Data Science assignment

General idea

Our customer is a major logistics company which makes use of ORTEC's route optimization software. Beforehand, drivers are assigned a set of locations at which they need to deliver packages, the drivers then use ORTEC's optimization software to create an *optimal route* which they can then use to deliver all packages in as little time as possible. However, the drivers are not always *100%* satisfied with the routes suggested by ORTEC's optimization engine. Therefore the drivers will perform 3 operations:

- Remove certain locations from their route.
- Add locations to their route.
- Change the order of locations on their route.

Drivers may do this based on their experiences/preferences, e.g. it could be that a driver knows that there is a school close to a certain location so he will avoid it when school starts/stops or a driver may want to deliver packages in a certain area around lunch because it is close to his home.

Goal

The objective of this project is to create model that is able to *learn* from a driver's preferences. An initial model could be developed to predict the likeliness of removing certain locations based on some features of the route & the location. It would also be interesting to think about how to implement adding locations and re-ordering routes. Before you delve into the details; think about:

- What features do you have available?
- What features can you compute?
- What features would you like to receive from the business to further frame the problem?

There are multiple deliverables for this project, we would like to see:

- An implementation of an ML model to predict the likelihood of removing locations:
    - Compute your methods accuracy
    - Create some visualizations to report on the performance of your model.
    - Ideally train multiple models and motivate which works better.
- How would you suggest to use insights gained here in the existing process.
- An action plan/suggested next steps for the remainder of the project (next steps for the business & for the consultants).

Available data

For this investigation, we have data available in the form of *requests/responses* between drivers and ORTEC's optimization engine. This data can all be found in the *data* folder.

Excel overview table

In the root of this folder, you can find the Excel file *ModifiedQueryRows.xlsx* which summarizes all requests that were made. The content of this Excel file looks like this:

	A	B	C	D	E	F	G	H
1	Date	Time	OptimizationRequestId	RouteId	TriggerType	ConfigurationName	Numbe	NumberOfTasksInInputPlan
2	5/30/2022	1:03:42.341 PM	ded486e8-cdda-49e7-8877-1419c09a6801	shiftId	FullOptimization	CreateSequence	1	0
3	5/30/2022	1:10:31.742 PM	58548c16-f60f-41d3-bf5e-b3a4fe1a3c16	0511_A89	TimeCalculation	EstimateTime	20	20
4	5/30/2022	1:11:07.500 PM	0fc8aa89-cf38-4a79-9514-d8c2915b3a25	0525_269	AdditionalOrders	AddToSequence	79	42
5	5/30/2022	1:12:01.156 PM	a64c4d13-0540-4383-86fa-60243dfd752c	0511_A89	TimeCalculation	EstimateTime	20	20
6	5/30/2022	1:18:43.056 PM	eae59070-4864-45d4-93c8-7985c5c90610	shiftId	FullOptimization	CreateSequence	1	0
7	5/30/2022	1:21:02.609 PM	1b9f771e-40e0-4d57-8ff5-f97ac4b4757c	0525_269	TimeCalculation	EstimateTime	78	78
8	5/30/2022	1:22:50.855 PM	39574f06-1d81-45b1-87b7-f9ab53bb2ac4	0525_269	TimeCalculation	EstimateTime	70	70
9	5/30/2022	1:24:40.934 PM	628698d8-7a33-4853-bd21-e2a11bc8d4c0	0525_269	TimeCalculation	EstimateTime	69	69

The columns have the following meaning:

- **Date:** The date on which the request was executed.
- **Time:** The time at which the request was initiated.
- **OptimizationRequestId:** Each call to the optimizer is automatically linked to a unique id; these ids do not have a special meaning.
- **RouteId:** A route id is generally of the form *code1_code2*, where both *code1* and *code2* are both combinations of letters and numbers. Here, *code1* signifies a depot and *code2* signifies a region assigned to the depot. In this experiment we are focusing on one depot; *0521*, therefore you can filter on **RouteId**'s of the form *0521_ **.
- **TriggerType:** This column explains the type of call that the driver has executed to the optimizer; we differentiate 3 different types of triggers:
    - **FullOptimization:** This indicates that the user does not provide any preferences and simply wants the optimizer to return a route with a driving time which is as low as possible.
    - **TimeCalculation:** When selecting this option, the user supplies a complete ordering of all selected locations. The used only wants to know the theoretical driving time for this route.
    - **AdditionalOrders:** When using this option, the user fixes a part of the solution. The relative order between the fixed locations needs to be respected but the order of the other locations can be chosen arbitrarily (may also be intertwined with the fixed locations).

      E.g.: Locations: 1,2,3,4,5 with fixed order 4→1→2 could lead to the solution 5→4→3→1→2
- **ConfigurationName:** For this case this is equivalent to Triggertype, with the translation step:

Triggertype	ConfigurationName
FullOptimization	CreateSequence
TimeCalculation	EstimateTime
AdditionalOrders	AddToSequence

- **NumberOfTasks:** This number indicates the total number of tasks that need to be supplied in the given route.

- **NumberOfTasksInInputPlan:** This number indicates the number of locations for which the user has fixed its order. Note that for createSequence runs, this number is always equal to zero while for EstimateTime this is always equal to NumberOfTasks. For AddToSequence, this should be somewhere between 0 and NumberOfTasks.

What does a typical optimization loop look like? In order to single out one optimization run; you need to filter on a unique RouteId (e.g. 0521_318) and a single date, what you will then typically see is a sequence of optimization calls:

Date	Time	OptimizationRequestId	RouteId	TriggerType	ConfigurationName	NumberOfTasks	NumberOfTasksInInputPlan
6/11/2022	6:02:52.131 AM	e443273e-ce59-4201-a244-25ca9d7c4cfa	0521_318	FullOptimization	CreateSequence	43	0
6/11/2022	6:08:08.766 AM	6e0096ef-72da-4c55-8b0f-0cc5d848b3ca	0521_318	TimeCalculation	EstimateTime	43	43
6/11/2022	6:32:45.756 AM	ce2dbfc4-3a2b-4814-8bf6-916a2a8d887e	0521_318	TimeCalculation	EstimateTime	38	38
6/11/2022	6:33:25.926 AM	921d4a07-e540-4622-852d-4483b7e38cad	0521_318	TimeCalculation	EstimateTime	38	38
6/11/2022	6:34:08.899 AM	9f4153ec-60a8-4298-be91-5fa9c0a705ed	0521_318	TimeCalculation	EstimateTime	38	38
6/11/2022	6:34:53.065 AM	994fdff0-18ca-4063-b586-c5c82ee93c45	0521_318	TimeCalculation	EstimateTime	37	37
6/11/2022	6:36:36.051 AM	45feaa25-0277-411e-b313-184b55d1c2be	0521_318	TimeCalculation	EstimateTime	36	36
6/11/2022	6:37:05.960 AM	5f50a187-0e30-4d67-bfde-9eae2caae3ba	0521_318	TimeCalculation	EstimateTime	36	36
6/11/2022	6:37:41.341 AM	2b558aab-87c4-4071-8680-438919dfdadd	0521_318	TimeCalculation	EstimateTime	36	36
6/11/2022	7:04:59.551 PM	9b7d059c-6ec5-4d39-a612-3229f18df827	0521_318	FullOptimization	CreateSequence	17	0

This is a typical pattern:

- In the morning the optimizer is called for a FullOptimization; that is: the driver was assigned a set of locations and he needs to create a route.
- After receiving an optimized route, the driver computes driving times for a set of manually adjusted routes throughout the remainder of the morning.
- The last request of the morning (at 06:37:41) represents the route that the driver will actually use.
- In the evening there is often another call to the optimizer, this one is for investigational purposes and does not have a role in this work.

In this case, we could argue that one should compare the route returned in the first call and the route used (that is, the route supplied/returned in the second to last call). There could however also be merit in investigating intermediate calls.

Request/Response data

In addition to the overview given in the Excel file, we also supply *json* & *txt* files which contain information on the given requests and responses. The following files are provided:

- **requests**: in this folder all requests are located, grouped per day & route id, a request file is structured as follows:
    - **id:** The unique request id
    - **configurationName:** configuration name of the request
    - **tasks:** list containing all tasks that need to be executed in this request, these tasks contain the following fields:
        - **id:** unique task id
        - **address:** Geocoordinates of location that needs to be delivered to
            - **latitude:** The location's latitude
            - **Longitude:** The location's longitude
        - **TimeWindow:** The time window in which this order must be delivered
            - **From:** The earliest hour at which the order may be delivered.
            - **Till:** The latest hour at which the order may be delivered.
    - **fixedTasks:** List consisting of all tasks for which the driver wants to fix the order.
        - **taskId:** the unique id of the task we want to fix.
        - **activityType:** Type of the activity; in this case, all activities have type *task*.
        - **fixedPosition:** Boolean which indicates whether this position is fixed, fixed means that its relative order to other fixed tasks should remain fixed.
- **responses**: In this folder you can find all responses provided by the optimizer. In this case, we simply return a text file consisting of a sequence of task id's. This order is the order suggested by the optimizer.