Medium article - Predicting Hazardous Seismic: Evaluating Scikit Learn Machine Learning models

A newbie evaluation on trying out new Machine Learning models for Classification and Data Augmentation to support better results.

First, we'd like to thank Nabanita Roy for pointing out this very interesting small dataset, doing the Data Exploration, Data Preparation and first models and performance analysis as Medium articles with a linked notebook!
The performance of those first models was very low for Recall: 0.06 the best performing model SVM with the raw dataset: this means that out of all the Actual Positive (hazardous) shifts, only 6% of the shifts have been predicted as Positive.

Predicting Positive here is predicting " the possibility of hazardous situation occurrence, [where] an appropriate supervision service can reduce a risk of rockburst (e.g. by distressing shooting) or withdraw workers from the threatened area. Good prediction of increased seismic activity is therefore a matter of great practical importance. "

We wouldn't want to send miners into a mine with a substandard model!

On the other hand, Precision for the best performing model is only 0.67, which means that out of all the shifts predicted as hazardous, (1 - 0.67) = 23% were actually low risk, which would mean a non insignificant number of shifts where miners may have been told to stay home, and thus a lower productivity.

Shared Colab notebook

Introduction to Scikit Learn

Understanding Classification Models for Supervised Machine Learning:
(1.4,1.5,1.6,1.9,1.10,1.11,1.17)

Understanding model performance metrics :

Data exploration

• Understand the feature data (data types, missing values, outliers, etc): see Data Exploration, Data Preparation
• Visualization (boxplots, scatter plots, correlation matrix, etc)- **see** Data Exploration, Data Preparation

Feature engineering

see [Data Exploration, Data Preparation]

As a small improvement, we replaced the One-hot encoder for the Categorical variables with Numerical encoding as
+the assessment coding being graded from Low to High, coding numerically adds meaningful information.

```
[Attribute
information](https://archive.ics.uci.edu/ml/datasets/seismic-bumps):
1. seismic: result of shift seismic hazard assessment in the mine
working obtained by the seismic
method (a - lack of hazard, b - low hazard, c - high hazard, d - danger
state);
2. seismoacoustic: result of shift seismic hazard assessment in the
mine working obtained by the
seismoacoustic method;
3. shift: information about type of a shift (W - coal-getting, N
-preparation shift);
```

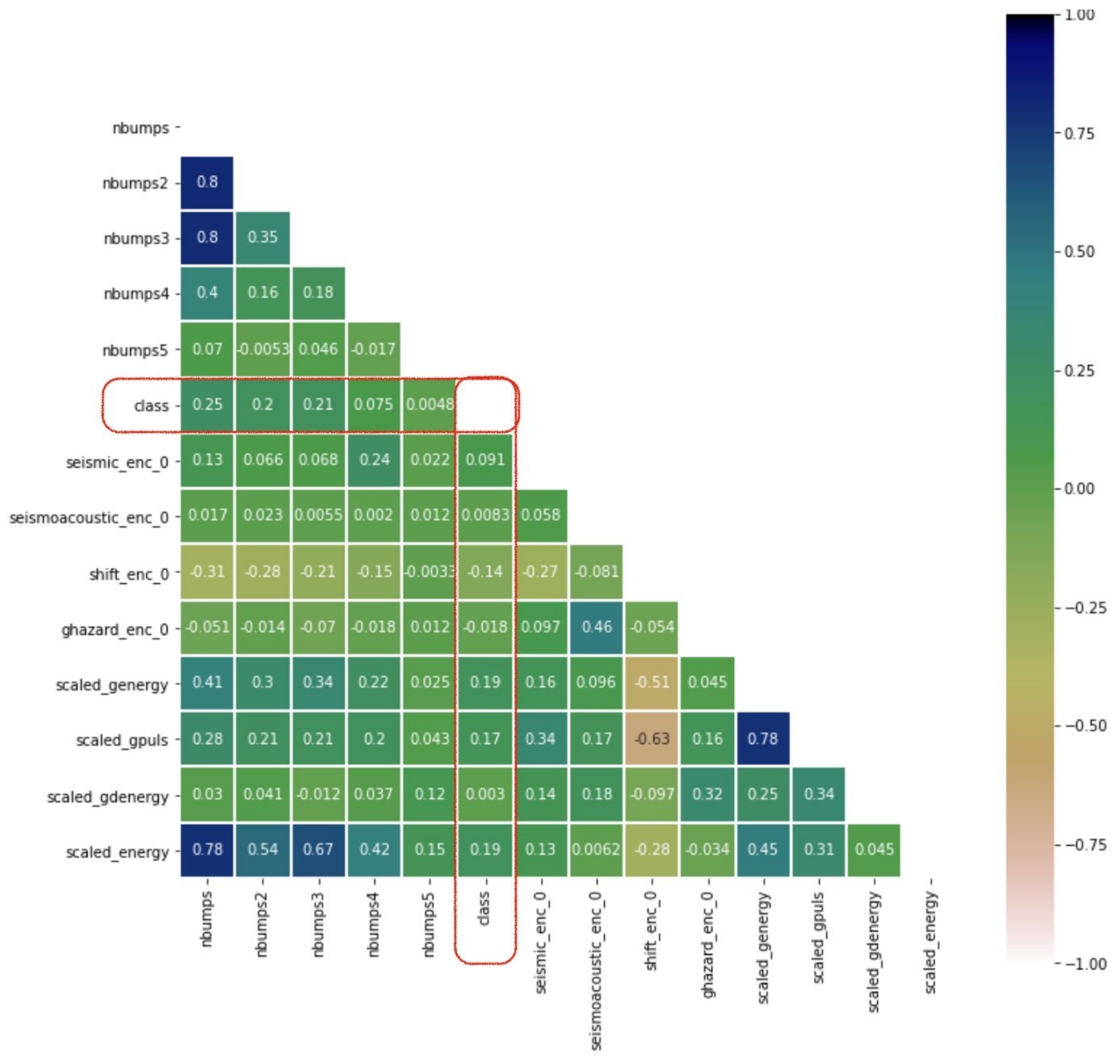
This is done with a simple mapping for the known values:

```
df[col_name + "_enc_0"] = df[col_name].map({'a': 0, 'b': 1, 'c' : 2,
'W' : 0, 'N' : 1})
```

It also makes interpretability easier as we will see later.

Correlations between features and the target

The final features correlated to the Target (Class):



There are strong correlations between some features which might need review if we had access to the data gathering process.

Data preparation

• Dealing with outliers -
• Dealing with missing values - no missing value
• Feature encoding (encoding categorical features): **see** Data Exploration, Data Preparation
• Feature selection
• Feature scaling: see **see** Data Exploration, Data Preparation
• Handling Imbalance:
see Data Exploration, Data Preparation : Stratified Shuffle Split to split Train and Test data and proportionally distribute the target classes.

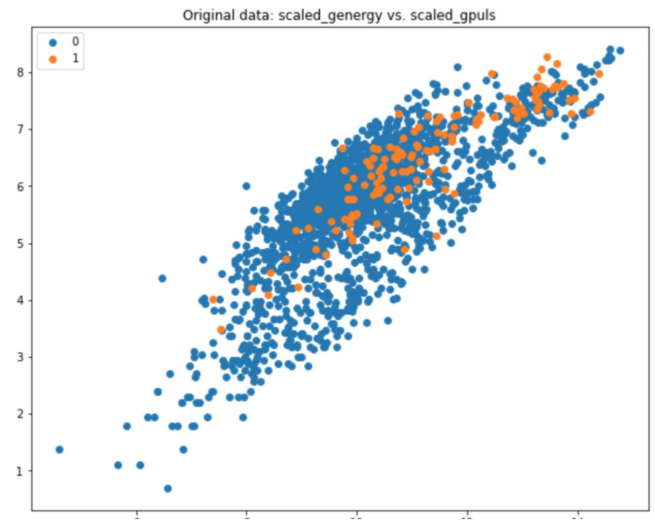
Augment data using Synthetic Minority Oversampling Technique (SMOTE) .

The original data has a lot more Class 0 than Class 1 points which is obvious in the data visualisations and is likely to impact the performance of all models. As suggested by Nabanita Roy , we tried the Synthetic Minority Oversampling Technique (SMOTE) oversampling technique.

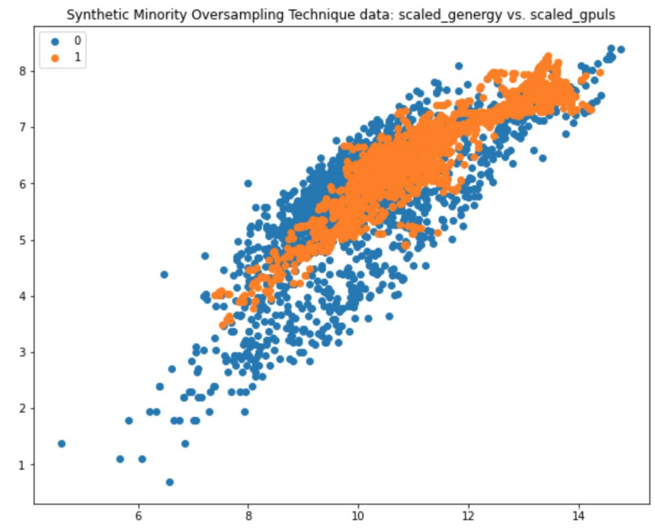
```
print('Original dataset shape %s' % Counter(y))


sm = SMOTE(random_state=42)
X_res, y_res = sm.fit_resample(X_train, y_train)


print('Resampled dataset shape %s' % Counter(y_res))
```

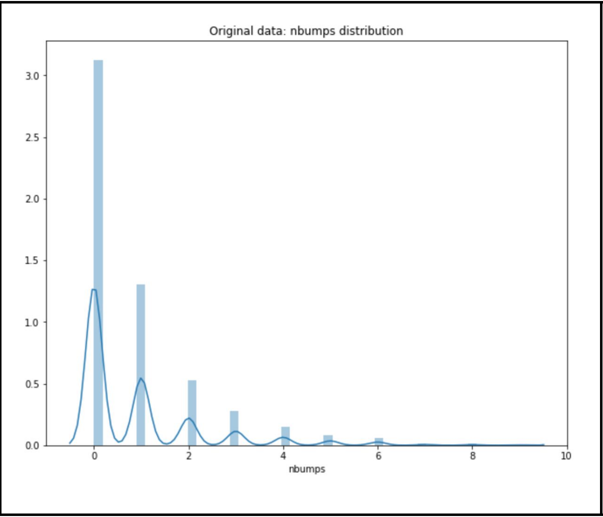


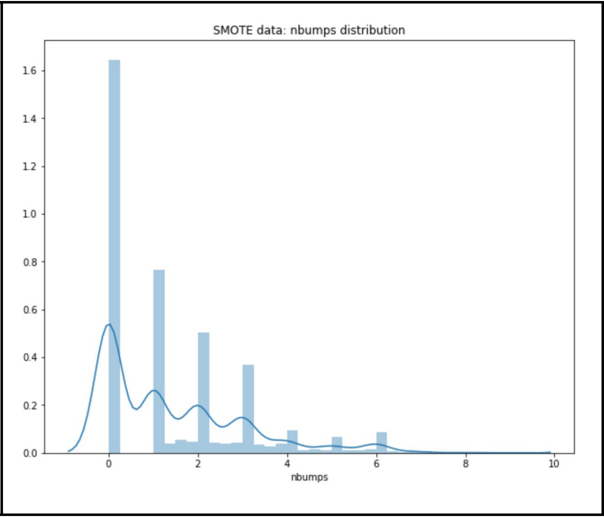
```
Original dataset shape Counter
({0: 2414, 1: 170})
```



```
Resampled dataset shape
Counter({0: 1931, 1: 1931})
```

Interestingly, for discrete features like nBumps (the number of seismic bumps recorded within the previous shift), new rows have some non integers values!





Note, we augment ONLY the Training set but we will test on the Original Test set, so as to make sure this transformation works in real-life.

Building and Evaluating Models

• Train many models from different categories (e.g., linear, naïve Bayes, SVM, kNN, decision trees, Random Forest, etc.) using standard parameters.
• Measure and compare their performance.
• Debug ML models and analyse the types of errors the models make.

Fine Tuning and Optimization

• Perform hyper-parameter optimization
• Incorporate transformation choices from part 2 as part of the hyper-parameter optimization
• Try Ensemble methods
• Finally assess the generalization capability of your model on the test set:
For example, for the Random Forest Model optimised for Recall with the SMOTE Training data:

- Training set Recall score: 0.9269808389435525,
- Test set Recall score: 0.38235294117647056
- => Overfitting.

Model Test Results

With the numerical encoding for the Categorical variables, we get the following results:

	scoring_technique	accuracy	precision	recall	f1-score
RFC SMOTE max_leaf_nodes=10	max_leaf10	0.84	0.23	0.62	0.34
DecisionTree SMOTE max_depth=3	None	0.86	0.24	0.50	0.32
XGBoost SMOTE	[Baseline]	0.86	0.24	0.50	0.32
XGBoost SMOTE recall	[recall]	0.86	0.24	0.50	0.32
DecisionTree SMOTE-shift max_leaf_nodes=10	None	0.85	0.22	0.53	0.31
RFC SMOTE max_depth=3	max_depth3	0.82	0.21	0.62	0.31
AdaBoost SMOTE data	None	0.89	0.24	0.32	0.28
AdaBoost SMOTE DTD_Max_Depth=3	None	0.92	0.32	0.24	0.27
RFC SMOTE	Recall	0.90	0.22	0.24	0.23
DecisionTree SMOTE max_leaf_nodes=10	None	0.72	0.14	0.62	0.22
Decision Tree SMOTE Data	None	0.89	0.16	0.18	0.17
AdaBoost SMOTE finetuned	f1	0.88	0.15	0.18	0.16
KNC Optimised SMOTE	[recall]	0.86	0.12	0.18	0.14
Decision Tree Raw Data	None	0.88	0.13	0.15	0.14
KNC SMOTE 1	None	0.83	0.08	0.15	0.10
KNC Model 1	['precision']	0.93	0.33	0.06	0.10
SVC Original Dataset	f1	0.90	0.12	0.09	0.10
SVC SMOTE	[recall]	0.90	0.12	0.09	0.10
KNC Baseline	None	0.93	0.33	0.03	0.05
KNC Model 5	['precision']	0.93	0.33	0.03	0.05
KNC Model 2	['precision']	0.93	0.25	0.03	0.05
AdaBoost Raw data	None	0.93	0.17	0.03	0.05
KNC Model 3	['recall']	0.89	0.04	0.03	0.03S
KNC Model 4	['f1']	0.89	0.04	0.03	0.03
XGBoost	[Baseline]	0.93	0.00	0.00	0.00

Note, with the previous One Hot encoders, we got slightly worse results:

	scoring_technique	accuracy	precision	recall	f1-score
XGBoost SMOTE	[Baseline]	0.85	0.21	0.47	0.29
XGBoost SMOTE recall	[recall]	0.85	0.21	0.47	0.29
RFC SMOTE max_leaf_nodes=10	None	0.81	0.19	0.59	0.29
DecisionTree SMOTE-shift max_leaf_nodes=10	None	0.80	0.18	0.59	0.28
DecisionTree SMOTE max_depth=3	None	0.82	0.19	0.53	0.28
DecisionTree SMOTE max_leaf_nodes=10	None	0.77	0.17	0.68	0.28
RFC SMOTE max_depth=3	None	0.79	0.18	0.59	0.27
AdaBoost SMOTE DTD_Max_Depth=3	None	0.91	0.25	0.18	0.21
Decision Tree Raw Data	None	0.90	0.19	0.18	0.18
AdaBoost SMOTE finetuned	f1	0.88	0.16	0.21	0.18
AdaBoost SMOTE data	None	0.86	0.14	0.21	0.16
KNC SMOTE 1	None	0.82	0.12	0.26	0.16
KNC Optimised SMOTE	[recall]	0.82	0.12	0.26	0.16
KNC Model 4	['f1']	0.93	0.30	0.09	0.14
RFC SMOTE	Recall	0.85	0.11	0.18	0.13
KNC Model 3	['recall']	0.89	0.13	0.12	0.12
SVC Original Dataset	f1	0.92	0.20	0.09	0.12
Decision Tree SMOTE Data	None	0.84	0.08	0.15	0.11
SVC SMOTE	[recall]	0.88	0.09	0.09	0.09
XGBoost	[Baseline]	0.94	1.00	0.03	0.06
AdaBoost Raw data	None	0.93	0.17	0.03	0.05
KNC Baseline	None	0.92	0.00	0.00	0.00
KNC Model 1	['precision']	0.93	0.00	0.00	0.00
KNC Model 2	['precision']	0.92	0.00	0.00	0.00
KNC Model 5	['precision']	0.93	0.00	0.00	0.00

These results summarise the finding of a number of options, and we could have tried much more with more time only with Scikit-learn options!

- RandomForestClassifier: RFC
- DecisionTreeClassifier: Decision Tree
- AdaBoostClassifier: AdaBoost
- C-Support Vector Classification: SVC
- K-Neighbors Classifier: KNC

- XGBoost (not in Scikit-learn)

They are ordered by f1-score, so as to try and make a good compromise between Recall and Accuracy, however if Recall is the most crucial criteria the order would have slightly changed.

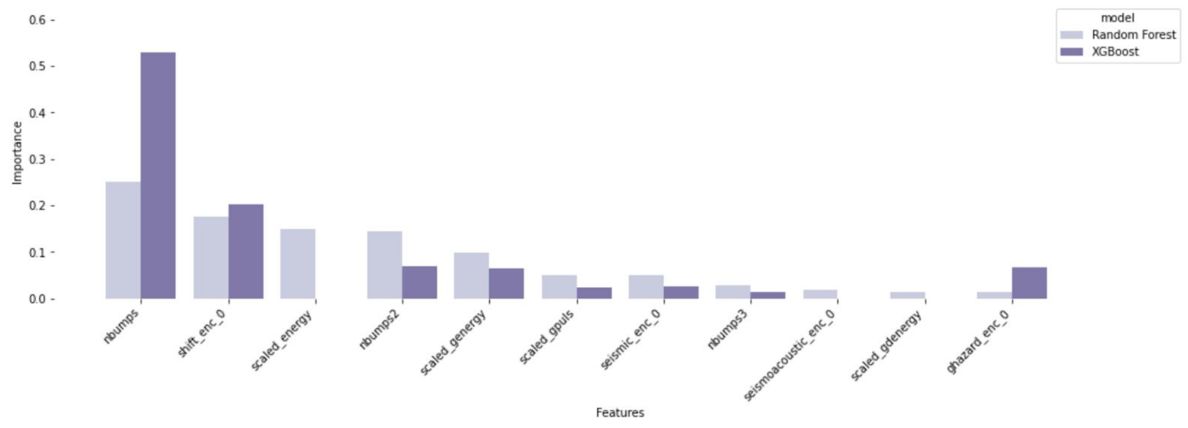
Note that the Test set is pretty small, as the full raw dataset is quite small: 2584 rows x 16 features. Sor for some of the Top performing results:

- The RFC SMOTE max_leaf_nodes=10 Confusion Matrix has the top Recall score: 20 / (20 + 14) = 0.59:
    - [[398 85]
    - [14 20]]
- The XGBoost SMOTE Confusion Matrix Recall score is: 16 / (16 +18) = 0.47
    - [[424 59]
    - [18 16]]

=> Pretty small difference in absolute numbers!

Explainability

Some models expose the Feature Importances which are great both to inform feature selection and checking for data leakage, as well as getting feedback and trust from the customer.
In this case we'll have a look at the top performing models with feature importance support, as well as the best simple Decision Tree:

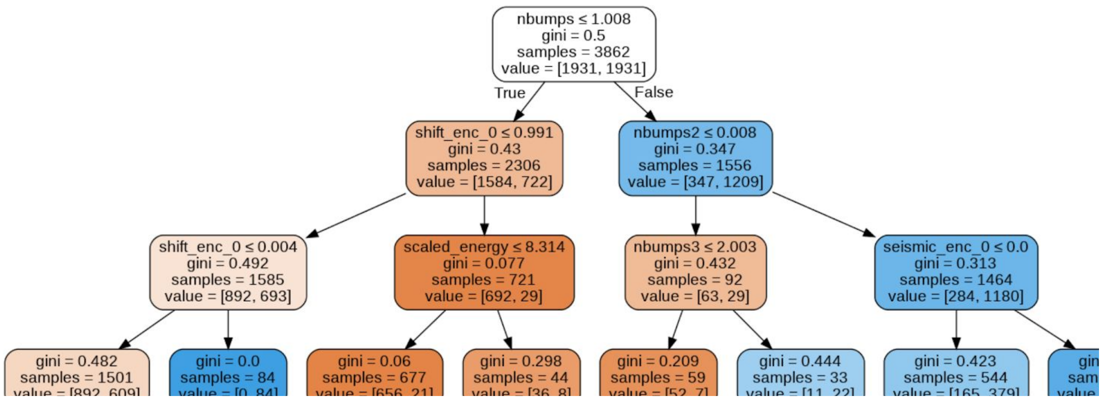


Nbumps (the number of seismic bumps recorded within the previous shift) is understandably a good predictor for hazard in the current shift, as well as nbumps2 (the number of seismic bumps in energy range [10^2,10^3]) registered within the previous shift).

Scaled energy: total energy of seismic bumps registered within the previous shift; highly correlated with nbumpsN, maybe should be removed from features.

Shift_enc_0: information about type of a shift (W / 0 - coal-getting, N / 1 -preparation shift) is a bit more concerning, could a Preparation shift be correlated to a Hazardous shift because the engineers thought it was risky? Could there be some Data leakage here?

.



Conclusions

1. Training any model on the augmented dataset (SMOTE) improves the model performance significantly
2. XGBoost wins by a small margin, based on shallow Decision Trees
3. The RandomForestClassifier and Decision Tree Classifier are doing well, THOUGH not with the GridSearchCV *Optimised* version! It seems like optimising against the SMOTE dataset can be counter-productive and choosing values like max_leaf_nodes=10 intuitively adapted to a small dataset works here.

What we have learned

- Finding a suitable dataset is a hard task: thanks again Nabanita!
- To validate a number of assumptions (what metrics is most important, why does there seem to be a correlation between the Shift type and the hazard Classification, could that be data leakage..), we would need to talk directly to the users.
- What we didn't expect: it takes a lot of time, both to run the optimisation and to try and understand the model options!
- All feedback is very much appreciated!

Team Member

Catherine Lalanne (Linkedin : https://www.linkedin.com/in/catherine-lalanne-85b5ba/)
Luciana Azubuike (Linkedin : https://www.linkedin.com/in/i-am-luciana-azubuike/)
Heejin Yoon (Linkedin : https://www.linkedin.com/in/heejin-yoon-429837190/)