Music classification mini project: Instructions

November 4, 2020, Version 3.0
David Widmann, david.widmann@it.uu.se
Department of Information Technology, Uppsala University

Abstract

This document contains the instructions for the mini project on classification for the course Statistical Machine Learning, 1RT700. The problem is to classify a set of 200 songs, and predict whether Andreas Lindholm would like them or not, with the help from a training data set with 750 songs. You are expected to (i) try some (or all) classification methods from the course and evaluate their performance on the problem, and (ii) make a decision which one to use and 'put in production' by uploading your predictions to a website, where your prediction will be evaluated and also compared to the performances of the other student groups. In addition, you should also (iii) discuss some ethical aspects of machine learning within your group. You will document your project by writing a report, which will be reviewed anonymously by your peers. A very well implemented and documented project will earn you a 'gold star' and a higher grade on the report.

0 Requirements

The project is to be done in groups of 3-4 students. All tasks described in this document have to be done in order to pass the project, and of course *all group members have to take part in the project*.

1 Problem: music classification

The technical problem is to tell which songs, in a dataset of 200 songs, Andreas Lindholm is going to like (see Figure 1). The data set consists not of the songs themselves, but of high-level features extracted using the web-API from Spotify[1]. These high-level features describe characteristics such as the acousticness, danceability, energy, instrumentalness, valence and tempo of each song.

Figure 1: Andreas Lindholm listening to music.

To your help, you are provided a training dataset with 750 songs, each of which Andreas has labeled with LIKE or DISLIKE. You are expected to use all the knowledge that you have acquired in the

[1] https://developer.spotify.com/web-api/get-audio-features/

Table 1: Details on the available features (from the Spotify API documentation)

Name	Type	Description
acousticness	float	A confidence measure from 0.0 to 1.0 of whether the track is acoustic. 1.0 represents high confidence the track is acoustic.
danceability	float	Danceability describes how suitable a track is for dancing based on a combination of musical elements including tempo, rhythm stability, beat strength, and overall regularity. A value of 0.0 is least danceable and 1.0 is most danceable.
duration	int	The duration of the track in milliseconds.
energy	float	Energy is a measure from 0.0 to 1.0 and represents a perceptual measure of intensity and activity. Typically, energetic tracks feel fast, loud, and noisy. For example, death metal has high energy, while a Bach prelude scores low on the scale. Perceptual features contributing to this attribute include dynamic range, perceived loudness, timbre, onset rate, and general entropy.
instrumentalness	float	Predicts whether a track contains no vocals. "Ooh" and "aah" sounds are treated as instrumental in this context. Rap or spoken word tracks are clearly "vocal". The closer the instrumentalness value is to 1.0, the greater likelihood the track contains no vocal content. Values above 0.5 are intended to represent instrumental tracks, but confidence is higher as the value approaches 1.0.
key	int	The key the track is in. Integers map to pitches using standard Pitch Class notation. E.g. 0 = C, 1 = C major/D minor, 2 = D, and so on.
liveness	float	Detects the presence of an audience in the recording. Higher liveness values represent an increased probability that the track was performed live. A value above 0.8 provides strong likelihood that the track is live.
loudness	float	The overall loudness of a track in decibels (dB). Loudness values are averaged across the entire track and are useful for comparing relative loudness of tracks. Loudness is the quality of a sound that is the primary psychological correlate of physical strength (amplitude). Values typical range between -60 and 0 db.
mode	string	Mode indicates the modality (major or minor) of a track, the type of scale from which its melodic content is derived.
speechiness	float	Speechiness detects the presence of spoken words in a track. The more exclusively speech-like the recording (e.g. talk show, audio book, poetry), the closer to 1.0 the attribute value. Values above 0.66 describe tracks that are probably made entirely of spoken words. Values between 0.33 and 0.66 describe tracks that may contain both music and speech, either in sections or layered, including such cases as rap music. Values below 0.33 most likely represent music and other non-speech-like tracks.
tempo	float	The overall estimated tempo of a track in beats per minute (BPM). In musical terminology, tempo is the speed or pace of a given piece and derives directly from the average beat duration.
time_signature	int	An estimated overall time signature of a track. The time signature (meter) is a notational convention to specify how many beats are in each bar (or measure).
valence	float	A measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track. Tracks with high valence sound more positive (e.g. happy, cheerful, euphoric), while tracks with low valence sound more negative (e.g. sad, depressed, angry).

course about classification algorithms, to come up with *one* algorithm that you think is suited for this problem and which you decide to put 'in production'.

1.1 Data sets

The data set to classify is available as `songs_to_classify.csv`, and the training data is available as `training_data.csv` on Studium. The columns in these tables represent extracted features, as specified by the header and documented in Table 1. The column "label" in `training_data.csv` is encoded as $1 =$ LIKE and $0 =$ DISLIKE.

1.2 Background

The problem of predicting user preferences is a hot research topic both in academia and industry: you have probably seen "you would perhaps also like ..." in online services. Within music, a big player has been the Echo Nest, founded in 2005 as a research spin-off from the MIT Media Lab and later acquired by Spotify. Their focus was methods for automated understanding of music, and in 2011 they released a popular benchmark dataset 'the million song dataset' (Bertin-Mahieux et al. 2011) which has become popular in the research community (see, for example, Fu et al. 2011; Oord, Dieleman, and Schrauwen 2013), and has similarities to this project. An overview of the scientific field of music recommendation is found in Kaminskas and Ricci (2012), and some pointers to recent advances can be found in Dieleman (2016) and Jacobson et al. (2016).

2 Technical tasks

2.1 Methods to explore

The course has (so far[2]) covered the five following 'families' of classification methods:

(i) logistic regression

(ii) discriminant analysis: LDA, QDA

(iii) K-nearest neighbor

(iv) Tree-based methods: classification trees, random forests, bagging

[2]Deep learning, which will be covered later, is also possible to use for classification. You are of course welcome to explore this as well, in addition to the minimum requirements.

(v) Boosting

In this project, you decide upon *at least* as many 'families' as you are group members, and decide in each 'family' *at least* one method to explore. Thus, if you are 4 group members, you explore at least 4 methods from at least 4 different 'families'.

In addition, you should *also* consider the very simple classifier of always predicting LIKE.

2.2 What to do with each method

For *each* method you decide to explore, you should do the following:

(a) Implement the method. We suggest that you use Python, and you may write your own code or use packages (the material from the problem solving sessions can be useful).

(b) Tune the method to perform well.

(c) Evaluate its performance using, e.g., cross validation. Note that each model needs to be evaluated using *only* the labeled data that is available in `training_data.csv`, i.e. for the purpose of model validation and selection you should *not* use the test data from `songs_to_classify.csv`. Exactly how to carry out this evaluation is up to you to decide.

Once you have completed the aforementioned tasks, you should with a good motivation (hint: cross validation) select which method you decide to use 'in production'. When you have decided which method to try 'in production', run it on the test data (for which you do not have the true labels) and submit your results to http://www.it.uu.se/edu/course/homepage/sml/project/submit/ to see how well it performs. You submit your results as a string like `010011011`, where `0` means DISLIKE and `1` means LIKE. The web site also contains a leader board, where you can see how well you are doing in predicting Andreas' music taste, compared to the other groups.

To pass the project, you only need to submit your final solution to the homepage ***once****!* However, for the sake of a fun competition during the project, we allow each group to submit up to one solution per day (only the latest submission each day will be considered). The leader board will be updated at midnight every night.

3 Reflection task

During the course of the project, you should also spend some time in your group discussing certain ethical aspects of machine learning based on the instructions below. You should summarize and include your discussions in the report. This part of the report should be no longer than 2 pages. You do not have to reach a consensus within the group, but make sure that the thoughts of all members are represented in the report.

Select one of the two following tasks:

(a) You may assume that the user Andreas never reads the "Terms of Use" or the "Privacy policy" of the services he is using, but he is aware that his music preferences are collected in order to provide him with better recommendations. Now assume that the music service is connected to some other common digital service (you choose and specify which one), so that the data set contains user data *both* for the music service *and* the other connected service. The user Andreas is aware that also the other service stores data about his usage, but he has never thought about the fact that they are possibly stored in the same database.

Your task is now to **suggest two usages of the collected data** (music preference + something else) **about which Andreas probably would be concerned and not give his consent**, if he was aware of it.

Example: The music service is connected to an online shopping portal, and the collected data is used to select music that tends to increase Andreas' shopping willingness whenever he is looking at products with high profit margin.

You should **also include your own thoughts and opinions** about what you have suggested. Finally, **also discuss whether you (as machine learning engineers) have to care about the users consent** if the data has already been collected and is available to you?

(b) In a sense machine learning can be understood as "programming through examples" (i.e., learning from training data). By feeding the examples/data into a model, the model automatically decides with no human intervention which features (and combinations thereof) are more important, and which are less so, in order to predict as well as possible. The limited human intervention in the process is one of the key reasons for the popularity of machine learning, but there are also reasons for being cautious.

There is a certain risk that prejudices and cultural biases in the training data are repeated, or even amplified, by the model. Popular examples from the media includes a racial bias in automated prediction of possible future criminals in the US, a Twitter bot from Microsoft which learned from other tweets and quickly became a racist and misogynist, and that Google Photo initially labeled black people as 'gorillas' (DeBrusk 2018; ProPublica 2016; USA Today 2015). In all these examples, the underlying problem is probably a cultural bias in the training data (we cannot expect the model to recognize black people, if the training data only contains photos of white people), which somehow is picked up by the model. We refer to this as *machine-learning bias*.

Suppose that you are working as machine learning consultants contracted by an insurance company. Your task is to design a machine learning system for decision support to the sales agents, based on a wide range of personal information of the potential customer, such as age, gender, yearly income, record of non-payments, family situation, housing standard, postal address, medical record, etc. All in all, **there is a high risk of obtaining (possibly subtle) machine-learning biases in the solution** which could have a big impact on the life of the customer. **Do you, as machine learning engineers, have a responsibility to inform and educate your client** (the insurance company) **about this risk?** Present at least two arguments for both sides (yes and no).

You should **also include your own thoughts and opinions** whether you think machine learning engineers have a general responsibility to make sure their solutions are carefully checked for machine-learning biases? Or is it just a side effect of the technology, which one has to live with?

As a ground for your arguments, you might find the following resources useful: IEEE Code of Ethics (IEEE 2018), The Code of Honour of The Swedish Association of Graduate Engineers (Sveriges Ingenjörer 2018) and an open letter on autonomous weapons from AI & Robotics Researchers (Tegmark et al. 2015). If you are interested in reading more, you might find O'Neil (2016) to be a good start.

4 Documentation

You should summarize your work by writing a report, which will be reviewed by your coursemates as well as the teaching assistants.

4.1 What to include in your report

The report should include the following:

(1) A brief introduction to the problem

(2) A concise description of each of the considered methods, and how they are applied to the problem. Even if you used high-level commands, such as `glm()` for logistic regression, you should explain what is 'under the hood' of the command! *Please, use your own words. We already know what is written in the book and on Wikipedia, so do not copy-paste from there. Writing concise summaries are for your own learning, and their quality is crucial to obtain a gold star.*

(3) How the methods were applied to the data (which inputs were used, if the inputs were considered as qualitative or quantitative, how parameters were tuned, etc), including motivations of the choices made.

(4) Your evaluation of how well each method performs on the problem.

(5) Which method you decided to use 'in production', and your (good) arguments for your choice!

(6) How well your method performs 'in production' (as obtained from http://www.it.uu.se/edu/course/homepage/sml/project/submit/).

(7) Your conclusions.

(8) Your discussion on the reflection task.

(9) Appropriate references.

(10) All code needed to reproduce your reported findings (in an appendix).

4.2 How to format your report

Your report should be submitted as a PDF-file following the style used for the prestigious machine learning conference Neural Information Processing Systems (NeurIPS), which also is the style used for this document. In the NeurIPS format, your report should be *no longer than 6 pages* (not counting the reference list and code appendix). Except for the page limitation, you should follow the NeurIPS style closely, including its instructions for figures, tables, citations, etc.

The recommended word processor to use is LaTeX. You can access the LaTeX files from the conference webpage https://neurips.cc/Conferences/2020/PaperInformation/StyleFiles. If you prefer not to install a LaTeX compiler on your computer, you can use online services such as Overleaf (https://www.overleaf.com/). In your .tex-file, add the lines
`\makeatletter`
`\renewcommand{\@noticestring}{}`
`\makeatother`
before `begin{document}` to suppress the conference-specific footnote.

If you instead prefer using Microsoft Word, OpenOffice, LibreOffice or similar, you may use the older NeurIPS style files available at https://neurips.cc/Conferences/2015/PaperInformation/StyleFiles.

When you submit your report for the first time, you should *not* include your own names in the report (since it will be reviewed anonymously by your colleagues)! This is the default status in the LaTeX files. If you later submit a revised report you should, however, include your names. In LaTeX this is achieved by the `final` option, i.e., use `\usepackage[final]{neurips_2020}`. In the .docx or .rtf format you have to do the changes manually.

The LaTeX template has line numbers in its draft mode. You should not remove these numbers. They can be useful for your reviewers when they want to refer to a specific part of your report (e.g., "the equation on line 54").

4.3 Specifying the number of group members

Since the report is submitted anonymously, you need to specify the number of group members in your group manually. Do this by adding "Number of group members: K", where K is the number of group members in your group, as the last sentence in the *abstract*.

4.4 How to submit your report

You submit your report using Studium. All submitted reports will automatically be checked for plagiarism using Urkund (http://www.urkund.com).

4.5 Contribution statements

As a separate document (a simple text file is enough), you should clearly state the contributions of each group member, clarifying who contributed to which part, etc. This document has to be submitted on Studium, separately from the report via a different submission page.

4.6 Peer review of the reports

Your report will be reviewed by students from other groups. Each student will also receive the report of another group, which you have to review. This means that the peer review is done individually and each group will receive multiple reviews. As a peer reviewer, you are expected to comment on the following aspects of the report:

(I) The subset of methods chosen to explore is sufficiently large (methods from at least as many 'families' as there is group members, plus the method of always predicting LIKE).

(II) All tasks (a)-(c) from Section 2.2 are made for each method.

(III) Make an assessment of the technical quality of the proposed solution. Have the considered methods been used in a relevant way to address the problem at hand? Are there any flaws in the reasoning and/or motivations used?

(IV) The report includes everything required from Section 4.1.

(V) The reflection task is discussed seriously

(VI) The quality of the language in the report is satisfactory.

(VII) The report follows the format requirements (correct template, page limitation, etc.).

The review process is "double blind", meaning that both the project report and the review are anonymous. The review is done by filling out scores in the rubric of the mini-project on Studium and by adding text comments in that rubric. Please follow the instructions on Studium for how to fill in and submit your review.

Of course, you should use a polite and constructive language in your review. (Tip: *think about how you would assess your own report before you submit it!*)

After the review deadline, each group will get the reviews on their report from other students, as well as comments and a grade (pass/revise/fail) from the teaching assistants.

5 Grading

The **first submission** of the report will be graded with one out of four possible grades:

- Fail, if the deadline is missed or the report is far from meeting the criteria. No revision is possible until next time the course is given.
- Revise, if there are only minor issues. A revised version should be handed in before the revision deadline.
- Pass, if the report fulfills *all* criteria (including the reflection task).
- Pass with gold star, if the report fulfills all criteria, and *in addition*
  - is written such that a thorough understanding of the methods is conveyed *and*
  - has a technical contribution beyond the minimum requirements.

  This will earn you a higher grade for the report and possibly a higher course grade. See the course web page on Studium for details.

If applicable, the **second submission** of the report (after revision) will be graded with one out of two possible grades:

- Fail, if the deadline is missed or the revised report still does not meet the criteria. No more revision is possible until next time the course is given.
- Pass, if the report fulfills all criteria.

Please not that sub-standard reports will not be given the chance to be revised, and gold stars are handed out only at the first submission.

6 Tip for getting started

Most background (methods, theory, Python commands, etc.) you need to complete the mini project have been/will be covered at lecture 3, 4, 5, 6 and 7 and problem solving sessions 3, 4, 5, 6, 7, 8 and 9. If you choose not to try boosting, you do not need lecture 7 and problem solving session 9, etc. *To get started, you only need the material from lecture 3 and 4.*

7 Deadlines

Check Studium for deadlines and other important dates.

Good luck!

References

Bertin-Mahieux, Thierry, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere (2011). "The million song dataset". In: *Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR)*.

DeBrusk, Chris (Mar. 2018). *The Risk of Machine-Learning Bias (and How to Prevent It)*. URL: https://sloanreview.mit.edu/article/the-risk-of-machine-learning-bias-and-how-to-prevent-it/.

Dieleman, Sander (2016). "Keynote: Deep learning for audio-based music recommendation". In: *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*.

Fu, Zhouyu, Guojun Lu, Kai Ming Ting, and Dengsheng Zhang (2011). "A survey of audio-based music classification and annotation". In: *IEEE Transactions on Multimedia* 13.2.

IEEE (2018). *IEEE Code of Ethics*. URL: https://www.ieee.org/about/corporate/governance/p7-8.html.

Jacobson, Kurt, Vidhya Murali, Edward Newett, Brian Whitman, and Romain Yon (2016). "Music Personalization at Spotify". In: *Proceedings of the 10th ACM Conference on Recommender Systems*.

Kaminskas, Marius and Francesco Ricci (2012). "Contextual music information retrieval and recommendation: state of the art and challenges". In: *Computer Science Review* 6.

O'Neil, Cathy (2016). *Weapons of Math Destruction: How Big Data Increases Inequality and Threatens Democracy*. Crown Publishing Group.

Oord, Aaron van den, Sander Dieleman, and Benjamin Schrauwen (2013). "Deep content-based music recommendation". In: *Advances in Neural Information Processing Systems 26 (NIPS)*.

ProPublica (May 2016). *Machine Bias – There's software used across the country to predict future criminals. And it's biased against blacks.* URL: https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing.

Sveriges Ingenjörer (2018). *Code of Honour*. URL: https://arkiv.sverigesingenjorer.se/Global/Dokumentbibliotek/Hederskodex%20ENG%20till%20webb.pdf.

Tegmark, Max et al. (July 2015). *Autonomous Weapons: an Open Letter from AI & Robotics Researchers*. URL: https://futureoflife.org/open-letter-autonomous-weapons/.

USA Today (July 2015). *Google Photos labeled black people 'gorillas'*. URL: https://eu.usatoday.com/story/tech/2015/07/01/google-apologizes-after-photos-identify-black-people-as-gorillas/29567465/.