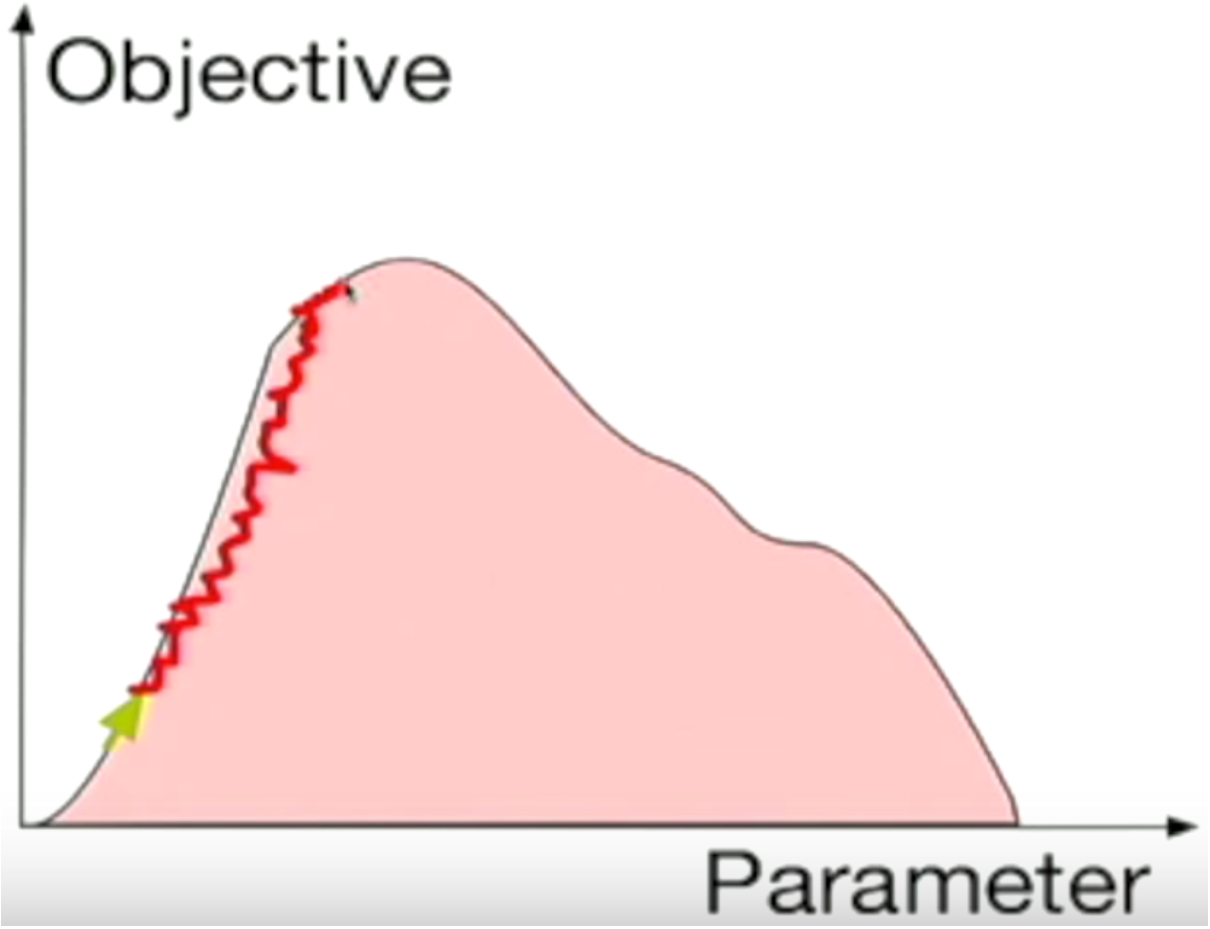
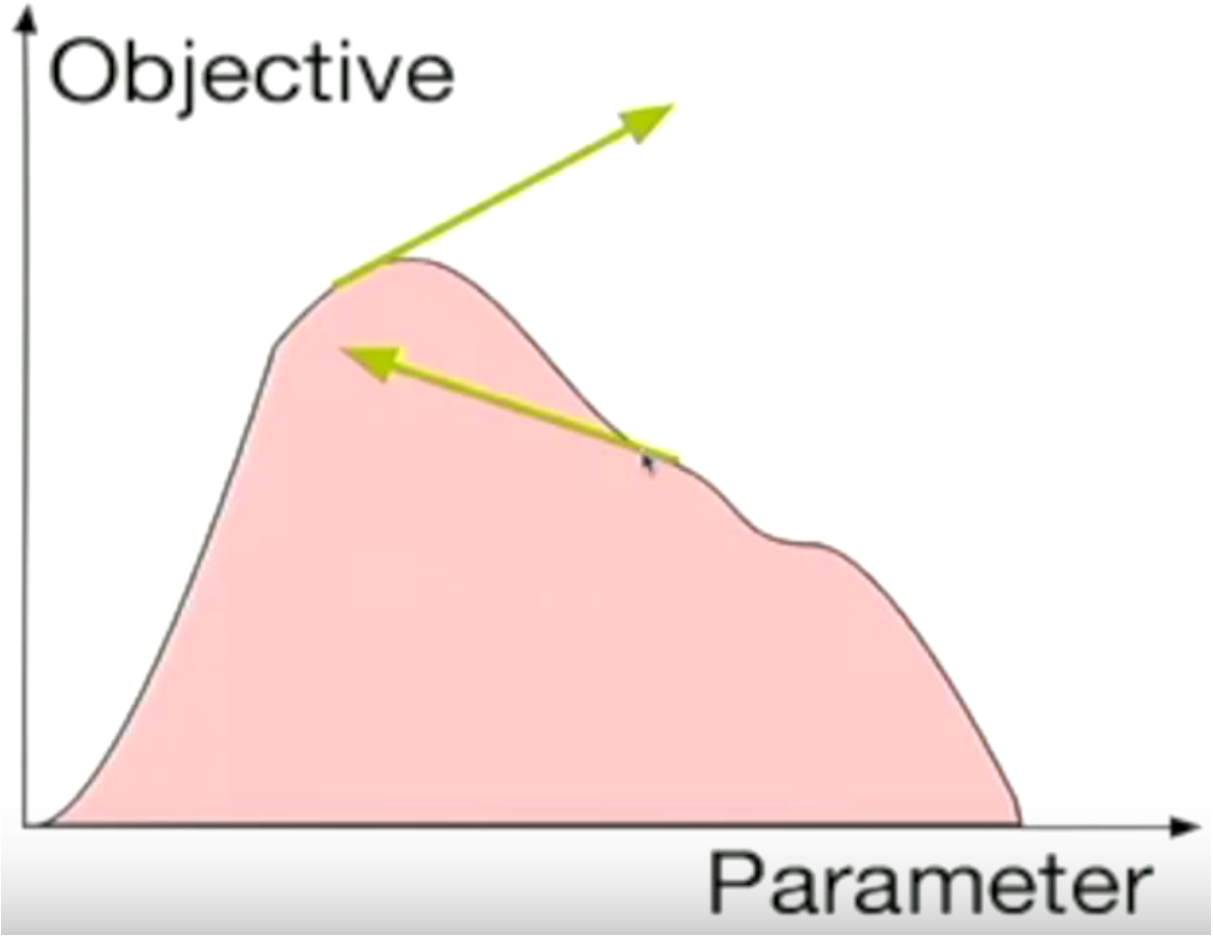
Nicholas Clement

1. Learning Rate

Learning rate is vital to the efficiency and accuracy of our logistic regression. If too small a learning rate is chosen then code will take thousands of iterations to execute.



If our step size is too large we will end up oscillating and never finding a good solution.

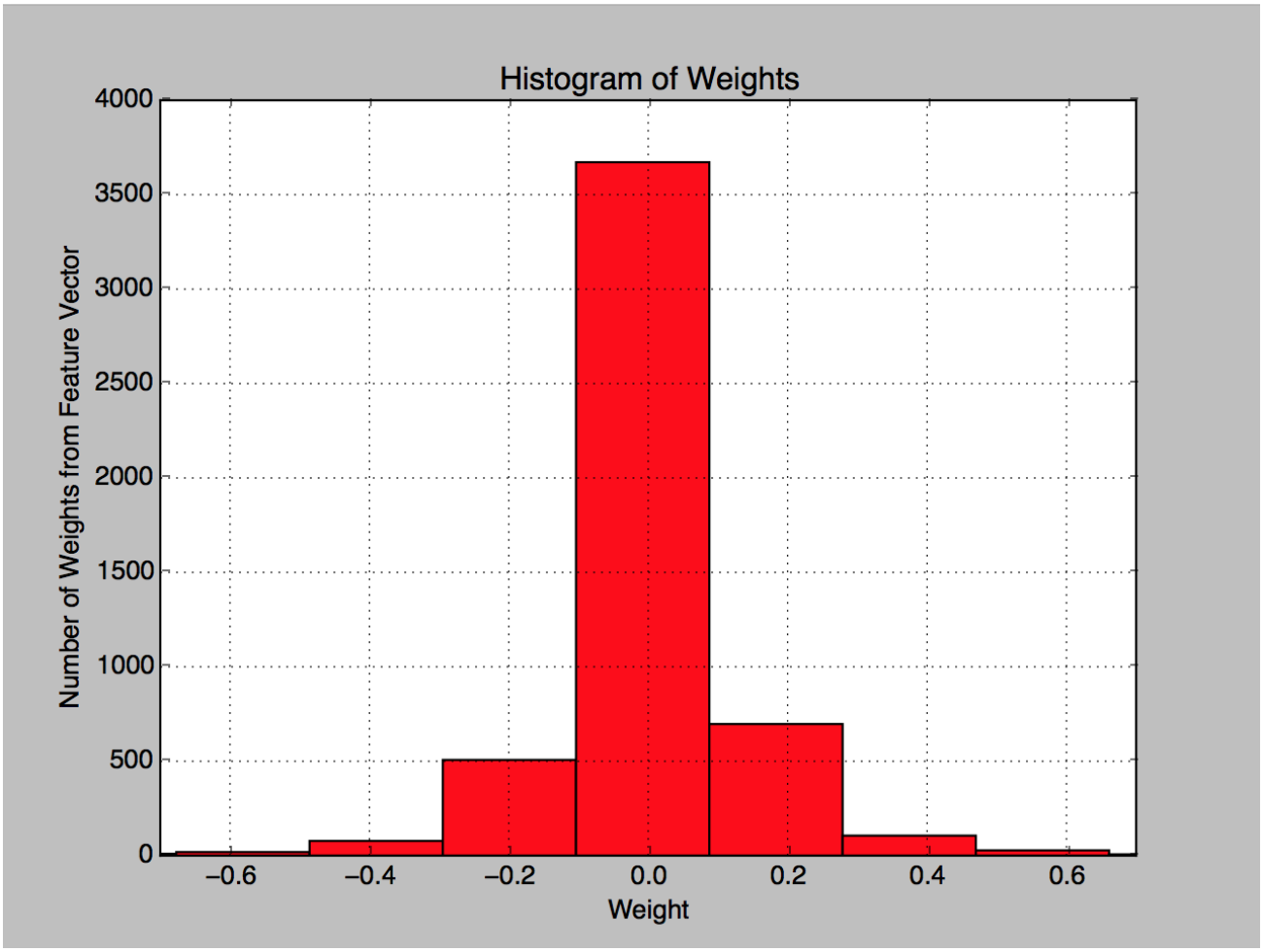


2. Number of Iterations

```
Update 1051     TP -125.489756  HP -25.138841   TA 0.974624     HA 0.939850
Update 1056     TP -126.303435  HP -24.990931   TA 0.971805     HA 0.939850
Update 1061     TP -123.245352  HP -24.924610   TA 0.974624     HA 0.947368
Time to execute = 9.475261547995615
Nicks-MacBook-Pro:logreg nicholas$
```

It can be seen that it took 1061 steps for us to reach a HA (Heldout accuracy) of about 95% and a TA (Train Accuracy) of about 97.5%. This counts as a single pass of our data, which is all we need for it to converge.

3. Good Predictors



This is a histogram of our final beta (weight) vector. It can be seen that the majority of weights fell between the -0.2 and +0.2 barriers. For this reason we can assume that any weight that falls beyond -0.2 or +0.2 would be a "good" indicator of the document. The absolute best indicators are going to lie near the -0.6 or +0.6 marks, because their weights are very strong and suggestive of the document they correlate to.

It is good to keep in mind that a positive probability is tied to baseball, while a negative probability correlates to hockey.

Best Predictors:

The best predictors for this dataset were "runs" and "hockey". In order to find these I used `numpy.argmin()` and `numpy.argmax()`. These functions returned the max/min indexes of the beta vector. With these indexes I used the vocabulary to find the best words to determine if the document was correlated to baseball or hockey.

```
Update 1056     TP -104.410624  HP -23.867281   TA
Update 1061     TP -99.038306   HP -23.416420   TA
77 33
Largest positive: runs Largest negative: hockey
Time to execute = 13.587157062999722
```