Problem 1 - Sentiment Analysis on movie reviews

We use **pipeline** and **grid search** to classify movie reviews as either positive or negative. Pipeline is used to chain multiple estimators into one. For pipeline, technically, there is a fixed sequence of steps in processing the data such as feature selection, normalization and classification. Here, we choose TfidfVectorizer and LinearSVC to construct pipeline. Further, we gird search over parameters of all estimators in the pipeline and fit the training data to get the best cross validation model. Since the parameters of the estimator are used, these methods are optimized by cross-validated grid-search over a parameter grid. Finally, we show its final evaluation score and offer a confusion matrix with the model we construct.

An example of a confusion matrix

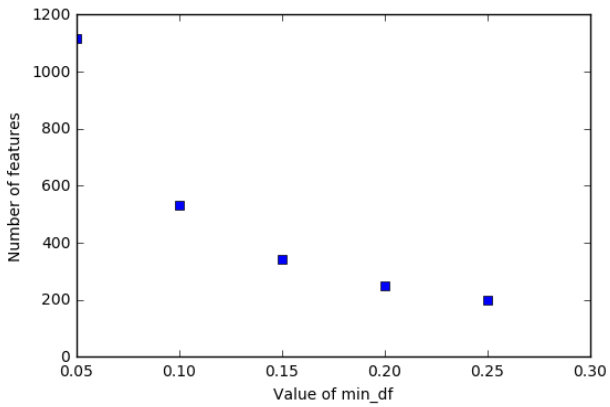
```
             precision    recall  f1-score   support

        neg       0.90      0.87      0.88       247
        pos       0.88      0.91      0.89       253

avg / total       0.89      0.89      0.89       500

[[215  32]
 [ 24 229]]
```

Consider an example of the table above, we find that the precision and recall are close to 90%, which means predicted data is accurate. So we can trust this model.

Problem 2 – Exploring the Scikit-Learn TfidVectorizer Class

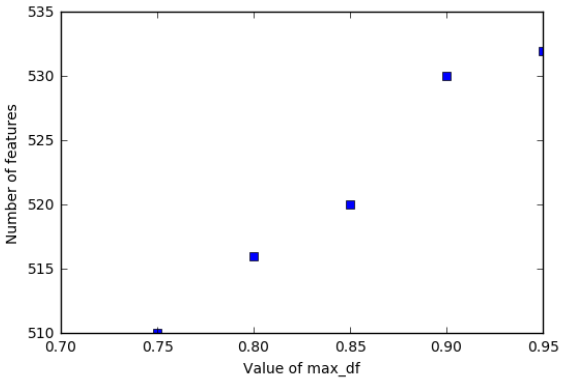
Definition: TF-IDF – term frequency- inverse document frequency

It is a numerical parameter which reflects how frequently a word repeats in a document. Here tf- calculates the raw frequency of each word in the document and idf is a measure of how much information the word provides. It can also be used for filtering stop-words in text-summarization and classification.

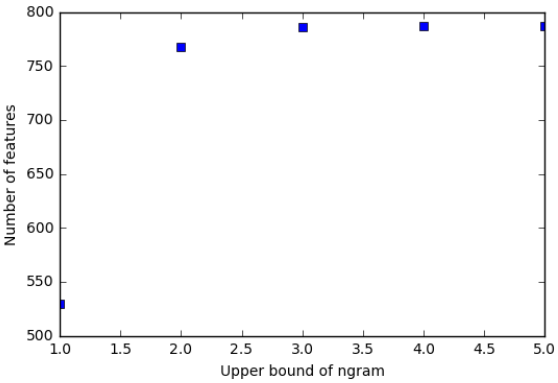


The above graph shows the number of features that were extracted for increasing values of the min_df. As we can see from this figure, when we increase the value of min_df, the number of extracted features will decrease since more and more features' document

frequency are lower than the threshold. By selecting a proper value of min_df, we can eliminate the word feature which shows up in a very limited number of reviews.



The above graph shows the number of features that were extracted for decreasing values of the max_df. As we can see from this figure, when we decrease the value of max_df, the number of extracted features decreases since more and more features have a higher document frequency than the threshold. Changing this value can help us to get rid of the most common word showed in almost every reviews.



The above graph shows the number of features that were extracted for increasing upper bound of ngram. We can see that the number of features increases as the value of upper bound increases before it reaches 3.0. This means higher upper bound of ngram allows us to extract more features. However, after a certain threshold value is reached, the increase of feature space is pretty small.

Problem 3: Machine learning algorithms

We know of the confusion matrix which contains True Positive, False Positive, True Negative and False Negative. The concepts of precision and recall are based on the confusion matrix. Here,

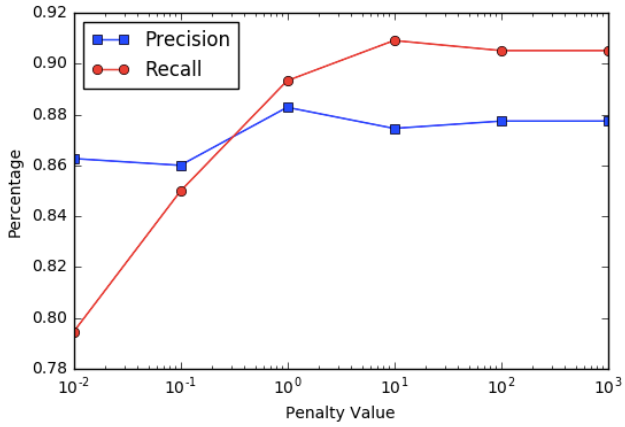
Precision (P) is defined as the number of True Positives upon the sum of True Positives and False Positives:

$$P = \frac{T_p}{T_p + F_p}$$

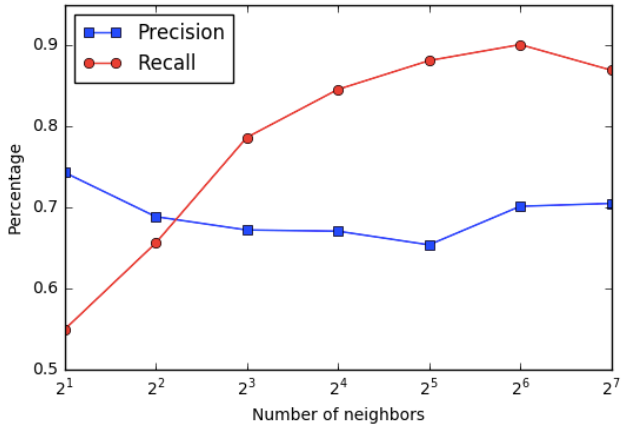
Recall (R) is defined as the number of True Positives upon the sum of True Positives and False Negatives:

$$R = \frac{T_p}{T_p + F_n}$$

We examined Precision and Recall using the Linear SVC under different penalty parameter C and the result is as follows:



We then examined Precision and Recall using KNeighbors Classifier with different number of neighbors and the result was a little different:



In this case, we noticed that when **penalty parameter C in LinearSVC and number of neighbors in KNN increases, recall will have a relatively large improvement especially in KNN result. It shows that a relatively less flexible model will benefit the recall in this dataset.**

Linear SVC's confusion matrix is as follows:

```
mean - 0.85; std - 0.01
             precision    recall  f1-score   support

        neg       0.87      0.85      0.86       252
        pos       0.85      0.87      0.86       248

avg / total       0.86      0.86      0.86       500

[[215  37]
 [ 33 215]]
```

KNeighbors Classifier's confusion matrix is as follows:

```
             precision    recall  f1-score   support

        neg       0.81      0.65      0.72       247
        pos       0.72      0.85      0.78       253

avg / total       0.76      0.75      0.75       500

[[161  86]
 [ 37 216]]
```

Besides LinearSVC and KNN, we also tried Random Forest and Multilayer Perceptron model to classify this dataset.

Random Forests

A random forest is an ensemble of decision trees which will output a prediction value. Each decision tree is constructed by using a random subset of the training data. After you have trained your forest, you can then pass each test row through it, in order to output a prediction.

The confusion matrix of Random Forests is as follows:

```
             precision    recall  f1-score   support

        neg       0.75      0.87      0.80       247
        pos       0.85      0.71      0.77       253

avg / total       0.80      0.79      0.79       500

[[214  33]
 [ 73 180]]
```

Multi-Layer Perceptron

An MLP can be viewed as a logistic regression classifier where the input is first transformed using a learnt non-linear transformation Φ. This transformation projects the input data into a space where it becomes linearly separable. This intermediate layer is referred to as a hidden layer. A single hidden layer is sufficient to make MLPs a universal approximator.

The confusion matrix of MLP is as follows:

```
             precision    recall  f1-score   support

        neg       0.90      0.88      0.89       247
        pos       0.89      0.91      0.90       253

avg / total       0.90      0.90      0.90       500

[[218  29]
 [ 23 230]]
```

We see that using MLP, our precision and recall values have been very high compared to other models. In fact, MLP has outperformed LinearSVC too!

However, compared with LinearSVC, we can say that Random Forest and KNN performed poorly. Taking the number of features in training dataset (around 1000) into consideration, we thought that the issue was that there are too many features inside it. Since Random Forest and KNN are more flexible model, when we apply them on a very high dimensional

data, its performance could be poor. Therefore, we want to add principle component analysis between vectorization and training. By introducing PCA in between, we hoped Random Forest and KNN to have better performance than before.

Confusion matrix of KNN is as follows:

```
             precision    recall  f1-score   support

        neg       0.61      0.81      0.70       247
        pos       0.73      0.50      0.59       253

avg / total       0.67      0.65      0.65       500

[[201  46]
 [127 126]]
```

And the confusion matrix of Random Forest after applying PCA is as follows:

```
             precision    recall  f1-score   support

        neg       0.63      0.81      0.71       247
        pos       0.74      0.55      0.63       253

avg / total       0.69      0.67      0.67       500

[[199  48]
 [115 138]]
```

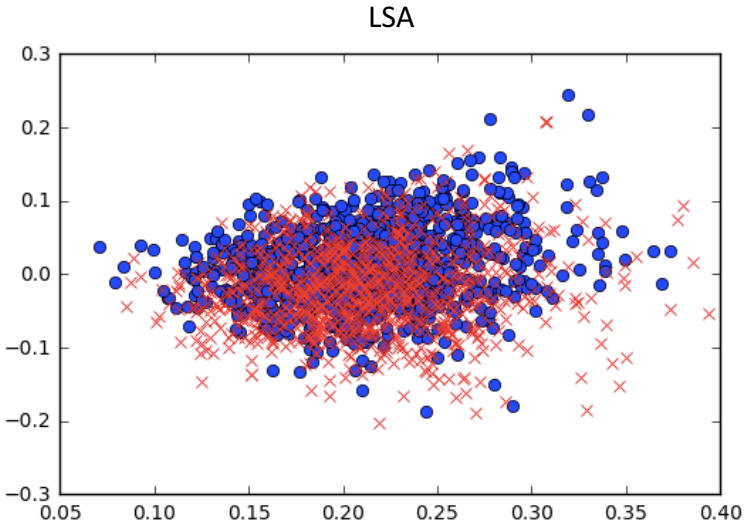
Compared with previous results, we found that it's counterintuitive that PCA make their performance even worse. We will keep the answer of this question and show it in the next section.

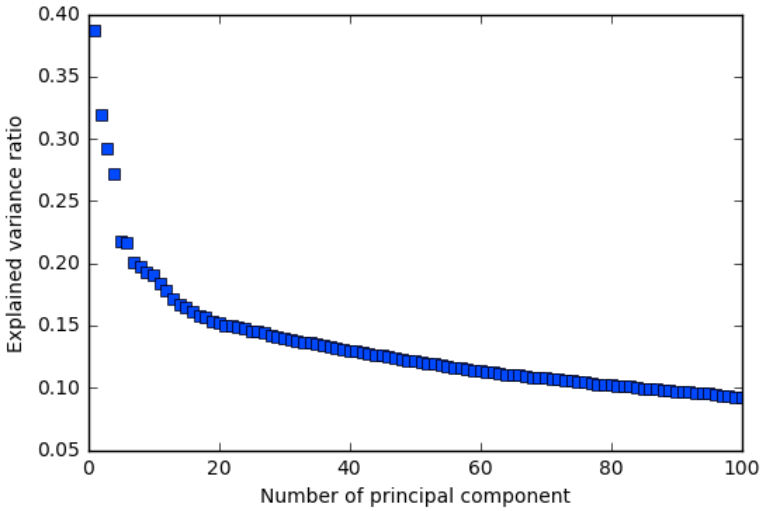
Problem 4: Finding the Right Plot

This was a tricky question and we spent a lot of time trying to classify properly the positive and negative reviews.

LSA

Latent Semantic Analysis is a technique for creating a vector representation of a document. Having a vector representation of a document gives you a way to compare documents for their similarity by calculating the distance between the vectors. This in turn means that we can classify the documents into different categories, in our case, positive and negative.





From these figures, we noticed that 2D PCA figure failed to give us a better classification boundary. To find the reason, we show the first 100 principal components and the ratio of variance they explained in the second figure. We can see that there is very limited information in each principal component and it's why the first figure doesn't help us to separate reviews in 2D plot. Also, the question why PCA decreases the overall performance of KNN and Random Forest is also answered.

Semantic Orientation Analysis using PMI Value

To get some insights of what happened in the dataset, we recall the semantic analysis method that we used in the first casestudy. Compared with principal component, we hope this method gives us more interpretation of the data.

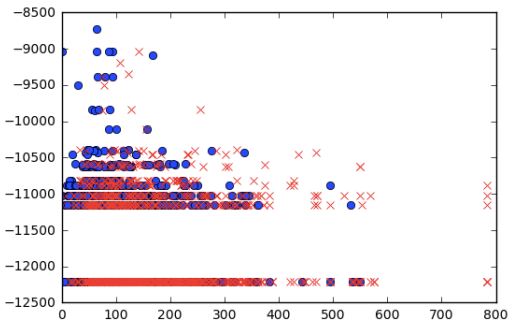
Two concepts were used here:

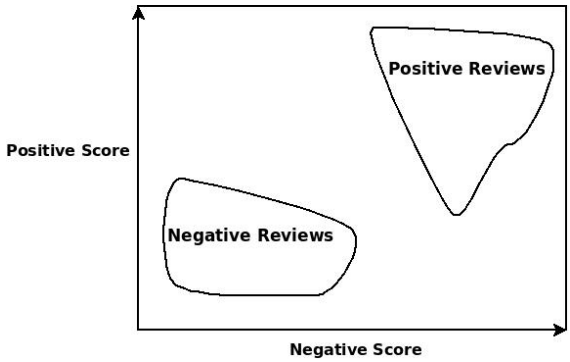
1. PMI Value The PMI value, also called Pointwise Mutual Information, is a mathematical function that gives people a general idea of how “close” two words could be by measuring the co- occurrence probability divided by their own probability to show up. The detailed function is showed in below.

$$PMI(w1, w2) = \log\left(\frac{P(w1 \wedge w2)}{P(w1)P(w2)}\right)$$

2. Semantic Orientation Value The basic idea of Semantic Orientation is introduced in Turney’s paper. A brief description is that given a positive and negative word pool, we can calculate PMI from a word to these two pools. By calculating the over-all distance from these pools, we can have a orientation of it. The advantage of Semantic Orientation is that once we have general positive and negative word pools, we don’t have to get any training data for particular input. Therefore, it’s a kind of unsupervised classification method. The general function of Semantic Orientation is also listed here.

$$SO(w) = \sum_{w' \in P^+} PMI(w, w') - \sum_{w' \in P^-} PMI(w, w')$$



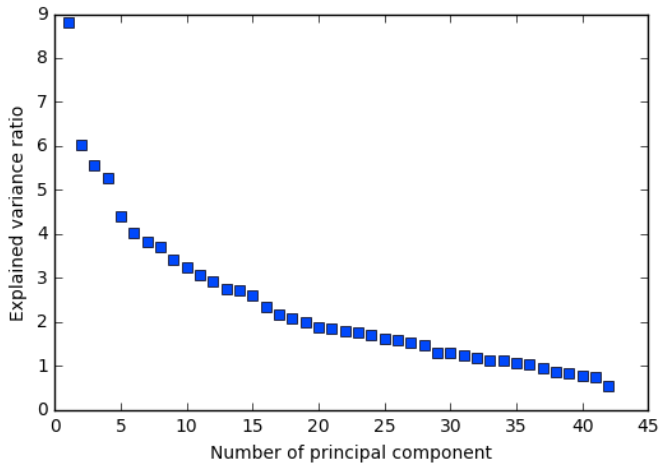


x axis is the max positive value in each review, y axis is the max negative value in each review

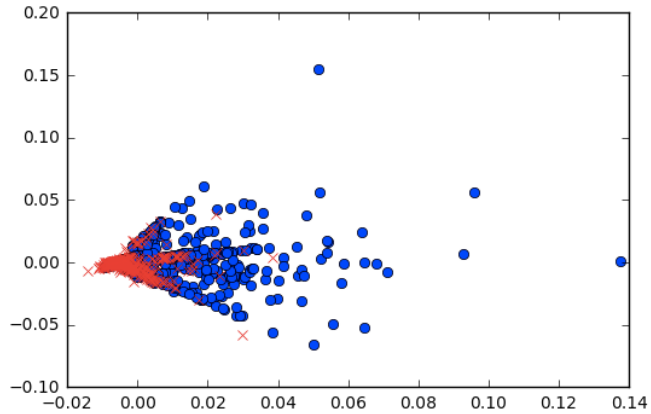
We see that in an ideal situation, the right side plot should stand. But in our dataset, we found out that people are using a lot of negative words in positive reviews too. So it becomes difficult to classify the reviews based on sentiment analysis.

Feature Selection

Based on previous study, we noticed that our feature space is too large to do a PCA and visualize in 2D. Therefore, we are tring to find out some feature selection method to help us filtering some important features out before we can use PCA to generate principal components.



From the figure above, we see that there is a significant improvement in explained variations in first few principal components. Therefore, we chose the first two Principal Components and plotted the figure which is shown below:



From this figure, there are a few zones containing the positive review. By joining some circles, we can somehow separate the positive reviews from the negative ones.

We tried our best to distinguish it into positive and negative and we were a little successful in doing that.

Summary

- **What is the relationship between this topic and Business Intelligence?**

  We can take an example of Netflix whether it wants to add a movie to its collection or not. Using some of the models above, it can create a collection of multiple movie reviews and put it in this model. After putting it in the model, our model will classify the reviews and give a general idea if the movie reviews are positive or negative. Based on this information, Netflix can then decide if it wants to add this movie to its collection or not.

- **How did you analyse the data?**

  We used the Scikit-learn package in Python to analyse our data. We identified the precision and recall using the Confusion Matrix and applied KNeighbours Classifier, PCA, Linear SVC, Random Forests and Multilayer Perceptron.
  For the visualizations where we attempted to classify the positive and negative reviews, we start from LSA and Semantic Orientation Analysis using PMI Value. For further improvement, we add feature selection model to help us determine the most important features.

- **What did you find in the data?**

  With the statistics of precision and recall, we found that the best performing method was Multilayer Perceptron and Linear SVC. Random Forest and KNeighbours Classifier provided decent results. We thought that Dimension Reduction using PCA for KNN and Random Forest should provide better results but it was the opposite of that. The reason is that the original feature space doesn't have a relatively high correlation between different features. Therefore, the top 100 principal component only contain very limited information.

- **What conjectures did you make and how did you support or disprove them using data?**

  Our main conjecture was that we thought that applying PCA to Random Forest and KNN would improve our model. But we were wrong. As described above previously, we found out that our model performed poorly after adding PCA to Random Forest and KNN. The reason is that the original feature space doesn't have a relatively high correlation between different features. Therefore, the top 100 principal component only contain very limited information.

- **Did you find anything surprising in the data?**

  Two things were rather surprising.
  Firstly, we thought that applying Dimension Reduction using PCA should improve the performance of our models but it failed.
  Secondly, by using the semantic orientation, we found that people are using lots of negative components in their positive reviews. At the same time, the positive components in positive reviews are not significantly stronger than those in negative reviews.
- **What business decision do you think this data could help answer? Why?**

  As mentioned above, this model will enable movie streaming/rental companies to decide if they wish to add the movie to their database or not by collecting all the reviews of the movies and inputting them into this model.