For this sentiment analysis task, the pipeline is as follows:

1) **Data Preprocessing and Baseline Model Selection:**
   - **Fixed steps:** transform into tf-idf matrix, split into 7:3 training set and test set
   - **Tuned steps:** lowercase, 1-3 grams, top n tf-idf words, top n chi-square words, SVD
   - **Candidates:** Multinomial Naïve Bayes, Logistic Regression, SVM, Trees, Simple NN

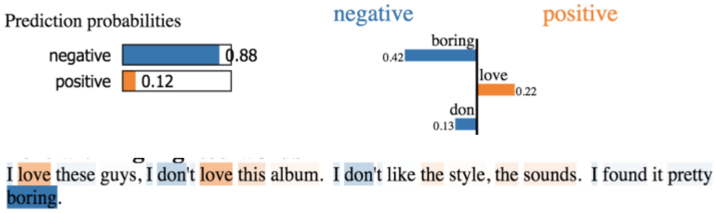
   Key Findings:

   a. **Limited Preprocessing is enough**: It turns out that lowercase and 2-grams always improved the model performance, but stemming (snowball) didn't significantly change the performance, and removing stopwords even gave me worse performance. The preprocessing should be tailored to the task and the dataset. Since stemming might be helpful select features using LIME, so I still stemmed the reviews.

   b. **Naïve Bayes is good for Data Preprocessing:** Multinomial Naïve Bayes trains faster compared with other models, so it is very suitable for comparing the performance of different data preprocessing methods and works as a baseline model.

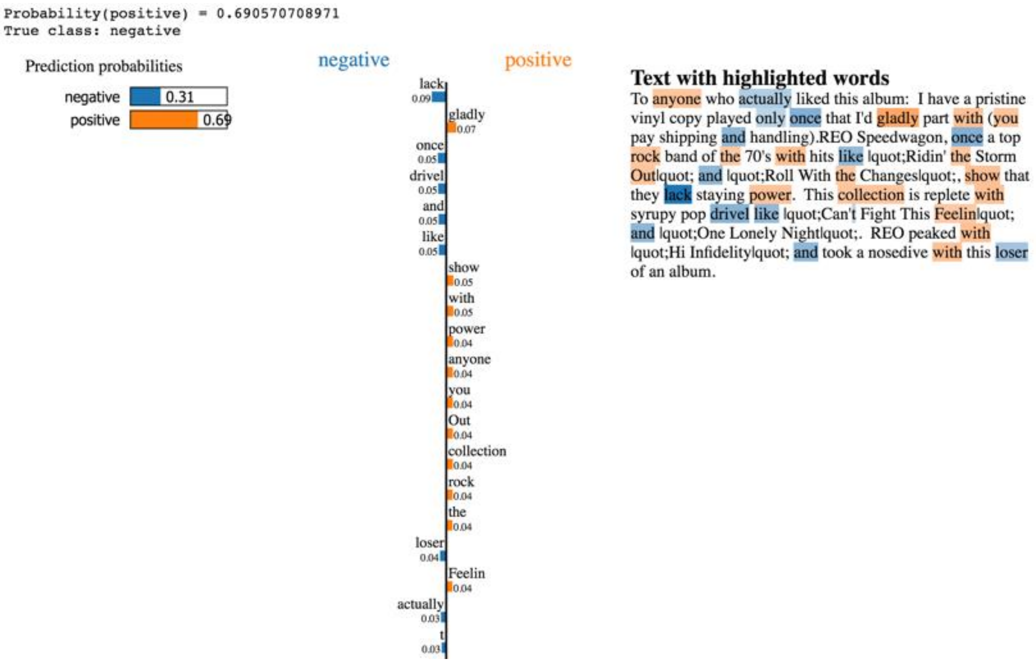
   c. **More Features works better than the more advanced model:** When I limited the features with top 300 highest chi-square words or use truncated SVD to reduce the dimension to 300, Gradient Boosting with 500 trees gave me the best performance but stuck at around 0.85 F-Score. However, when I increased the feature size to 100,000+, although I could not afford to train a large Gradient Boosting Trees, the simple Naïve Bayes outperformed greatly with 0.91+ F-Score, and linear SVM was 0.92+. As expected, logistic regression performed in between, but it models P(Y|X), so I used LR to do feature selection using LIME.

2) **Feature Selection using LIME:**
   - Firstly, I checked several reviews to get a general sense of the result. I was impressed by some good features, which have high scores to decide the sentiment of the reviews, like:



I love these guys, I don't love this album. I don't like the style, the sounds. I found it pretty boring.

   - Then I focused on the misclassified reviews to find some potential bad features

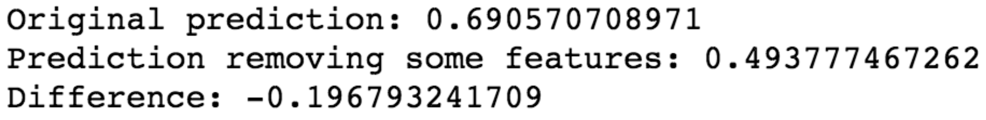


```
Probability(positive) = 0.690570708971
True class: negative
```

Text with highlighted words

To anyone who actually liked this album: I have a pristine vinyl copy played only once that I'd gladly part with (you pay shipping and handling).REO Speedwagon, once a top rock band of the 70's with hits like |quot;Ridin' the Storm Out|quot; and |quot;Roll With the Changes|quot;, show that they lack staying power. This collection is replete with syrupy pop drivel like |quot;Can't Fight This Feelin|quot; and |quot;One Lonely Night|quot;. REO peaked with |quot;Hi Infidelity|quot; and took a nosedive with this loser of an album.

For example, this review is wrongly assigned as positive, but there are a couple of words quite misleading both in positive and negative. After I removed “and”, “70”, “with”, “you”, “is”, ”rock”, “the”, “feelin”, “roll”, the result would change to the right class:



```
Original prediction: 0.690570708971
Prediction removing some features: 0.493777467262
Difference: -0.196793241709
```

3) Retrain the Model after Removing the Bad Features:

I repeated the (2) process for several misclassified reviews and removed bad features using “stopwords list” when reconstructing the tf-idf matrix. However, although the single misclassified review’s score improved, in total the accuracy didn’t improve. Maybe a single iteration and checking couple reviews are not enough for the real task, and more feature selections are required to improve the result.

At last, the model with the best performance is the soft ensemble Logistic Regression and Multinomial Naïve Bayes, which gave me 0.89916 on the test dataset, and 0.90249 as the final score. The ensemble method balances bias and variance, so it seems not over-fitting and has good prediction accuracy.