Accident prediction in time and space

Timothée Guédon, Antoine Hébert

Concordia University
Computer Science and Software Engineering Department

April 15, 2019

Table of Contents

Table of Contents

Road accident prediction

Context:

- Road accidents implies millions of people deaths and injuries every year.
- Human and economic impact on society.
- Master's thesis of Antoine Hébert in a different context.

Our goals:

- Dataset analysis.
- Finding key insights in accident prediction.
- Identifying issues related to class imbalance and geo-spatial data.

Datasets

We used open datasets from Open Canada.

National Collision Database (NCDB):

- All vehicle collisions reported by the police from 2012 to 2017 in Montreal.
- Date and time, severity, number of death and injury, number of vehicles, localization, speed limit etc.

National Road Network:

- Shape of the road, how roads are connected together, the mail addresses on the street, the number of lanes in each direction and the type of pavement.

Climate data - Hourly Data Report:

- For each position and hour, the dataset provides the temperature, the humidity, the wind speed and direction, the visibility, the atmospheric pressure etc.

Tree-based algorithms

We chose to focus on tree-based algorithms because it has proven its efficiency on accident prediction.

Related work on accident prediction:

- Extensive use of decision trees in several form. (Athanasios Theofilatos, A. 2017, Joaquín Abellán, 2013, Lei Lin 2014, Li-Yen Chang 2005)
- Recently, deep learning algorithms (LSTM and CNN architectures for example) (Zhuoning Yuan, 2018, Quanjun Chen 2016)

Chosen algorithms:

- Random Forest
- Balanced Random Forest
- Gradient Boosted Trees (GBT) - XGBoost implementation

Table of Contents

Generating the negative samples

- Every time an accident did not happen is considered a negative sample.
- A sample from the cartesian product between all dates (between 01/01/2012 and 31/12/2017) and all roads.
- Generation of 5 million negative samples for about 200 000 positive samples.

Some feature engineering

- Computing the distances between GPS coordinates (taking into account earth curvature)
- Making the dates and times cyclic
- Creating missing points on the roads
- Interpolating weather statistics by averaging data from several stations
- etc...

Area under PR curve

- ROC Curve : false positive rate VS true positive rate

$$FPR = \frac{FP}{FP + TN}, TPR = \frac{TP}{TP + FN}$$

- PR Curve : true positive rate VS precision

$$TPR = \frac{TP}{TP + FN}, Precision = \frac{TP}{TP + FP}$$

- Data imbalance imply decrease of TP and increase of FN
- Therefore, area under PR exhibits larger difference than are under ROC

Table of Contents

Remainder: Random Forests

Remainder

- Random Forests (Breiman [2001]) are combinations of fully-grown decision trees.
- Each tree is trained on a random sample of the data and a random sample of the features.

Weighted Random Forests (WRF):

- A solution to imbalance.
- Add weights to the classes when growing the tree (in the Gini coefficient).
- Add weights to the classes at the voting time.

Solution to imbalance

Balanced Random Forests (BRF):

- Another solution to imbalance.
- Combine undersampling and ensemble method versus loss of information.
- For each tree : Randomly draw sample of same size from majority (with replacement) and minority class.
- The rest of the algorithm is identical to Random Forests

BRF versus WRF:

- No clear winner ! But...
- WRF - More vulnerable to noise on minority class
- BRF - More computationally efficient because use less data (subsampling)
- BRF - Easier to implement

Implementation of BRF in Spark

Existing implementation of RF in Spark

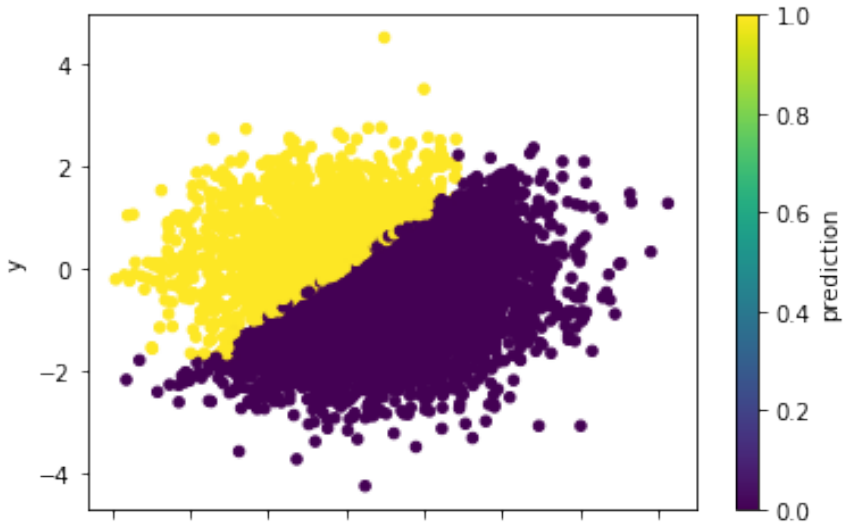
- To manage the subsampling of the dataset, Spark's scala implementation use the Poisson distribution.
- The Poisson distribution is used to compute the probability of an event to occur at a given time.
- To know how many times a given sample will be in the subsample of a given tree (can be 0 times), we use "distribution sampling" of 1 element.
- (Distribution sampling: Get a random sample following a given distribution)

Our modifications:

- Instead of creating one Poisson distribution (with the "p" parameter and a seed) we create two Poisson distributions:
  - One for the distribution of the "positive" samples,
  - Another for the distribution of the "negative" samples.

Experimenting with our BRF implementation

- Area Under PR : 99,3% for WRF VS 99,7% for BRF
- F1 score : 99% for WRF VS 98% for BRF (which proves that F1 score can be misleading)



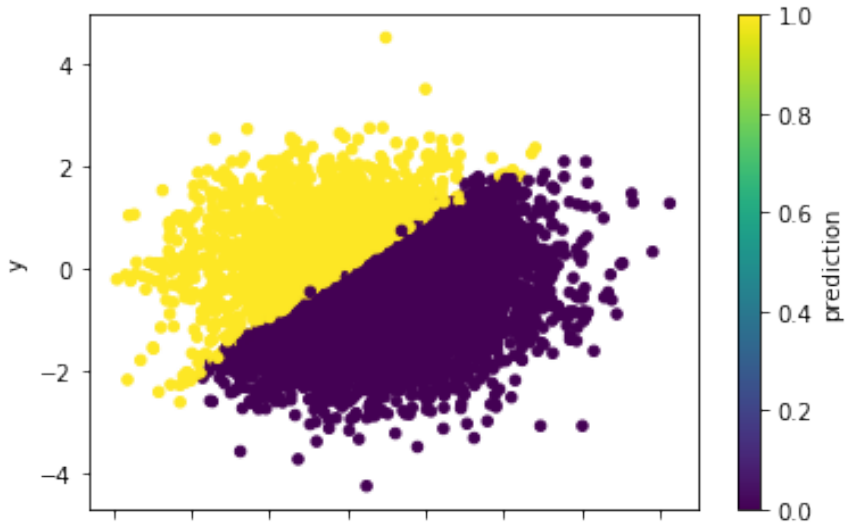
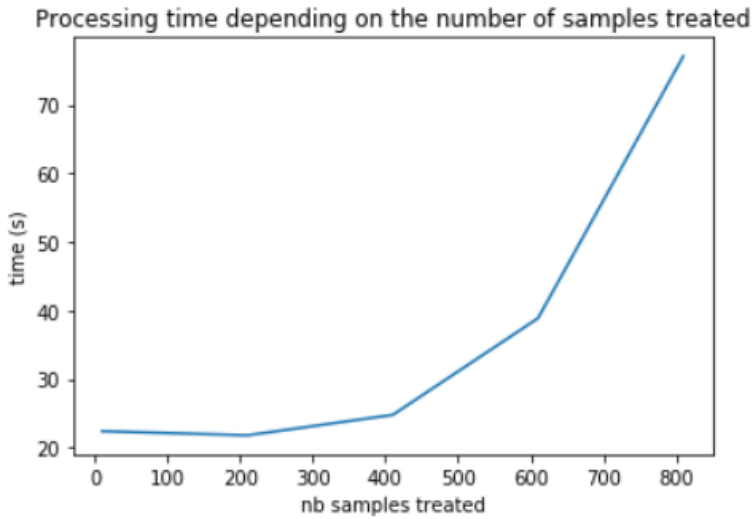
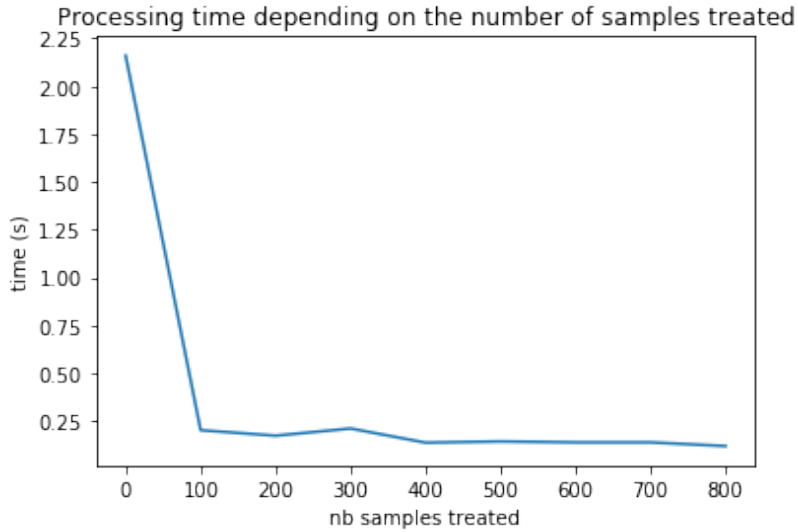


Table of Contents

Dataset generation

- Using slurm-based (scheduler) cluster to increase the dataset generation's speed.
- Tricks to increase the generation speed and reduce the memory consumption.





Spark VS Dask

Advantages of using Dask over Spark

- Mimic the pandas dataframe API
- Lighter than Spark
- Designed to be more flexible in terms of applications and algorithms

Why we switched for Spark

- We found ourselves writing a lot more code using Dask than using Spark
- We found Spark easier to use for processing the dataset in batch
- Spark's map-reduce and shuffling operations are very handy
- Spark ML pipelines allow to create pipelines of preprocessing easily

Table of Contents

Comparison between the algorithms

- Random forest
- Balanced random forest
- XGBoost

Key features identified

- The most important features seem to be the time and location of the vehicle, followed by some road's features like the type of road (highway for example) and the street length.
- Finally, comes wome weather features like the visibility, the wind and the humidity.

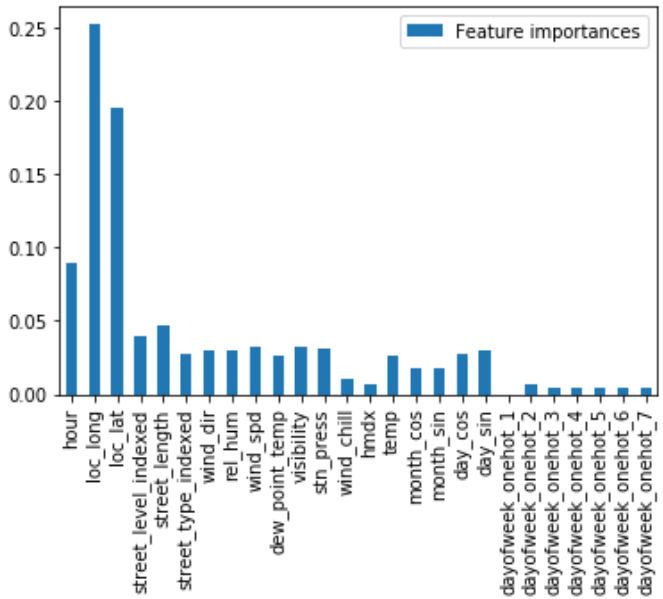


Table of Contents

Conclusion

- We found some key features that can explain the occurence of accidents.
- We identified some issues related to class imbalance and big data analytics.
- Future work would include tuning the XGBoost classifier, testing lightGBM classifier, assing some datasets like a road works dataset.
- The code can be found on Github at https://github.com/GTimothee/accident-prediction-montreal

References I

Leo Breiman. Random forests. Machine Learning, 45(1):5–32, Oct 2001. ISSN 1573-0565. doi: 10.1023/A:1010933404324. URL https://doi.org/10.1023/A:1010933404324.