Section 3

slide pack by

Leo Valberg

Data

We have one year’s historic data for 500 posts

Inputs: category, type, month, hour, weekday, paid, page total likes

Outcomes include: total impressions, reach, consumers and interactions

Task

Produce a classifier that can make recommendations to optimise outcomes of future Facebook posts

Outcome

An algorithm that can say which strategy has the best change to maximise post performance

The classifier is available as a Jupyter notebook for testing at XXXXXXX

Issues

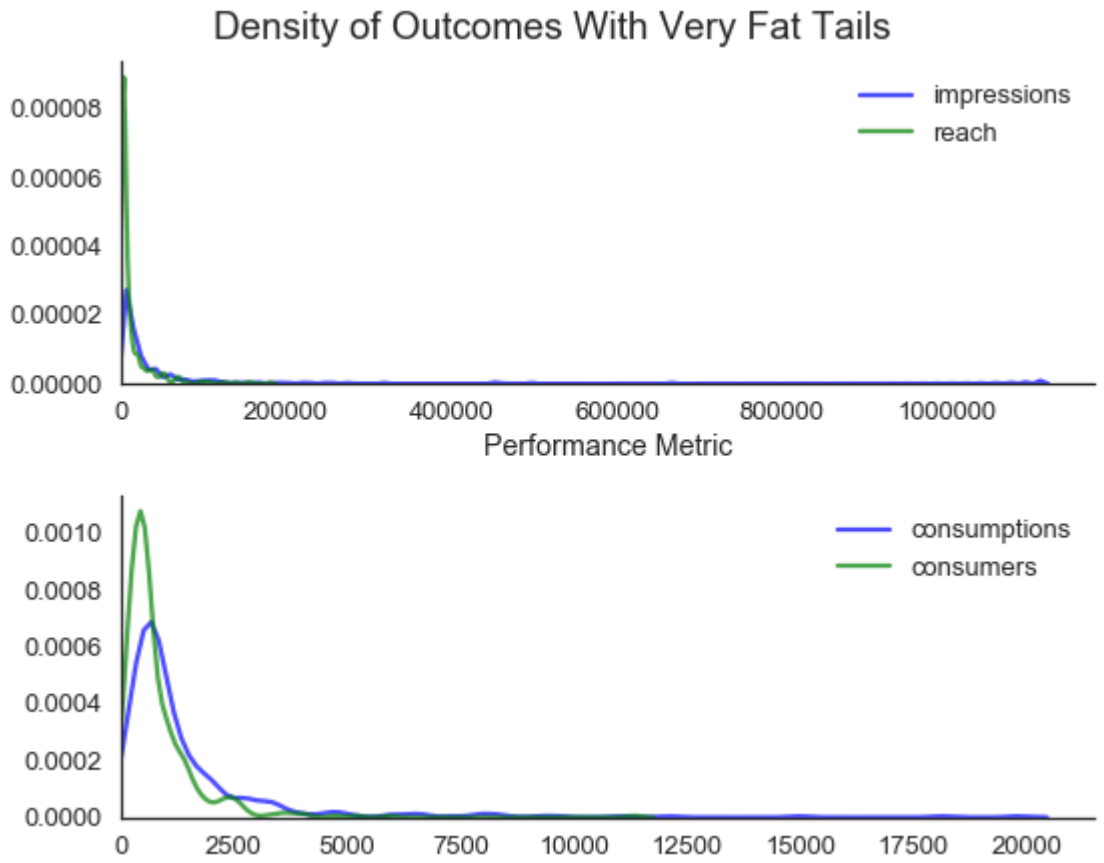
The State Space is several factors larger than the Data Volume

- Easy to overfit
- And find random correlations

Missing key information:

- What was in the post
- Information about the company's network
- Information about external campaigns

Very skewed data with fat tails



Data not used

We only have one year's data so "month" is unreliable

"Page total likes" is a proxy for time and contains some information about the company's network but how and why past page total likes will consistently impact future outcomes is not clear

Reduce the problem

The previous paper[1] dealt with the problem of fat tails by excluding the top 5% of extreme events

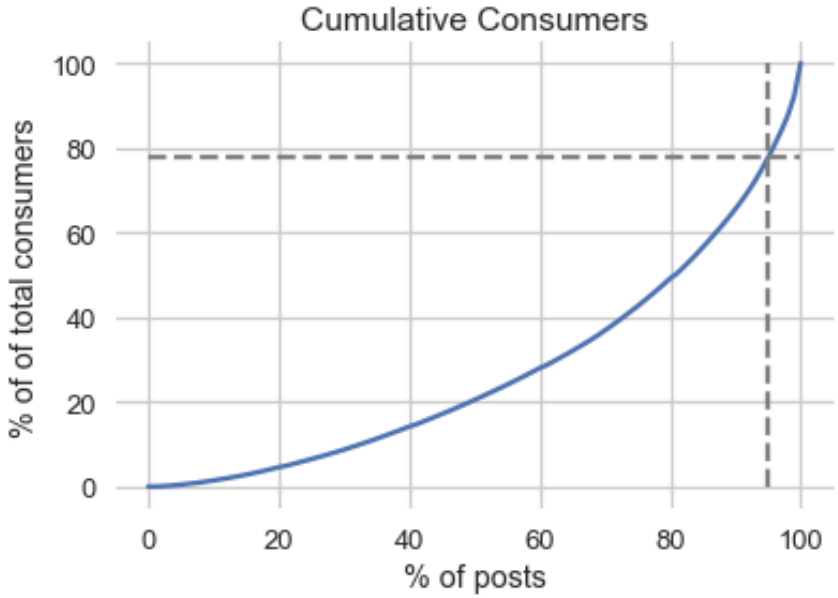
However this excludes 20% of all post "consumers"

So instead of trying to predict the absolute outcomes. We will look to predict the relative outcome of different inputs

Specifically we will try to predict which quintile a post will achieve

So if we are trying to optimise a post for "impressions" we can estimate if it will do badly (1st quintile), average (3rd) or go viral (5th)

To try and avoid overfitting and random correlations we will use a Naïve Bayes classifier. Which allows independent treatment of the features and transparent interrogation of the predictions



[1] Moro et al., 2016, Predicting social media performance metrics and evaluation

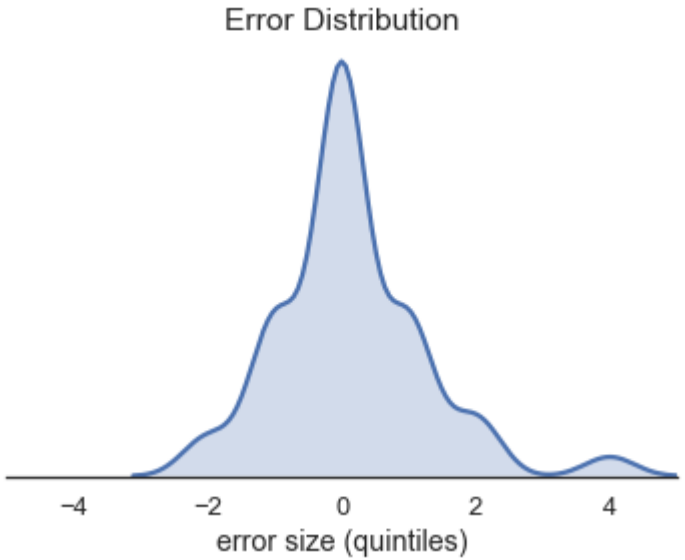
Performance of the algorithm

As said before, predicting how well a post will do, without an assessment of the content is tricky

To evaluate the performance we will look at the standard deviation of the errors from an out of sample data set

We perform multiple cross validations and average the variance.
Then take the square root to get an ‘average’ standard deviation of errors

The plot shows an example error distribution from one run



If the algorithm was no better than random we would expect an stdev score of 2

When we run our algorithm across all the data we get a stdev score of 1.5

And if we drill down a bit, for example only looking at “impressions” for “status” posts we improve on that with a stdev score of 1.1

To put it another way: **66% of the time the post will predicted to within 1 quintile**

An algorithm that can recommend “**What**” and “**When**” to post

The algorithm has been designed to flexibly accept different questions

The user can specify what they want to do, for example: post a link

And then ask the algorithm:

“What day is it best to post a link on; if I want to maximise likes?”

The user gets back the top three recommendations

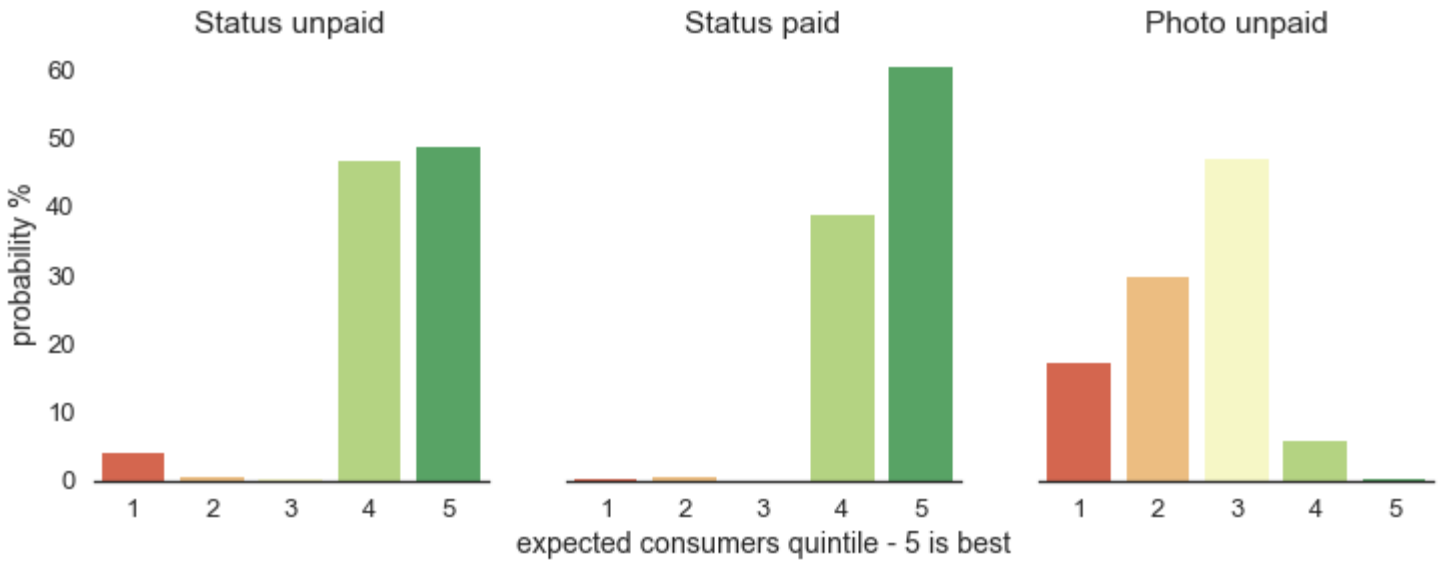
The visualisation is currently a bit clumsy, but it shows the user the top three suggestions and what the probability of the outcome being in each quintile

An algorithm that can recommend “**What**” and “**When**” to post

I want to post a **product** (direct advertisement or explicit brand content) and maximise **consumers**

“Should I post a status update, photo, link or video; and it should be paid or unpaid?”

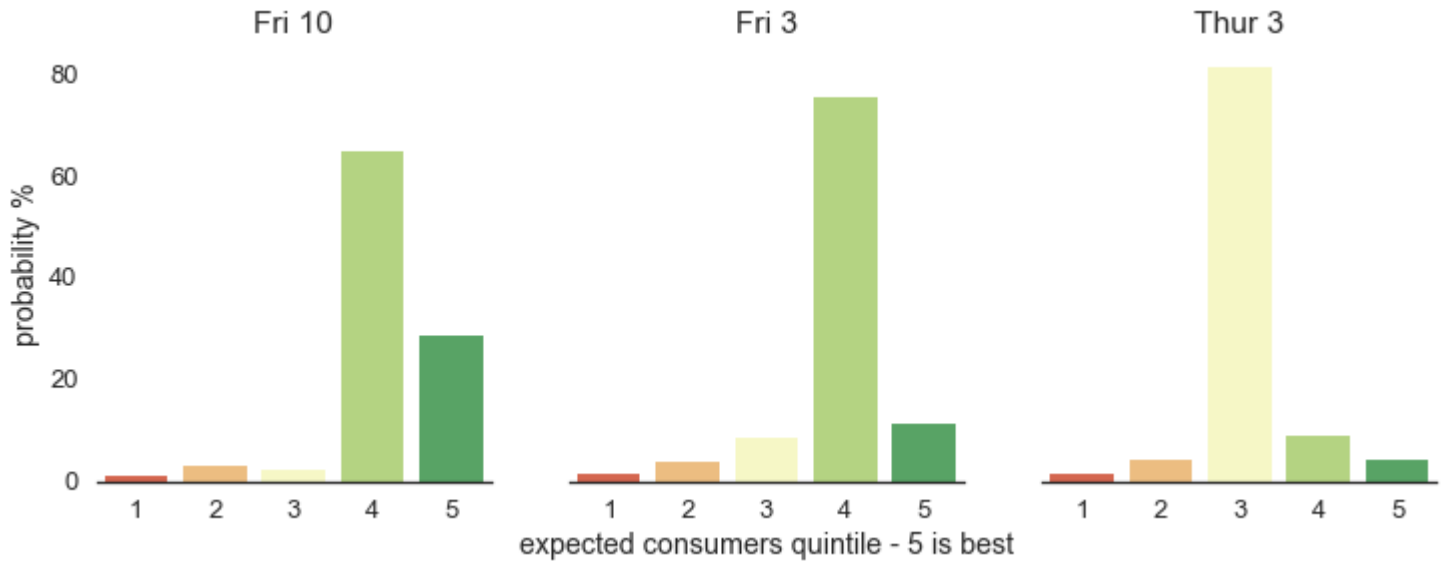
The algorithm tells me a **paid**, **status** update is my best option



An algorithm that can recommend “**What**” and “**When**” to post

If I decided on a **paid status** post. I can then ask “**When is the best time to post this?**”

And I can see that either **Friday** at **10** or **3** is best.



Conclusions

The model performs relatively well given the limitations of the data and the unpredictability of the outcomes

The model can be applied to any input features and output metrics

You can ask the model intuitive real world questions

The model is simple to understand and quick to run

It provides framework to benchmark future posts

As more data becomes available the predictive ability of the model will increase

The model is easily extended if new features become available

With more data it would be possible to model complex interactions by calculating conditional probabilities on feature combinations

Any questions, please contact leovalberg@gmail.com