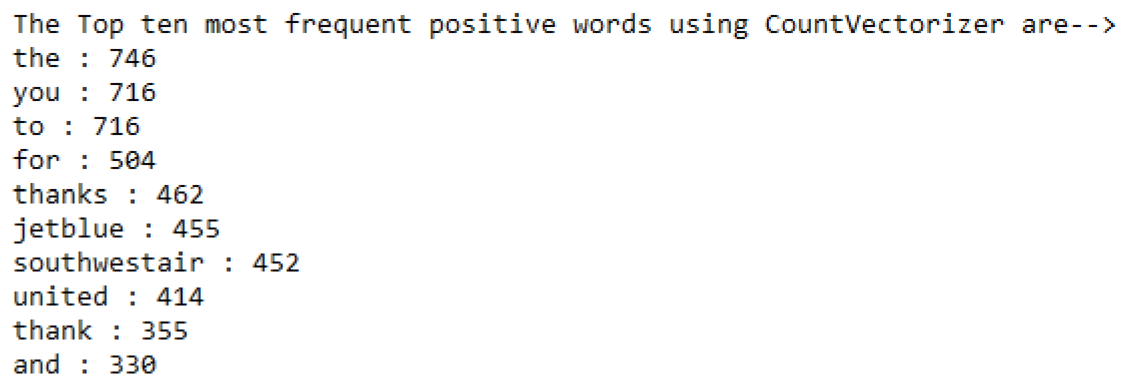
IA3 Report – Sentiment analysis with support vector machines

G10 - Opeyemi Ajibuwa

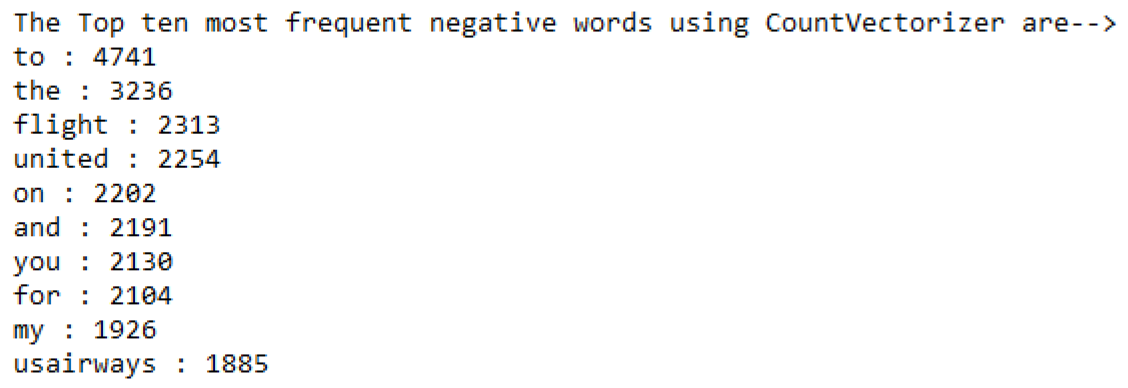
Part 0: Preprocessing

(a) **Report the ten most frequent words (frequency is measure by the total counts of each word are maximal) for the positive tweet and negative tweet respectively. Do you find these words to be informative of the classes?**
Except for the stopwords in this list, it is somewhat representative of words that expresses positive sentiment.



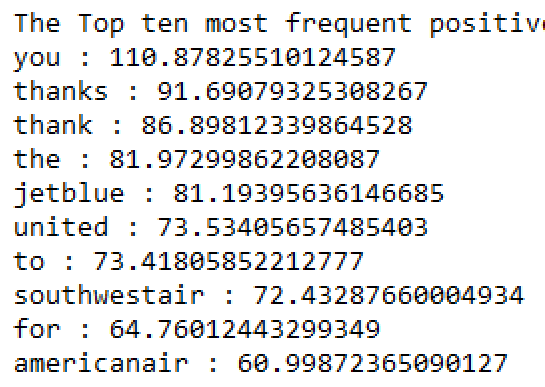
```
The Top ten most frequent positive words using CountVectorizer are-->
the : 746
you : 716
to : 716
for : 504
thanks : 462
jetblue : 455
southwestair : 452
united : 414
thank : 355
and : 330
```

Without some contextual information, this list is not quite informative of words with negative sentiments even if the stopwords are removed.



```
The Top ten most frequent negative words using CountVectorizer are-->
to : 4741
the : 3236
flight : 2313
united : 2254
on : 2202
and : 2191
you : 2130
for : 2104
my : 1926
usairways : 1885
```

(b) **Report the ten words with the highest total TF-IDF's for the positive tweet and negative tweet respectively. How do they compare to the list in 0(a)? Which one do you think are more informative and why?**
This list is more informative of words with positive sentiments. It has fewer stopwords and is quite representative of words with positive sentiments.



```
The Top ten most frequent positive words using TFIDF vectorizer-->
you : 110.87825510124587
thanks : 91.69079325308267
thank : 86.89812339864528
the : 81.97299862208087
jetblue : 81.19395636146685
united : 73.53405657485403
to : 73.41805852212777
southwestair : 72.43287660004934
for : 64.76012443299349
americanair : 60.99872365090127
```

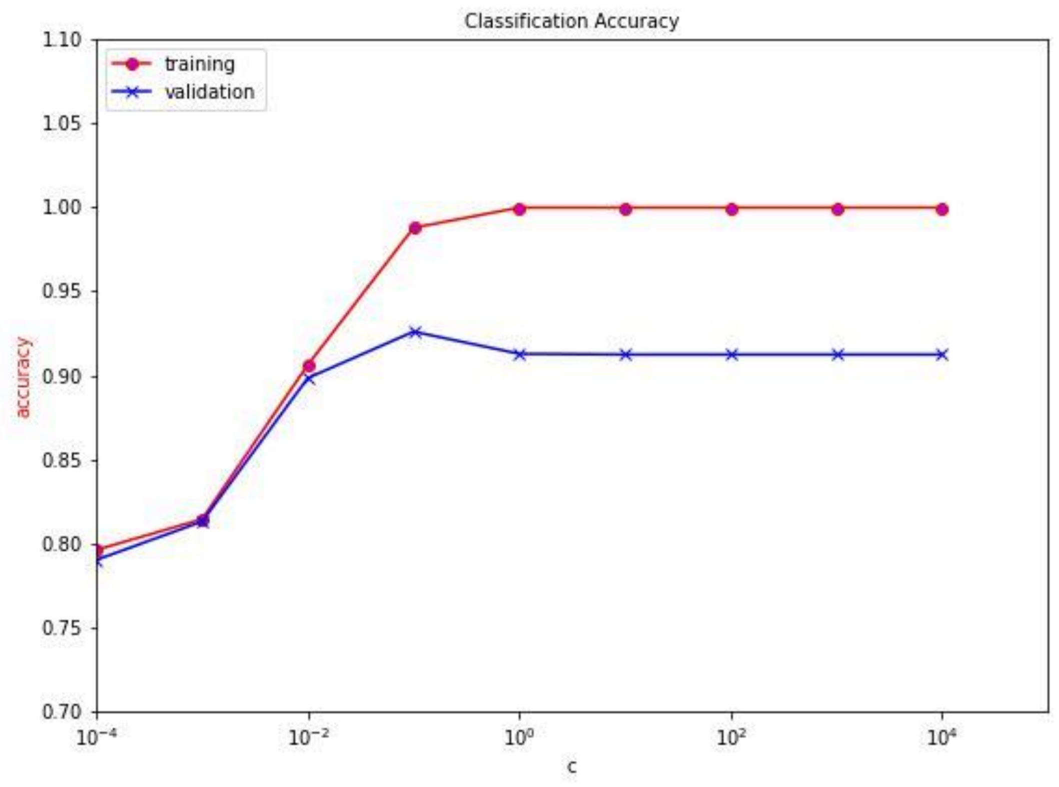
This list is still not quite representative of words with negative sentiments without having some contextual information.

```
The Top ten most frequent negative words using TFIDF vectorizer-->
to : 359.2906838073978
the : 295.37105839383537
flight : 241.58371560220323
you : 238.11512888455562
united : 237.01120827396764
on : 229.74896490472437
and : 222.17548084678558
for : 221.41670018189913
usairways : 214.98042505497082
my : 214.8651738529854
```

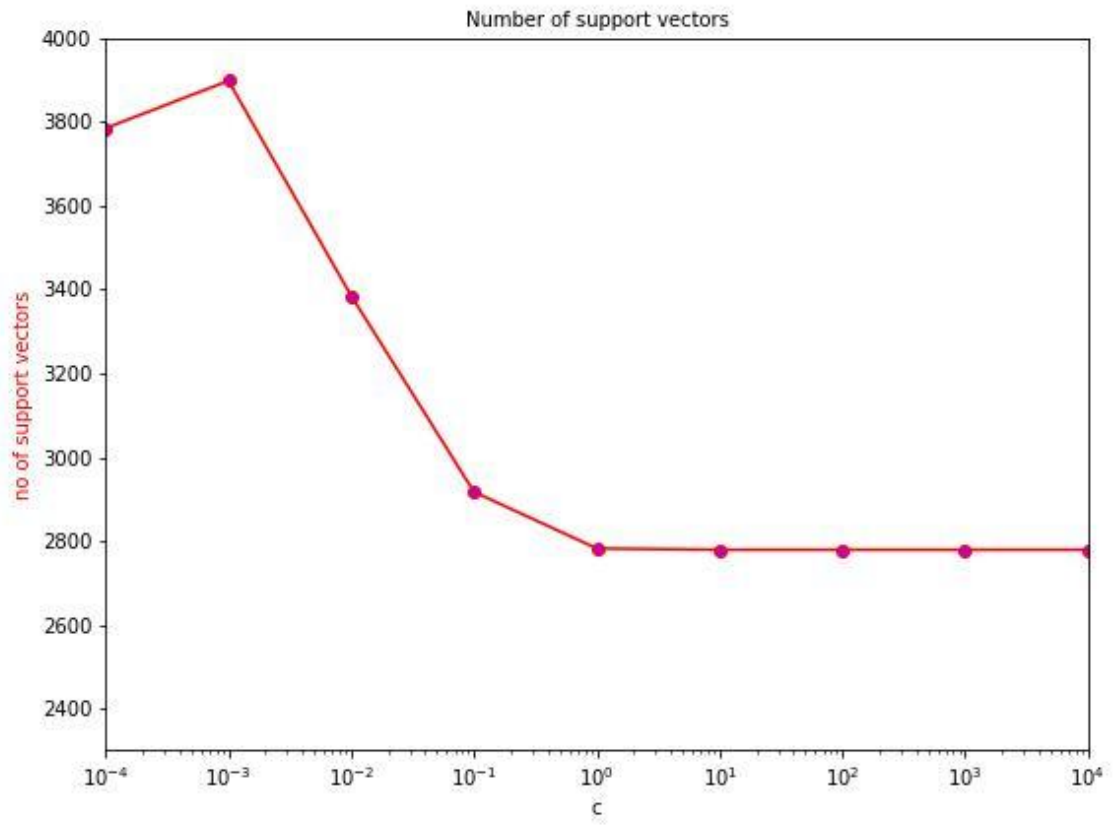
<u>Part 1:</u> Linear SVM

(a) **What is the best validation performance you are able to achieve with linear SVM? What c value is used?**
The best validation performance I achieved has classification accuracy = 92.6% for C value = 1

(b) **What trend do you theoretically expect to see for training and validation performance as we increase c? Plot the training and validation accuracy against different values of c. Does the trend you observe match your expectation?**
Theoretically, as we increase c, I expect the training performance to increase as the optimization algorithm will try to reduce |w| as much as possible leading to a hyperplane which tries to classify each training example correctly. However, the validation performance will decrease with increasing c due to an overfitting of the trained classifier (loss in generalization properties). My observation from the implementation is not very far from the theoretical expectation. The training accuracy increases with increasing c and stayed constant after a certain point, while the validation accuracy also initially increases until c =10^-1, then decreased and then stayed relatively constant as c increased.



(c) **What relationship do we theoretically expect between c and the number of support vectors? Plot the number of support vectors against the different values of c. Does this trend you see match your theoretical expectation?**
Theoretically, I expect the number of support vectors to reduce as we increase the value of the hyperparameter c, and vice versa. This trend matches my expectation as observed from the plot below.

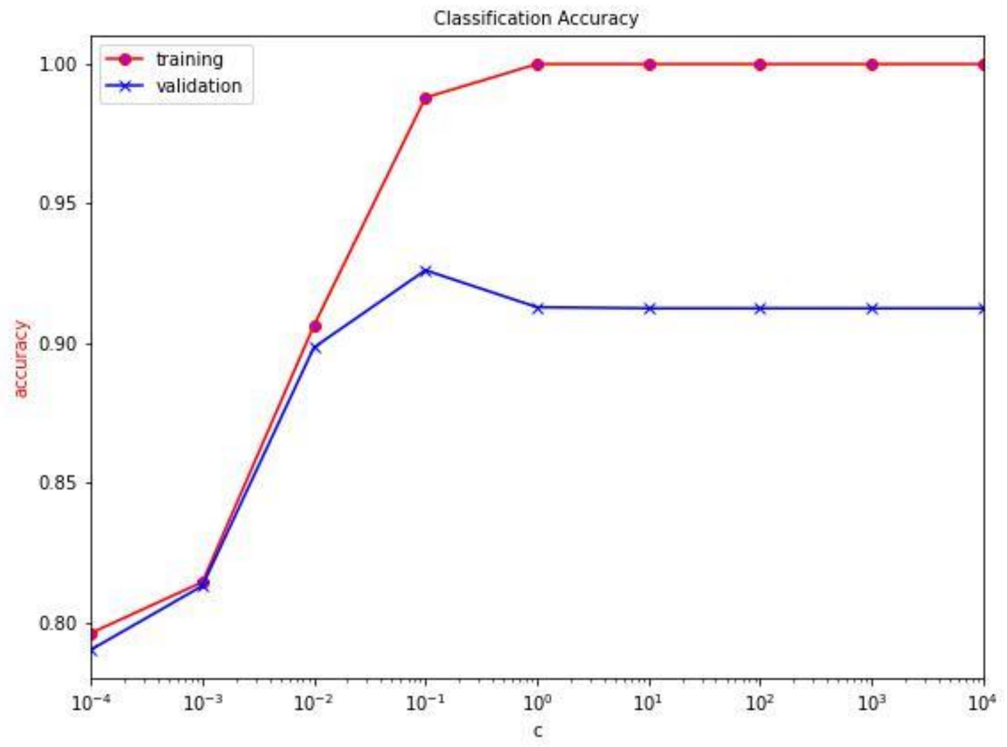


<u>Part 2:</u> Quadratic SVM

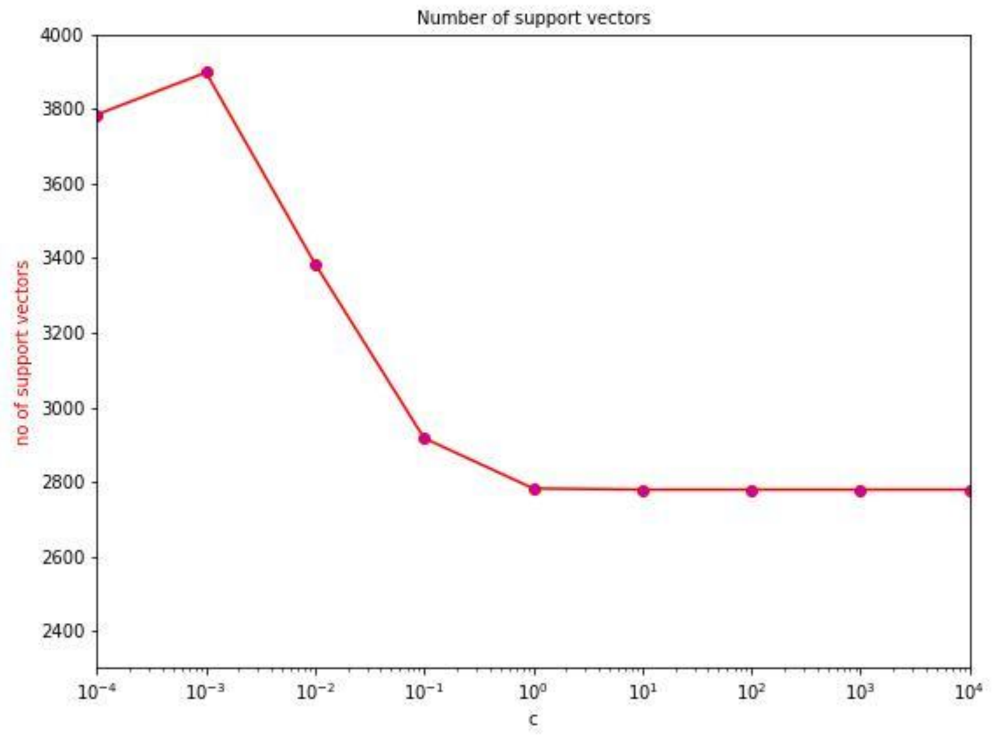
(a) **What is the best validation performance you are able to achieve with quadratic kernel? What c value is used?**
The best validation performance I achieved has classification accuracy = 92.6% for C value = 0.1

(b) **What trend do you theoretically expect to see for training and validation performance as we increase c? Plot the training and validation accuracy against different values of c. Does the trend you observe match your expectation?**
Theoretically, as we increase c, the training performance of the quadratic SVM will increase as it tries to minimize the number of misclassified examples and create a decision boundary with a smaller margin. Consequently, we expect the validation performance to decrease with increasing c values due to the overfitting properties of the trained classifier. My observation from the implementation is not very far from the theoretical expectation and very similar to the linear SVM case. The training accuracy increases with increasing c and stayed constant after a certain point,

while the validation accuracy also initially increases until c =10^-1, then decreased and then stayed relatively constant as c increased.



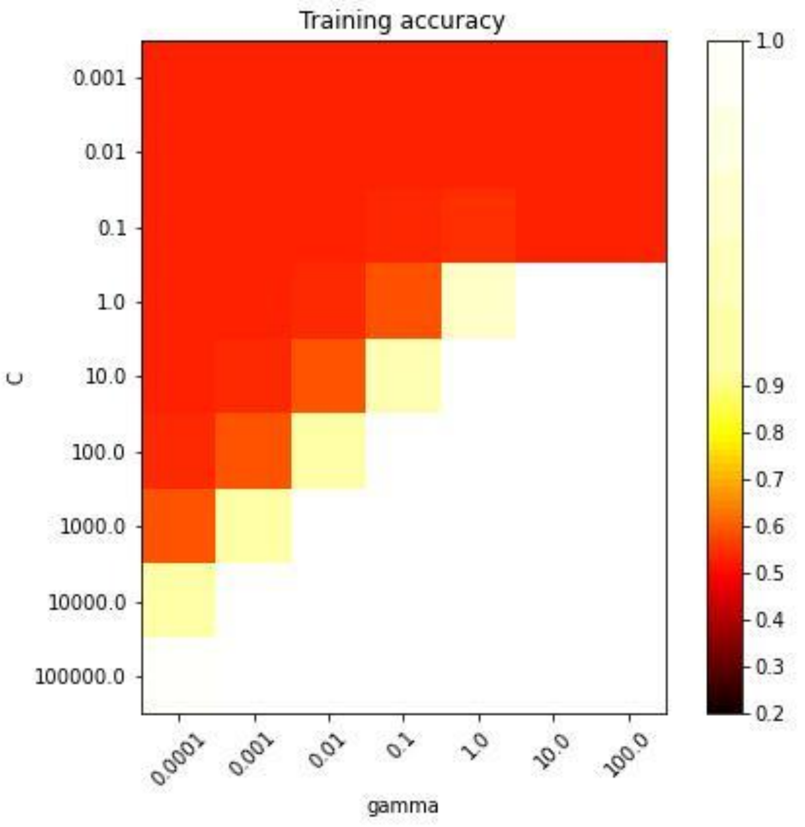
(c) **What relationship do we theoretically expect between c and the number of support vectors? Plot the number of support vectors against the different values of c. Does this trend you see match your theoretical expectation?**
Theoretically, I expect the number of support vectors to reduce as we increase the value of the hyperparameter c, and vice versa. This trend matches my expectation as observed from the plot below.

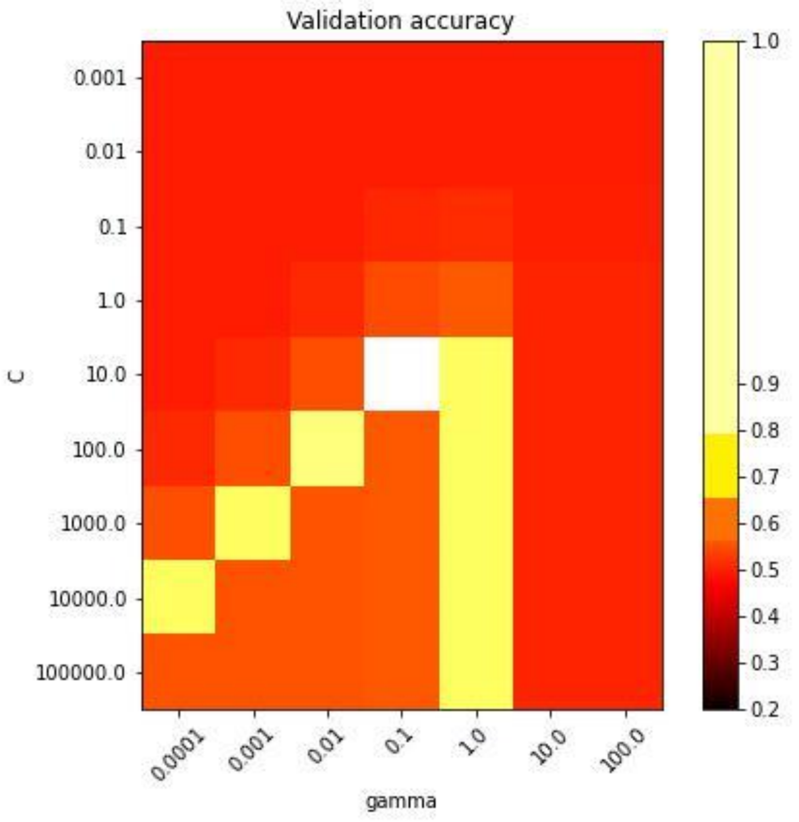


Part 3: SVM with RBF Kernel

(a) What is the best validation performance you are able to achieve for RBF kernel? What c and gamma parameters are used?

The best validation performance I achieved has classification accuracy = 92.76% for c value = 10 and γ = 0.1





(b) What trend do you theoretically expect to see for training and validation performance as we increase c with fixed gamma value? Does this trend you observe match your expectations?

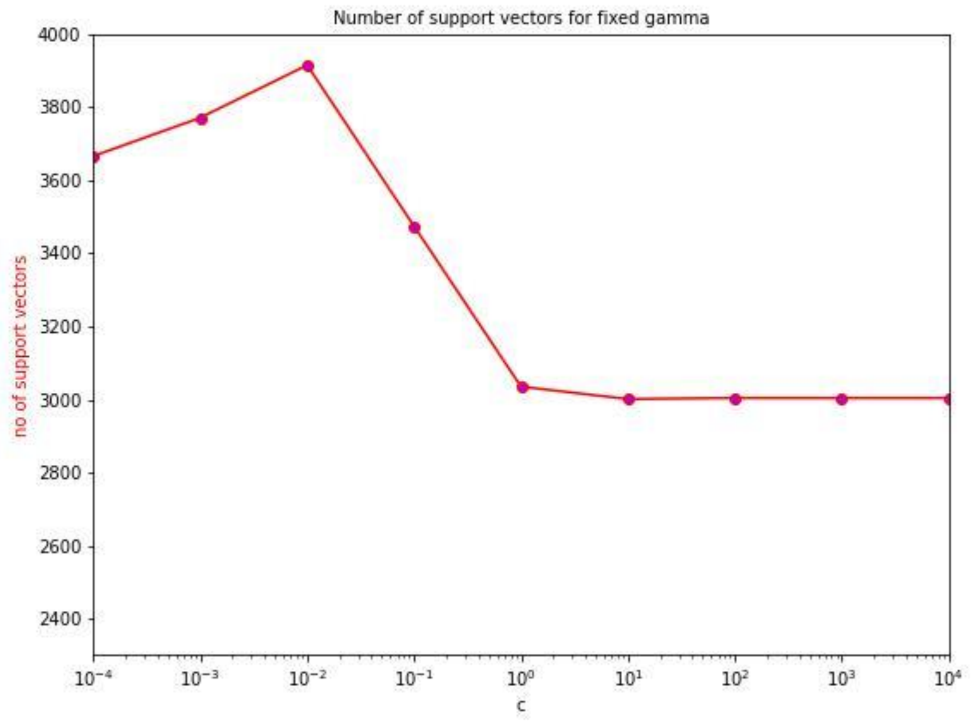
Theoretically, the training accuracy should increase with increasing c values for a small, fixed gamma value. This means that the validation accuracy will decrease with increasing c values due to the overfitted trained classifier. On the other hand, with large, fixed gamma value, the training accuracy will initially be low and then starts to increase as c increases. This is because large gamma value will over smooth and underfits the model while increasing c values will tend to gradually overfit the model thereby creating a balancing act between the two opposing forces. As a result, the validation accuracy will decrease with increasing c value under a large gamma value. This trend does indeed match my observation from the results obtained from the implementation.

(c) What trend do you theoretically expect to see for training and validation performance as we decrease gamma with fixed c? Does this trend you observe match your expectation?

Theoretically, with large, fixed c value, training accuracy will increase, and validation accuracy will decrease with decreasing gamma value. With small, fixed c value, training accuracy will decrease, and validation accuracy will stay relatively constant for decreasing gamma value. This trend follows my observations as I expected.

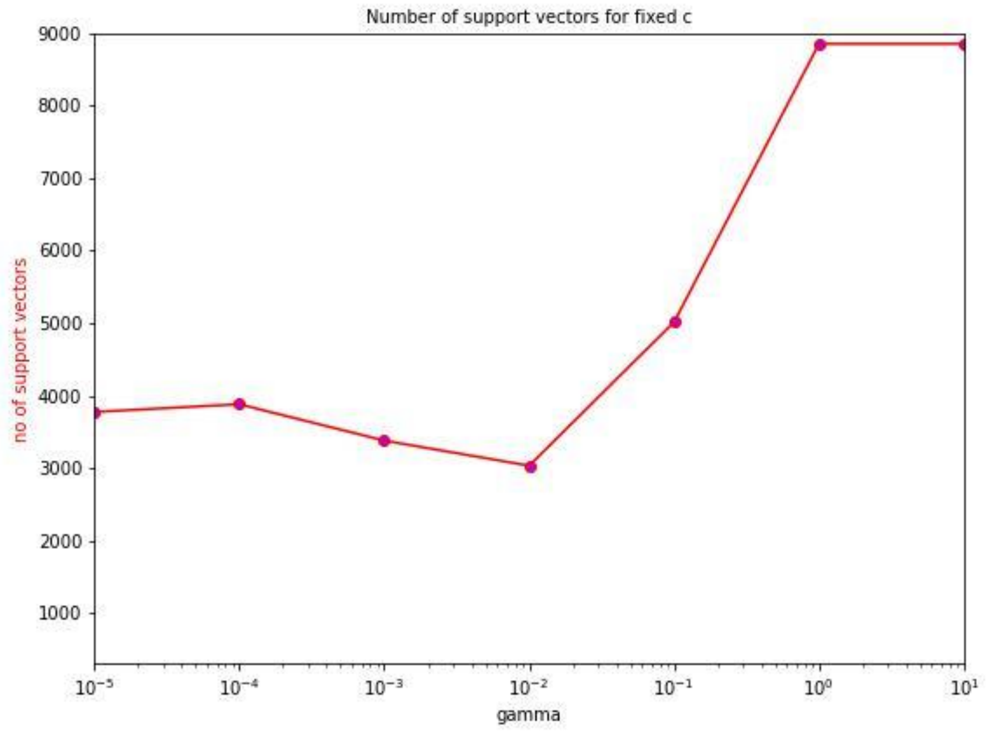
(d) With fixed gamma, what relationship do we theoretically expect between c and the number of support vectors? Plot the number of support vectors as a function of c for gamma = 0.1. Does this trend you see match your theoretical expectations?

For a small, fixed gamma value, theoretically, the number of support vectors will decrease as c increases. The trend observed from the plot below matches the theoretical expectation.



(e) With fixed c, what relationship do we theoretically expect between gamma and the number of support vectors? Plot the number of support vectors as a function of gamma for c = 10. Does this trend you see match your theoretical expectations?

For large, fixed c, theoretical expectations is that the number of support vectors will increase as gamma increases. As seen from the plot below, our observations match the theoretical expectation.



<u>Part 4:</u> Final discussion

Question - Comparing across the three different models and their performances, please discuss, for each model, the main sources for their error. Specifically, do you think for each case, the error is primarily the modeling, estimation, optimization or Bayes error? Or a combination of multiple types of errors? Please provide a clear rationale for your claims.

Errors in the linear model can be attributed to modelling, optimization, Bayes and estimation errors. The modelling error is caused by the use of the overly simplistic linear SVM algorithm. The optimization error might have resulted from not tuning some of the hyperparameters which result in an imperfectly optimized objective for the linear model. The Bayes error is due to the inherent nature of the dataset, non-removal of the stopwords and lack of feature engineering on the dataset. The estimation error might be explained by the unbalanced nature of our dataset which results in our learned model being overfitted to a certain class (in this case negative sentiment).

Error in the polynomial model is attributable to estimation, modelling, and Bayes error. The class imbalance problem of our dataset still affects the quadratic kernel SVM despite being a more complex model than the linear SVM, and this account for the errors due to estimation. Modelling error also affects the polynomial kernel SVM as using an overly complex model (degree 3 polynomial) had comparably lower performance on the test dataset (i.e., it overfitted) than a degree 2 polynomial. Bayes error also affects the polynomial model due to the inherent flawed nature of the dataset (i.e the stopwords which provides no meaning to our classification problem).

Error in the RBF kernel SVM model might be attributable to estimation, optimization, and Bayes errors. The estimation error is also still due to the imbalanced nature of our dataset causing the model to be overfitted to a certain class than another. Optimization error might be due to not fully exploring the solution space including the hyperparameters to achieve full convergence, and similar to the previous models, the RBF kernel is also still subject to Bayes error due to inherent problems with our datasets including the non-removal of the stopwords and lack of feature engineering.