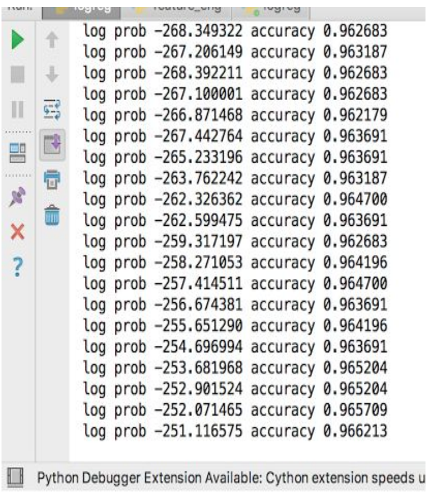
1.2)

1) The below mentioned diagrams are for epoch 1 and learning rates as mentioned beside. The log probabilities are printed after training every 100 instances(progress is called). At smaller learning rate , It takes lot of iterations to converge due to small steps.(ex 0.001). At higher learning rate , the log probabilities oscillates continuously and we will miss the minimum and it becomes difficult to converge.(ex0.1). Higher or smaller learning rate are not efficient for converging. We need to have high learning rate initially. Gradually our learning rate need to adapt and decrease accordingly in order to avoid missing the minimum.

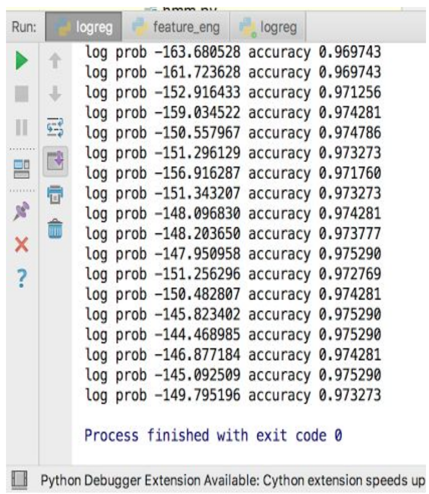
0.001 - eta



```
log prob -268.349322 accuracy 0.962683
log prob -267.206149 accuracy 0.963187
log prob -268.392211 accuracy 0.962683
log prob -267.100001 accuracy 0.962683
log prob -266.871468 accuracy 0.962179
log prob -267.442764 accuracy 0.963691
log prob -265.233196 accuracy 0.963691
log prob -263.762242 accuracy 0.963187
log prob -262.326362 accuracy 0.964700
log prob -262.599475 accuracy 0.963691
log prob -259.317197 accuracy 0.962683
log prob -258.271053 accuracy 0.964196
log prob -257.414511 accuracy 0.964700
log prob -256.674381 accuracy 0.963691
log prob -255.651290 accuracy 0.964196
log prob -254.696994 accuracy 0.963691
log prob -253.681968 accuracy 0.965204
log prob -252.901524 accuracy 0.965204
log prob -252.071465 accuracy 0.965709
log prob -251.116575 accuracy 0.966213
```

Python Debugger Extension Available: Cython extension speeds u

0.01 - eta

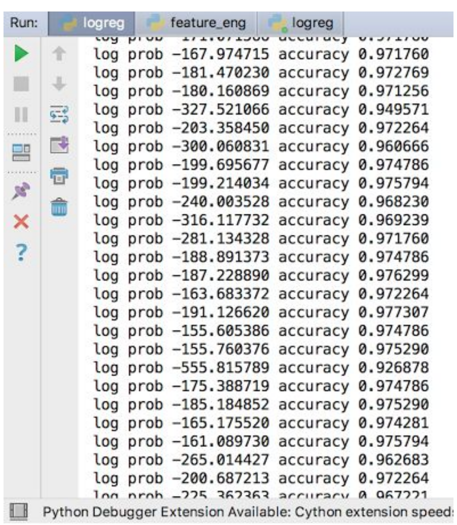


Run: logreg feature_eng logreg

```
log prob -163.680528 accuracy 0.969743
log prob -161.723628 accuracy 0.969743
log prob -152.916433 accuracy 0.971256
log prob -159.034522 accuracy 0.974281
log prob -150.557967 accuracy 0.974786
log prob -151.296129 accuracy 0.973273
log prob -156.916287 accuracy 0.971760
log prob -151.343207 accuracy 0.973273
log prob -148.096830 accuracy 0.974281
log prob -148.203650 accuracy 0.973777
log prob -147.950958 accuracy 0.975290
log prob -151.256296 accuracy 0.972769
log prob -150.482807 accuracy 0.974281
log prob -145.823402 accuracy 0.975290
log prob -144.468985 accuracy 0.975290
log prob -146.877184 accuracy 0.974281
log prob -145.092509 accuracy 0.975290
log prob -149.795196 accuracy 0.973273

Process finished with exit code 0
```

Python Debugger Extension Available: Cython extension speeds up



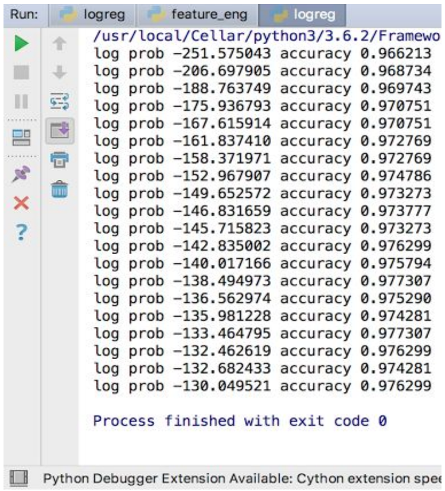
Run: logreg feature_eng logreg

```
log prob -167.974715 accuracy 0.971760
log prob -181.470230 accuracy 0.972769
log prob -180.160869 accuracy 0.971256
log prob -327.521066 accuracy 0.949571
log prob -203.358450 accuracy 0.972264
log prob -300.060831 accuracy 0.960666
log prob -199.695677 accuracy 0.974786
log prob -199.214034 accuracy 0.975794
log prob -240.003528 accuracy 0.968230
log prob -316.117732 accuracy 0.969239
log prob -281.134328 accuracy 0.971760
log prob -188.891373 accuracy 0.974786
log prob -187.228890 accuracy 0.976299
log prob -163.683372 accuracy 0.972264
log prob -191.126620 accuracy 0.977307
log prob -155.605386 accuracy 0.974786
log prob -155.760376 accuracy 0.975290
log prob -555.815789 accuracy 0.926878
log prob -175.388719 accuracy 0.974786
log prob -185.184852 accuracy 0.975290
log prob -165.175520 accuracy 0.974281
log prob -161.089730 accuracy 0.975794
log prob -265.014427 accuracy 0.962683
log prob -200.687213 accuracy 0.972264
```

Python Debugger Extension Available: Cython extension speed

- 0.1 eta

2) The below diagram shows log probabilities for epoch 1 to 20(eta - 0.001). The test accuracy increases initially with epoch and later oscillate with epoch (It neither increases nor decreases continuously.)



```
Run:  logreg  feature_eng  logreg
/usr/local/Cellar/python3/3.6.2/Framewo
log prob -251.575043 accuracy 0.966213
log prob -206.697905 accuracy 0.968734
log prob -188.763749 accuracy 0.969743
log prob -175.936793 accuracy 0.970751
log prob -167.615914 accuracy 0.970751
log prob -161.837410 accuracy 0.972769
log prob -158.371971 accuracy 0.972769
log prob -152.967907 accuracy 0.974786
log prob -149.652572 accuracy 0.973273
log prob -146.831659 accuracy 0.973777
log prob -145.715823 accuracy 0.973273
log prob -142.835002 accuracy 0.976299
log prob -140.017166 accuracy 0.975794
log prob -138.494973 accuracy 0.977307
log prob -136.562974 accuracy 0.975290
log prob -135.981228 accuracy 0.974281
log prob -133.464795 accuracy 0.977307
log prob -132.462619 accuracy 0.976299
log prob -132.682433 accuracy 0.974281
log prob -130.049521 accuracy 0.976299

Process finished with exit code 0

Python Debugger Extension Available: Cython extension spee
```

2.2)

1) a) I added features count of invited quotes, count of exclamations in the review. This implies he is stressing many points in the review.(indicates it may be positive review). These features increased accuracy above baseline .

b) I added feature count of ((what|where|when|where|how|whose) .*\ ?) regular expression in the review. So this type of question implies he may not like the movie. This feature increased accuracy above baseline.

c)I added feature count of years mentioned in review. I assumed the mentioned of years may imply that he liked the movie and gave citations of other movie which he related. But this feature decreased accuracy because even in case of not liking the movie he gave citations.

d)I added feature count of conjunctions in the review. This implies he may be giving reasons for not liking the movie. This feature increased accuracy above baseline because of above assumption.

e) I added feature number of () brackets in the review. This may imply that he is elaborating something more in detail to explain the things he liked in the movie. This feature didn't increase accuracy because above assumption didn't hold.(he explained which he didn't like in the movie also) .

f) I added features number of positive words and negative words in the review using a set of words from net. This feature didn't increase accuracy because of the limitedness of set of words.

2) unigrams - set of sequnce of one word from the given text.(review)
Bigrams - set of sequence of two words from given text.(review)
N-gram - set of sequence of n words from given text.(review).

These features increased performance to above 80%. These features help because when same unigram , bigrams or found are found in training sample it assumes same class label. Higher the match of count of ngrams between test review and training review higher the probability it has same class label. So it does based on count of matched ngrams.

3) $$P(y=c/x) = \frac{\exp(\beta_c^T x)}{\sum_{c'=1}^{C} \exp(\beta_{c'}^T x)}$$

Negative log likelihood $= -\sum_{i=1}^{N} \log\left(P(y^{(i)}/x^{i})\right)$

$$= -\sum_{i=1}^{N} \log\left(\frac{\exp(\beta_{c_i}^T x_i)}{\sum_{c'=1}^{C} \exp(\beta_{c'}^T x_i)}\right)$$

c_i is correct label of x_i training sample

$$= -\sum_{i=1}^{N}\left(\left(\log\left(\exp\left(\beta_{c_i}^T x_i\right)\right) - \log\left(\sum_{c'=1}^{C} \exp(\beta_{c'}^T x_i)\right)\right)\right)$$

Negative log likelihood $(L) = -\sum_{i=1}^{N}\left(\beta_{c_i}^T x_i - \log\left(\sum_{c'=1}^{C} \exp(\beta_{c'}^T x_i)\right)\right)$

Assume $$\pi_{c_i i} = \frac{\exp(\beta_{c_i}^T x_i)}{\sum_{c'=1}^{C} \exp(\beta_{c'}^T x_i)}$$

$$\frac{\partial L_i}{\partial \beta_{c_i, j}} = \frac{\partial\left(-\log\left(\frac{\exp(\beta_{c_i}^T x_i)}{\sum_{c'=1}^{C} \exp(\beta_{c'}^T x_i)}\right)\right)}{\partial \beta_{c_i, j}}$$

$$= \frac{\partial(\pi_{c_i, i})}{\partial \beta_{c_i, j}}$$

$$\frac{\partial(-\log \pi_{c_i i})}{\partial \beta_{c_i j}} = -\frac{1}{\pi_{c_i i}} \frac{\partial\left(\pi_{c_i i}\right)}{\partial \beta_{c_i j}}$$

$$= \frac{\partial\left(\pi_{c_i, i}\right)}{\partial \beta_{c_i j}} = \frac{\partial\left(\frac{\exp(\beta_{c_i}^T x_i)}{\sum_{c'=1} \exp(\beta_{c'}^T x_i)}\right)}{\partial \beta_{c_i, j}}$$

$(x_i)_j$ is j^{th} feature of x_i sample

$$= \frac{(x_i)_j \exp(\beta_{c_i}^T x_i) \times \left(\sum_{c'=1} \exp(\beta_{c'}^T x_i)\right) - (x_i)_j \{\exp(\beta_{c_i}^T x_i)\} \times \exp(\beta_{c_i}^T x_i))}{\left(\sum_{c'=1}^{C} \exp(\beta_{c'}^T x_i)\right)^2}$$

$$= \frac{(x_i)_j \exp(\beta_{c_i}^T x_i) \left(\sum_{c'=1}^{C} \exp(\beta_{c'}^T x_i) - \exp(\beta_{c_i}^T x_i)\right)}{\left(\sum_{c'=1}^{C} \exp(\beta_{c'}^T x_i)\right)^2}$$

$$= (x_i)_j \, (\pi_{c_i i})(1 - \pi_{c_i i})$$

$$\frac{\partial(\log \pi_{c_i i})}{\partial \beta_{c_i j}} = -\frac{1}{\cancel{\pi_{c_i i}}} (x_i)_j \times \cancel{\pi_{c_i i}} \times (1 - \pi_{c_i i}) = (x_i)_j (1 - \pi_{c_i i})$$

Scanned by CamScanner

$$\frac{\partial L}{\partial \beta_{c_i j}} = -(x_i)_j \left(1 - \pi_{c_i i}\right)$$

$\pi \rightarrow$ is for class c_i & i sample

c_i is class of i instance