Airline Departure
Data Analysis and Regression

Lucchi Manuele & Tricella Davide

September 1, 2022

Instructors: Professor Cesa-Bianchi & Professor Malchiodi

We declare that this material, which we now submit for assessment, is entirely our own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of our work. We understand that plagiarism, collusion, and copying are grave and serious offences in the university and accept the penalties that would be imposed should I engage in plagiarism, collusion or copying. This assignment, or any part of it, has not been previously submitted by us or any other person for assessment on this or any other course of study.

Contents

Abstract

The purpose of this paper is to evaluate the usage of a Logistic Regression model on a airlines dataset to predict flight cancellation or diversion, in a scalable and time/space efficient implementation.

1 Definitions

Label

Model

2 Dataset

The initial dataset, [Airline Delay and Cancellation Data] CITAZIONE is made of 9 years of airlines flights data, composed by 10 files (one for each year from 2009 to 2018) of around 6 milions records each. The files presents 28 columns, of which we only took the 9 more relevant

FL_DATE The flight date.

OP_CARRIER The carrier code.

ORIGIN The departure place.

DEST The destination place.

CRS_DEP_TIME The.

CRS_ARR_TIME The.

CANCELLED If the flight has been canceled.

DIVERTED If the flight has been diverted.

CRS_ELAPSED_TIME TODO

DISTANCE The distance the flight has to cover.

In the case the prediction is about the cancellation, the DIVERTED column will be ignored, while if the prediction is on if the flight would be diverted or not, the CANCELLED column will be ignored.
The carrier code is a two characters alphanumeric code, the origin and destination places are a three characters alphanumeric code.
Flight date, departure time and arrival time are dates, while the elapsed time and the distance are real numbers.

Cancelled and diverted are either 0 or 1.

One milion of records equally distributed between the files were taken to perform the training.

3 Preprocessing Techniques

3.1 Algorithms and Techniques

Multiple preprocessing techniques were used.
First, the dataset has been balanced in regard of the evaluated property, be it being canceled or diverted, so that there are an equal number of uniformly drawn positives and negatives. MIGLIORARE
Then the data not already represented as real numbers has been converted; places and carriers, that were alphanumeric codes, had a number assigned based on the code, dates were splitted between the year and the rest, with the latter being hashed MIGLIORARE.
The data (now completely composed of real numbers) was then normalized between 0 and 1, to avoid exploding values.
Lastly, the data was splitted between the training set (75%) and the test set (25%).

DIVISO PER MEDIA E VARIANZA??? L2 REGULARIZATION

3.2 Parallelization

Keeping in mind that the implementation has to be space and time efficient and scale up to large datasets, the preprocessing part has been carried out using the library PySpark. PySpark is a wrapper for Python of the library Apache Spark, originally written in Java.
The purpose of this library is the handling of parallelized data processing, particularly regarding the Distributed File System Hadoop, also created by the Apache Foundation. The library handles automatically the work distribution on the available nodes that the system provide, it can be composed of a single machine with multiple cores or a cluster of machines, this improves significantly the scalability of the solution, which can be run on competely different system without code modification.
The usage of this library created some compatibility issues, because the data structures used by PySpark were not compatible with various parts of the preprocessing section, which had been written initially suing the data science library Pandas. To solve these problems, it wasn't possible to simply use a conversion and leave the parts written in Pandas as they were, because the computation would have run on a single machine, without parallelization, making the use of PySpark completely pointless. The issue has been addressed using the PySpark.SQL functionality, which allow to execute queries on a distributed dataframe. For our purposes various UDF(User Defined Functions), have been created, which then have been applied to every column containing certain types of data.

The library PySpark is also able to handle the csv file handling, so it has been used to save the preprocessed data to speed up multiple runs on the dataset. To carry out the writing of the various distributed dataframes, various files are created, then at the time of reading, the data is distributed to the various nodes.
The preprocessing part of the solution is therefore computed in a distributed manner, using the PySpark dataframe as main data structure to perform the various calculations on the columns of the dataset. At the end of this section of the solution, the distributed dataframes are merged into one using the collect method. This operation can create memory problems if the dataset is extremely large, but it is not possibile to distribute efficiently the model training and computing as easily as the preprocessing, so the main part of the solution will be computed using standard Python data structures.

4 Model

4.1 Parameters initialization

Parameters such as Weights and Bias are initialized using a uniform distribution between 0 and 1, with the first one having the same length as the number of columns and the second being a scalar value.

4.2 Iterations

4.3 Algorithm

differenze con sklearn

4.4 Regularization

Regularization is a technique used to prevent the overfittings. A regularization term is added to the optimization problem (i.e. the gradient calculation) to avoid overfitting. The used version is called **L2**, also known as **Ridge Regression**. MIGLIORARE + BIBLIO

- descrizione inizializzazione parametri - descrizione batching e iterazioni - descrizione forward propagation - descrizione loss - descrizione calcolo gradiente - descrizione update

5 Performances

- come scalano le operazioni di numpy effettuate

6 Experiments

6.1 Canceled Flights

6.2 Diverted Flights

Iterations	LR	L2	Loss
100	0	0	0
500	0	0	0
1000	0	0	0

- nostro modello dopo v1 iterazioni con learning rate v1 - nostro modello dopo v1 iterazioni con learning rate v2 - nostro modello dopo v2 iterazioni con learning rate v1 - nostro modello dopo v2 iterazioni con learning rate v2 - confronto modello di sklearn con i valori migliori

7 Results and Conclusions

7.1 Space and Time