Supervised Learning - Regression

Iago García Suárez[1,1*]

Corresponding author(s). E-mail(s): iago.garcia@alu.uclm.es;

Abstract

For this project, a dataset with records related to an insurance company was given. The objective is to design a model which will be able to predict the final Insurance Cost for a given client, using its personal data, economic situation, working conditions and claim description.

Keywords: Supervised Learning, Regression, Boosting, Hyperparameter Optimization

1 Introduction

The purpose of this document is to expose the methods and steps followed to create a model that should be able to predict the value of the cost to an insurance company in case of an accident. For this, a dataset with several records has been provided. These records show these features:

- ***ClaimNumber***: Unique policy identifier.
- ***DateTimeOfAccident***: Date and time of accident.
- ***DateReported***: Date that accident was reported.
- ***Age***: Age of the worker.
- ***Gender***: Gender of the worker.
- ***MaritalStatus***: Marital status of the worker. Can be (M)arried, (S)ingle or (U)nknown.
- ***DependentChildren***: The number of dependent children the worker has.
- ***DependentsOther***: The number of dependants excluding children.
- ***WeeklyWages***: Total weekly wage.
- ***PartTimeFullTime***: Binary (P) or (F).
- ***HoursWorkedPerWeek***: Total hours worked per week.
- ***DaysWorkedPerWeek***: Number of days worked per week
- ***ClaimDescription***: Free text description of the claim.

- ***InitialIncurredClaimCost***: Initial estimate by the insurer of the claim cost.
- ***UltimateIncurredClaimCost***: Total claims payments by the insurance company.

The objective field is *UltimateIncurredClaimCost*, the cost for the insurance company, and to achieve the best results, this dataset has been preprocessed, removing irrelevant and potentially incorrect data, while transforming the remaining values to match the types needed. Then, the resulting data was divided into a training data and a test data, followed by its use in the model training using different strategies.

2 Preprocessing

During the preprocessing, two main tasks are performed. The first one is the claim description processing. For this, Natural Language Processing techniques are used. The text is parsed to lower case, incorrect words are corrected, stopwords are removed and the resulting terms are lemmatized. Once the dataframe is ready, it is saved to a file to be used in the preprocessing.

The second step is the preprocessing of the data. First, the resulting data from the claim description processing is concatenated and then the null values are dropped. Some values are checked in order to look for possible errors, like happens in the feature *HoursWorkedPerWeek* that is found to have some incoherent values. A week has a maximum of 168 hours, so values above and equal should be removed. Once removed, the maximum value for this field is 93 hours, which may be very high but possible.

Once the numeric values are checked, the next step is to get rid of the categorical ones. **OneHotEncoder** is used on the fields *Gender*, *MaritalStatus* and *PartTimeFullTime* and then concatenated to the original dataframe.

3 Exploring the data

Once the preprocessing is done, it is possible to explore and analyze the data to spot possible outliers or usual values. To do this, both **Histograms** and **Correlation Diagrams** are plotted.

3.1 Histograms

With histograms, it is possible to see which values are more common and their distribution. 3 groups of histograms are displayed, being these **Personal Data** histograms, **Working Conditions** histograms and the histograms representing the **Economic Factors**.

For each group, some hypothesis could be made:

- Personal Data (fig. 1): *Age* is very distributed among plausible values, even though the lower limit is concerning, with workers registered as extremely

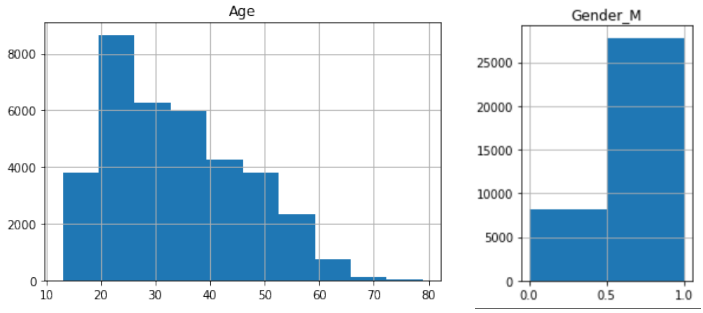

Fig. 1 Personal Data Histograms

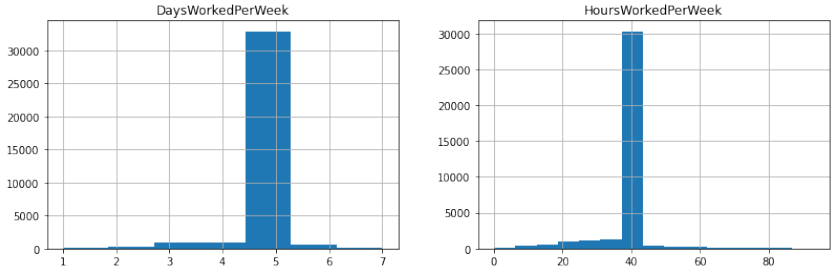

Fig. 2 Working Conditions Histograms

young workers (10-18 years old). But the *Gender* feature could be a matter of interest, as the most part of the incidents occur to males.

- Working Conditions (fig. 2): These histograms represent the values for *DaysWorkedPerWeek*, *HoursWorkedPerWeek* and *PartTimeFullTime*. The histogram showing the days worked per week is clearly represented by the hours worked per week, so the correlation should be very close to 1, thus it could be ignored. The same thing will happen with the histogram of *PartTimeFullTime* even with more correlation, so the value of the hours worked per week could be used to represent these three features.
- Economic Factors (fig. 3): In this group of histograms it's noticeable some outliers. For example, in the *WeeklyWages* histogram, the vast majority is grouped between 0 and 1000, but there are values reaching more than 7000. This is probably an error, but it won't be dropped because it is not clear enough. It will probably affect negatively to the error metric and predictions accuracy. The same happens with *InitialIncurredClaimsCost* but in a higher scale.

3.1.1 Correlation Diagram

Once known the distribution for each feature, a correlation diagram is made to select the most correlated features that could affect the final insurance cost.

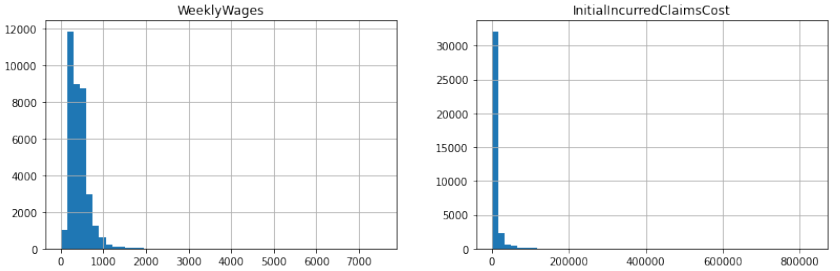

Fig. 3 Economic Factors Histograms

Using Pearson's coefficient, the resulting diagram (fig. 4) shows that the initial claim cost, the weekly wages and the age are the three most correlated features.

With these results, the next step is creating the models and finding out which one is the best for the chosen strategy.

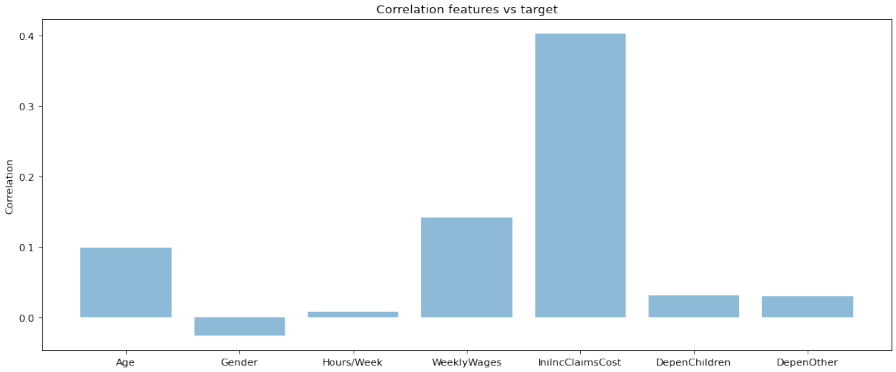

Fig. 4 Pearson Correlation Diagram

4 Transformation

Once the data is preprocessed and explored, test and train splits are needed. The claim description, already processed, is vectorized using TF-IDF. The resulting dataframe is concatenated with the columns corresponding to the three most correlated features (explained in subsection 3.1.1). With the new dataframe, *selectKBest* algorithm is applied to obtain the 10 best features. Now, the objective field is joined and the result is saved in a file to be used in ***Hyperparameter Optimization*** (section 7.1) and ***Boosting*** (section 7.2). Within the same process, Knn and Decision Trees are calculated for this new dataset.

5 K Neighbours Regressor

Through this model, predictions will be based on the values of the k nearest neighbours. To find the best k value, some models need to be trained and then compared to determine which one has the best accuracy. Using ***Cross Validation*** with *KFold* and 10 splits, K is calculated from 1 to 50 for both strategies ***uniform*** and ***distance***.

For the error metric MAE will be used, and the results indicate that the best k value is 49. The error keeps decreasing with diminishing returns so, in order to avoid overfitting, 49 is chosen. This lead the model to a MAE of 7804.2599 and a R^2 score of 0.2808. These results are very unaccurate, so a Decision Tree model is used instead (section 6), as the data pool is very large for Knn to be used.

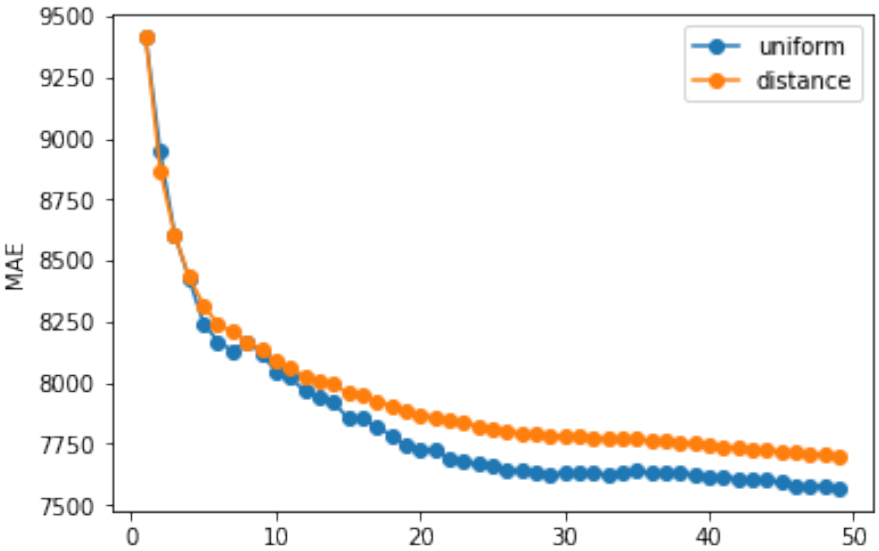

Fig. 5 Cross Validation for Knn results.

The model is trained and then tested. For the predictions to be readable, a range of 300 samples is applied and plotted (fig. 6). Here, it's possible to affirm that the average values are the most accurate ones. This could be caused by the outliers values being extremely different, and most could be input errors. As the good ones can't be distinguished, none is removed, being the most probable cause of the high unaccuracy.

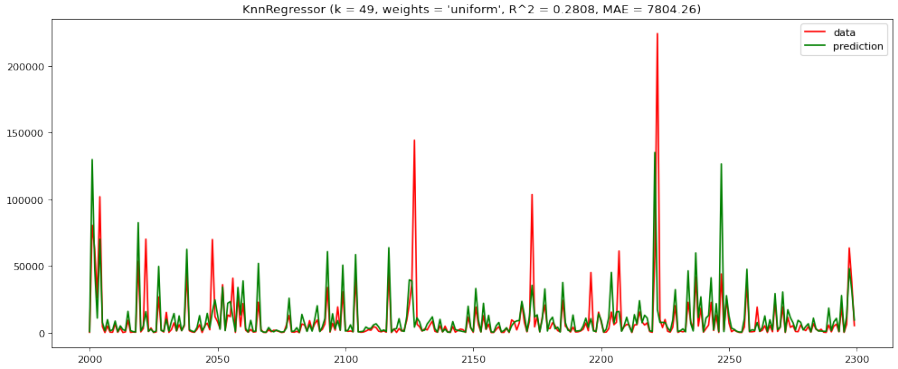

Fig. 6 Predictions vs Real Data for kNN

6 Decision Tree

With the objective of knowing more about how the model is calculated, Decision Trees are used. MAE is used as the error metric, and two predictions are done.

- **Train vs train** (fig. 7): The trained model is applied to the train dataset, and the results are almost perfect as expected with a MAE = 85.60, so the adjustment of the model could be valid.
- **Train vs test** (fig. 8): If the model is now applied to the test dataset, it's observable that the results are better in terms of average values predictions. The outliers raise the MAE value to a higher point than the knn one, probably caused by the outliers predictions being too high.

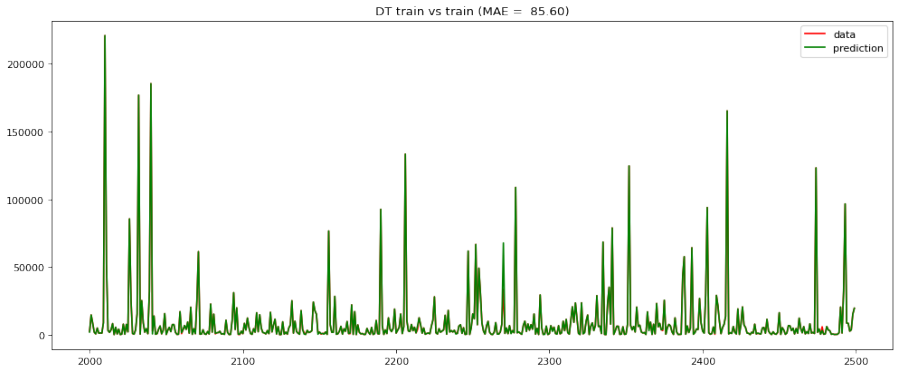

Fig. 7 Train vs train predictions

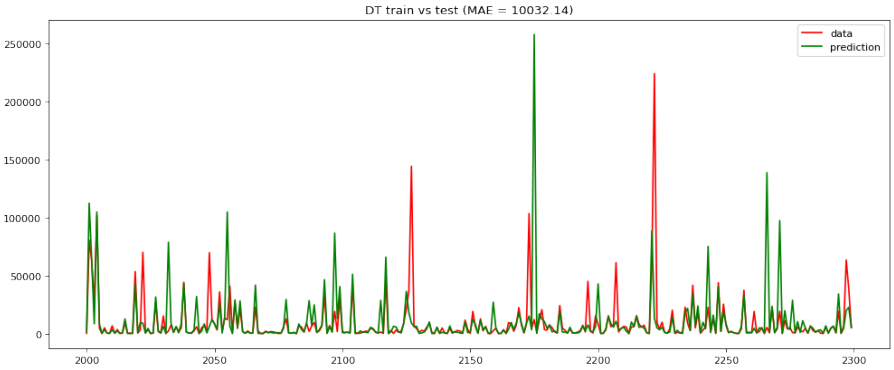

Fig. 8 Train vs test predictions

With cross validation, maximum depth value is intended to be improved, and the results (fig. 9) show that a maximum depth of 55 would perform the best. After applying this value, the results are not improved (fig. 10), so a ***Random Forest Regressor*** will be used.

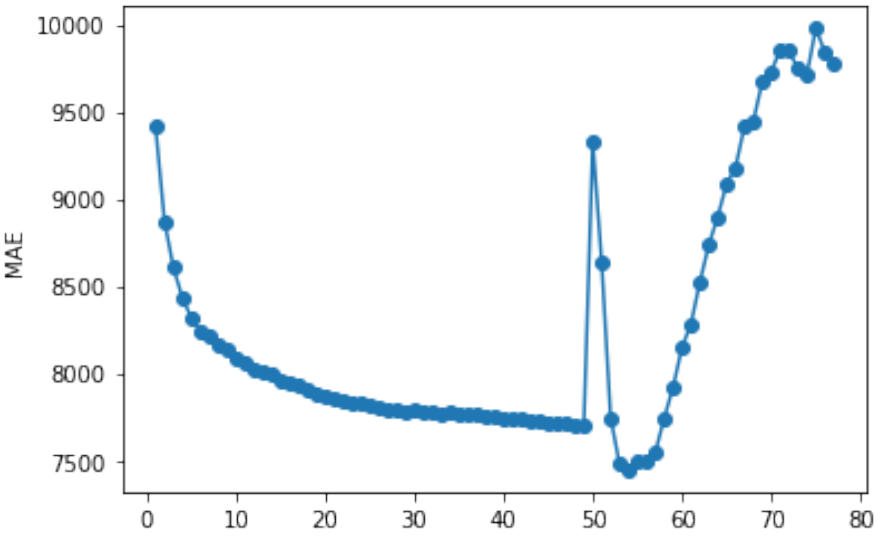

Fig. 9 Cross validation to improve max_depth value

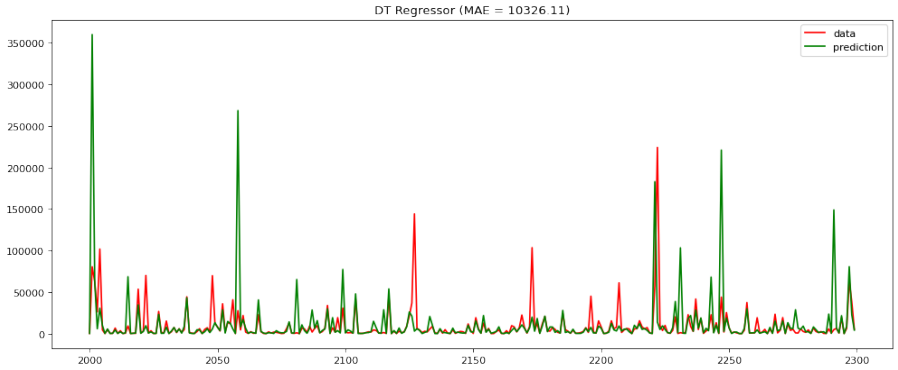

Fig. 10 DT predictions with max_depth = 55

The feature relevancy table (fig. 11) shows that, as expected from the correlation diagram, the initial claim is the most relevant feature with a value of 0.499, followed by the weekly wages with 0.160 and then the age with an importance of 0.131.

	Attributes	Decision Tree
0	0	0.055240
1	1	0.017475
2	2	0.004669
3	3	0.036376
4	4	0.003736
5	5	0.072590
6	6	0.018780
7	InitialIncurredClaimsCost	0.499282
8	WeeklyWages	0.160122
9	Age	0.131729

Fig. 11 Feature relevancy for DT

7 Random Forest

In this model training, the objective is to create multiple decision trees and then use the results to train the model. This strategy may improve the overfitting and, therefore, the results. Using 100 estimators with a maximum depth of 55, the results show some improvement, but not very noticeable (fig. 12).

The reason could be the wrong hyperparameter selection, so a hyperparamater optimization is performed.

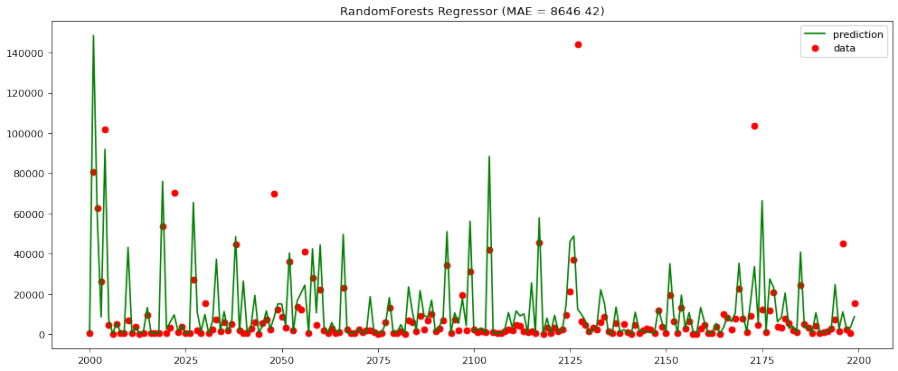

Fig. 12 Random Forest predictions

7.1 Hyperparameter Optimization

In order to find the best parameters for the *Random Forest Regressor* algorithm, a ***Grid Search Cross Validation*** is performed with the next parameters values:

- ***n_estimators*** = [50, 100, 300]
- ***max_depth*** = [50, 100, 150]
- ***min_samples_split*** = [2, 4, 6]
- ***min_samples_leaf*** = [8, 12, 16]
- ***boostrap*** = [True, False]

The top 3 configurations obtained (fig. 13) show that the best maximum depth is closer to 50 (probably 55), while the second and the third use a maximum depth value of 150. Meanwhile, the number of estimators is 100 for the three of them.

```
Model with rank: 1
Mean validation score: 0.280 (std: 0.102)
Parameters: {'bootstrap': True, 'max_depth': 50, 'min_samples_leaf': 16, 'min_samples_split': 6, 'n_estimators': 100}

Model with rank: 2
Mean validation score: 0.279 (std: 0.101)
Parameters: {'bootstrap': True, 'max_depth': 150, 'min_samples_leaf': 16, 'min_samples_split': 4, 'n_estimators': 100}

Model with rank: 3
Mean validation score: 0.279 (std: 0.101)
Parameters: {'bootstrap': True, 'max_depth': 150, 'min_samples_leaf': 16, 'min_samples_split': 2, 'n_estimators': 100}
```

Fig. 13 GridSearchCV top 3 results

With this obtained values, the predictions improve slightly (fig. 14), with a MAE value equal to around a third of the obtained with the decision tree regressor model.

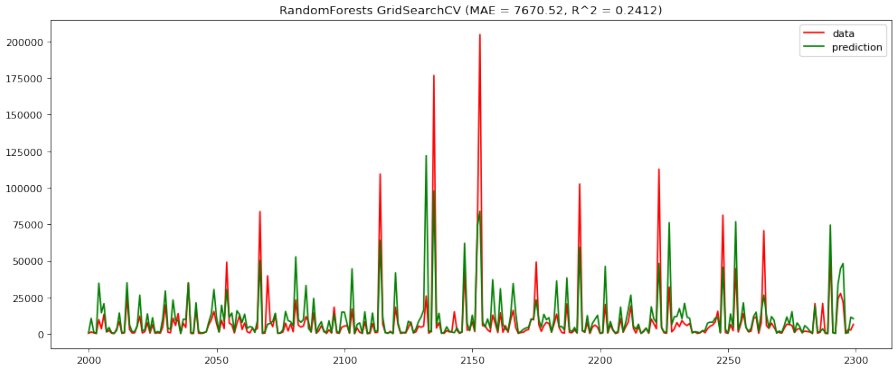

Fig. 14 Random Forest with the best configuration found with Grid Search

7.2 Boosting

Once the best configuration is found, boosting technique is applied through ***AdaBoostRegressor*** and ***GradientBoostingRegressor***. The best performance is gotten by the *AdaBoostRegressor* model (fig. 15) improving the decision tree error and even the Random Forest results slightly. The *GradientBoostingRegressor* (fig. 16) in the other hand, gets the worst results.

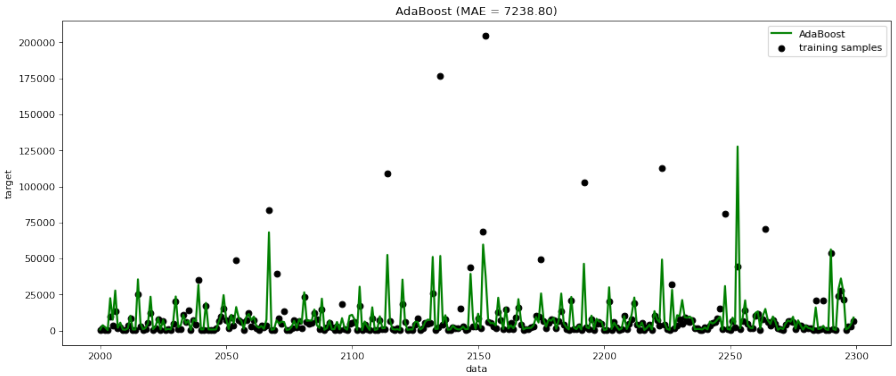

Fig. 15 AdaBoost results

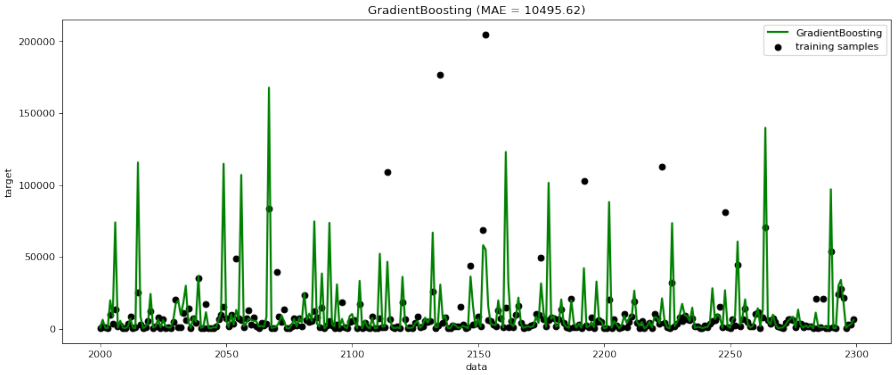

Fig. 16 Gradient Boosting results

8 Conclusion

This project has shown the use of multiple regression models to predict insurance costs on accidented workers, beginning with a baseline with the Knn regressor and decision tree regressor, to be extended with Random Forest, and improving the results with hyperparamter optimization and boosting.

The results are very unaccurate, but the average values are noticeable better than the outlier values. Observing the original distribution, it is highly probable that most of the outliers are input errors when recording the data, as some errors were found in the preprocessing like values in the hours worked per week being much higher than the actual hours in a week. If every outlier were removed, the results may be, with a high chance, much better and with a higher accuracy, but outliers exist and can't be removed, due to they may be the point of interest in the predictions. As it is not possible to distinguish the wrong values from the real ones, they all are left to be studied, causing the high error values.