Supervised Learning: Workers Compensation Claims

María Blanco González-Mohíno, José Alberto Seco Sánchez Camacho, Pablo Velasco Crespo and Adrián Ruíz Esteban

Contributing authors: Maria.Blanco4@alu.uclm.es; JoseAlberto.Seco@alu.uclm.es; Pablo.Velasco2@alu.uclm.es; Adrian.Ruiz6@alu.uclm.es;

Abstract

Predicting workers' compensation claims is not an easy task, but it is not impossible either. To predict it we will be using some different algorithms to make models of our data which is going to help us predict the workers' compensation. The result of this work will facilitate an approximation of how much they will claim.

1 Introduction

During this essay, we will show how we create a model to infer different claims. In order to do it, we will explain all iteration through the model that we did until we achieve the final model, metric, errors and outliers will be also explained.

2 Problem description

Our duty is predicting workers compensation claims. To do so it is going to be necessary some preprocesing and some modeling:

- **Preprocessing**. We transform the data in order to allow us the posterior use of the algorithms to build prediction models, for example, missing values, from categories to numbers, ...
- **Basic Modeling**.This is a basic modeling provided by the teacher.

- **BaseLine**. We will do the feature selection, validation, KNN and Decision Trees, we also made some iteration in order to improve our model.
- **Optimized Model**. Where we use RandomForest and Boosting + Hyperparameter Optimization, in this part we improve our model.
- **Improvements**. Where we will predict the UltimateIncurredClaimCost using the Natural Language.

3 Iterations through model - Baseline

We performed a first iteration, in order to get an idea about the dataframe, in this iteration several parts of the project were carried out, a first KNN and a Decision Tree were performed, at this point we could check which were the different errors that we obtained such as a MSE of 574524523 and a MAE of 8950 (rounding). At this point there were very few variables that had a significant relevance and we decided to perform the correlation between variables, to have a background apart from the relationship between variables extracted from the decision tree.

The next iteration we performed we decided to remove the less relevant characteristics, we kept the features that were most relevant to the decision tree (*WeeklyWages*, *InitialIncurredCalimsCost*, *Age*, *DependentChildren*) and on the other hand we took the next most dependent variable from *UltimateIncurredClaimCost*, *DependentsOther*.

Even with this we still had an error of 7608.666773466037 MAE, 559747196.1366212 MSE.

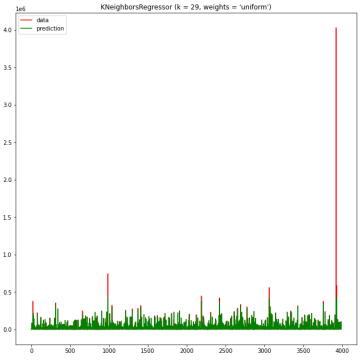
At this point we realized that *InitialIncurredCalimsCost* had a relevance in the model of 0.965311 almost 100% so we decided to test the model by removing this feature.

The model without *InitialIncurredCalimsCost*, whose results were even worse, MAE 12236.085, MSE 758963246.99, the relevance of the leftover features increased the most, although *WeeklyWages* became the one with the most weight of a 0. 773891 followed by *DependentChildren* with 0.115925, *WeeklyWages* became the most representative, 77.38% very high value so we decided to get the *InitialIncurredCalimsCost* feature, the model seems more complete for us.

On the other hand, *DependentsOther* was relevant, although not very much, so we decided to leave it to make the model more accurate.

Once these steps were done, we tried with another testing percentage, 35%, the results were better, but we found that the outliers influenced more than we imagined:

With this information we decided to remove the outliers according to the final claim, we decided to remove them in a more statistical way for two reasons, if we used an unsupervised model, as could be the case of DBSCAN, the RAM would overflow so we could not do much more, on other side we wanted to remove the outliers that presented a great distance from the rest, so we simply used the necessary quantiles, after several revisions, using the 5 chosen characteristics we found the following relevance of characteristics in decision tree:

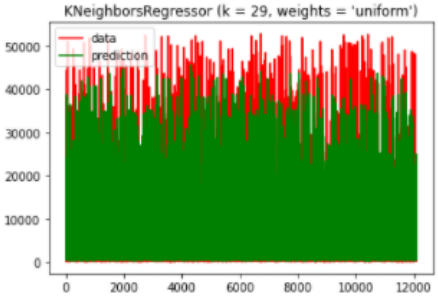


Attributes	Decision Tree
WeeklyWages	0.018920
InitialIncurredCalimsCost	0.976629
Age	0.003860
DependentChildren	0.000591
DependentsOther	0.000000

Metrics obtain in decision tree:

METRICS	VALUE
MAE	2395.8227165465128
MAPE	0.5727312529093795
MSE	25380261.47579692
R^2	0.5685115299064054

We found these values for KNN and decision tree to be quite good, so we decided to leave this model as definitive. The prediction obtained graphically:



3.1 Outliers

As mentioned above, the outliers had a significant impact on the final results so we decided to remove them. As they had such an impact we decided to analyze them and make predictions using only outliers with Decision Tree and KNN algorithm.

To predict using outliers we created a new dataframe that had only outliers and made the same process as we did before.

METRICS	VALUE
MAE	65895.65584747432
MAPE	0.3236995161773757
MSE	10948484637.222652
R^2	-4.522701392437229

As we can see it has a higher error than the predictions done before. Here we got the relevancies of the features:

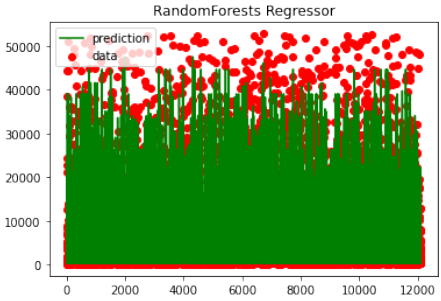
Attributes	Decision Tree
WeeklyWages	0.902407
InitialIncurredCalimsCost	0.009843
Age	0.082083
DependentChildren	0.005667
DependentsOther	0.000000

4 Optimized model

4.1 Random Forest

Now we are going to predict *UltimateIncurredClaimCost* using random forest. The features selected to predict are: *WeeklyWages*, *HoursWorkedPerWeek* and *InitialIncurredCalimsCost*. Test size is going to be a 33%. With those values we get a MAE (Mean Absolute Error) of: 2147.1003923182784

This is how the prediction looks compared to the data:



And now we calculate the relevance of the features:

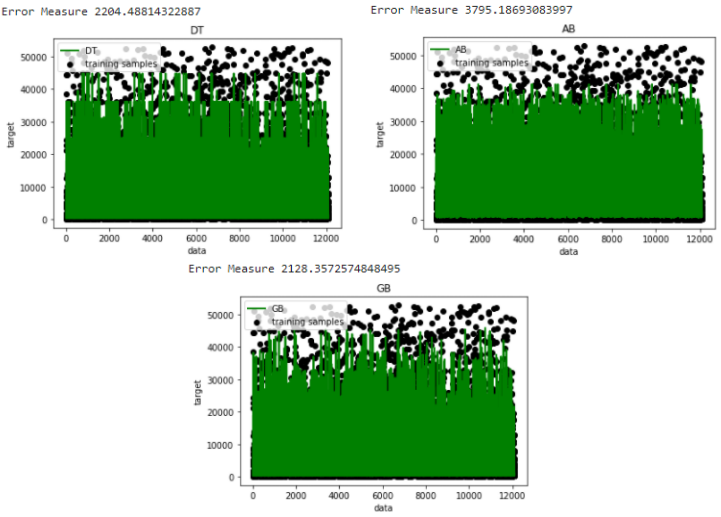
Attributes	**Decision Tree**
WeeklyWages	0.034362
InitialIncurredCalimsCost	0.959713
Age	0.005357
DependentChildren	0.000540
DependentsOther	0.000028

As we can see Random Forest feature relevance's are more distributed and errors are lower than KNN or Decision Trees, so we can confirm that using RF is better than do the prediction with CV and DT.

4.2 Boosting

For the boosting part we compute three different optimizations, a simple Decision Tree Regressor, the Ada Boost Regressor[1] and Gradient Boosting Regressor[2].

Here we left the result of each boosting and their respective error metrics, the first one is the Decision Tree Regressor, then the Ada Boost and down the Gradient Boosting:



Feature relevancies of boosting:

Attributes	DTR	AR	GBR
WeeklyWages	0.033240	0.303584	0.046715
InitialIncurredCalimsCost	0.963867	0.472977	0.927470
Age	0.002806	0.211738	0.023046
DependentChildren	0.000086	0.011674	0.002092
DependentsOther	0.000000	0.000027	0.000677

Feature relevancies are more distributed in Adaboost regressor but we also obtain more error.

[1] An AdaBoost regressor is a meta-estimator that begins by fitting a regressor on the original dataset and then fits additional copies of the regressor on the same dataset but where the weights of instances are adjusted according to the error of the current prediction. As such, subsequent regressors focus more on difficult cases

[2] Gradient Boosting Regressor builds an additive model in a forward stage-wise fashion; it allows for the optimization of arbitrary differentiable loss functions. In each stage a regression tree is fit on the negative gradient of the given loss function.

4.3 Hyper-parameter Optimization

For the hyperparameter optimization we use a grid search and a random search.

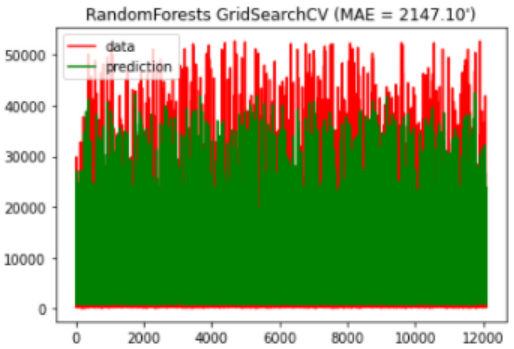
4.3.1 GridSearch

Grid parameters:

```
param_dist = {"n_estimators": [8, 16, 32, 64],
              "max_features": ['auto', 'sqrt'],
              "max_depth": [16,8,4,2],
              "bootstrap": [True, False]
             }
```

Grid best estimator:

Mean validation score: 0.717 (std: 0.024)



Feature relevancies of the best grid estimator:

Attributes	Decision Tree
WeeklyWages	0.065653
InitialIncurredCalimsCost	0.906829
Age	0.024580
DependentChildren	0.002551
DependentsOther	0.000386

Those values are quite good respect all others we get in the models, so this is the "better" model we have so far.

4.3.2 RandomSearch

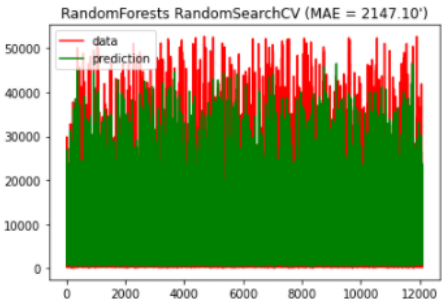
Random search grid:

```
param_dist = {"n_estimators": [4, 8, 16, 32],
              "max_features": ['auto', 'sqrt'],
              "max_depth": [16,8, 4, 2],
              "min_samples_split": sp_randint(2, 25),
              "min_samples_leaf": sp_randint(1, 25),
              "criterion": ["squared_error", "mae"]}
```

Feature relevancies of the best grid:

METRICS	VALUE
MAE	2338.0816692924436
MAPE	0.5693560449611733
MSE	25130276.180869564
R^2	0.5400964455000256

Graphic representation:



5 Improvements

To do the improvements we have to preprocess the Natural Language in column *ClaimDescription*.We do a first iteration including outliers in the model. Results were wuite poor, MSE of 12300... So as we see the nlp model was not good, so we decide extract the outlier just like we did in our baseline.

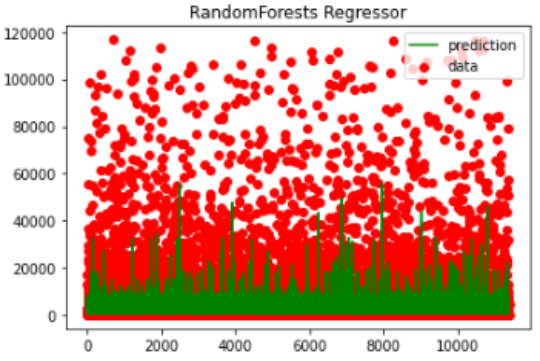
Next steps were done with the outlier dataframe and with the dataframe without outliers: Firstly we removed useless symbols, capital letters and repeated words. Then we replaced all contractions (for example, we replaced are not for are not). Next we tokenized, lemmatized and vectorized the sentences. Now that we have the tokens we tagged the words with its type, so now we have the words classified as nouns, verbs...

It is time to model, to make the prediction we use the dataframe after preprocessing steps.

Now that we have the features selected we will calculate the depth of the tree. Next we predicted the *UltimateIncurredClaimCost* using RandomForest, metrics obtained:

METRICS	VALUE
MAE	6419.971224679538
MAPE	2.3296211044707604
MSE	233487012.88729078
R^2	-18.183556542327885

And this is how the comparison between the prediction and the original data would look like:

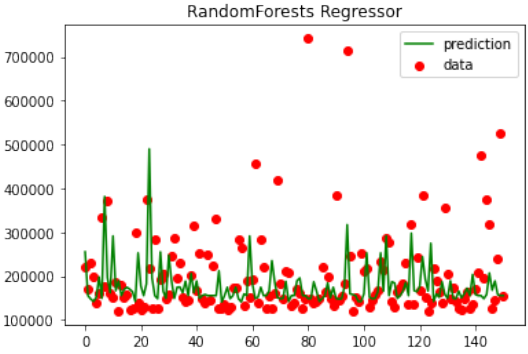


As we can see metrics are not as good as in the baseline but with this dataset this are the better values that we can obtain.

5.1 Outlier model

Like we did before we will predict the same way (using Natural Language) but using only outliers. We used the same methods as before, first calculate the depth, then Random forest. Results obtained:

METRICS	VALUE
MAE	66101.18734544703
MAPE	0.37657178765877464
MSE	11725170181.108683
R^2	-4.408736372409759



Attributes	Decision Tree
and	0.00
and cc	0.00
back	0.00
back rb	0.00
back rb strain	0.00
cc	0.00
fell	0.25
fell vbd	0.25
in	0.00
injury	0.00
...	...

6 Summary

We have introduced our approach to the supervised learning regression problem of workers' compensation claims prediction. Multiple regression models were tested and optimised to obtain the most accurate parameters possible in a reasonable time.

Manual optimisation, boosting... were used to generate a collection of models from which the best one was chosen.

As a result, we think that the best model to choose is the one made with GridSearchCV, since we have the best errors so far and one of the best relevance distributions.