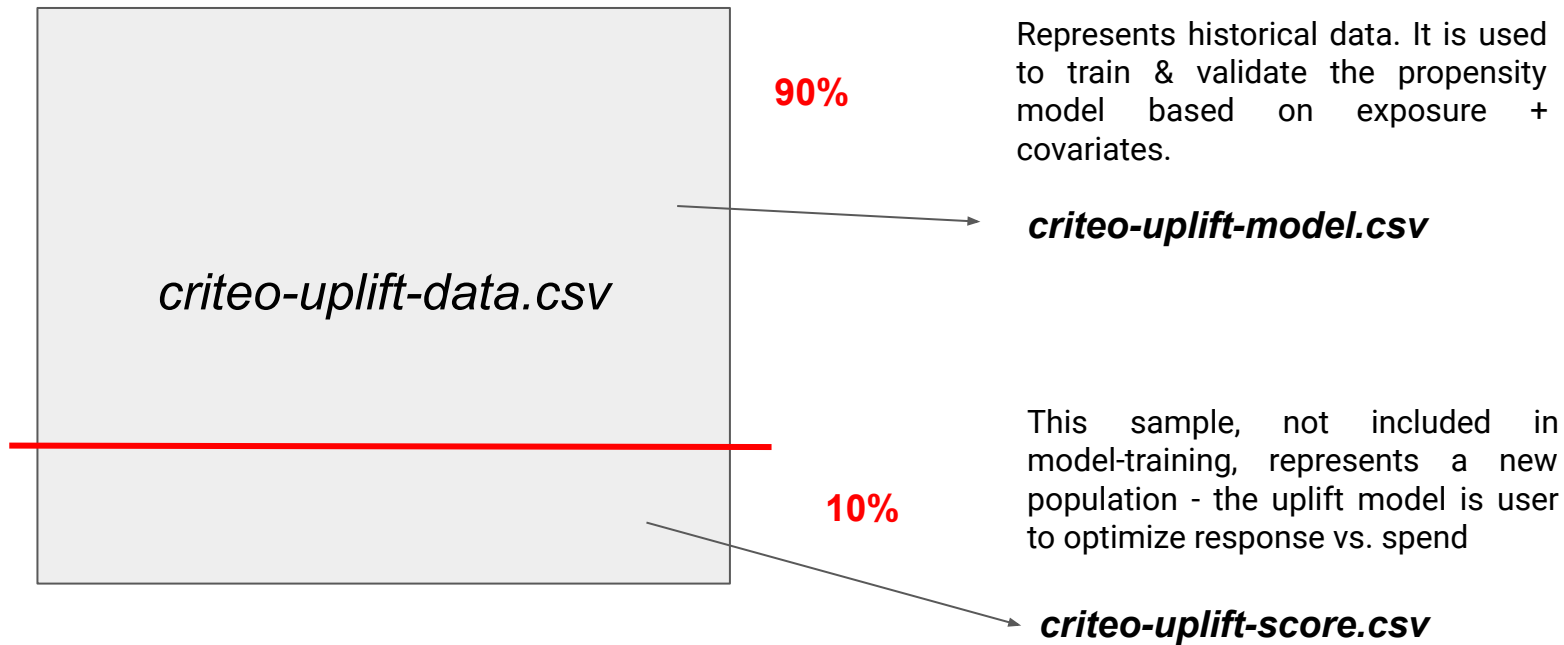


0. Split original data to simulate uplift execution



Traditional Uplift Model Setup

Menu Search Feature List: All Features View Raw Data + Create feature list 1-16 of 16										
<input type="checkbox"/> Feature Name	Data Quality	Index	Var Type	Unique	Missing	Mean	Std Dev	Median	Min	Max
<input type="checkbox"/> exposure		10	Numeric	2	0	0.03	0.17	0	0	1
<input type="checkbox"/> visit		15	Numeric	2	0	0.05	0.21	0	0	1
<input type="checkbox"/> conversion		14	Numeric	2	0	0.00	0.05	0	0	1
<input type="checkbox"/> treatment		13	Numeric	2	0	0.85	0.36	1	0	1
<input type="checkbox"/> f11		12	Numeric	80	0	-0.17	0.02	-0.17	-1.28	-0.17
<input type="checkbox"/> f10		11	Numeric	95,101	0	5.33	0.17	5.30	5.30	6.47

1. Launch AutoPilot to predict **conversion**

Advanced Options

- Partitioning
- External Predictions
- Smart Downsampling
- Time Series
- Feature Constraints
- Bias and Fairness
- Clustering
- Additional

Smart Downsampling

☒ Downsample Data

For classification problems this will allow you to downsample the majority class in order to build faster models with similar accuracy.

Majority class downsampling percentage:
This must be between 1% - 100%

16

Minority rows (6,698 of 6,698)
Majority rows (359,958 of 2,249,738)

Results of downsampling for EDA sample 2256436 nonmissing rows:

Minority rows 6,698 (of original 6,698)
Majority rows 359,958 (of original 2,249,738)

If your target is binary and minority class is very small ($<1\%$), consider **downsampling**. It's unlikely to improve performance but will definitely speed up training.

What would you like to predict?

conversion CLASSIFICATION

No target?

Number of rows

conversion

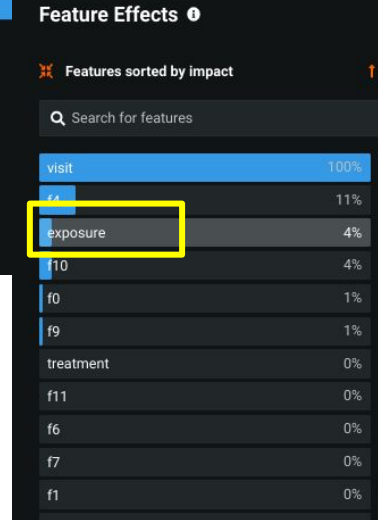
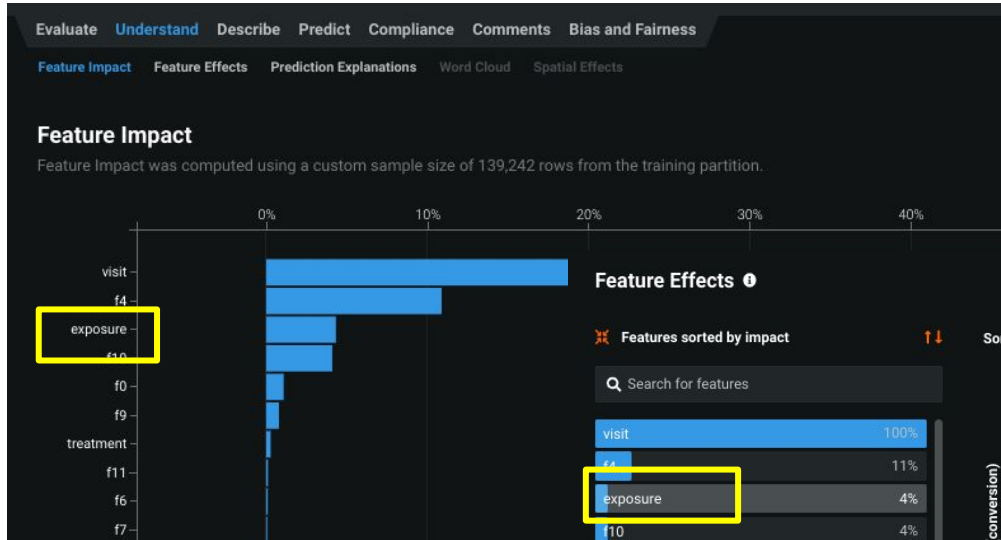
Start

Modeling Mode: Quick

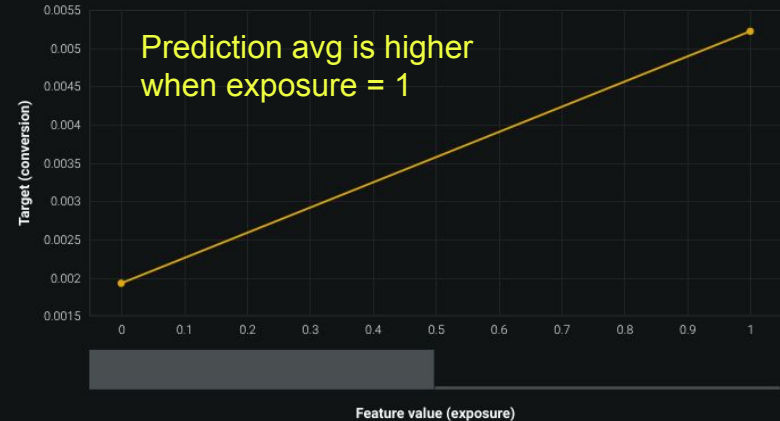
Feature list: Informative Features

Optimization Metric: Weighted LogLoss

2. Confirm AutoPilot Results



Sort by: Numeric Data selection: Validation More



Verify the following:

1. Feature **impact** includes the exposure feature
2. The exposure has a *positive effect* on the score.

3. Deploy the Model to generate uplift scores

The screenshot displays the DataRobot web interface. At the top, a navigation bar includes tabs for Evaluate, Understand, Describe, Predict (highlighted), Compliance, Comments, and Bias and Fairness. Below this, a sub-navigation bar shows Make Predictions, Deploy (highlighted), DataRobot Prime, Downloads, and Portable Predictions. The main section is titled 'Deploy model' with the subtitle 'Deploy a model to leverage it in your business applications.' A 'Prediction threshold' input field is set to 0.2114. Two buttons are visible: 'Set up new deployment' (highlighted with a yellow box) and 'Add to Model Registry'. A yellow text overlay states 'This creates an API in MLOps'. On the right, a secondary panel shows tabs for Overview, Service Health, Data Drift, Accuracy, Fairness, Humility, Challengers, Predictions (highlighted), and Settings. Below these are links for Make Predictions, Job Definitions, Integrations, Prediction API (highlighted), and Monitoring. The 'Prediction API Scripting Code' section provides instructions on using scripts for batch and real-time predictions, with a link to 'Open documentation'. It features a table with 'Prediction Type' (Batch, Real-time) and 'Interface' (CLI, API Client, HTTP). The 'API Client' interface is selected, and a 'Copy script to clipboard' button (highlighted) is present. Below the button, the script usage is shown:

```
"""Usage:
python datarobot-predict.py <input-file.csv> <output-file.csv>

We highly recommend that you update SSL certificates with:
pip install -U urllib3[secure] certifi

Details: https://app.datarobot.com/docs/predictions/batch/batch-prediction-api/index.html
"""
```

To calculate uplift we need a script for the model API. This is where **convert.py** comes from

4a. Prepare score file and calculate uplift

Use the API to create **propensity*** and **response*** scores.

```
audience = pd.read_csv('./customers.csv')

audience['exposure']=0

audience.to_csv(path_or_buf='./audience')
os.system("./convert.py audience
wo_exp.csv")

audience['exposure']=1

audience.to_csv(path_or_buf='./audience')
os.system("./convert.py audience
with_exp.csv")
```

Difference of prop. and resp = **uplift score**.

```
noexp =
pd.read_csv('./wo_exp.csv').rename(columns={'conversion_1_PREDICTION': 'convert_exp_no'})

exp =
pd.read_csv('./with_exp.csv').rename(columns={'conversion_1_PREDICTION': 'convert_exp_yes'})

scores =
pd.concat([noexp[['convert_exp_no']],exp[['convert_exp_yes']]],axis=1)

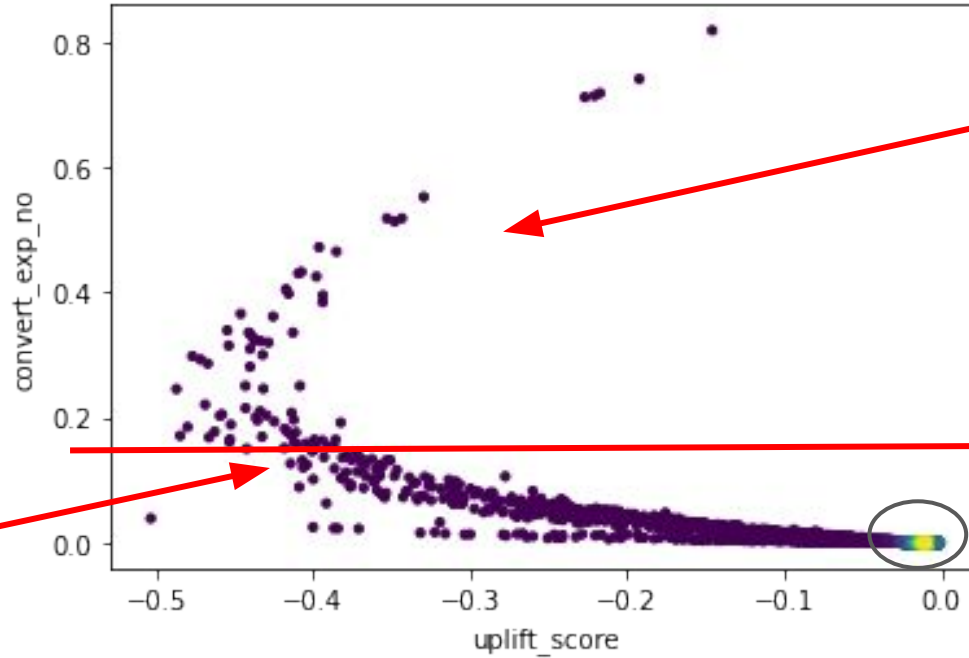
scores['uplift_score']=scores['convert_exp_no']-
scores['convert_exp_yes']

scores.plot.scatter(x = 'uplift_score', y =
'convert_exp_no');
```

** Propensity = $Pr(\text{purch.} \mid \text{no intervention})$. Response = $Pr(\text{purch.} \mid \text{intervention})$*

4b. Uplift Scatter plot

These are the **persuadables** (where you should spend ad money starting at top-left)



These customers will respond regardless of exposure (**sure things + sleeping dogs**)

Note: this is a density plot. Most of the scores appear in this small oval

Appendix: Uplift open-source package

The DataRobot approach to the automation of uplift modeling is based on the Causal ML methodology (originally developed at uber).

Documentation: <https://causalml.readthedocs.io/en/latest/about.html>

Youtube pres: <https://www.youtube.com/watch?v=2J9j7peWQgI>

Python package: <https://github.com/uber/causalml>

White paper: <https://arxiv.org/abs/2002.11631>



Causal**ML**

Goal of Uplift Modeling

Uplift model estimates heterogeneous treatment effects with ML algorithms

Conditional average treatment effect: $CATE = E[Y | \text{Intervention}, X] - E[Y | \text{No Intervention}, X]$



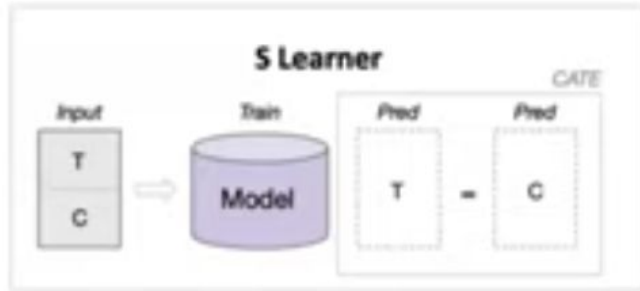
Sure things & Lost causes – Will behave the same no matter what you do. Including them as a target in the model is okay, but will make our targeting inefficient.

Sleeping dogs - These people are turned off by your intervention. Definitely don't include them, ideally you would even downrank them.

Persuadable - This is the population you actually care about because they exhibit the ideal behavior **because** you intervened. Ideally you uprank them as much as possible

Source: <https://www.youtube.com/watch?v=2J9j7peWQgI>

S Meta-Learner for Uplift Modeling



Procedure

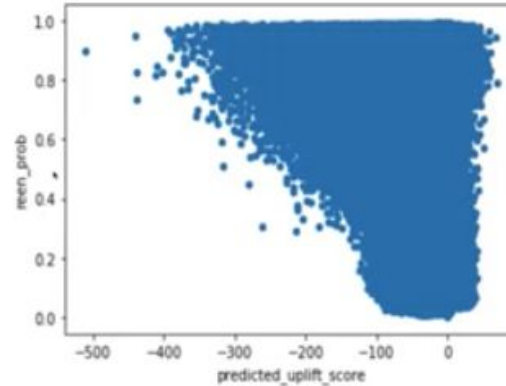
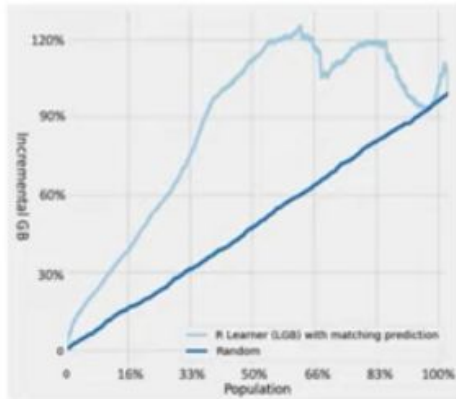
1. Create a binary feature `is_treatment`, indicating whether a user is from the treatment group
2. Train a single (S) model
3. For all users, set `is_treatment` to 1 and calculate $\text{yhat}_{\text{is_treatment}=1}$
4. For all users, set `is_treatment` to 0 and calculate $\text{yhat}_{\text{is_treatment}=0}$
5. $\text{CATE} = \text{yhat}_{\text{is_treatment}=0} - \text{yhat}_{\text{is_treatment}=1}$

Source: <https://www.youtube.com/watch?v=2J9j7peWQgI>

Optimization with Uplift Model

Uplift Modeling @ Uber

- Maintaining > 100% campaign incremental values while targeting only ~ 50% population by identify the persuadable users
- Propensity score and uplift score are weakly negatively correlated . While propensity score model is easier to build in general, it cannot substitute the uplift model.



Source: <https://www.youtube.com/watch?v=2J9j7peWQgI>