CentraleSupélec - 2EL1730 - MACHINE LEARNING

Guilherme Mertens de Andrade Ricardo Chiquetto do Lago Lucas Tramonte João Pedro Regazzi

1 Introduction

In order to classify a given geographical area into the classes ' Demolition', 'Road', 'Residential', 'Commercial', 'Industrial', 'Mega Projects', this study proposes the use of different machine learning models, such as Logistic Regression, Random Forest, Xgboost and neural networks.

The database contains geographic features obtained from satellite images of 100 different cities between January 2014 and July 2020 [1]. Therefore, feature engineering techniques were proposed with the aim of discretizing classes more efficiently.

2 Data Treatment

2.1 One-hot encoding

The first token decision was to apply hot encoding in the flags listed as "urban types" and "geograph types". Then, eleven binary features were created from the first category and 5 other features from the second one. However, it should be pointed out that this practice leads to greater sparsity, which can also cause problems for the classification, and deserves due attention when choosing features later on.

2.2 Fixing dates out of order

The group noted that the dates were not in order, i.e., the date 0 not always occurred before the date 1, and the same to the others. So, it was decided to reorganize the data in order to increment sequentially the dates. Then, the features linked to the dates as "$(change\_status)$" and the colors means and standard deviation were relocated together.

2.3 Solving lack of data

It was also noted that there was a lack of data in some features, so that a lot of samples had "nan" values inside. So, to avoid loosing data, it was used k-Nearest Neighbors algorithm to replace the "nan" values for estimated ones. The idea was simple: apply k-NN this $k = 5$ to obtain the approximation between samples from the same class.

2.4 Strings to integer Maps

The first obvious mapping action was the replacement of the "$(change\_type)$" categories to the related integers, as defined in Kaggle platform. In addition, the "$(change\_status)$" in all dates were replaced appropriately too for creating a numerical scale from the vegetation to the constructed building in operation, as described in table 1

Table 1: Substituition dictionary for 'change status'

Status	Code
Greenland	1
Land Cleared	2
Materials Introduced	3
Prior Construction	4
Excavation	5
Construction Started	6
Construction Midway	7
Materials Dumped	8
Construction Done	9
Operational	10

3 Feature Engineering

3.1 New Geometric Features

From the given polygons, it was possible to obtain a lot of important features that were used to distinguish between the change types. The centroid values of x and y were extracted, as so the area and the perimeter that together can build to another feature called "compactness" defined as $4\pi\frac{\text{area}}{\text{perimeter}^2}$ that helps one to distinguish between thin structures (as "roads") from the other polygons closer to the circle. Other proprieties were extracted too, as the "angle", defined as the $arctg$ of the ration of the projections in y and x axis, and the number of vertices of the polygon that usually is four, but not always.

To compare the quadrilles, the sum of the differences between the opposite sides of the figures was calculated to check if it was a parallelogram or not, and with these differences, two features were created with a different strategy when the number of vertices was not four, as expressed in equation 1 , assuming that L is the constant threshold that separate parallelograms and non-parallelograms, N is the number of vertices and a, b, c and d are adjacently the sides of the quadrille.

$$f_1(x) = \begin{cases} |(a-b)| + |(c-d)| & \text{if } N(x) = 4 \\ 10 \cdot L & \text{otherwise} \end{cases} \tag{1}$$

$$f_2(x) = \begin{cases} |(a-b)| + |(c-d)| & \text{if } N(x) = 4 \\ 4 \cdot N(x) \cdot L & \text{otherwise} \end{cases} \tag{2}$$

To analyze mostly the non-quadrilles, three features were established: the area of the minimum rectangle that area of the smallest rectangle that encloses the polygon (to obtain the real dimension of a "Mega project", for example), the boolean convexity of the polygon and the proportional difference between the convex hull's and the polygon's areas. Finally, the centroid x and y of each sample's polygon is used to cluster the sample's by geographical position using the k-means method. Using the elbow-method, the best value of k is chosen as four, capturing the graphical positioning.

3.2 Deltas

Just to check later the usability of all the possible features in discrimination action, a lot of delta variables were created subtracting time-adjacent values of colors' means and standard deviations, values of dates or total changes, comparing the first value with those from the last date. It was also created the number of unique values in the status change columns.

3.3 Slopes and Rates

Analyzing the change of values during the time, there were calculated the rate of changing the color proprieties. It was done with a linear regression of two parameters with the five given values of each propriety and the subsequent extraction of the line's slope. In the same way, the "construction rate" or "civilizing rate" was created analyzing the change in the "$(change\_status)$" during the time.

It was also created the Coefficient of Variation for each image, which was the ratio of the sum of standard deviations to the sum of means of each color.

Finally, another k-means model (k=4) was created to segregate all the features related to the mean of each color in classes, reducing the dimensionality of the dataset but maintaining the important information.

4 Feature Selection

To select the best features, we have conducted an iterative process of testing the model, examining the confusion matrix for both the training and testing data, and identifying features that effectively discriminate between classes exhibiting false negatives or positives, by graphical analysis and finally testing the model again. Additionally, we employed other methods outlined below.

4.0.1 Avoiding Correlation

The first important criteria during feature selection is avoiding correlation between the selected features. Although, analyzing the combination of each pair from the 150 features could be very exhausting. So, the group decided to use other criteria before and select from the result the independent ones, using the knowledge of features generation or some specific analyses as the correlation coefficient.

4.0.2 ANOVA

ANOVA is a statistic test to check if the distributions are equal or not. Assuming $H_0 : \mu_0 = \mu_1 = \mu_2 = \mu_3 = \mu_4 = \mu_5$, such that μ_i is the mean value of a feature in the class i, the test was used to check what features have the biggest F-value to reject H_0. It could be used to compare all the classes, or to select a feature that distinguish well two classes i and j if $H_0 : \mu_i = \mu_j$. In table 2 there are the features with biggest F-value to reject the first H_0.

Table 2: The biggest F-values that rejects $H_0 : \mu_0 = \mu_1 = \mu_2 = \mu_3 = \mu_4 = \mu_5$

Feature	F-value
Dense Urban	835.72
num_vertices	901.36
change_status_value_date2	1314.37
Industrial	1423.08
change_status_value_date0	3683.61
centroid_x	4310.79
centroid_y	5405.23
change_status_value_date3	6660.29
compactness	13095.25
civilizing_rate	20371.15
change_status_value_date4	25579.74

4.0.3 Graphical analysis

To inspect the discrimination power of an specific feature, the graphical visualization of the mean and the std deviation of the feature in each class. This approach is not scalable, but is simpler to human comprehension and useful in punctual analyses. For example, in figure 2 it is visible that the feature "civilizing rate" and "rect area" are good to identify classes 0 and 5 respectively ("Demolition" and "Mega projects").

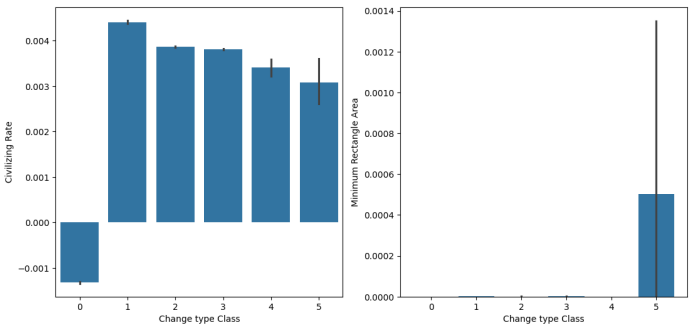

Fig. 1: Bar plot of mean and standard deviation of two features for the six classes

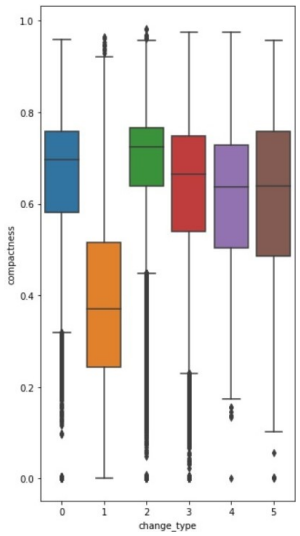

Fig. 2: Box plot of mean and standard deviation of the compactness features

4.0.4 RF and XGboost return: Information Gain

The Random Forest and the XGboost algorithms provided by the Scikit-learn library in Python have en attribute called "$feature\_importances\_$" that returns, after the data fitting, the importance that each feature had in the decision process in all trees. Then, after a program execution, it is possible to check the real contribution that each feature gave to the discrimination task.

4.0.5 Confusion Matrix

Another useful tool for feature selection is the confusion matrix, which allows us to examine instances of class confusion. In the image below, we observed that the model struggles to differentiate between the "Commercial" and "Residential" classes. Armed with this insight, we were able to search for features that effectively distinguish between these two classes. Additionally, the confusion matrix provided valuable insights into the initial model's failure to predict the "Industrial" and "Mega Projects" classes, which was attributed to misrepresentation.

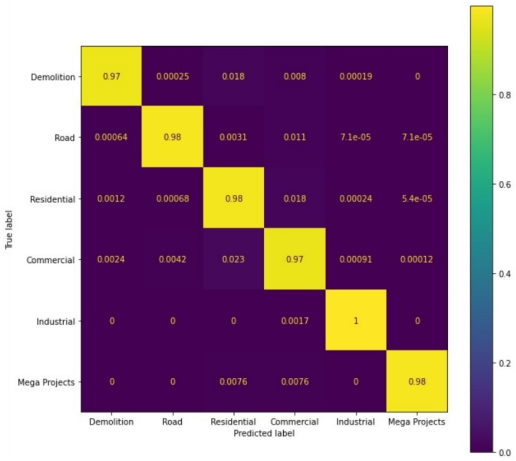

Fig. 3: Confusion Matrix plot with normalization

4.0.6 Recursive Feature Elimination

The Recursive Feature Elimination (RFE) is a wrapper-type feature selection algorithm that removes the least important features from the dataset until the desired number of features is reached.

Additionaly, we conducted a RFE with cross-validation (RFECV) aiming to optmize the number of features selected. In this regard,

due to computational constraints, RFECV was not feasible for all features, so we ran it only on 42 features previously identified as useful through prior feature selection methods. The algorithm identified 25 features to obtain optimal performance but, despite that, we found that using additional features not included in this selection contributed to a higher Kaggle score. This observation emphasizes the importance of a comprehensive approach to feature selection, considering both automated techniques like RFECV and empirical validation through performance evaluation.

Table 3: Features used in the model

Features using the RFE method	Features
geography_typeDense Forest	geography_typeCoastal
geography_typeGrass Land	geography_typeLakes
geography_typeSparse Forest	urban_typeUrban Slum
geography_typeFarms	change_status_date4
urban_typeSparse Urban	date0
urban_typeDense Urban	img_red_mean_prop_rate
centroid_x	img_green_mean_prop_rate
centroid_y	img_blue_mean_prop_rate
angle	area
compactness	date_delta_total
num_vertices	change_status_date3
length_dif1	date1
length_dif2	img_red_std_rate
dif_convex_prop_area	paralelogram
geo_cluster	change_status_date0
date_delta_total	rect_area
civilizating_rate	convex
length	color_cluster
urban_typeIndustrial	covdiff1
	covdiff2
	covdiff3
	covdiff4
	covdiff5
	n_status_change

5 Model

Random Forest showed the best results among the models used, which can be seen in Table 4. The fact that it is a bagging model helped to avoid overfitting in the prediction, producing a good F1-score.

Table 4: F1-Score comparison of each model.

Model	F1-Score	Kaggle score
Random Forest	0.72	0.96736
3x RF	0.76	0.96770
Logistic Regression	0.33	0.46
Neural network	0.6	0.39

It was noticed that there was an imbalance of classes in the dataset, which meant that the models did not correctly classify the 'Industrial' and 'Mega Projects' classes. In view of this, the following weight distribution was used to balance the less represented classes : $class\_weight = \{0:1, 1:1, 2:1, 3:1, 4:10, 5:100\}$.However, a higher F1-score was not obtained with this method

A pipeline with the model and the RandomUnderSampler function was created to under-sample the majority classes[2], but without good results. Although the random resampling strategy improved the results in the test, there was no increase in the F1-score in the kaagle, so it was not used in the final prediction

The little different model was that called "3x RF" in the table 4. The idea was to use a Random Forest (RF) algorithm to avoid overfitting and to divide firstly the data between two groups: one with the classes 0,1,2 and 3 and the other with the less represented 4 and 5. Then, another RF was modeled to decide the class of the samples in each group. That was interesting, because permitted to better chose the features in each classification model, maximizing the gain of information. In addition, the lack of data for the last two classes was not a problem in a model with only this two classes. This method was particularly intriguing because, with a single tree, it required development until each leaf contained only one sample to predict the misrepresented classes, leading to overfitting. With this alternative method, it became feasible to construct trees with a minimum sample per leaf greater than 1.

For the initial tree, which distinguished between less representative and highly representative classes, the optimal model was attained with 200 estimators. Subsequently, for the tree distinguishing among the highly representative classes, 300 estimators were utilized. Finally, for the last tree, 100 sample trees were employed.

6 Avoiding overfitting

The split between the training data and the test data was 80% and 20% respectively. However, as the latter classes were less represented in the model, there was a loss of generalization from these classes to the test data, which causes a reduction in the F1-score.

The first important strategy to avoid overfitting was well selecting the features used. So, all the methods presented in section 4 were essential to not increasing the variance.

Some dimension reduction algorithms were tested, but the feature selection was much more effective than FDA, for example, because that one increased the bias of the model prediction.

Last but not least, the hyperparameters used in RF was also relevant to void it. For example, in each tree had just 60% of the features and the number of trees was also limited to not dick the model.

Finally, we also adapted our strategies and model hyperparameters according to the discrepancy between the kaagle result and the training result, in order to control overfitting.

7 Conclusion

Although strategies related to classifiers are efficient, such as tuning hyperparameters and using aggregation models, we realized that the pre-processing and feature engineering stages were essential to achieving good results. A more in-depth study of how features discretize classes, using feature importance techniques, was important to understand their impact on classification. Of all the models used, the Random Forest showed the best results, together with the selection of features from the RFE.

References

[1] S. Verma, A. Panigrahi, and S. Gupta, "Qfabric: Multi-task change detection dataset," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2021, pp. 1052–1061.

[2] M. A. Tahir, J. Kittler, and F. Yan, "Inverse random under sampling for class imbalance problem and its application to multi-label classification," *Pattern Recognition*, vol. 45, no. 10, pp. 3738–3750, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0031320312001471