Using Fault Data to Predict Vehicle Derates

Big-G Express

Smokey and the Bandits:
Maria, Andrew, Jeff

Presentation Outline

- Research Objective
- Data Challenges
- Classifying Target Variable
- Feature Engineering
- Model Selection and Results

Research Objective

Predict when a full derate is imminent with enough time to get the truck off the road (2+ hours)

➔ **Caught Full Derate (True Positive)**

+$4,000

➔ **False Derate Prediction (False Positive)**

-$ 500

➔ **Evaluating Results**

False Positive Cost = False Positives * $500

True Positive Savings = True Positives * $4,000

Net Savings = True Positive Savings - False Positive Cost

Results

Random Forest -$99k / Logistic Regression: -$83k / ***YDF GBL: $0***

Data Challenges

Diagnostics Data (Long to Wide)

Id	Name	Value	FaultId
1	IgnStatus	False	1
7	IntakeManifoldTemperature	78.8	1
11	LampStatus	1023	1
14	Speed	0	1
19	ParkingBrake	True	1

IgnStatus	IntakeManifoldTemperature	LampStatus	Speed	ParkingBrake
False	78.8	1023	0	True
True	NaN	1279	NaN	NaN
NaN	NaN	1279	NaN	NaN
True	NaN	1279	NaN	NaN
NaN	NaN	16639	NaN	NaN

Removing Service Stations

```
threshold_miles = 0.5
threshold_meters = threshold_miles * 1609.34
```

IsServiceStation	
False	0.889795
True	0.110205

Identifying Derate Rows

	IsFullDerate	Count
0	False	1055549
1	True	355

Full Dataset

	IsFullDerate	Count
0	False	111288
1	True	43

Test Data

```
faults['IsFullDerate'] = (faults['spn'] == 5246)
```

There is no baseline / ‘normal’ truck operation indicator - everything is some kind of fault.

There were instances of multiple derate events for a specific vehicle in a very short period of time.

Classifying Target Variable

MH Note: in its raw form, the data does not have "labels", so you must define what labels you are going to use and create those labels in your dataset. Also, you will likely need to perform some significant feature engineering in order to build an accurate predictor.

➔ **Identified valid vehicle derates**

Kept only the first derate for a given 24-hour window

➔ **Identified a derate window**

Up to *x* hours prior to a valid derate (most of our models used a 2-hour window)

```
Sample rows where derate_window is True:
       EquipmentID       EventTimeStamp   spn   next_trigger_time  derate_window
996835   105349576 2018-07-06 09:42:48  5246 2018-07-06 09:42:48           True
972882   105427203 2018-04-27 06:07:55  5246 2018-04-27 06:07:55           True
5712          1329 2015-02-25 13:53:08  4344 2015-02-25 13:53:08           True
5713          1329 2015-02-25 13:53:08  5246 2015-02-25 13:53:08           True
83425         1339 2015-06-12 15:35:22  5246 2015-06-12 15:35:22           True
```

Feature Engineering

➔ **Converting data to usable types**

- Used string substitution to make some object type features numeric.
- Some features were numerically represented discrete variables (‘spn’, ‘fmi’), so we left those alone.
- Had some boolean, float, int, and datetime cols which were converted to seconds

➔ **Imputing NaNs**

- Since time series data, NaNs filled with .ffill() and .bfill() methods
- EXCEPT found better results imputing using mean for logreg model

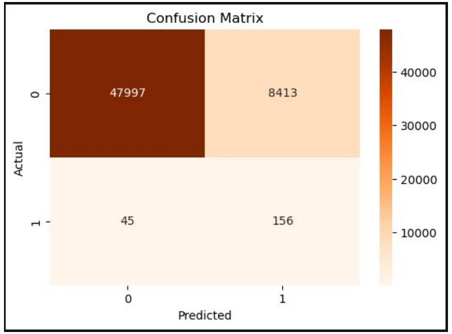
➔ **Additional Features**

- fault_frequency, time_since_last_fault, derate_window, SeverityLevelFeature

Model Selection

➔ **Logistic Regression**

- Basic, dip-your-toes-in-the-water, logreg model
- Target was a five-hour derate window (tried multiple, and this provided the best results)
- Results were overall…not great (raw matrix pictured)
    - After filtering out duplicates, there were 22 TP and… 3925 FP
    - **-$1,874,500** in "savings"
    - BUT this improved to **-$82,500** after raising decision threshold to 90% (20 TP and 325 FP)



Model Selection

➔ **Random Forest**

- n_estimators=50,
- max_depth=15

Model Net Savings

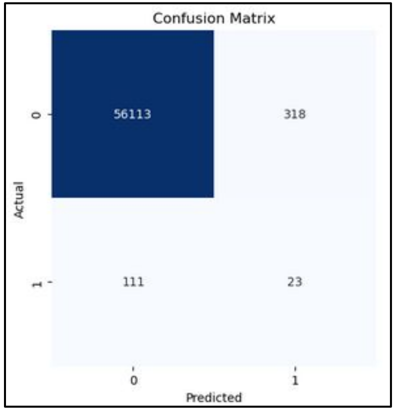
True Negatives: 56,113

False Positives: 318

False Negatives: 111

True Positives: 15

Money "Saved": **-$99,000**



Top 10 Important Features:

	Feature	Importance
12	SeverityLevelFeature	0.214960
0	DistanceLtd	0.127416
10	FuelTemperature	0.108458
1	EngineOilTemperature	0.077515
9	IntakeManifoldTemperature	0.071135
5	EngineOilPressure	0.067904
8	EngineRpm	0.066278
7	BarometricPressure	0.055489
6	EngineCoolantTemperature	0.052904
4	EngineLoad	0.045936

Model Selection

➔ **YDF Gradient Boosted Learner**

- A set of shallow decision trees that attempt to correct for the error in previous trees.
- We used 500 trees with max depth of 10 and applied ridge regularization.
- Target was derate_window, which was the 2 hours before a derate occurred at least 24 hours after the last derate occurred (or the first derate)
- We extracted predicted probabilities and converted to boolean with threshold of 0.9 (increased precision) to identify predicted class (the target)

Model Evaluation

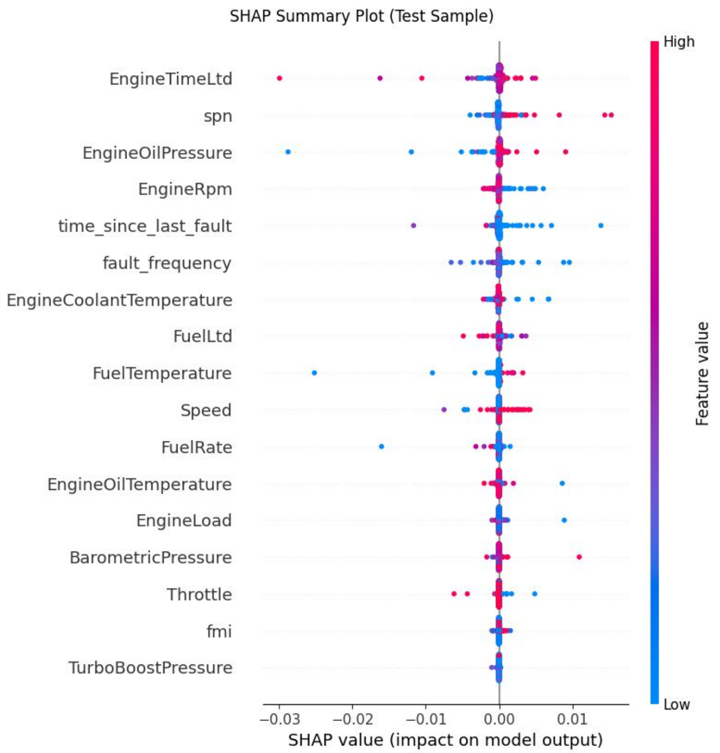
➔ **YDF Gradient Boosted Learner**

```
Classification Report (sklearn):
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     56523
           1       1.00      0.61      0.76        88

    accuracy                           1.00     56611
   macro avg       1.00      0.81      0.88     56611
weighted avg       1.00      1.00      1.00     56611
```

Random sample of 500 values ordered by importance

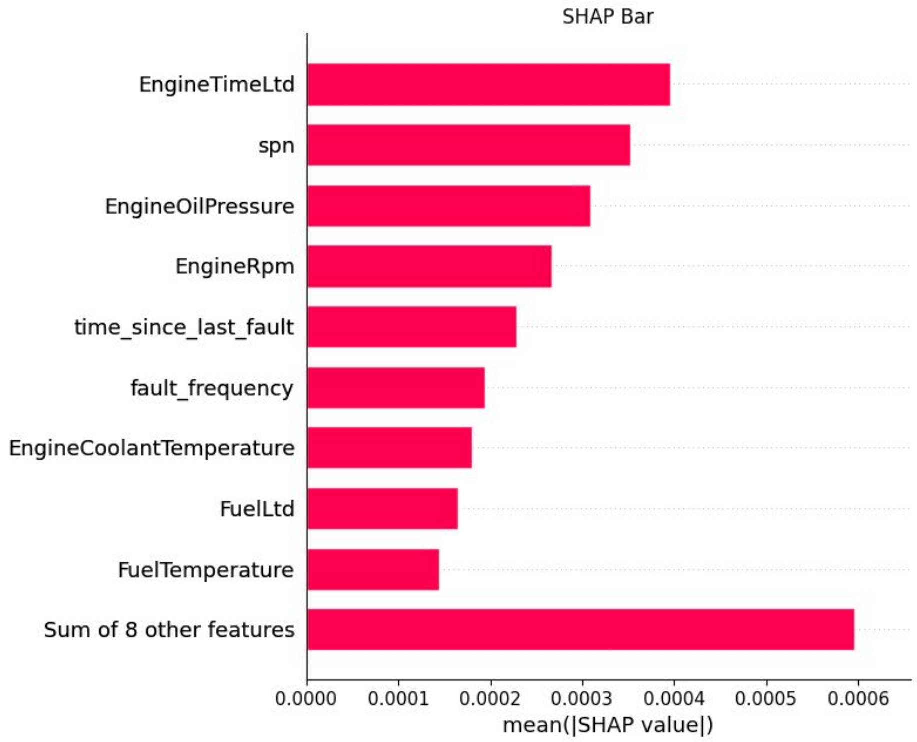
Model Evaluation

➔ YDF Gradient Boosted Learner



Model Evaluation

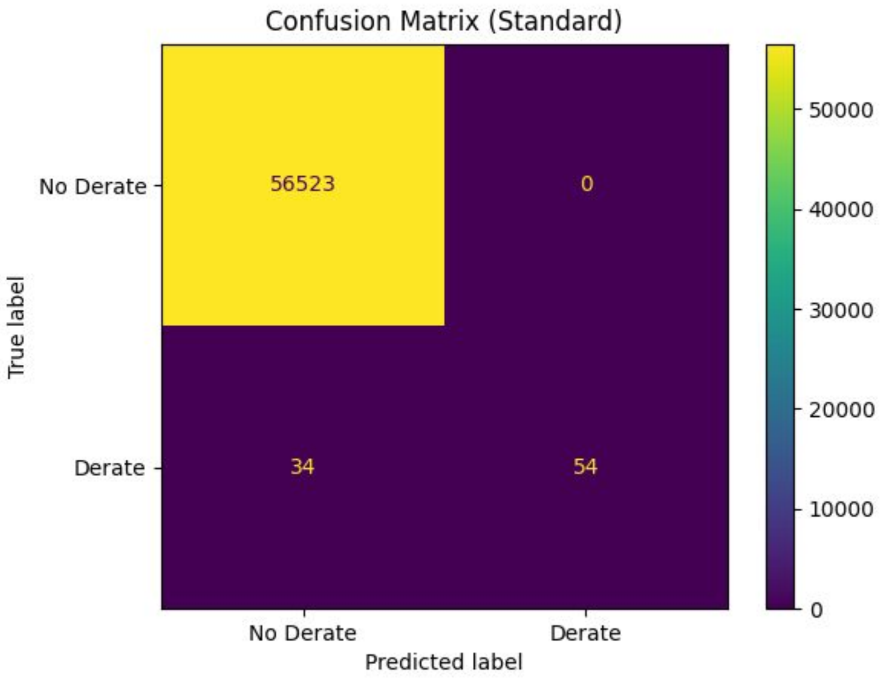
➔ YDF Gradient Boosted Learner



Model Evaluation

➔ **YDF Gradient Boosted Learner**

```
Standard Confusion Matrix:
  True Negatives (TN): 56523
  False Positives (FP): 0
  False Negatives (FN): 34
  True Positives (TP): 54
```



Model Evaluation

- **YDF Gradient Boosted Learner**
  - “Valuable TPs” were true positives that were correctly predicted at least 2 hours before the next derate and at least 24 hours after the previous derate event.
  - “Costly FPs” were false positives that occurred in isolation (were lonely) and at least 24 hours after the previous derate and at least 24 hours before the next derate.

--- Final Cost/Savings Analysis ---

➔ YDF Gradient Boosted Learner

Valuable True Positives (Savings): 0

Costly False Positives (Costs): 0

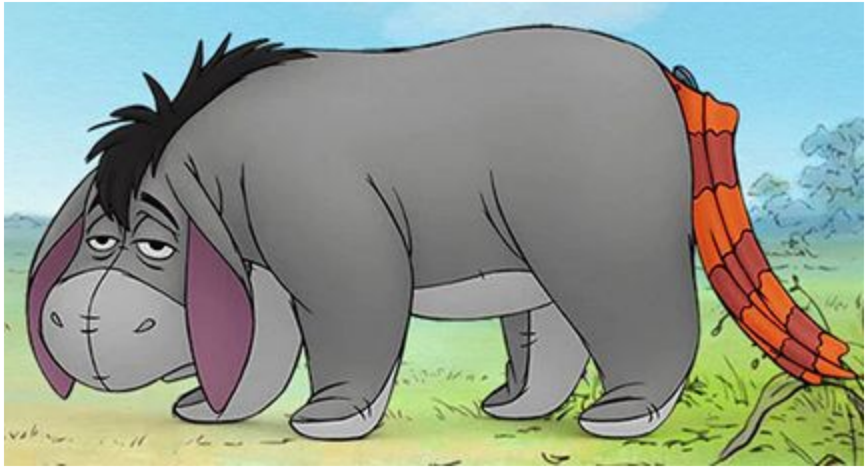
Total Savings: $0

Total Costs: $0

Net Savings (Custom Definition): $0

--- Conclusion ---

➔ It's all pointless...



Questions

Maria, Andrew, Jeff