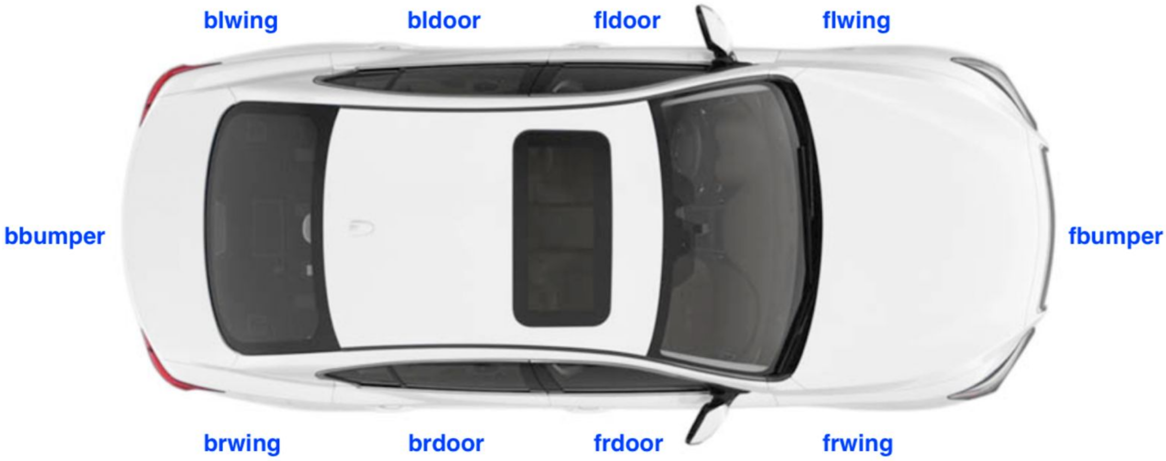
Take home task

Data for this task:
https://s3-eu-west-1.amazonaws.com/tractable-interview-test/tractable_exercise_data.zip

Background

Tractable is a startup specialising in applying machine learning breakthroughs from the last decade to disaster recovery. Our current focus is the auto insurance industry, where we want to automate as much of the accident assessment and cost estimation processes as possible. In order to achieve this, we have obtained millions of detailed accident reports / claims, containing hundreds of millions of images. We have used these images and metadata to train visual models to detect various parts in a vehicle and the extent of the damage on them.

In this toy problem, we consider the following 10 parts in a car - **fbumper** (Front Bumper), **flwing** (Front Left Wing), **frwing** (Front Right Wing), **fldoor** (Front Left Door), **frdoor** (Front Right Door), **bldoor** (Back Left Door), **brdoor** (Back Right Door), **blwing** (Back Left Wing), **brwing** (Back Right Wing) and **bbumper** (Back Bumper).



Assume that these parts can be detected visually with 100% accuracy. We are interested in determining if a part is damaged and if so, whether it needs to be repaired or replaced. We therefore provide you the output of an Undamaged-Repair-Replace (URR) classifier that per part detects whether the part is undamaged or whether it needs to be repaired or replaced. Note that parts that are lightly damaged are typically repaired and parts that are heavily damaged are typically replaced. You can therefore treat this as an ordinal model where undamaged < repair < replace.

Based on this classifier output and some ground-truth metadata, you are expected in this exercise to find the right thresholds / decision boundaries that distinguish the three classes -

undamaged, repair and replace. See the following pages for details on the data, and the specific tasks and deliverables.

Data

You are given the following two data dumps for 100,000 claims:

Metadata

The metadata folder contains sharded, gzipped comma-separated files with the following fields:

- **Claim data**:
  - **claim_id**: A unique ID for a claim
  - **make**: Make description of the vehicle
  - **model**: Model description of the vehicle
  - **year**: Model year of the vehicle
  - **poi**: The main point of impact (eg. Front Centre, Right Rear Corner, etc.)
- **Line data**:
  - **line_num**: Number of the line item
  - **part**: Name of part (eg. fbumper, bbumper, etc.)
  - **operation**: Name of operation (eg. repair, replace)
  - **part_price**: Total price of the part if replaced in $
  - **labour_amt**: Total labour amount to perform the operation in $

Note that a claim can have multiple line items (the claim data columns will be the same, and the line data columns differ). The line data contains information on the damaged parts for the claims: the operation (repair or replace) performed on the part and the cost associated with the operations. If there isn't a line for a part in a claim, assume that that part is undamaged. Also assume that the vehicle details (make, model, year) and point of impact (poi) are known at inference time.

Classifier Output

classifier_output.csv is a comma-separated file containing the following fields:

- **claim_id**: A unique ID for the claim
- **part**: Name of part (eg. fbumper, bbumper, etc.)
- **urr_score**: Undamaged / Repair / Replace score, float value ranging from 0 to 1
- **set**: The set (train/val/test) that the claim belongs to, where 0 => train, 1 => val and 2 => test

Note that for each claim, the classifiers output scores for all 10 parts. So there will be 10 lines per claim in this file. If the classifier score is missing, that means that the part was not identified by the AI in the images provided.

Tasks and Deliverables

The first part of the task is to improve the performance of the URR classifier by determining the right thresholds/decision boundaries between the three classes (undamaged/repair/replace). The purpose of the final system is to predict which parts need to be repaired and replaced in a given claim, but your task stops at finding the thresholds. This task contains of the following sub-tasks:

1. Determine ground-truth labels from the metadata and merge the two data dumps for analysis (expected content in hand-in: code)
2. Analyse the performance of the classifiers (expected content in hand-in: code, 2-5 bullet-points and 1-3 tables/figures)
3. Find the optimum thresholds to distinguish the undamaged/repair/replace classes. You are free to choose any objective here (accuracy, true positive rate, etc.) but make sure to justify your choice, and that the justification is sensible (expected content in hand-in: code, 2-5 bullet-points and 1-3 tables/figures)

The next part of the task is to discuss how to take this solution closer to something that could be used live. This is a design task and we're interested in understanding how you would approach solving these problems:

4. Discuss ways you would scale your code to ingest more data, i.e. of the order of millions (expected content in hand-in: 2-5 bullet-points)
5. Discuss ways you would further improve the performance of the classifiers if you had more time (expected content in hand-in: 2-5 bullet-points)
6. You are now tasked with predicting the cost of performing the operations for a particular claim as a function of the URR scores and the claim data. How would you design a system that can predict these costs? Which metadata fields would you use? Would you also require any additional data (not provided in the data dump) that will help you improve the accuracy of your estimate? (expected content in hand-in: 5-10 bullet-points)

Please submit all the code used for this exercise and a report with answers to questions and figures etc (can be a Jupyter notebook). Apart from the bullet-points (with one, maybe two sentences per bullet point), no more text/motivations need to be written.

Constraints

We would prefer if you used Python for this exercise but apart from that, there are no constraints. You are free to use any third-party libraries. We however request you give us details of your setup like the version of Python and the libraries you used, so that we can easily run your code and replicate the results.

Evaluation Criteria

You will be evaluated on the following:

- Python coding skills (for task 1-3)
- ML/statistical understanding (mainly on task 3)
- Systems design (for task 4-6)
- Quantity of work is **not** part of the evaluation (“expected content in hand-in” is there to help you limit the amount of work needed)