MusicMood

Machine Learning in Automatic Music Mood Prediction Based on Song Lyrics

Sebastian Raschka
December 10, 2014
Music Mood Prediction

• We like to listen to music [1][2]
• Digital music libraries are growing
• Recommendation system for happy music (clinics, restaurants ...) & genre selection

Predictive Modeling

- Reinforcement learning
- Hidden Markov models
- Unsupervised learning
- Clustering
- Supervised learning
- Ranking
- Classification
- Regression

- DBSCAN on a toy dataset
- Naive Bayes on Iris (after LDA)
Supervised Learning In a Nutshell
MusicMood - The Plan

- Million Song Dataset
- Sampling and Labeling
  - Training Dataset
  - Validation Dataset
- Mood Classifier
- Web App
The Dataset

Home » Getting the dataset

Getting the dataset

The logistics of distributing a 300 GB dataset are a little more complicated than for smaller collections. We do, however, provide a directly-downloadable subset for a quick look.

Before you start, you might want to review exactly what the dataset contains. Here is a page showing the contents of a single example file. You can download the corresponding raw HDF5 file here: TRAXLZU12903D05F94.h5.

http://labrosa.ee.columbia.edu/millionsong/
Sampling

Lyrics available?  
http://lyrics.wikia.com/Lyrics_Wiki

Lyrics in English?  
Python NLTK

1000 songs for training

200 songs for validation
Mood Labels

Downloading mood labels from Last.fm

Manual labeling based on lyrics and listening

sad if ...

- Dark topic (killing, war, complaints about politics, ...)
- Artist in sorrow (lost love, ...)

Why so sad?
The mood of music over the last 50 years

[based on the 1000-song training dataset]
https://github.com/rasbt/musicmood

Sebastian Raschka 2014

This work is licensed under a Creative Commons Attribution 4.0 International License.
Word Clouds

happy:

sad:
A Short Introduction to Naive Bayes Classification
Naive Bayes - Why?

- Small sample size, can outperform the more powerful alternatives [1]

- "Eager learner" (on-line learning vs. batch learning)

- Fast for classification and re-training

- Success in Spam Filtering [2]

- High accuracy for predicting positive and negative classes in a sentiment analysis of Twitter data [3]

---


Bayes Classifiers
It’s All About Posterior Probabilities

\[ P(\omega_j | x_i) = \frac{P(x_i | \omega_j) \cdot P(\omega_j)}{P(x_i)} \]

posterior probability = \frac{\text{conditional probability \cdot prior probability}}{\text{evidence}}

objective function: maximize the posterior probability

\[
predicted \text{ class label } \leftarrow \arg \max_{j=1,...,m} P(\omega_j | x_i)\]
The Prior Probability

Maximum Likelihood Estimate (MLE)

\[ \hat{P}(\omega_j) = \frac{N_{\omega_j}}{N_c} \]

- $N_{\omega_j}$: Count of samples from class $\omega_j$.
- $N_c$: Count of all samples.
The Effect of Priors on the Decision Boundary

If $P(\omega_1) < P(\omega_2)$

If $P(\omega_1) = P(\omega_2)$

If $P(\omega_1) > P(\omega_2)$
Class-Conditional Probability

Maximum Likelihood Estimate (MLE)

\[ \hat{P}(x_i | \omega_j) = \frac{N_{i,c}}{N_i} \quad (i = (1, \ldots, d)) \]

- \( N_{i,c} \): Count of feature \( x_i \) in class \( \omega_j \).
- \( N_i \): Count of feature \( x_i \) in all classes.

"chance of observing feature \( x_i \) given that it belongs to class \( \omega_j \)."
Evidence

\[
P(x_i) = P(x_i \mid \omega_j) \cdot P(\omega_j) + P(x_i \mid \omega_j^c) \cdot P(\omega_j^c)
\]

just a normalization factor, can be omitted in decision rule:

\[
\frac{P(x_i \mid \omega_1) \cdot P(\omega_1)}{P(x_i)} > \frac{P(x_i \mid \omega_2) \cdot P(\omega_2)}{P(x_i)}
\]

\[
\alpha P(x_i \mid \omega_1) \cdot P(\omega_1) > P(x_i \mid \omega_2) \cdot P(\omega_2)
\]
Naive Bayes Models

Gaussian Naive Bayes

for continuous variables
Naive Bayes Models

Multi-variate Bernoulli Naive Bayes

\[
P(x|\omega_j) = \prod_{i=1}^{m} P(x_i|\omega_j)^b \cdot (1 - P(x_i|\omega_j))^{(1-b)} \quad (b \in 0, 1)
\]

for binary features
Naive Bayes Models

Multinomial Naive Bayes

\[
\hat{P}(x_i \mid \omega_j) = \frac{\sum tf(x_i, d \in \omega_j)}{\sum N_{d \in \omega_j} + \alpha} \cdot V
\]

\[
P(x \mid \omega_j) = P(x_1 \mid \omega_j) \cdot P(x_2 \mid \omega_j) \cdot \ldots \cdot P(x_n \mid \omega_j) = \prod_{i=1}^{m} P(x_i \mid \omega_j)
\]

- \(x_i\): A word from the feature vector \(x\) of a particular sample.
- \(\sum tf(x_i, d \in \omega_j)\): The sum of raw term frequencies of word \(x_i\) from all documents in the training sample that belong to class \(\omega_j\).
- \(\sum N_{d \in \omega_j}\): The sum of all term frequencies in the training dataset for class \(\omega_j\).
- \(\alpha\): An additive smoothing parameter (\(\alpha = 1\) for Laplace smoothing).
- \(V\): The size of the vocabulary (number of different words in the training set).
Naive Bayes and Text Classification
**Feature Vectors**

**The Bag of Words Model**

- $D_1$: "Each state has its own laws."
- $D_2$: "Every country has its own culture."

<table>
<thead>
<tr>
<th></th>
<th>each</th>
<th>state</th>
<th>has</th>
<th>its</th>
<th>own</th>
<th>laws</th>
<th>every</th>
<th>country</th>
<th>culture</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_{D_1}$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$x_{D_2}$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$\sum$</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Tokenization and N-grams

“a swimmer likes swimming thus he swims”

- unigram (1-gram):
  a swimmer likes swimming thus he swims

- bigram (2-gram):
  a swimmer swimmer likes likes swimming swimming thus ...

- trigram (3-gram):
  a swimmer likes swimmer likes swimming likes swimming thus ...

Stemming and Lemmatization

Porter Stemming

A swimmer likes swimming, thus he swims.

A swimmer like swim , thu he swim .

Lemmatization

A swimmer likes swimming, thus he swims.

A swimmer like swimming , thus he swim .
Stop Word Removal

A swimmer likes swimming, thus he swims.

swimmer likes swimming , swims .
Term and Frequency

normalized term frequency = \frac{tf(t, d)}{n_d}

where

- $tf(t, d)$: Raw term frequency (the count of term $t$ in document $d$).
- $n_d$: The total number of terms in document $d$. 
Term Frequency - Inverse Document Frequency (Tf-idf)

\[ \text{Tf-idf} = \text{tf}_n (t, d) \cdot \text{idf} (t) \]

Let \( \text{tf}_n (d, f) \) be the normalized term frequency, and \( \text{idf} \), the inverse document frequency, which can be calculated as follows

\[ \text{idf} (t) = \log \left( \frac{n_d}{n_d (t)} \right), \]

where

- \( n_d \): The total number of documents.
- \( n_d (t) \): The number of documents that contain the term \( t \).
Grid Search and 10-fold Cross Validation to Optimize F1

\[ F1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \]

\[ \text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \]

\[ \text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \]

- TP = true positive (happy predicted as happy)
- FP = false positive (sad predicted as happy)
- FN = false negative (happy predicted as sad)
**K-Fold Cross Validation**

*Complete dataset*

- **Training dataset**
- **Test dataset**

**k-fold cross-validation (k=4):**

<table>
<thead>
<tr>
<th>fold 1</th>
<th>fold 2</th>
<th>fold 3</th>
<th>fold 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test set</td>
<td>Test set</td>
<td>Test set</td>
<td>Test set</td>
</tr>
</tbody>
</table>

1st iteration → calc. error

2nd iteration → calc. error

3rd iteration → calc. error

4th iteration → calc. error

*calculate avg. error*
10-Fold Cross Validation After Grid Search

Receiver Operator Characteristic of the Lyrics Classifier

True Positive Rate vs. False Positive Rate for each fold:
- ROC fold 1 (area = 0.86)
- ROC fold 2 (area = 0.75)
- ROC fold 3 (area = 0.69)
- ROC fold 4 (area = 0.80)
- ROC fold 5 (area = 0.66)
- ROC fold 6 (area = 0.78)
- ROC fold 7 (area = 0.71)
- ROC fold 8 (area = 0.81)
- ROC fold 9 (area = 0.71)
- ROC fold 10 (area = 0.74)
- Random Guessing (area = 0.50)
- Mean ROC (area = 0.75)

(final model)
10-fold Cross Validation (mean ROC)
Multinomial vs Multi-variate Bernoulli Naive Bayes

- 1: MV Bernoulli NB, stop words, porter stemmer, uni-gram, df (ROC AUC = 0.73)
- 2: Multinomial NB, stop words, porter stemmer, uni-gram, tf (ROC AUC = 0.72)
- 3: Multinomial NB, stop words, porter stemmer, uni-gram, tf-idf (ROC AUC = 0.75)
- Random Guessing
10-fold Cross Validation (mean ROC)
Multinomial Naive Bayes & Hyperparameter Alpha

Multinomial NB, stop words, porter stemmer, uni-gram, tf-idf

- alpha = 0.05 (ROC AUC = 0.72)
- alpha = 0.1 (ROC AUC = 0.74)
- alpha = 1.0 (ROC AUC = 0.75)
- alpha = 2.0 (ROC AUC = 0.75)
- Random Guessing
10-fold Cross Validation (mean ROC)
Multinomial Naive Bayes & Vocabulary Size

Multinomial NB, stop words, porter stemmer, uni-gram, tf-idf

True Positive Rate

False Positive Rate

- max features = 1000 (ROC AUC = 0.75)
- max features = 3000 (ROC AUC = 0.76)
- max features = 5000 (ROC AUC = 0.75)
- max features = all (=8550) (ROC AUC = 0.75)
- Random Guessing
10-fold Cross Validation (mean ROC)
Multinomial Naive Bayes & Document Frequency Cut-off

Multinomial NB, stop words, porter stemmer, uni-gram, tf-idf

- no cutoff (ROC AUC = 0.75)
- min. df = 0.1 (ROC AUC = 0.68)
- min. df = 0.01 (ROC AUC = 0.75)
- Random Guessing
10-fold Cross Validation (mean ROC)  
Multinomial Naive Bayes & N-gram Sequence Length
Contingency Tables of the Final Model

<table>
<thead>
<tr>
<th>Performance metric</th>
<th>Training</th>
<th>Validation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>80.00%</td>
<td>54.50%</td>
</tr>
<tr>
<td>Precision</td>
<td>99.60%</td>
<td>88.89%</td>
</tr>
<tr>
<td>Recall</td>
<td>55.38%</td>
<td>15.24%</td>
</tr>
<tr>
<td>F1-score</td>
<td>71.18%</td>
<td>26.02%</td>
</tr>
<tr>
<td>ROC auc</td>
<td>77.60%</td>
<td>56.57%</td>
</tr>
</tbody>
</table>
Live Demo

http://sebastianraschka.com/Webapps/musicmood.html

Artist name: Bob Dylan
Song title: Blowing in the wind

Lyrics:
How many roads must a man walk down Before you call him a man? Yes, 'n' how many seas must a white dove sail Before she sleeps in the sand? Yes, 'n' how many times must the cannon balls fly Before they're forever banned? The answer, my friend, is blowin' in the wind
The answer is blowin' in the wind How many times must a man look up Before he can see the sky? Yes, 'n' how many ears must one man have Before he can hear people cry? Yes, 'n' how many deaths will it take till he knows That too many people have died? The answer, my friend, is blowin' in the wind
The answer is blowin' in the wind How many years can a mountain exist Before it's washed to the sea? Yes, 'n' how many years can some people exist Before they're allowed to be free? Yes, 'n' how many times can a man turn his head Pretending he just doesn't see? The answer, my friend, is blowin' in the wind
The answer is blowin' in the wind

Prediction: sad (probability 63.83%)

Try again

Listen to this song on YouTube.

I am looking forward to your feedback in order to improve the mood classifier! Clicking one of the two buttons below will add a new mood label to existing songs in the database, or a new database entry will be created if this song was not included in the training dataset: yet.

I think this song is  happy  sad

- 1332 lyrics
- 1350 mood labels

Flask
web development, one drop at a time
Future Plans

- Growing a list of mood labels (majority rule).

- Performance comparisons of different machine learning algorithms.

- Genre prediction and selection based on sound.
Thank you!