10.4 Document Classification

Document Classification

- In this section, we think about predicting (tex) document attributes.
  - For example: sentiment analysis
- We use IMDb reviews with the binary response `positive` or `negative`.
  - Each review has a different length and may include slang, typos, non-words, etc.
  - We need to find a way to create *features* out of the reviews.

Document Classification

Simplest, common approach is the *bag-of-words* model:

- Each document is scored for the presence of absence of each of the words in some language dictionary.
- We create a binary feature vector and score a 1 for each word present and 0 otherwise.
  - If the dictionary contains M words, we will have M features.
  - This results in *sparse* feature matrices.
- If we used the entire English language, M would be enormous, so we limit to the K most commonly occurring words in the training corpus.
  - Often, we do this after removing common (but unhelpful in this context) words like "the", "and", "a", etc.

IMDb reviews

Here's the beginning of a positive review that has been limited to the $10,000$ most commonly occurring words in the $25,000$ review training corpus:

> *this film was just brilliant casting location scenery story direction everyone's really suited the part they played and you could just imagine being there robert ⟨UNK⟩ is an amazing actor and now the same being director ⟨UNK⟩ father came from the same scottish island as myself so i loved . . .*

- Many words have been omitted.
- Unknown words have been marked as such.

IMDb reviews

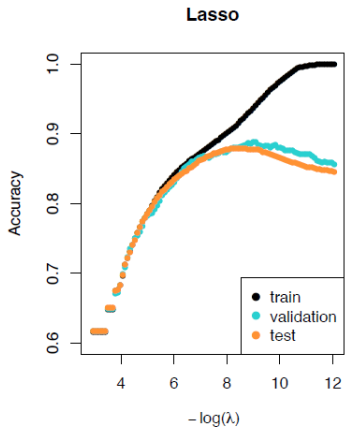
- Training and test set, each with $25,000$ reviews.
  - Each set balanced with regard to sentiment.
- Each feature matrix has dimension $25,000 \times 10,000$
  - Only 1.3/ of the binary entries are nonzero!
  - We store these in *sparse matrix format*, where only the location (and possibly values) for the nonzero entries are stored.
- Here we score only for presence or absence of a word, but we could also score for frequency.

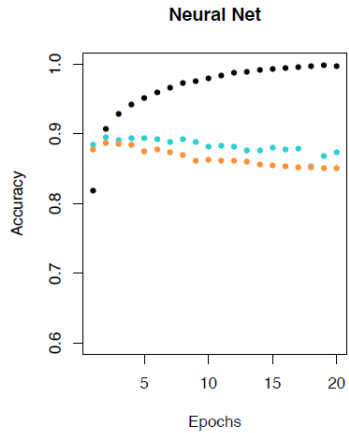
IMDb reviews

We fit two models:

1. A lasso logistic regression using `glmnet`
2. A two-class neural network with two hidden layers, each with 16 ReLU units.

IMDb reviews





IMBd reviews

- Each method produces a sequence of solutions.
- The lasso sequence is indexed by the regularization parameter λ
- The neural net sequence is indexed by the number of gradient descent iterations used in the fitting.
- We can use cross validation or a validation set to select a good solution from each sequence.
- Both models achieve similar accuracy.

Nonlinear Logistic Regression?

- A two-class neural network amounts to a nonlinear logistic regression model

$$\log\left(\frac{P(Y=1|X)}{P(Y=0|X)}\right) = Z_1 - Z_0$$
$$= (\beta_{10} - \beta_{00}) + \sum_{l=1}^{K_2}(\beta_{1l} - \beta_{0l})A_l^{(2)}$$

Bag-of-Words

The bag-of-words approach ignores the context of the words.

Two popular ways to include context:

1. The *bag-of-n-grams* model. Records the consecutive co-occurrence of every distinct group of n words.
   - Ex: distinct pairs “blissfully long” ad “blissfully short” have different sentiments.
2. Treat the document as a sequence, taking account of all the words in the context of those that preceded and those that follow.