Earthquake prediction using python

AI _ phase 3

Features:

The initial acoustic signal is decomposed into segments with 150000 rows per segment, which suggests that the training dataset has 4194 rows. Features are calculated as aggregations over segments. For more details see, for example, here and here.

Baseline model:
Before we start with the feature selection, we calculate feature importance as it is explained here and train the baseline model on the 15 most important features.

```
From earthquake import config, utils:

# load training set
data = utils.read_csv(config.path_to_train)
# create list of features
features = [column for column in data.columns if column not in ['target', 'seg_id']]
# display importance
best_features = utils.feature_importance
```

We train the model using CatboostRegressor with default parameters and evaluate the performance with a stratified KFold (5 folds) cross-validation.

```
Import numpy as np
from sklearn.model_selection import cross_val_score
from catboost import CatBoostRegressor

# set output float precision
np.set_printoptions(precision=3)
# init model
model = CatBoostRegressor(random_seed=0, verbose=False)
# calculate mae on folds
mae = cross_val_score(model, data[best_features], data['target'],
   cv=5, scoring='neg_mean_absolute_error', n_jobs=8)
# print the results
print('folds: {}'.format(abs(mae)))
print('total: {:.3f}'.format(np.mean(
```

To avoid a potential overfitting, we employ a genetic algorithm for feature selection. The genetic context is pretty straightforward. We suppose that the list of features (without duplicates) is the chromosome, whereas each gene represents one feature. n_features is the input parameter controlling the amount of genes in the chromosome.

```
Import random:
```

```
class Chromosome(object):
    def __init__(self, genes, size):
        self.genes = random.sample(genes, size)
```