I report some experiments on using a Naive Bayes classifier for classifying the Movie Preference problem. In general, the Naive Bayes classifier has achieved F1 scores in the range of 70%-85%.

1 Methodology

The following parameters were tested:

- The smoothing coefficient. Values from $10^{-4}$ up to $10^2$ were tested on a power scale.
- The choice of the model. Both Bernoulli and Multinomial models were tested.
- The prepossessing of the input. I restricted myself to the default tokenizer from ScalaNLP, and some different windows of n-grams (1, 2, and 3). Note that n-grams included also all the k-grams ($k < n$). I also applied the Porter Stemmer before building the n-grams to keep only the roots of the words (Algorithm 1). It marginally improved scores compared to using plain words.

For all the combination of choices, the classifier was run 10 times using cross-validation. The cross-validation batches were generated by shuffling the complete of examples to ensure a good spread, and then bucketing in a round-robin fashion (Algorithm 2). I found that the preshuffling step made a difference in the spread of the scores. Indeed, the implementation starts by concatenating the list of positive and negative example, so the cross-validation procedure would create batches with exactly the same proportion of negatives and positives. The pre-shuffling gives a bit more diversity do the batches and gives more indication about the robustness of the algorithm.

2 Results

All the runs are presented in the figures, for different values of the smoothing coefficient $\theta$.

The best results were obtained for 2-grams with a binomial distribution (the “bag of words” model), with a median F1 score of 85% for the best parameters. On average, the multinomial models always performed more poorly (in terms of F1 score) than the binomial models for every choice of input and every choice of smoothing. This difference is much more pronounced for 1-grams and less so for 2- and 3-grams. The performance of the classifiers substantially drops if the value of the smoothing coefficient is too large: the features are damped by the strong prior and produce too weak a classifier. For 2-grams and 3-grams, a small value for the smoothing does not seem to have much impact on the performance of the classifier. It would have been interesting to take even smaller values to see some drop in performance.

Due to running time, I limited the experiments up to 3-grams. Scala seems to choose some weak structures for representing n-grams (collection.immutable.Vector) that negatively impacted the performance of the classification. Using 3-grams did not significantly improve the results, probably due to the size of the training data.
Algorithm 1 Data preprocessing procedure

def file_to_data_grams(f: File,
    gram_size: Int,
    counting: Int => Double): Counter[String, Double] = {
    val ps = PorterStemmer()
    val wstk = WhitespaceTokenizer()
    val words: Seq[String] = ps.apply(Source.fromFile(f, "UTF-8")
        .getLines()
        .flatMap((s: String) => wstk.apply(s)).toIterable).toSeq
    val grams = words ++ (2 to gram_size).flatMap(gram => {
        words.sliding(gram).map(_.mkString(" "))
    })
    Counter(grams.groupBy(s => s).map({ case (s, l) =>
        (s, counting(l.length)) })))
}

Algorithm 2 Creation of the cross-validation batches

type DataSet = Seq[Example[Int, Counter[String, Double]]]
val z: Seq[DataSet] = Random.shuffle(data)
    .zipWithIndex
    .groupBy(_._2 % num_batches)
    .values
    .toSeq.map(_._1)

Figure 1: Results for different values of $\theta$, when using a binomial distribution on a 1-gram dataset. On
the left, F1 scores, and ROC plot on the right. The solid line is the median of the F1 score for each
cross-validation experiment, the dotted lines are the extremal values and the crosses are the individual
results for each cross-validation.
Figure 2: Results for different values of $\theta$, when using a multinomial distribution on a 1-gram dataset. On the left, F1 scores, and ROC plot on the right. The solid line is the median of the F1 score for each cross-validation experiment, the dotted lines are the extremal values and the crosses are the individual results for each cross-validation.

Figure 3: Results for different values of $\theta$, when using a binomial distribution on a 2-gram dataset. On the left, F1 scores, and ROC plot on the right. The solid line is the median of the F1 score for each cross-validation experiment, the dotted lines are the extremal values and the crosses are the individual results for each cross-validation.
Figure 4: Results for different values of $\theta$, when using a multinomial distribution on a 2-gram dataset. On the left, F1 scores, and ROC plot on the right. The solid line is the median of the F1 score for each cross-validation experiment, the dotted lines are the extremal values and the crosses are the individual results for each cross-validation.

Figure 5: Results for different values of $\theta$, when using a binomial distribution on a 3-gram dataset. On the left, F1 scores, and ROC plot on the right. The solid line is the median of the F1 score for each cross-validation experiment, the dotted lines are the extremal values and the crosses are the individual results for each cross-validation.
Figure 6: Results for different values of $\theta$, when using a multinomial distribution on a 3-gram dataset. On the left, F1 scores, and ROC plot on the right. The solid line is the median of the F1 score for each cross-validation experiment, the dotted lines are the extremal values and the crosses are the individual results for each cross-validation.

<table>
<thead>
<tr>
<th>Positive class</th>
<th>Negative class</th>
</tr>
</thead>
<tbody>
<tr>
<td>outstand</td>
<td>incoher</td>
</tr>
<tr>
<td>seemless</td>
<td>predat</td>
</tr>
<tr>
<td>hatr</td>
<td>schumach</td>
</tr>
<tr>
<td>strongest</td>
<td>illog</td>
</tr>
<tr>
<td>mulan</td>
<td>ludicr</td>
</tr>
<tr>
<td>gump</td>
<td>hudson</td>
</tr>
</tbody>
</table>

Table 1: Strongest word signals, positive class

I also report in Table 1 a few examples of the strongest words associated to each class by the classifier. Interestingly, it is mostly stemmed adjectives and a few keywords from movies: “Mulan” and “Forrect Gump” for the positive reviews are notable. The classifier is influenced both by domain-specific vocabulary (movie titles) and by general description vocabulary (“outstanding”, “seemless”, “strongest”). I wonder if using LDA instead of Naive Bayes would have found these different classes. Also, a certain number of keywords from the StarWars trilogy were close (“Vador”, “Obi-Wan”, “Kenobi”) (around the 10th position, not reported in the table). I could not figure out which movies were referred to in the second set: “Predator” for “Predat”? I am not sure for “Schumach” and “Hudson”.