GA SF Data Science 20

The Forever War

by Andrew Lovett-Barron

One Caltrain Rider's

Data-Driven Cry for Help

github.com/readywater/caltrain-predict

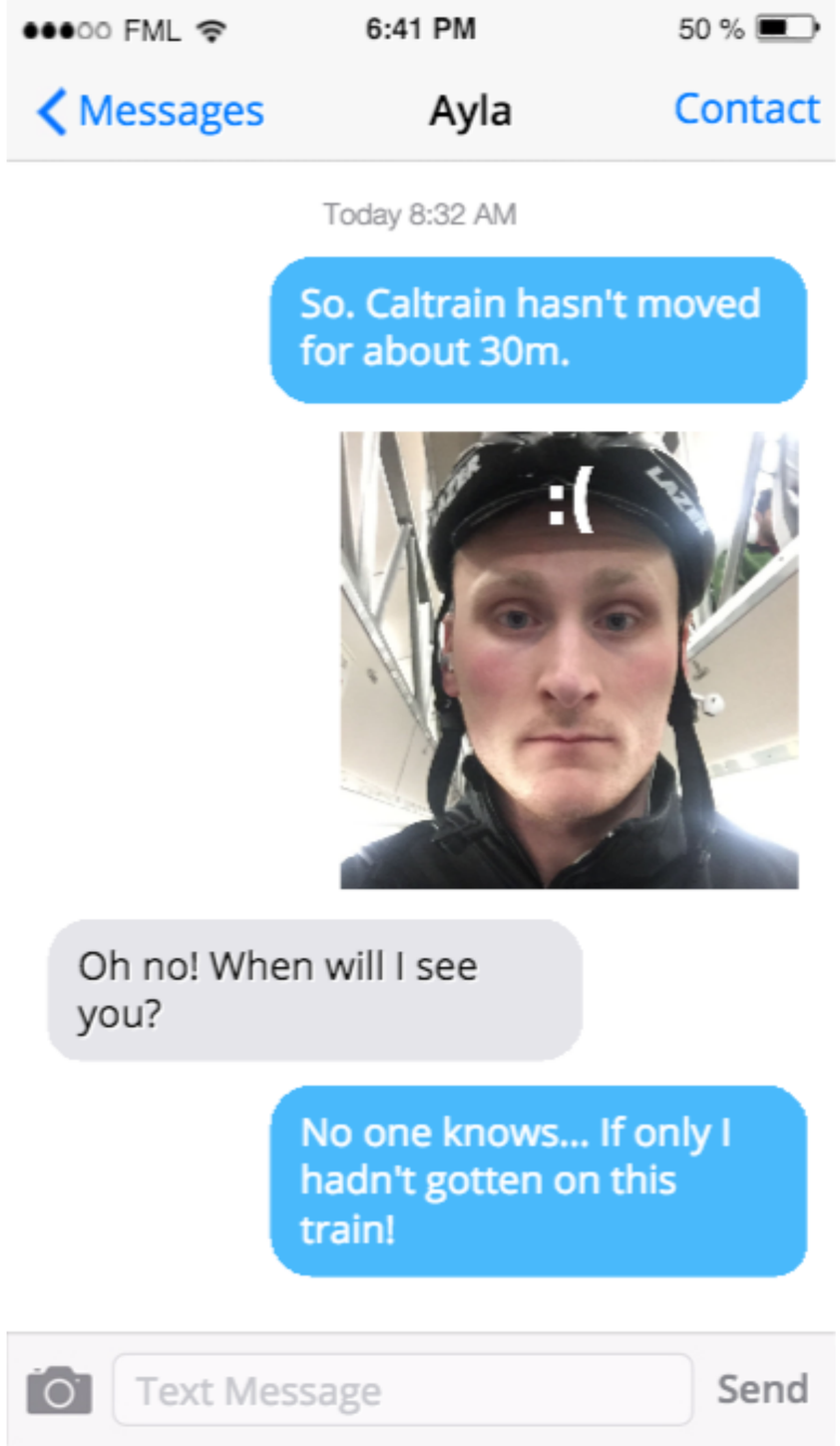
@readywater

The Problem

Daily, the nomadic San Franciscan dons their backpack to travel into the valley. They find sustenance and productivity there, only to return with offerings to their 7x7 landlords.

But to travel south, they face an indomitable and unpredictable challenge:

The Caltrain.



My Story

I commute to Palo Alto for my work at IDEO.

Most of our community lives in the city and commutes in to PA. We rely on the Caltrain: that fickle red demon of the peninsula.

If only we had a **better intuition for knowing what caused delay** or **when a delay might be coming.**

We might stick around in the office longer, grab a beer, make something interesting, leave a bit earlier, or spend $70 on an uber.

Building Blocks

Just as with travel, the Caltrain does not make data easy.

They report delays via **@caltrain** né **@Caltrain_news**.

Their schedule is via **PDF schedule of its routes.**

A schedule API is available, but not relevant to this analysis.

On the week of this presentation, Caltrain also just changed their schedule to spite me.

A million miles

Scrape Data from Twitter (twitter api)

Separate Events (custom functions)

Hand Clean Data (Numbers/Data dictionary)

Merge hand Truth w/ Twitter (pandas)

Generate schedule matrix (Pandas/custom functions)

Add Weather (forecast.io)

Merge with twitter truth data (Pandas)

Twitter Data

I wrote functions to clean and detect mentions of specific trains within tweets.

From these, I manually identified delay types given the following data dictionary, based on the observation that @caltrain did not give reliable continuous ranges.

is_delay: 1 = true, 0 false (ignore 0)
delay_minor: >=10min = 1 else 0
delay_med: < 10m & >=20m = 1 else 0
delay_major: < 20m & >=40m = 1 else 0
delay_catastrophic:< 40m = 1 else 0
is_backlog: multiple trains IN SAME DIRECTION delayed or passing else 0
is_canceled: Train Canceled else 0
is_passing: Train indicates passing/rescheduling else 0
is_accident: Mention of accident, vehicle on tracks, or fatality else 0
is_mechanical: Mention of mechanical issue else 0
is_customer: Mention of rider-caused disruption (eg fare evasion) else 0
is_event: Mention of event-caused disruption (eg Giants game) else 0
NOTE: is_delay with no indication of magnitude means UNKNOWN but reported

Schedule Data

I generated a complete weekday schedule from the pdf caltrain schedule, with Palo Alto as the counterpoint of that matrix.

This was populated with **historical weather data from forecast.io** and parsed train IDs for **direction**, **time of day**, and **speed.**

This “scheduled train” set was then merged with the twitter-driven delay data set based on **train ID** and **Date.** This was accomplished by an unnecessarily complicated for loop.

This gave me a dataset that had **21021 observations** and **33 features.**

By the Numbers

The Caltrain runs **91 times per weekday**.

I sampled data from twitter between 5pm on **June 8th, 2015** and midnight on **January 25th, 2016**.

That's **291 days** of delays.

Across ~**3200** and **291 days** (the API limit), there were **584 delayed trains** reported.

During those 291 days, there were **~21000 individual train trip**.

So only **1.56% of all trains were delayed.**

Delay, Actually

The challenge embedded in all of this is that we have a massive POSITIVE data set where everything (according to what I could scrape) appears fine.

As a baseline, I will be approaching certain scores with suspicion.

Delay Only Categories

del_min: 27.8%

del_med: 43.4%

del_maj: 8.5%

del_cat:4.9%

Uncategorized: 15.4%

Total Set

is_delay: 1.56%

del_min: 0.43%

del_med: 0.68%

del_maj: 0.13%

del_cat: 0.076%

Approach

First

I have suspicions about what affects the Caltrain's delays. How valid are these?

Second

Can I predict (with a a reasonable degree of accuracy) the likelihood of a delay and its magnitude given the current data-set?

Hypotheses

1) **Weather** creates delay, but not major ones.

2) The **day of the week** affects delay, likely because of drinking, fatigue, or city events.

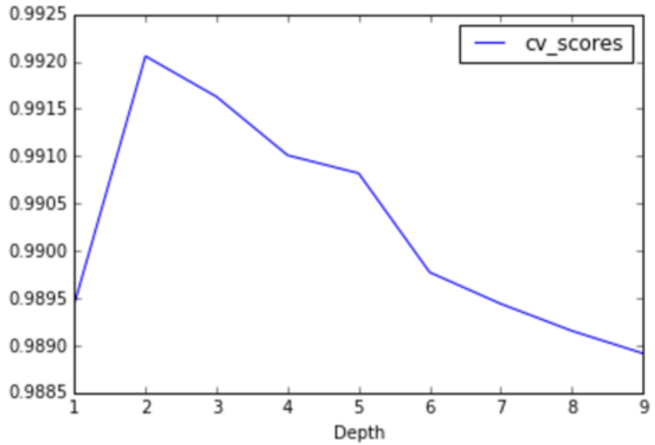
3) **Northbound** trains are more likely to be delayed.

4) **Certain Trains** (with a train ID) are more likely to be delayed than others (likely because of rush hour)

5) Delay is more likely at certain **times of the year**.

Decision Tree

Took the approach of running a decision tree on the following set.



Trees: 2

CV_score: 0.992959

Trained MSE: 0.007

X

'temp','precipiation','visability','windspeed','humidity','cloudcover','is_bullet','is_limited','t_northbound',

Dummy Vars: Train_ids (91 cat, 'tid_101' base), Weekdays (7 days, 'd_sunday' base)

Y

'is_delay'

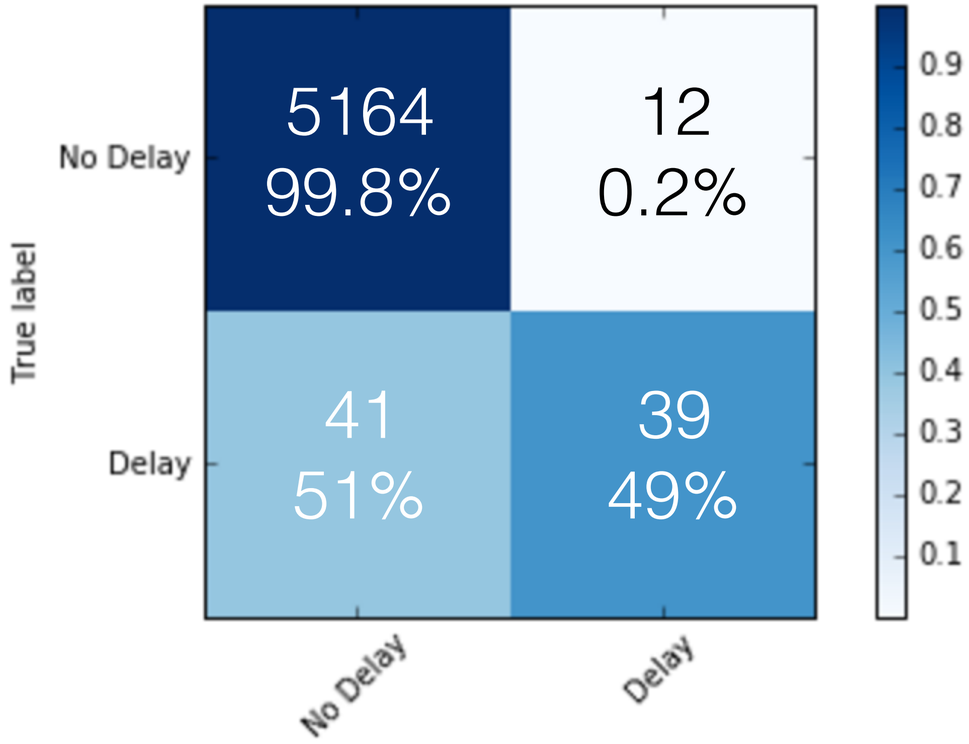
Source Code

https://github.com/readywater/caltrain-predict/blob/master/07focus_decision_tree.ipynb

Decision Tree

Because we have an overwhelming number of No Delay observations, we are focusing mostly on the Delay events.

ROC = 0.838



Accuracy	Misclass	**Sensitivity**	Specificity
99.77 %	0.23 %	**48.75%**	99.77 %

Key Features

Decision Trees

	Gini_Import	feature
7	0.521470	is_limited
6	0.293549	is_bullet
0	0.051288	temp
5	0.033127	cloudcover
4	0.032764	humidity
3	0.016853	windspeed
10	0.010351	d_wednesday
2	0.008541	visability
17	0.006534	tid_135.0
12	0.005773	d_friday

Gradient Boosting

	Importance	feature
0	0.169679	temp
4	0.167738	humidity
5	0.152647	cloudcover
3	0.149600	windspeed
2	0.083236	visability
12	0.037309	d_friday
7	0.028038	is_limited
9	0.025530	d_tuesday
1	0.021733	precipiation
11	0.019582	d_thursday

Decision Tree

This model validated a few assumptions I had:

1) **Train ID** and the **Type of train** were major variables. I suspect this has to do with time of day.
   eg) 267 is Northbound at 454pm in PA
   eg) 283 is Northbound at 624pm in PA

2) Cloud Cover and Temperature have an impact, but unsure to what degree. Cloud cover seems counterintuitive to me, so I want to push this.

Hypotheses

1) **Weather** creates delay, but not major ones.

2) The **day of the week** affects delay, likely because of drinking, fatigue, or city events.

3) **Northbound** trains are more likely to be delayed.

4) **Certain Trains** (with a train ID) are more likely to be delayed than others (likely correlates with time/speed)

5) Delay is more likely at certain **times of the year**.

Prediction

Take the approach of using a gradient boosting classifier to try and identify the greatest predicability for True Delay events.

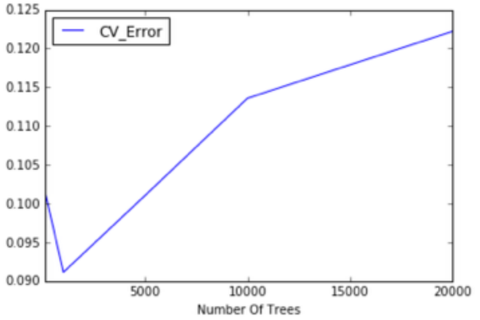
I care a lot less about the false negative (False No Delay) identification, so I'm willing to accept a wider margin of error there.

I'm using the same data set from my decision tree exploration.

Training

I tuned the Gradient Boosting algorithm using cross-validation on the tree number and the depth.

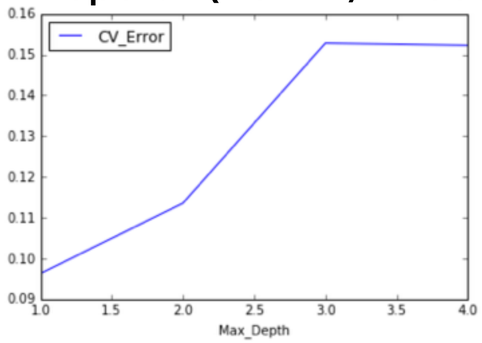
Number of Trees : 1000



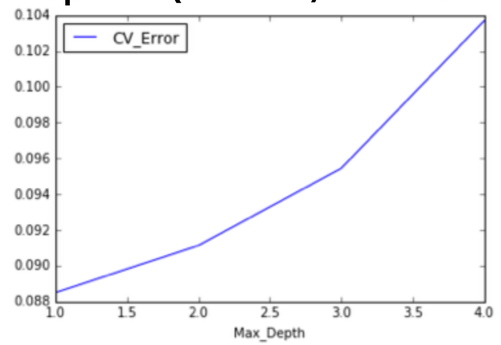
<u>Best Outcomes</u>
Lambda: 0.01
Trees: 1000
Depth: 1

MSE = 0.00852

Depth: 1 (Run w/ Trees: 10k)

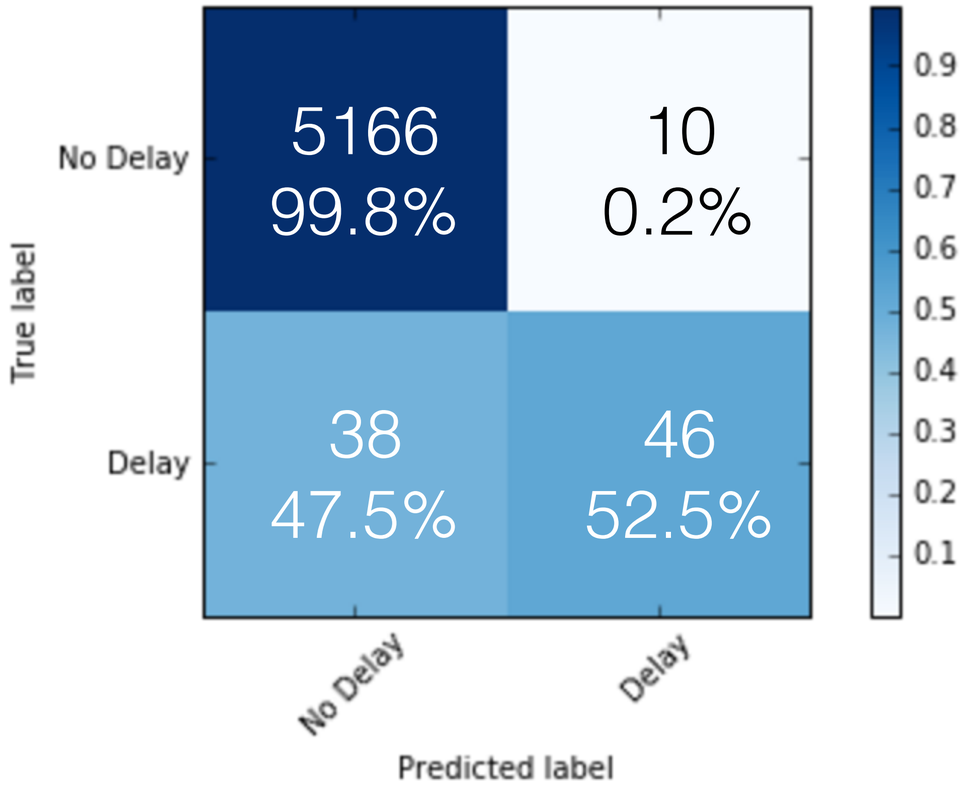


Depth 1 (Run w/ Trees: 1000)



Gradient Boosting

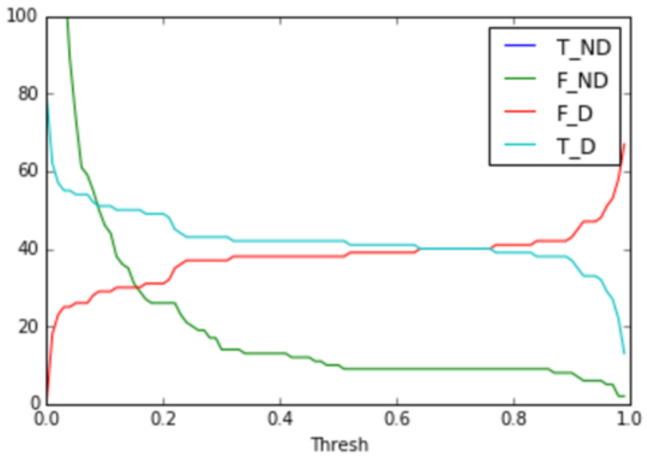
Gradient Boosting did see an increase in sensitivity over decision trees, but it was minor.

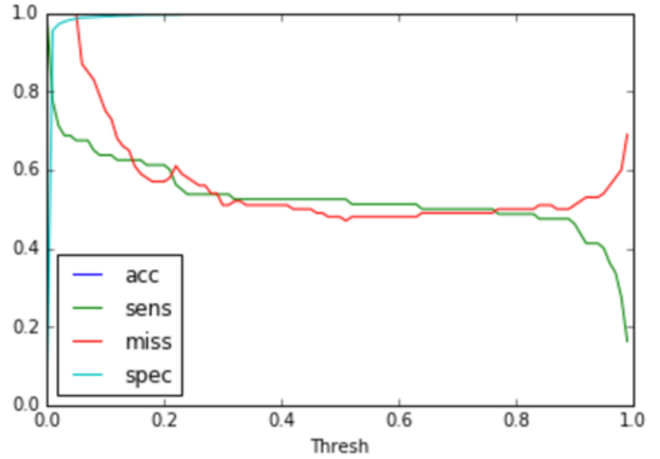


Accuracy	Misclass	**Sensitivity**	Specificity
99.09 %	0.091 %	**52.5 %**	99.81 %

Modified Threshold

When selecting the threshold, I looked for points where the TRUE DELAY was higher than FALSE DELAY, and then didn't worry too much about non-delay events.



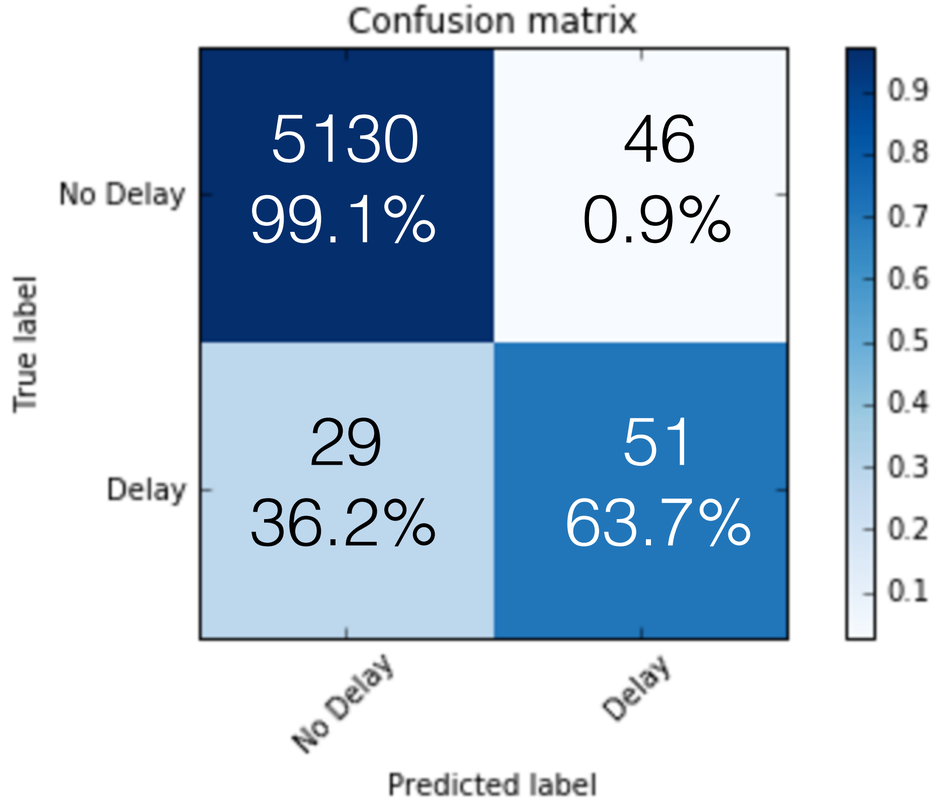


Modified Threshold

To focus on correctly predicting DELAY though, I changed the threshold for “delay" prediction to 90% probability.

Throwing caution to the wind, I also saw 71% when testing at 98% threshold.

Accuracy	Misclass	**Sensitivity**	Specificity
98.57 %	1.4 %	**63.7 %**	99.11 %



Will I be delayed?

I wrote a function to ask the very simple question:

“Will the train I’m hoping to take be delayed today?”

With a simple train ID input and a probability threshold option to override the default of 0.2, it extracts the basic features from the train ID to map it to the schedule, and queries forecast.io for the current weather.

Based on this model, it estimates the probability of delay and provides an opinion based on the threshold.

Will I be delayed?

```
def get_train_prediction(trainid,threshold=0.2):
    The_train = str(trainid)

    time = datetime.datetime.now()
    forecast = forecastio.load_forecast(api_key, lat, lng,time=time)
    current = forecast.currently()

    Predict_me = pd.DataFrame({
            'temp': [current.d['apparentTemperature']],
            'precipiation': [current.d['precipIntensity']],
            'visability': [current.d['visibility']],
            'windspeed': [current.d['windSpeed']],
            'humidity': [current.d['humidity']],
            'cloudcover': [current.d['cloudCover']],
            'is_bullet': 1 if str(The_train)[0] == '2' else 0,
            'is_limited':1 if str(The_train)[0] == '3' else 0,
            't_northbound': int(The_train)%2,
            'd_tuesday': 1 if time.weekday == 1 else 0,
            'd_wednesday':1 if time.weekday == 2 else 0,
            'd_thursday':1 if time.weekday == 3 else 0,
            'd_friday':1 if time.weekday == 4 else 0
            })


    t = pd.DataFrame(columns=tid_col)
    t = t.append([0]).fillna(0)
    t['tid_'+The_train+'.0'] = 1
    t['tid_'+The_train+'.0']
    Predict_me = pd.concat([Predict_me, t],axis=1)
    del Predict_me[0]
    pprob = GBC_Tree.predict_proba(Predict_me).T
    pred = 1 if pprob[1] >= threshold else 0 # GBC_Tree.predict(Predict_me)
    print "Will be delayed:",pred
    print "Probability:",np.round(pprob.T,4)
    return [pred,pprob]
```

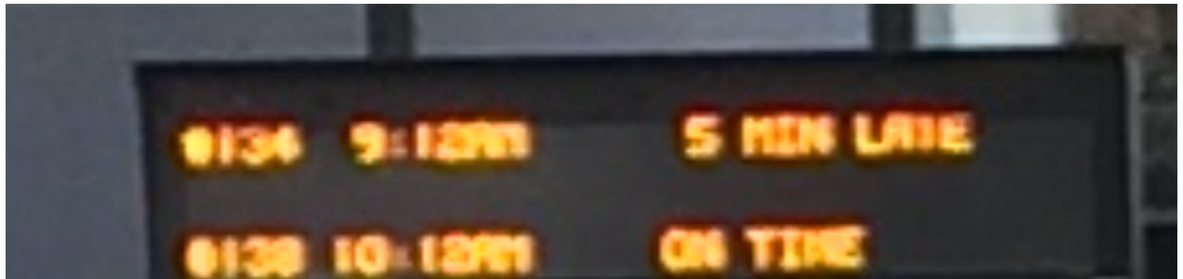
Will I be delayed?

So you ask, will it work?

On a test of caltrain 134, going south to Palo Alto, my model told me that there was a 79% chance that I would be delayed. It did not identify that there was a delay, however.

The train was 8 minutes late to 22nd street.

The delay was not reported on @caltrain.



#kegofglory

So while modest,

this small anecdotal victory made the whole damn thing incredibly worthwhile.

I danced to Palo Alto that morning.

Model Matters

Frustration

Momentum

Anticipation

Disappointment

Work-Life Balance

Social Responsibility

Agency Within a System

Form Factor

I want to help my friends and colleagues feel a bit less annoyed by the common but disruptive delays of our primary mode of transportation.

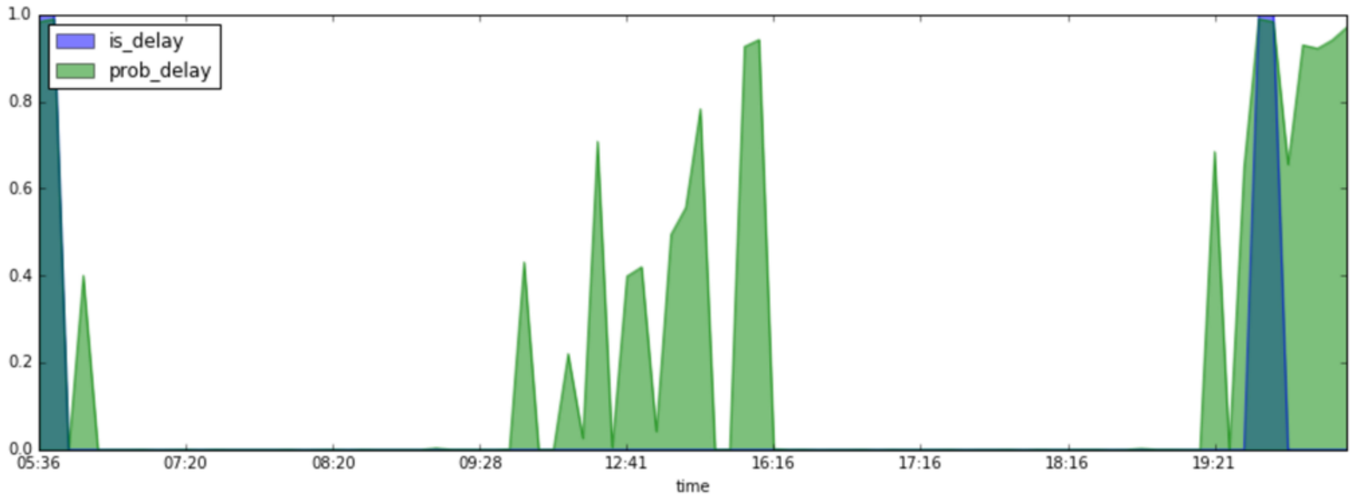
In the end, my model isn't that strong. So is it possible to get value from a weak model?

How might we house a predictive model? Instead of classification, but instead **help users intuit probability?**

Form Factor

Beyond prediction at a glance, **can we interpret the risk within a system as a whole?**

As a test, I ran the model across an entire day. The probability increased over time, but only predicted delay twice.



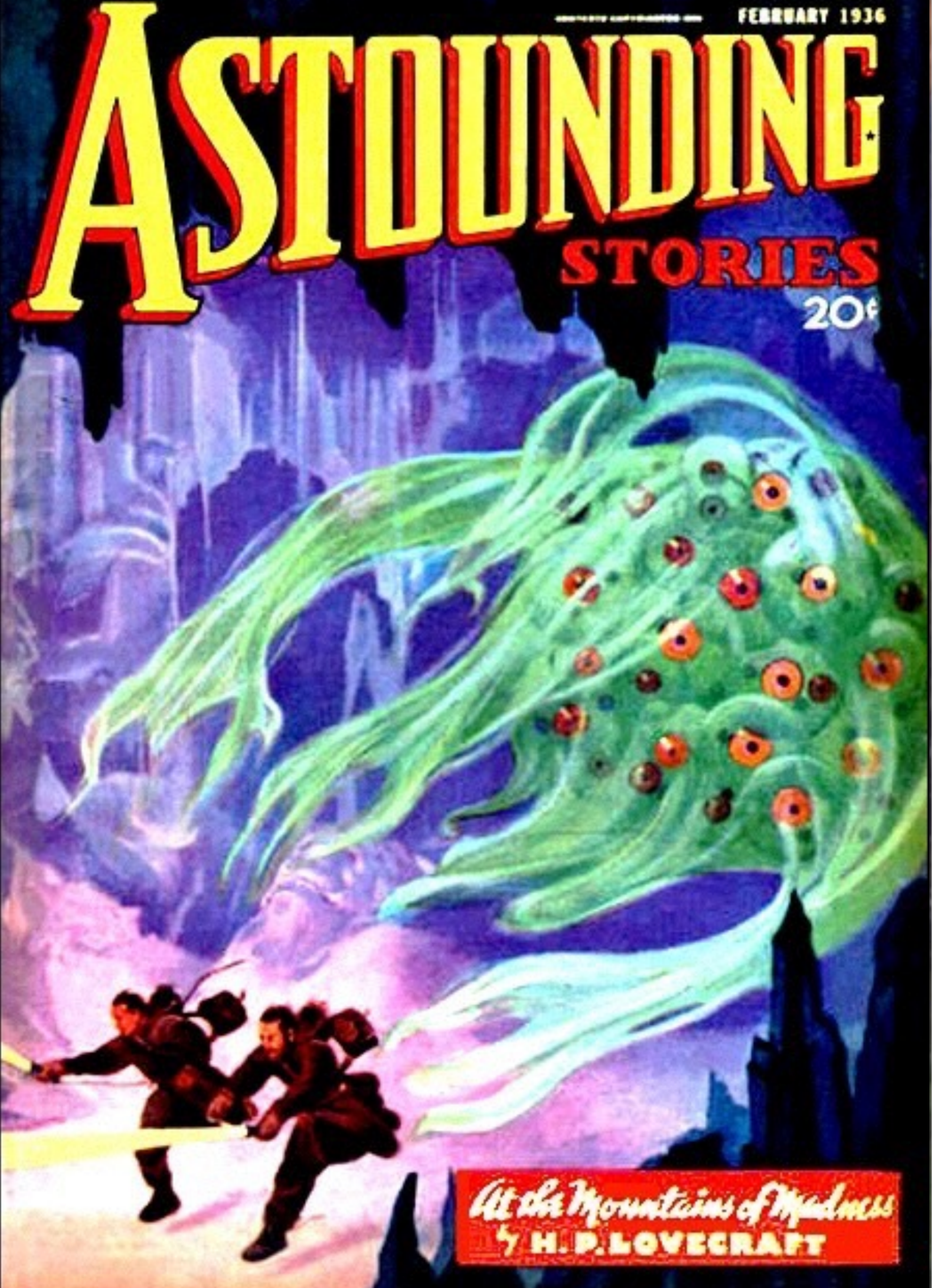
Next Steps

Data Challenges

In the end, I didn't love my data. Next time, I would also include event data like Giants games, school schedules for delays and mental health, and take an NLP approach to realtime twitter parsing. I would also integrate delay magnitude and type.

Model Challenges

Really interested in exploring ensemble approaches. PCA + KNN for looking at train category trends. Would love to try SVM as a next step. Constructing a supporting model to identify delays via twitter as well to feed dataset.



Designing Reality

End of the day, I am interested in how we can use data and probability-driven prediction to affect our lived experience.

Might we someday develop the literacy necessary to intuit and internalize the multiple worlds presented through predictive models?

Can design help bridge that gap?

Looking forward to ascending the mountains of madness with you all.

Keep in Touch

🐦 @readywater 🐦

📄 alb@ideo.com 📄

👔 linkedin.com/in/andrewlb 👔

Project

🔬 github.com/readywater/caltrain-predict 🔬