UPTAKE

Drew Fustin data scientist

insurance marketing case study

using random forests

task

optimize marketing target customers to maximize profit

drew fustin

task

optimize marketing target customers to maximize profit

constraint

marketing costs $30 per customer, even if they don't respond

task

optimize marketing target customers to maximize profit

constraint

marketing costs $30 per customer, even if they don't respond

desire

identify which customers are most likely to respond

task

optimize marketing target customers to maximize profit

constraint

marketing costs \$30 per customer, even if they don't respond

desire

identify which customers are most likely to respond

maximize

profit = $\sum$ profit / customer – \$30 × customers marketed to

data

21 different customer characteristics, including:

- age
- marital status
- type of job
- number of times previously contacted
- outcome of previous marketing campaign
- euribor 3 month rate
- consumer price index
- etc.

data

21 different customer characteristics, including:

- age
- marital status
- type of job
- number of times previously contacted
- outcome of previous marketing campaign
- euribor 3 month rate
- consumer price index
- etc.

subset of the data (training set) also contains key variables:

- did the customer respond to this campaign?
- how much profit did the company make on the purchase?

data

In [6]:

```
train = pd.read_csv('data/training.csv')
train
```

Out[6]:

	custAge	profession	marital	schooling	default	housing	loan	contact	month	day_of_week	...	emp.var.rate	cons.price.idx
0	34	admin.	single	university.degree	no	no	yes	cellular	apr	wed	...	-1.8	93.075
1	31	services	single	high.school	no	no	no	cellular	jul	thu	...	1.4	93.918
2	NaN	admin.	single	high.school	no	no	no	telephone	jun	NaN	...	1.4	94.465
3	52	admin.	divorced	university.degree	unknown	yes	no	cellular	jul	tue	...	1.4	93.918
4	39	blue-collar	single	NaN	unknown	yes	no	cellular	jul	tue	...	1.4	93.918
5	40	entrepreneur	married	NaN	no	yes	no	telephone	jun	thu	...	1.4	94.465
6	50	technician	single	NaN	no	no	no	cellular	jul	tue	...	1.4	93.918
7	41	technician	married	professional.course	no	no	no	cellular	oct	thu	...	-3.4	92.431
8	23	blue-collar	single	basic.4y	no	yes	no	telephone	jun	fri	...	1.4	94.465
9	29	technician	married	professional.course	no	yes	no	cellular	aug	mon	...	1.4	93.444
10	57	retired	married	NaN	unknown	no	no	telephone	jun	NaN	...	1.4	94.465
11	33	blue-collar	married	unknown	no	no	no	cellular	jul	mon	...	1.4	93.918
12	NaN	technician	married	university.degree	no	yes	no	telephone	oct	wed	...	-1.1	94.601
13	59	housemaid	married	NaN	unknown	no	no	telephone	jun	thu	...	1.4	94.465

data

In [6]:

```python
train = pd.read_csv('data/training.csv')
train
```

Out[6]:

	custAge	profession	marital	schooling	default	housing	loan	contact	month	day_of_week	...	emp.var.rate	cons.price.idx
0	34	admin.	single	university.degree	no	no	yes	cellular	apr	wed	...	-1.8	93.075
1	31	services	single	high.school	no	no	no	cellular	jul	thu	...	1.4	93.918
2	NaN	admin.	single	high.school	no	no	no	telephone	jun	NaN	...	1.4	94.465
3	52	admin.	divorced	university.degree	unknown	yes	no	cellular	jul	tue	...	1.4	93.918
4	39	blue-collar	single	NaN	unknown	yes	no	cellular	jul	tue	...	1.4	93.918
5	40	entrepreneur	married	NaN	no	yes	no	telephone	jun	thu	...	1.4	94.465
6	50	technician	single	NaN	no	no	no	cellular	jul	tue	...	1.4	93.918
7	41	technician	married	professional.course	no	no	no	cellular	oct	thu	...	-3.4	92.431
8	23	blue-collar	single	basic.4y	no	yes	no	telephone	jun	fri	...	1.4	94.465
9	29	technician	married	professional.course	no	yes	no	cellular	aug	mon	...	1.4	93.444
10	57	retired	married	NaN	unknown	no	no	telephone	jun	NaN	...	1.4	94.465
11	33	blue-collar	married	unknown	no	no	no	cellular	jul	mon	...	1.4	93.918
12	NaN	technician	married	university.degree	no	yes	no	telephone	oct	wed	...	-1.1	94.601
13	59	housemaid	married	NaN	unknown	no	no	telephone	jun	thu	...	1.4	94.465

problem #1: missing data

data

In [6]:
```
train = pd.read_csv('data/training.csv')
train
```

Out[6]:

	custAge	profession	marital	schooling	default	housing	loan	contact	month	day_of_week	...	emp.var.rate	cons.price.idx
0	34	admin.	single	university.degree	no	no	yes	cellular	apr	wed	...	-1.8	93.075
1	31	services	single	high.school	no	no	no	cellular	jul	thu	...	1.4	93.918
2	NaN	admin.	single	high.school	no	no	no	telephone	jun	NaN	...	1.4	94.465
3	52	admin.	divorced	university.degree	unknown	yes	no	cellular	jul	tue	...	1.4	93.918
4	39	blue-collar	single	NaN	unknown	yes	no	cellular	jul	tue	...	1.4	93.918
5	40	entrepreneur	married	NaN	no	yes	no	telephone	jun	thu	...	1.4	94.465
6	50	technician	single	NaN	no	no	no	cellular	jul	tue	...	1.4	93.918
7	41	technician	married	professional.course	no	no	no	cellular	oct	thu	...	-3.4	92.431
8	23	blue-collar	single	basic.4y	no	yes	no	telephone	jun	fri	...	1.4	94.465
9	29	technician	married	professional.course	no	yes	no	cellular	aug	mon	...	1.4	93.444
10	57	retired	married	NaN	unknown	no	no	telephone	jun	NaN	...	1.4	94.465
11	33	blue-collar	married	unknown	no	no	no	cellular	jul	mon	...	1.4	93.918
12	NaN	technician	married	university.degree	no	yes	no	telephone	oct	wed	...	-1.1	94.601
13	59	housemaid	married	NaN	unknown	no	no	telephone	jun	thu	...	1.4	94.465

problem #1: missing data

problem #2: non-numeric data

data

In [10]:
```
train
```

Out[10]:

	custAge	profession	marital	schooling	default	housing	loan	contact	month	day_of_week	...	emp.var.rate	cons.price.idx	cons.conf.idx	eur
0	34	0	0	0	0	0	0	0	0	0	...	-1.8	93.075	-47.1	1.4
1	31	1	0	1	0	0	1	0	1	1	...	1.4	93.918	-42.7	4.9
2	33	0	0	1	0	0	1	1	2	2	...	1.4	94.465	-41.8	4.9
3	52	0	1	0	1	1	1	0	1	2	...	1.4	93.918	-42.7	4.9
4	39	2	0	5	1	1	1	0	1	2	...	1.4	93.918	-42.7	4.9
5	40	3	2	1	0	1	1	1	2	1	...	1.4	94.465	-41.8	4.8
6	50	4	0	2	0	0	1	0	1	2	...	1.4	93.918	-42.7	4.9
7	41	4	2	2	0	0	1	0	3	1	...	-3.4	92.431	-26.9	0.7
8	23	2	0	3	0	1	1	1	2	3	...	1.4	94.465	-41.8	4.9
9	29	4	2	2	0	1	1	0	4	4	...	1.4	93.444	-36.1	4.9
10	57	5	2	6	1	0	1	1	2	4	...	1.4	94.465	-41.8	4.9
11	33	2	2	4	0	0	1	0	1	4	...	1.4	93.918	-42.7	4.9
12	30	4	2	0	0	1	1	1	3	0	...	-1.1	94.601	-49.5	0.9
13	59	6	2	5	1	0	1	1	2	1	...	1.4	94.465	-41.8	4.9

problem #1: missing data

solution: guess missing values

problem #2: non-numeric data

data

```
In [10]:  1 train
```

Out[10]:

	custAge	profession	marital	schooling	default	housing	loan	contact	month	day_of_week	...	emp.var.rate	cons.price.idx	cons.conf.idx	eur
0	34	0	0	0	0	0	0	0	0	0	...	-1.8	93.075	-47.1	1.4
1	31	1	0	1	0	0	1	0	1	1	...	1.4	93.918	-42.7	4.9
2	33	0	0	1	0	0	1	1	2	2	...	1.4	94.465	-41.8	4.9
3	52	0	1	0	1	1	1	0	1	2	...	1.4	93.918	-42.7	4.9
4	39	2	0	5	1	1	1	0	1	2	...	1.4	93.918	-42.7	4.9
5	40	3	2	1	0	1	1	1	2	1	...	1.4	94.465	-41.8	4.8
6	50	4	0	2	0	0	1	0	1	2	...	1.4	93.918	-42.7	4.9
7	41	4	2	2	0	0	1	0	3	1	...	-3.4	92.431	-26.9	0.7
8	23	2	0	3	0	1	1	1	2	3	...	1.4	94.465	-41.8	4.9
9	29	4	2	2	0	1	1	0	4	4	...	1.4	93.444	-36.1	4.9
10	57	5	2	6	1	0	1	1	2	4	...	1.4	94.465	-41.8	4.9
11	33	2	2	4	0	0	1	0	1	4	...	1.4	93.918	-42.7	4.9
12	30	4	2	0	0	1	1	1	3	0	...	-1.1	94.601	-49.5	0.9
13	59	6	2	5	1	0	1	1	2	1	...	1.4	94.465	-41.8	4.9

problem #1: missing data — solution: guess missing values

problem #2: non-numeric data — solution: transform into integer factors

drew tuslin

UPTAKE

data

```
In [10]:  1  train
```

Out[10]:

	custAge	profession	marital	schooling	default	housing	loan	contact	month	day_of_week	...	emp.var.rate	cons.price.idx	cons.conf.idx	eur
0	34	0	0	0	0	0	0	0	0	0	...	-1.8	93.075	-47.1	1.4
1	31	1	0	1	0	0	1	0	1	1	...	1.4	93.918	-42.7	4.9
2	33	0	0	1	0	0	1	1	2	2	...	1.4	94.465	-41.8	4.9
3	52	0	1	0	1	1	1	0	1	2	...	1.4	93.918	-42.7	4.9
4	39	2	0	5	1	1	1	0	1	2	...	1.4	93.918	-42.7	4.9
5	40	3	2	1	0	1	1	1	2	1	...	1.4	94.465	-41.8	4.8
6	50	4	0	2	0	0	1	0	1	2	...	1.4	93.918	-42.7	4.9
7	41	4	2	2	0	0	1	0	3	1	...	-3.4	92.431	-26.9	0.7
8	23	2	0	3	0	1	1	1	2	3	...	1.4	94.465	-41.8	4.9
9	29	4	2	2	0	1	1	0	4	4	...	1.4	93.444	-36.1	4.9
10	57	5	2	6	1	0	1	1	2	4	...	1.4	94.465	-41.8	4.9
11	33	2	2	4	0	0	1	0	1	4	...	1.4	93.918	-42.7	4.9
12	30	4	2	0	0	1	1	1	3	0	...	-1.1	94.601	-49.5	0.9
13	59	6	2	5	1	0	1	1	2	1	...	1.4	94.465	-41.8	4.9

problem #1: missing data

solution: guess missing values

problem #2: non-numeric data

aside: nope. should have used one-hot encoding

how do we guess missing data?

the same way we are going to predict response rates

how do we guess missing data?

- the same way we are going to predict response rates
  - using random forests

random forests

- build a random decision tree on some of the characteristics
  - e.g. if they're over 35, go left. if not, go right.
  - continue down the tree branching on different factors
  - once done, send training set customers through the tree
  - count the number of 'yes' responses at each end

random forests

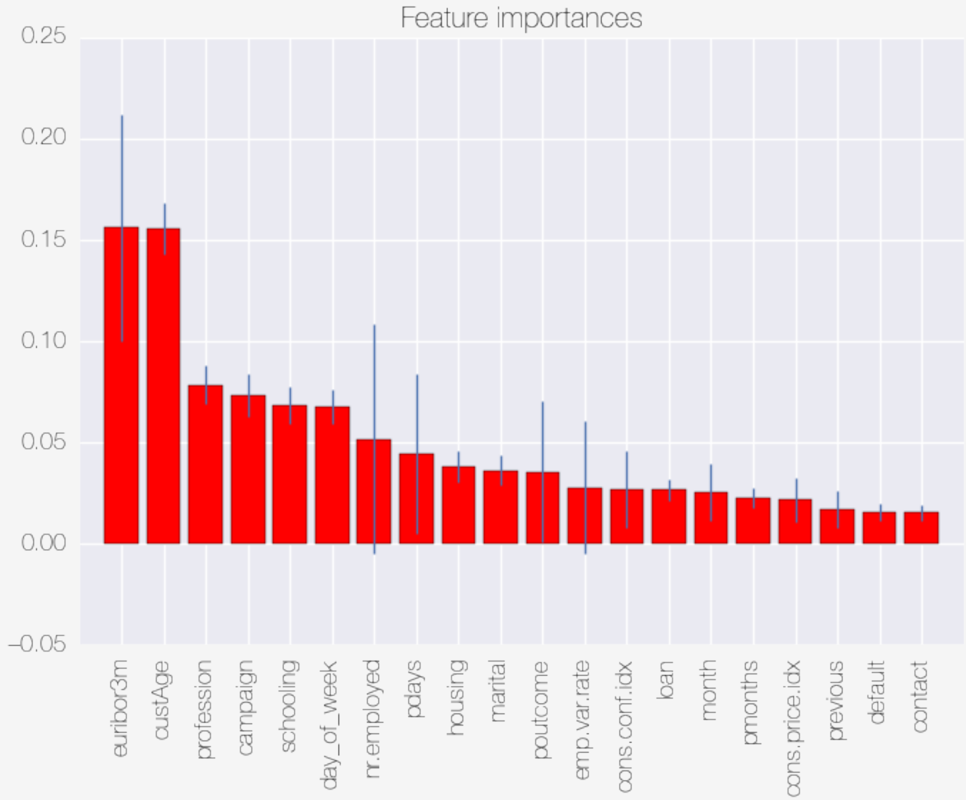
aside: should have done proper feature selection

- build a random decision tree on some of the characteristics
  - e.g. if they're over 35, go left. if not, go right.
  - continue down the tree branching on different factors
  - once done, send training set customers through the tree
  - count the number of 'yes' responses at each end

random forests

aside: should have done proper feature selection

- build a random decision tree on some of the characteristics
  - e.g. if they're over 35, go left. if not, go right.
  - continue down the tree branching on different factors
  - once done, send training set customers through the tree
  - count the number of 'yes' responses at each end



I didn't properly handle this necessarily – I just passed all features through the forest. I could have more intelligently performed feature selection to only use important ones.

random forests

- build a random decision tree on some of the characteristics
    - e.g. if they're over 35, go left. if not, go right.
    - continue down the tree branching on different factors
    - once done, send training set customers through the tree
    - count the number of 'yes' responses at each end
- build *lots* of decision trees on different sets of characteristics

random forests

- build a random decision tree on some of the characteristics
  - e.g. if they're over 35, go left. if not, go right.
  - continue down the tree branching on different factors
  - once done, send training set customers through the tree
  - count the number of 'yes' responses at each end
- build *lots* of decision trees on different sets of characteristics

aside: how many? what kind? should have performed a more robust grid search to best determine the hyperparameters of the random forest

random forests

- build a random decision tree on some of the characteristics
  - e.g. if they're over 35, go left. if not, go right.
  - continue down the tree branching on different factors
  - once done, send training set customers through the tree
  - count the number of 'yes' responses at each end
- build *lots* of decision trees on different sets of characteristics
- change the cutoff for the branch each time
  - e.g. split on 28 instead of 35 this time

random forests

- build a random decision tree on some of the characteristics
  - e.g. if they're over 35, go left. if not, go right.
  - continue down the tree branching on different factors
  - once done, send training set customers through the tree
  - count the number of 'yes' responses at each end
- build *lots* of decision trees on different sets of characteristics
- change the cutoff for the branch each time
  - e.g. split on 28 instead of 35 this time
- some trees won't pick good predictors, some will

random forests

- build a random decision tree on some of the characteristics
  - e.g. if they're over 35, go left. if not, go right.
  - continue down the tree branching on different factors
  - once done, send training set customers through the tree
  - count the number of 'yes' responses at each end
- build *lots* of decision trees on different sets of characteristics
- change the cutoff for the branch each time
  - e.g. split on 28 instead of 35 this time
- some trees won't pick good predictors, some will
- average over all trees, bad branches mostly cancel out

drew fustin

UPTAKE

random forests

- build a random decision tree on some of the characteristics
    - e.g. if they're over 35, go left. if not, go right.
    - continue down the tree branching on different factors
    - once done, send training set customers through the tree
    - count the number of 'yes' responses at each end
- build *lots* of decision trees on different sets of characteristics
- change the cutoff for the branch each time
    - e.g. split on 28 instead of 35 this time
- some trees won't pick good predictors, some will
- average over all trees, bad branches mostly cancel out
- send the customers you want to predict through the forest

decision tree example

drew fustin

decision tree example

decision tree example

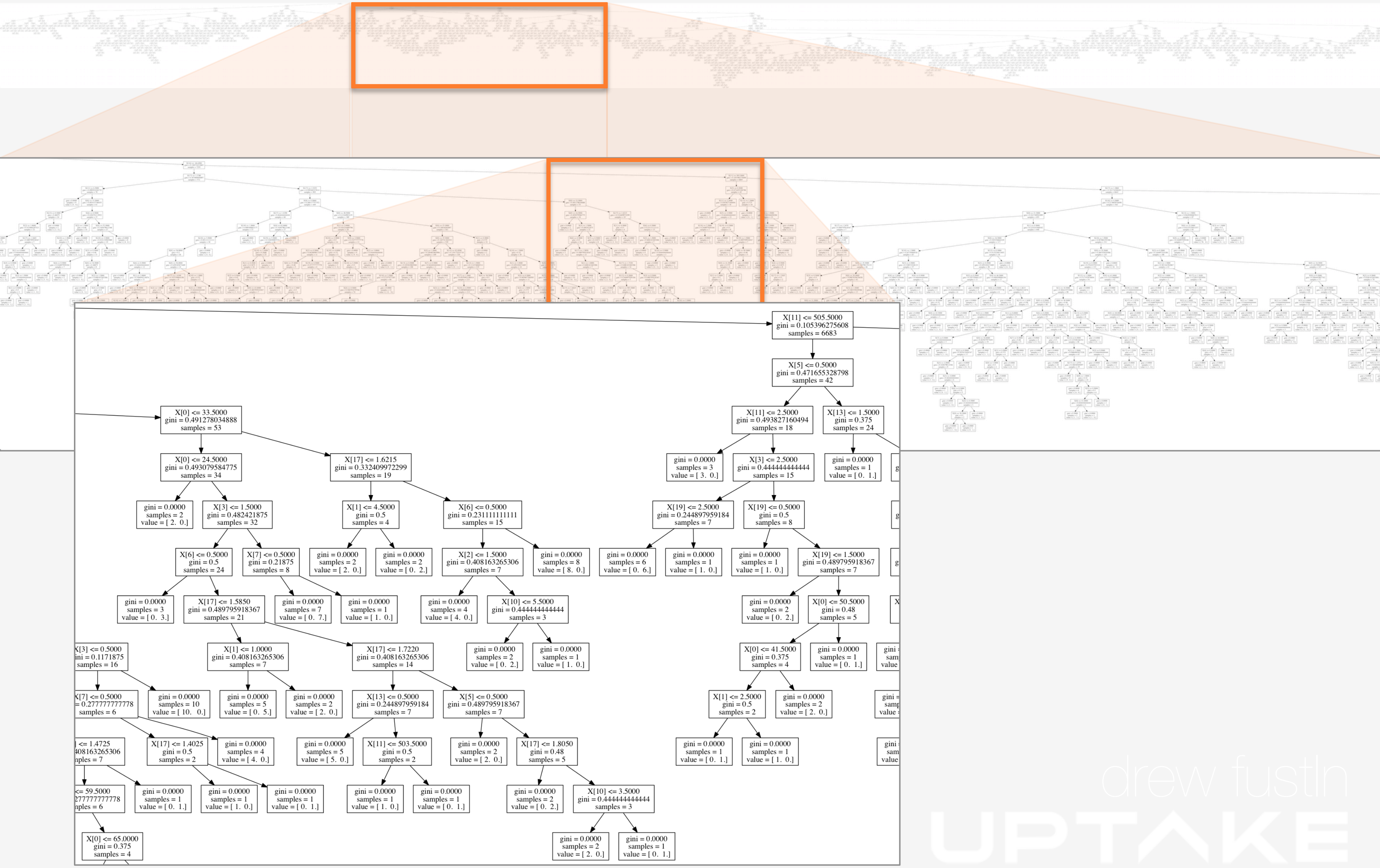
decision tree example

drew fustin

UPTAKE

decision tree example

decision tree example



how do we know this forest is good?

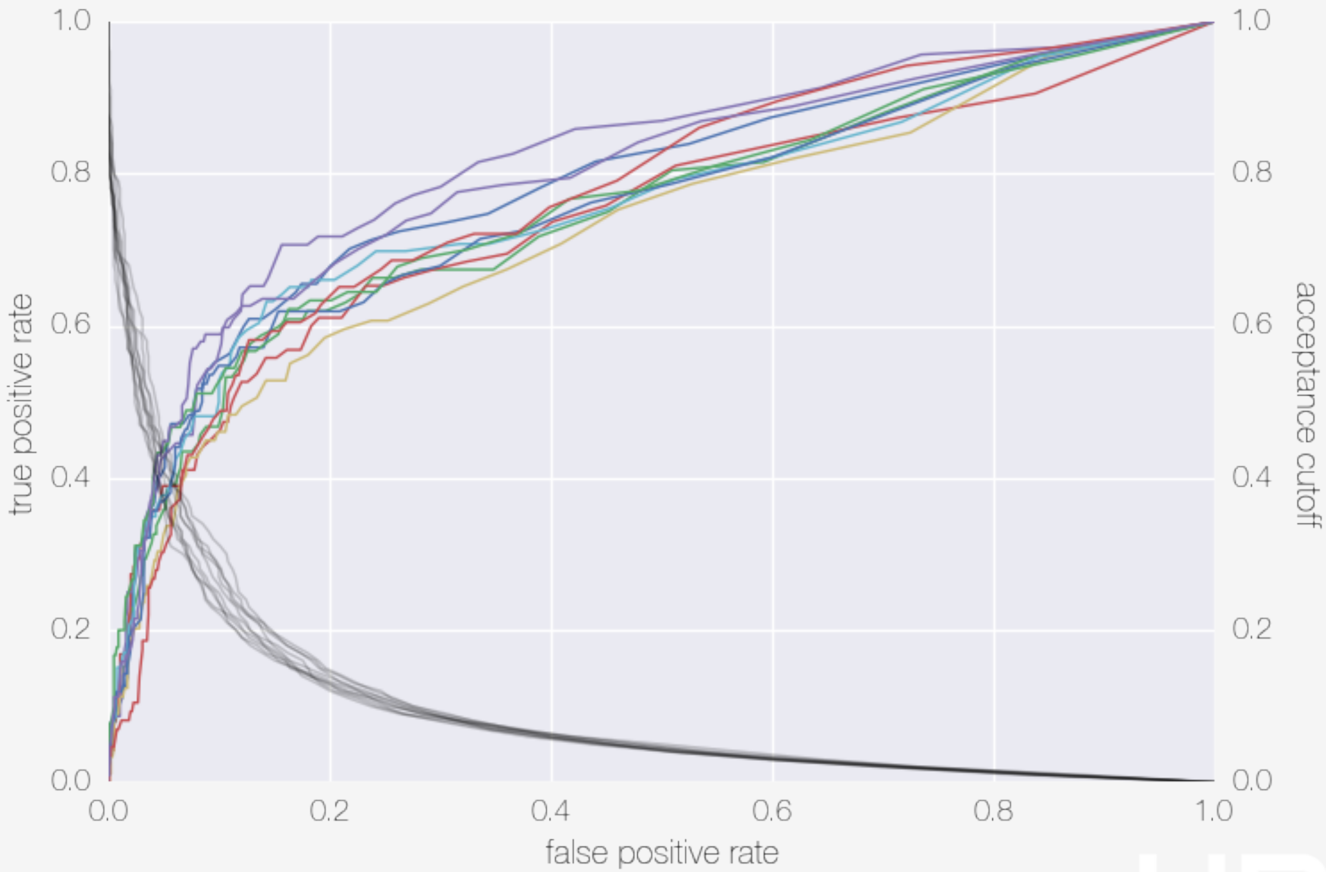
- split the data we *know* (training set) into multiple pieces
- pretend we don't know the answers for one piece
- predict what our forest would guess
- compare to the actual answer

picking customers

- passing customer through forest gives ‘yes’ probability
- market to customers with probability > cutoff

picking customers

- passing customer through forest gives 'yes' probability
- market to customers with probability > cutoff

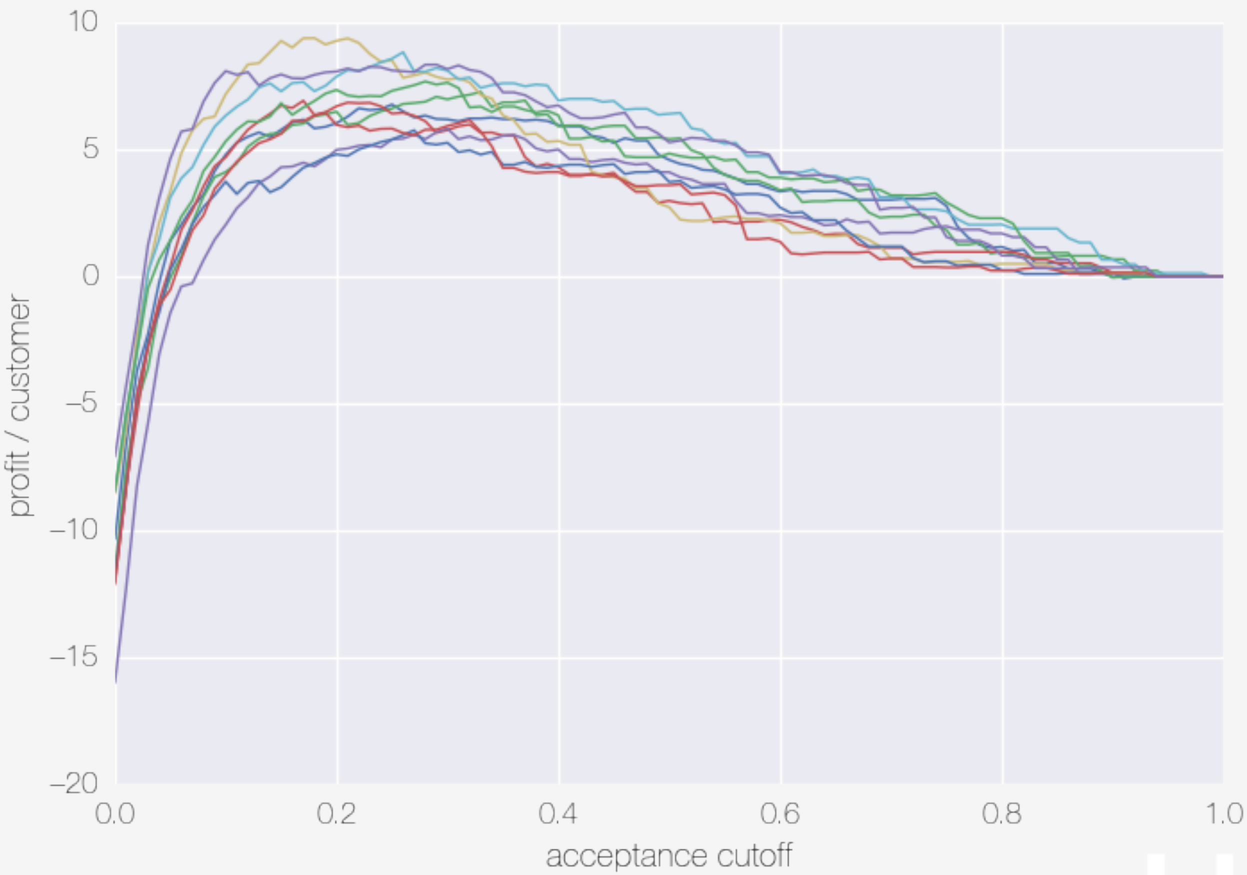


drew fustin

UPTAKE

picking customers

- passing customer through forest gives 'yes' probability
- market to customers with probability > cutoff
- cutoff is chosen such that it maximizes profit in training set



future ready model

passing a single customer through the forest is easy, and we can always retrain the model as more data comes in to keep it robust and predictive


UPTAKE
Drew Fustin data scientist