling. If we remember, only PERSON_AGE is numeric, and all other features were symbolic but then transformed into numerical ones. What this means is that, since our features distributions aren't Gaussian, and they are bounded, min-max scaling should work better. In this particular case, it didn't. However, we should also notice that the difference between z-score and min-max is quite small, in practice is the difference between having two Killed records being misclassified!

Ultimately, and by analysing the plots, indeed some of the features were able to be 'used' as having a Gaussian distribution simply because on the granularity part of the work we set values that were closer together and the result was a Gaussian distribution even if that wasn't our objective.

This way, and because the results were better with z-score, and because some features have indeed a Gaussian distribution (or similar), we are going to continue with the z-score scaling.

Balancing:

Regarding Data Balancing, we are going to test SMOTE, SMOTE + UnderSampling and OverSampling. Just using UnderSampling held week results, as such, here, we do undersampling of the majority class to 10 thousand entries, and SMOTE for the minority one, to match the majority class entries.

Previous best configuration	Balancing Method	Naive Bayes		KNN	
		Precision	Recall	Precision	Recall
Custom Imputation + Custom Ordinal Encoding	SMOTE + UnderSampling	**0.09**	**0.95**	**0.29**	**0.82**
	SMOTE	0.09	0.95	0.37	0.76
	OverSampling	0.06	0.95	0.50	0.72

Now we need to choose what balancing technique to use. We could, in theory, not use any, however, since our data is so unbalanced, it is not correct. One could argue that if without balancing, the results were so close, then, maybe, the models were already generalising the data, however, we can't be sure of that.

With this said, there's a moto that we want to follow: prevent every death we can! What this means is that, yes, for some of the configurations on the table above, our true positives (Killed being identified as Killed) decrease a little (from 70 to 56 in one case) but the false positives (Injured being identified as Killed) decreased from around 700 to 100. The change magnitude of the false positives was much bigger (and better), however, we need to make a choice, and here, ours is to preserve every life as best as we can! (Obviously, depending on the context, we could argue that the monetary cost of trying to save so many more lives - the false positives - could maybe not be bearable and put in risk the resources for the true positives, but that's not the choice we are making).

This way, and to maximize the true positive rate for KNN, we want the model with the best recall, which would be using SMOTE + UnderSampling. Moreover, for Naive Bayes, the best model is a tie between SMOTE and SMOTE + Undersampling. Since we choose SMOTE + Undersampling for KNN, we do the same for Naive Bayes.

A note must be done that without balancing the data, the results were better **for the Naive Bayes**. One can probably assume that this is due to points on the boundaries of the decisions between the two classes. Without balancing, the recall was the same, but the precision was 0.10 against 0.09. However, and because the gains for the KNN were quite impactful, and also because the difference for the Naive Bayes is quite small, we will proceed with the balancing method **SMOTE + UnderSampling**.

2.1 Feature Engineering:

Feature Selection:

Previous best configuration	Feature Selection	Naive Bayes		KNN	
		Precision	Recall	Precision	Recall
Custom Imputation + Custom Ordinal Encoding + (SMOTE + UnderSampling Balancing)	Redundant Variables: 0.9	0.11	0.95	0.29	0.82
	Redundant Variables: 0.7	0.17	0.92	0.29	0.82
	Variance: 0.9	0.13	0.91	0.01	0.86

With a threshold of 0.9, for the redundant features, the features to be dropped are: ['POSITION_IN_VEHICLE']. With 0.7 are: ['CONTRIBUTING_FACTOR_2', 'CONTRIBUTING_FACTOR_1', 'POSITION_IN_VEHICLE', 'PED_ACTION'].

For KNN the results are already an improvement over the last step. Between the twos, and by observing the Confusion Matrix it is possible to observe that with a threshold of 0.7 the results are the best (4 less false positives). For the Naive Bayes, if we want to, again, maximize the number of True Positives, then we go with the threshold of 0.9, at the expense of Precision.

Since the maximisation of True Positives was the last step criteria we are going to proceed with it, as such, and also because the difference for both thresholds for KNN is negligible, the best threshold is 0.9 for the redundant variables. Also, comparing with the variance study feature selection, for kNN the results got significantly worse, and for Naive Bayes they didn't improve either.

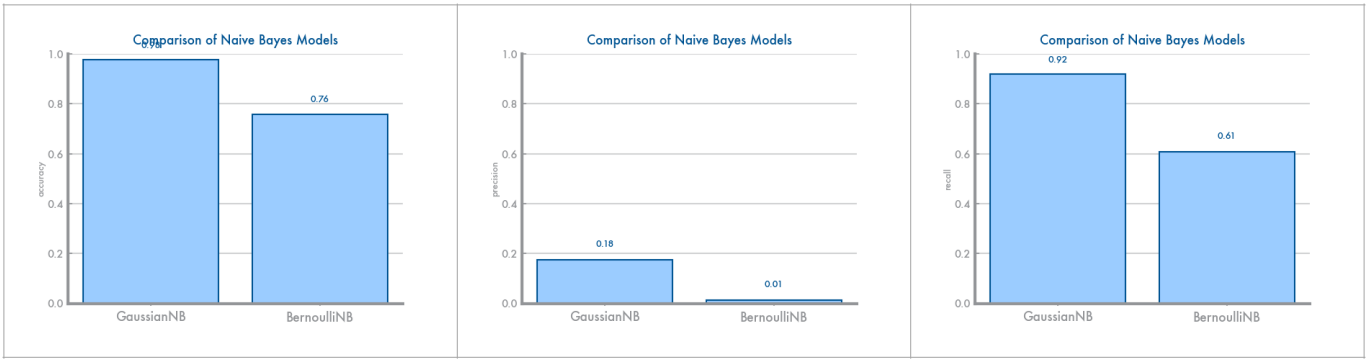
This way, we are going to proceed with the feature selection method using the redundant analysis with a threshold of 0.7.

Feature Extraction:

Feature Generation:

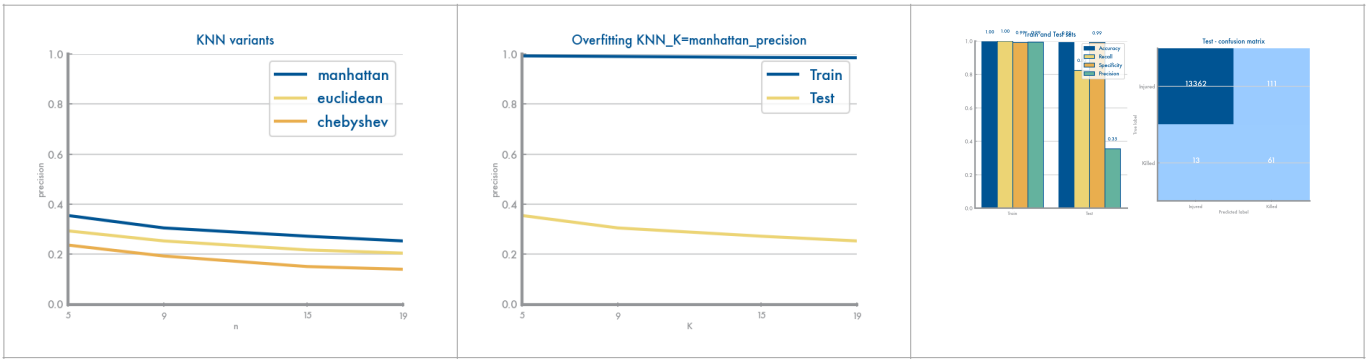
3 Classification:

Naive Bayes:



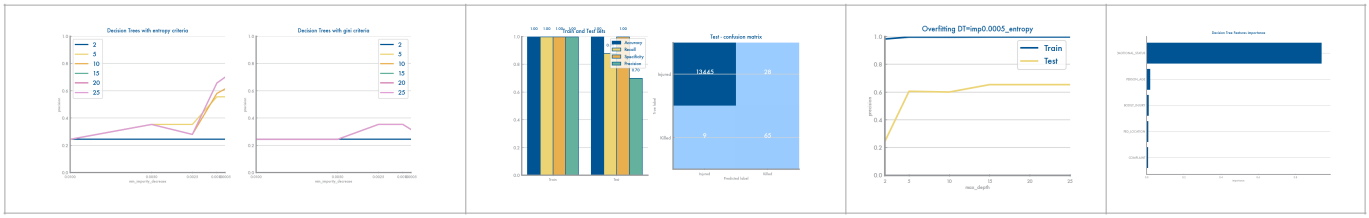
The best model is with the GaussianNB. As we've said in the data scaling section, some features have a gaussian distribution, that's why the GaussianNB works the best.

KNN:



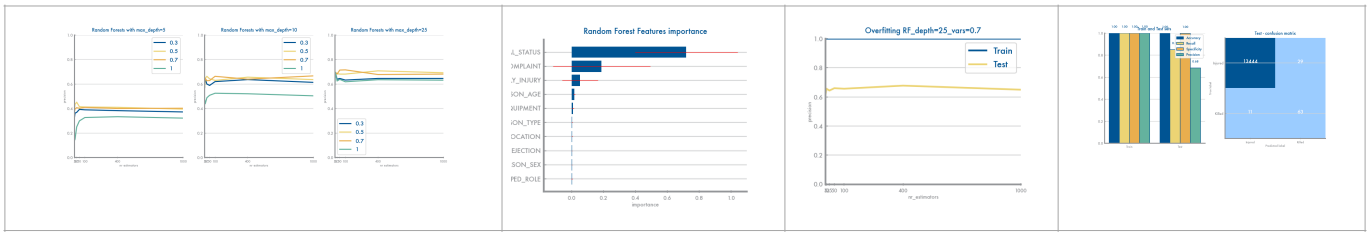
Best model with 5 neighbours and manhattan measure. The model entered in overfitting for the accuracy, precision and score. This was expected due to the SMOTE.

Decision Trees:



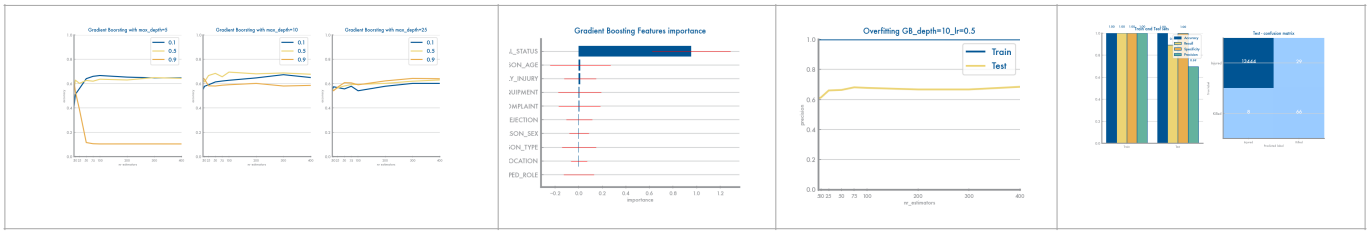
Best results achieved with entropy criteria, depth=15 and min_impurity_decrease=0.0005 ==> precision=0.65

Random Forests:



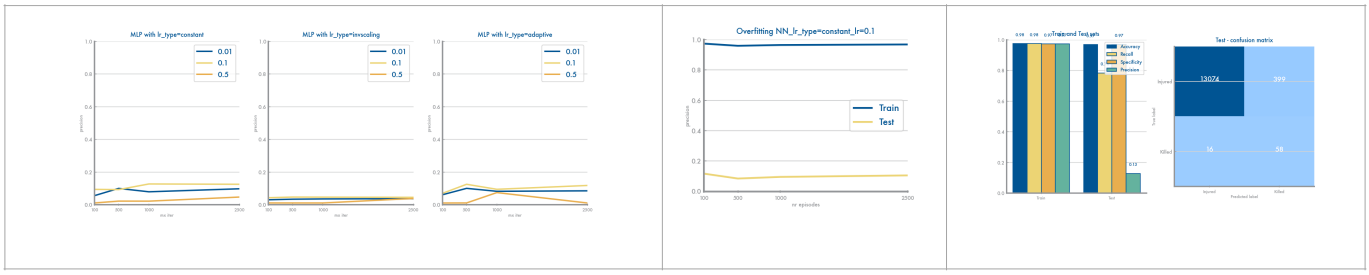
Best results with depth=25, 0.7 features and 10 estimators, with precision=0.65

Gradient Boosting:



Best results with depth=10, learning rate=0.50 and 100 estimators, with precision=0.69

Multi-Layer Perceptrons:



Best results with lr_type=constant, learning rate=0.1 and 100 max iter, with precision=0.37

For all classifiers that have a feature importance graph, the most important feature always was the EMOTIONAL_STATUS.

4 Clustering:

5 Association Rules:

6 Time Series Analysis:

Matrix Profile:

Forecasting:

7 Critical Analysis: