Dealing with overconfidence and bias in low-cost evaluation of audio music similarity

Tim Buckers

Dealing with overconfidence and bias in low-cost evaluation of audio music similarity

by

Tim Buckers

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Friday March 5, 2021 at 14:00 AM.

Student number:	4369459	
Project duration:	February 1, 2020 – March 5, 2021	
Thesis committee:	Prof. Alan Hanjalic,	TU Delft, Responsible Full Professor
	Dr. Julián Urbano,	TU Delft, Daily Supervisor
	Dr. Joana Gonçalves,	TU Delft

Acknowledgements

This work is the result of 13 months research on evaluation of audio music similarity. This thesis is the last part of my 6 years at the Delft University of Technology. Consisting of my Bachelor and Master in computer science. Throughout these years I was always interested in the computer science perspective on questions in our daily lives. In particular the situations where most people just know the answer, but are not able to clearly breakdown the substantiation. Let alone, have an understanding of how a computer would solve this. The topic of audio music similarity is a beautiful example of this. It is for humans easy to hear if songs sound similar, but it becomes difficult when asked to explain or substantiate the decision-making. A step further is defining precise rules that a machine could interpret, this is significantly harder. Personally, I find this process interesting because when one succeeds in translating this to these precise rules we often obtain additional insights. This is one of the aspects that makes the topic of this working interesting. Music is often considered as a very subjective and emotional subject which creates a nice contrast with the scientific approach of evaluation.

Before diving in, I would first like to thank everybody that was involved in this work or helped in any way. Especially considering the unusual circumstance we were all in due to COVID-19. Only a few weeks after my start the first consequences became visible. The third meeting with my supervisor Julián Urbano was the last non-digital meeting. All the following months I was working every day from home and all communication was digital. Which is far from ideal when trying to efficiently discuss the technical details of a project this size. Besides the direct personal impact, the needed digitization of all work was also a time-consuming challenge for my supervisor. Nevertheless, the thesis continued and the digital communication was very useful. Therefore my gratitude for all your help. Besides, the technical help I also want to express my appreciation to everybody in my social environment that positively influenced my mental health and contributed to my motivation. My friends, including fellow students, provided the highly valued distraction when needed. They truly made my Delft experience lively and memorable. Naturally, I would like to thank my parents and brothers for their support. They created an ideal environment that allowed me to accomplish all this. A special thanks to Eline who supported me every day, even after constantly hearing oddly specific details of this thesis. Details that were arguably not always the most interesting topics of conversation. A project of this size contains regular setbacks and also included some scheduling conflicts with my social life. This all has indirect consequences for the people around you which can be demanding. The level of support and understanding from everyone around me was heartwarming. That is why I would like to end this preface by expressing my appreciation for this in particular, thank you all.

Tim Buckers
Delft, March 2021

Contents

List of Figures iii

List of Tables vii

1 Introduction 1
- 1.1 Application . . . 1
- 1.2 Evaluation of AMS systems . . . 2
- 1.3 Low-cost evaluation model . . . 3
  - 1.3.1 Overconfidence . . . 4
  - 1.3.2 Overestimation . . . 4
  - 1.3.3 Contributions . . . 5

2 State-of-the-art in AMS evaluation 7
- 2.1 Evaluation of information retrieval systems . . . 7
  - 2.1.1 Metrics . . . 8
- 2.2 The low-cost evaluation model . . . 9
  - 2.2.1 Predicting the judgment label . . . 10
  - 2.2.2 Performance estimation . . . 11
  - 2.2.3 Selection of songs to judge . . . 13
- 2.3 Problems in the low-cost evaluation model . . . 14
  - 2.3.1 Overconfidence . . . 14
  - 2.3.2 Overestimation . . . 14

3 Overconfidence in the performance scores 15
- 3.1 Quantifying overconfidence . . . 15
  - 3.1.1 Overconfidence in performance prediction . . . 15
  - 3.1.2 Overconfidence in relevance prediction . . . 16
- 3.2 Reducing overconfidence via more reliable features . . . 19
  - 3.2.1 Inclusion of own judgment label . . . 21
  - 3.2.2 Uniqueness of attribute . . . 21
  - 3.2.3 Number of systems . . . 22
  - 3.2.4 Number of judged documents . . . 23
- 3.3 Reducing overconfidence via the judgment selection method . . . 25
- 3.4 Reducing overconfidence via better prediction algorithms . . . 27
  - 3.4.1 Comparing performance of different prediction algorithms . . . 28
- 3.5 Results . . . 29
  - 3.5.1 Probabilistic predictions . . . 31
  - 3.5.2 Absolute performance estimation . . . 32
  - 3.5.3 Relative performance estimation . . . 32
  - 3.5.4 Rank estimation . . . 33
- 3.6 Summary . . . 35

4 Overestimation of absolute performance metrics 39
- 4.1 Quantifying the overestimation . . . 39
- 4.2 Underestimation of optimal rank . . . 39
- 4.3 Solutions to reduce the overestimation . . . 40
  - 4.3.1 Label of max probability . . . 41
  - 4.3.2 Weighted sampling . . . 43
- 4.4 Summary . . . 44

5 Conclusion and future work 47
5.1 Conclusion 47
5.2 Future Work. 48

Bibliography 51

List of Figures

1.1 Conceptual overview of how an AMS system is evaluated. In the MIREX setting (black) by human experts and by the Low-cost evaluation model (green). The black path shows the high-level steps taken to obtain the performance score solely based on human judgments. In green, the low-cost model is illustrated. Based on meta-data and the needed human judgments the model makes similarity predictions. These predictions are then translated to an estimation of the performance score. The human judgment components are red to emphasize that this is the most resource-expensive part of the evaluation process. 2
1.2 Predicted performance score plotted against the actual value. All points from one color represent all the different systems in the years 2007, 2009, 2010 and 2011. The predictions are made by the output-based model. A diagonal line is shown because all estimations that match the actual score lie on this line. 5

2.1 An example query for music retrieval: 1 song is the query document, the retrieval system retrieves 5 songs which it expects to be relevant. Then the labels indicate how similar they are to the query song. The 5 judgment labels are used to calculate the performance score. In this particular example it is simply taking the average label of the 5 retrieved documents. 8
2.2 No overlap in possible performance score 13
2.3 Overlap in possible performance score 13

3.1 The reliability of the original judgment-based model. Calculated on different amount of predictions sampled from the 2007, 2009, 2010 and 2011 dataset. The different colors show distributions when the predictions are rounded to different amounts of distinct probability values. . . . 20
3.2 The distribution of group sizes for the predictions made for 2007 with the judgment-based model. The dataset 2007 has 5998 retrieved songs that can be predicted. The different colors show distributions when the predictions are rounded to different amount of distinct probability values. 20
3.3 The reliability of the original judgment-based model. Calculated on different amount of predictions sampled from the 2007, 2009, 2010 and 2011 dataset. The predictions are rounded to 10 distinct values. The different colors show the minimal group size the groups are taken into account. 20
3.4 Average artist feature calculation. In green the original way of calculation. In blue the newly proposed method called Excluding Self. 21
3.5 The reliability of the output-based predictions for the 4 different years. The colors indicate the different models used. 24
3.6 The influence of judgment ratio on reliability. Comparing the original model, exclude self and exclude self & feature count. Only judgment-based predictions are taken into account. 25
3.7 The distribution of songs over 6 genres. Blue shows the distribution of all 2007 songs that belong to those 6 genres. Red shows the distribution of the same genres, but for the first 10% judged songs. 25
3.8 Estimation error of average system performance with a subset of the jugments. 26
3.9 The distribution of difference in average performance between all system pairs. 26
3.10 RPS error of the judgment-based predictions. The horizontal axis indicates the number of judgments used in the feature calculation for the average system. Note that the horizontal axis is not linear. 26
3.11 RMSE of the different prediction models. The predictions are made with the output-based features. The error is the average of the years 2007, 2009, 2010 and 2011. 27
3.12 Reliability error of the different prediction models. The predictions are made with the output-based features. The error is the average of the years 2007, 2009, 2010 and 2011. 27

3.13 RMSE of the different prediction models. The predictions are made with the judgment-based features and all judgments available. The error is the average of the years 2007, 2009, 2010 and 2011. 28
3.14 Reliability error of the different prediction models. The predictions are made with the judgment-based features and all judgments available. The error is the average of the years 2007, 2009, 2010 and 2011. 28
3.15 Prediction variance compared to variance in the error. For the different models and the output-based predictions. 29
3.16 Prediction variance compared to variance in the error. For the different models and the output-based predictions. This time the predictions are calibrated based on variance. 29
3.17 The reliability error of the ctree similarity prediction on the training set itself. Plotted against the Min Criterion value. 30
3.18 The RMSE error of the ctree similarity prediction on the training set itself. Plotted against the Min Criterion value. 30
3.19 The reliability error of the ctree similarity prediction on the training set itself. Plotted against the Max Tree Depth. 30
3.20 The RMSE error of the ctree similarity prediction on the training set itself. Plotted against the Max Tree Depth. 30
3.21 The RPS score of the judgment-based predictions comparing the logistic regression against ctree. The RPS score is the average of the 4 years combined. On the horizontal axis, the judgment ratio ranges from 0.05 to 1. 30
3.22 RMSE, Reliability and RPS error metrics. The output-based original model in red is compared against the proposed system-based model in blue. 31
3.23 RMSE, Reliability and RPS error metrics of judgment-based predictions in a setting with all judgments available. 32
3.24 The accuracy and confidence of judgment-based predictions. In case the judgment-based features are not available a uniform distribution is assumed as prediction. 35
3.25 The accuracy and confidence of judgment-based predictions. In case the judgment-based features are not available the output-based model is used to predict. 35

4.1 Estimated vs. actual performance scores of the teams in MIREX 2007, 2009, 2010 and 2011. On the left side the predictions are made with the improved output-based model. On the right side the predictions are made with the improved judgment-based model with all judgments available. 40
4.2 Estimated vs. actual performance scores in MIREX 2007, 2009, 2010 and 2011. Predictions are made with output-based model. Left Broad scale is used and on the right side the Fine scale. . . 41
4.3 Red is the distribution of the actual judgment labels in the years 2007, 2009, 2010 and 2011 combined. On the left side the broad scale is used thus the possible values are 0, 0.5 and 1.0. On the right side the fine scale is used thus the possible values range from 0.0 to 1.0 with a step size of 0.1. The blue bars in both plots are again the 4 years combined, but instead of the judgment label the label with the maximum probability is used from the output-based predictions. 42
4.4 Estimated vs. actual optimal query scores in MIREX 2007, 2009, 2010 and 2011. Predictions are based on the max label of the output-based model predictions. Left the Broad scale is used, on the right the Fine scale is used. 42
4.5 Estimated vs. actual optimal query scores in MIREX 2007, 2009, 2010 and 2011. Predictions made with the output-based model. For both NDCG and RBP indicates with different symbols. In blue the estimations are based on the expected value and in red they are based on the label with the highest probability. The saturation of the color indicates which scale is used. 43
4.6 Estimated vs. actual optimal query scores in MIREX 2007, 2009, 2010 and 2011. Predictions are based on the proposed sampling method of the output-based model predictions. Left the Broad scale is used, on the right the Fine scale is used. 44
4.7 Estimated vs. actual query scores in MIREX 2007, 2009, 2010 and 2011. Predictions made with the output-based model. For both NDCG and RBP indicates with different symbols. In blue the estimations are based on the expected value and in red the proposed sampling method is used. The saturation of the color indicates which scale is used. 45

4.8 The estimation of the metrics NDCG and RBP compared to the actual values. On the left the expected value is used to estimate. On the right the proposed sampling method is used. The colors indicate the judgment ratio. 45
4.9 The estimation of the normalized metrics compared to the actual value. The colors indicate the judgment ratio used. The estimations are made with the available judgments, judgment-based predictions and output-based predictions. 46

List of Tables

1.1 Relative performance predictions grouped on confidence into 4 bins. The predictions are made with 0% of the judgments available. The lower and upper bound indicates the lowest and highest confidence of the bin. The confidence and accuracy columns are the average of the bin. The last column is the ratio of predictions that fall into this bin. 4

2.1 The general properties of the 4 different MIREX datasets. 11

3.1 The ratio of actual performance scores that fall in a confidence interval of the model estimation. The model estimates the absolute performanceof all systems in the 4 years of the MIREX dataset. Both the output-based an judgment-based models are used. 17
3.2 The ratio of actual performance scores that fall in a confidence interval of the model estimation. The model estimates the relative performance between all systems pairs in the 4 years of the MIREX dataset. Both the output-based an judgment-based models are used. 18
3.3 The error of judgment-based predictions for the 4 years. Comparing the original model with the excluding self adjustment. The numbers that are bold are better in terms of error to clearly illustrated the difference. 22
3.4 Some properties of the 4 different datasets. The columns fART, fGEN and fSYS show the maximal value this feature in the dataset. The columns RPS, RMSE, Variance and Reliability are all calculated of the output-based predictions for that specific year. 22
3.5 The ratio of actual performance scores that fall in a confidence interval of the model estimation. The model estimates the absolute performance of all systems in the 4 years of the MIREX dataset. Both the output-based and judgment-based models are used. The better numbers are marked bold. 33
3.6 The ratio of actual performance scores that fall in a confidence interval of the model estimation. The model estimates the relative performance between all systems pairs in the 4 years of the MIREX dataset. Both the output-based and judgment-based models are used. The better numbers are marked bold. 34
3.7 Relative performance predictions grouped on confidence into 4 bins. The predictions are made with the output-based model. The judgment values themselves are not used in making the prediction. The lower and upper bound indicate the lowest and highest confidence of the bin. The confidence and accuracy columns are the average of the bin. The last column is the ratio of predictions that fall into this bin. The closer matching confidence and accuracy pairs are marked bold. 36
3.8 Relative performance predictions grouped on confidence into 4 bins. The predictions are made with the judgment-based model and 100% of the judgments available. The judgment values themselves are not used in making the prediction. The lower and upper bound indicate the lowest and highest confidence of the bin. The confidence and accuracy columns are the average of the bin. The last column is the ratio of predictions that fall into this bin. The closer matching confidence and accuracy pairs are marked bold. 37

List of Symbols

In general, uppercase italic letters (e.g. X, Y) denote random variables and lowercase italic letters (e.g. x, y) denote scalars. Lowercase bold letters (e.g. $\vec{x}$, $\vec{y}$) denote vectors, and Greek letters (e.g. μ, σ) represent parameters. Calligraphic letters (e.g. $\mathcal{Q}$, $\mathcal{S}$) represent sets, and sans serif letters (e.g. A, B) represent retrieval systems. When necessary though, this notation is explicitly ignored or simplified.

$\mathcal{D}, n_\mathcal{D}$	A corpus of documents and its size.
$\mathcal{Q}, n_\mathcal{Q}$	A set of queries and its size.
$\mathcal{R}, n_\mathcal{R}$	A set of relevance judgments and its size.
$\mathcal{S}, n_\mathcal{S}$	A set of retrieval systems and its size.
$\mathcal{L}, n_\mathcal{L}$	A set of possible relevance levels and its size; $\mathcal{L} := \{0, 1, \dots, n_\mathcal{L} - 1\}$
r_d	The relevance judgment for document d; $r_d \in \mathcal{L}$.
R_d	Random variable representing the relevance of document d.
$\mathcal{R}^\ell$	The subset of documents whose relevance is ℓ; $\mathcal{R}^\ell = \left\{d \in \mathcal{D} : r_d = \ell\right\}$.
A	The ranking of documents by system A; $\mathsf{A} := \{\mathsf{A}_1, \dots, \mathsf{A}_{n_\mathcal{D}}\}$.
A_i	The document retrieved at rank i by system A; $\mathsf{A}_i \in \mathcal{D}$.
A_d^{-1}	The rank at which document d is retrieved by system A; $\mathsf{A}_{\mathsf{A}_d^{-1}} = d$.
I	An ideal ranking of documents; $\mathsf{I} := \left\langle \mathsf{I}_1, \dots, \mathsf{I}_{n_\mathcal{D}} : \forall i : r_{\mathsf{I}_i} \geq r_{\mathsf{I}_{i+1}} \right\rangle$.
Λ	An arbitrary effectiveness measure or a distribution of effectiveness scores.
λ	The effectiveness score according to some arbitrary measure Λ.
$\lambda_{q,\mathsf{A}}$	The effectiveness score of system A for query q according to measure Λ.
$\overline{\lambda}_{\mathcal{Q},\mathsf{A}}$	The average score of system A with query set $\mathcal{Q}$ according to measure Λ.
$\Delta\lambda_{q,\mathsf{AB}}$	The difference between systems A and B for query q according to measure Λ.
$\overline{\Delta\lambda}_{\mathcal{Q},\mathsf{AB}}$	The average difference between systems A and B with query set $\mathcal{Q}$ according to measure Λ.
$\mathrm{E}[X]$	The expectation of random variable X.
$\mathrm{Var}[X]$	The variance of random variable X.
$\vec{y}_k$	Vector of the cumulative prediction probabilities.
$\vec{o}_k$	Vector representing the cumulative observations.

1 Introduction

This chapter is an introduction to evaluation of audio music similarity. Starting with explaining the field of audio music similarity and why it is useful to have computer systems that can calculate this similarity. To make it more concrete some example applications of music similarity are given. Then section 1.2 briefly goes into the evaluation component. Proper evaluation of these systems is costly. Then section 1.3 introduces a more efficient evaluation method. However, this method is not perfect and therefore this thesis addresses problems of this method, two in particular. A summary of the proposed improvements in this thesis is listed in the last section 1.3.3.

1.1. Application

Nowadays music is often stored digitally. Creating your own playlist is becoming more accessible for people all over the world. This progression comes along with users who want to search in this library of music[22]. Searching for songs is mostly done by searching on title-, album- or artist-name. However, this limits the searching process. How to find a song of which a user does not know the name or artist? How to find songs that perfectly fit your playlist? A possible solution for this is querying by song. Using a song as input and getting similar songs as a result. Or in some cases using a list of songs as input such as an entire playlist.

To obtain matching songs audio music similarity (AMS) plays an important role. This is simply put how musically similar songs are when only considering the audio signal. Besides querying songs AMS is also used in playlist generation, music recommendation systems, and visualization of music collections. AMS is one component of the more general concept music similarity. This is the quantitative measure to which degree two songs are similar or different. Music similarity consists of a wide range of components, there are more contributing factors besides the audio content itself. Factors such as artist, language and marketing are used to convey the song. Music similarity is also subjective and user-dependent. The opinion, mood and context of the user heavily influence it. Not only taking the audio signal into account is proven to be beneficial[21].

Some methods try to calculate music similarity as a whole, without breaking it down into the different components. One of the most successful approaches to building recommender systems is collaborative filtering[35]. This is also a commonly used method to obtain music similarity. The similarity is based on the behavior of other users. For example, if many users listen to song A and song B it is likely that they are similar to some degree. Another option is looking at playlists of users. Songs that are added to the same playlist are again more likely to be similar. Collaborative filtering has the advantage of capturing multiple components at once. Besides the similarity on an audio level it also takes contextual factors into account, such as personal preference and cultural consideration[1]. However, there are limitations and problems. The cold-start problem is problematic for new songs. This is the lack of user information when a song is just added. Nobody has ever listened to this song nor is it in any playlist[22]. When there is enough data available methods that consider these contextual factors and predict music similarity as a whole are suitable. However, the lack of enough reliable data is exactly one of the difficulties that need to be dealt with. In general for new, undiscovered or unique artists, an audio-based technique may be more beneficial[1].

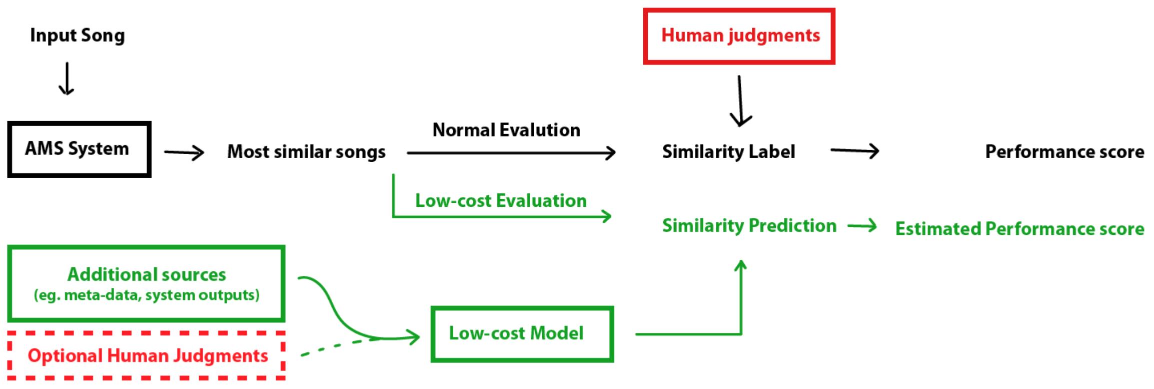

Figure 1.1: Conceptual overview of how an AMS system is evaluated. In the MIREX setting (black) by human experts and by the Low-cost evaluation model (green). The black path shows the high-level steps taken to obtain the performance score solely based on human judgments. In green, the low-cost model is illustrated. Based on meta-data and the needed human judgments the model makes similarity predictions. These predictions are then translated to an estimation of the performance score. The human judgment components are red to emphasize that this is the most resource-expensive part of the evaluation process.

Calculating music similarity is not a solved problem; this is still an ongoing research topic. Breaking music similarity down into different factors is beneficial for a better understanding. This is one of the reasons why there is a need for audio music similarity in research. It is often a significant component in music systems such as recommendation and search. Obviously, calculating audio music similarity by surveying users is not feasible for large quantities[22]. Popular digital music libraries contain millions of songs. Asking people to judge the similarity of most song pairs is simply not feasible. In addition this process never stops. Every newly added song needs to be labeled. Automatic measurement of the similarity between two songs based only on an analysis of their audio content is valuable[1]. There are many different algorithms and automated methods which try to calculate this similarity. We call these AMS systems. Such a system looks at attributes of the audio signal to compare songs. Example audio features are melody, volume, timbre and rhythm. Based on these kinds of features the algorithm will output the expected audio music similarity of songs. Creating an audio-based music similarity algorithm is not trivial and a still an ongoing research topic. To accomplish this, evaluation is needed. This is to verify the output a system produces. Verification is needed to quantify how accurate and precise the system is. By having a proper evaluation method, systems can be improved and compared.

1.2. Evaluation of AMS systems

Ideally, evaluation would be done with a large dataset of songs where all possible song pairs have a label indicating how similar they are. Judging song pairs in terms of audio music similarity is costly. Multiple users have to listen to at least a fragment of both songs and judge how similar they are. A dataset of only 100 songs already has 4950 unique song pairs. Fortunately, there are some substantial efforts made towards creating evaluation data sets. For example by Music Information Retrieval Evaluation eXchange (MIREX), which is a community-based formal evaluation framework. The first edition was in 2005, organized by IMIRSEL (International Music IR Systems Evaluation Laboratory) [12]. They create test collections on a yearly basis. Including some audio music similarity test collections[20]. They are not judging all possible song pairs. In each dataset, they take a fixed number of systems to evaluate. For 100 songs each system will try to find the 5 best matching songs. In the case of 2 systems, this gives at most $100 * 5 * 2 = 1000$ unique song pairs. Based on the labels of these song pairs a performance score for the AMS system can be calculated. Figure 1.1 shows the general steps to obtain the performance score of an AMS retrieval system for 1 song.

The MIREX dataset is more efficient than simply judging all song pairs. However, only a fixed amount of systems are evaluated. Due to music copyright, submissions must send their algorithm and let MIREX run them. Which is computationally and time expensive[12]. This means that the evaluation of a new or improved algorithm is not straightforward with these kinds of datasets. System evaluation is also used for verifying if a change to the algorithm is beneficial or not. This is similar to comparing different systems. Even small changes to a system can make the algorithm retrieve other documents. When there are new documents retrieved these also need annotations. Due to the high cost of judgment, the size of these judged datasets is often limited. This can impact the reliability of the evaluation and how repeatable the results are. For MIREX in particular we already see that the differences between some of the systems are not significant to conclude which system is actually better performing [10].

Simply put evaluation data is costly and often because of this, there is a lack of enough evaluation data. Large public music datasets such as the Million Song Dataset[2] often lack evaluation labels due to the high cost. Important to consider is that a test collection is created for a particular task in a chosen context. For example, depending on the type of music selected some algorithms may outperform others. Also, new songs and new genres are created each year. Perhaps a specially trained algorithm for audio music similarity for classical music is outperforming a more general algorithm in the specific genre of classical music. This means that not one test collection is going to be applicable for all problems and research in the audio music information retrieval field. The music information retrieval community has long recognized the need for a comprehensive evaluation paradigm[9][40]. The literature of Music Information Retrieval research is primarily one of development, not evaluation[11]. The International Society for Music Information Retrieval (ISMIR) is a society focused on processing, searching, organizing and accessing music-related data. This society also expressed its concerns. “The ISMIR community is really concerned about how we evaluate our systems. Individuals are often frustrated because the current evaluation practices we follow do not fully allow us to work and improve as much as we wish.”[30]

The evaluation done by MIREX is direct, this is called explicit evaluation. Explicit evaluation is produced by controlled processes. Implicit evaluation, on the other hand, is indirect and sometimes even without the intention or awareness of the user. To illustrate implicit evaluation, we use the case of search engines. When a user clicks on a document or site this probably indicates that the user thinks this is relevant. By observing this user behavior, we can implicitly obtain feedback useful for the evaluation of systems[28]. Implicit evaluation can often be achieved in a cost and time-effective manner[33]. The retrieval of this information can often be automated. However, implicit evaluation is often less accurate compared to explicit evaluation. When making the evaluation less costly, a lot more values can be evaluated. If the costs go down enough, compared to the loss in accuracy, the trade-off is beneficial. Often this kind of approach is useful in a setting where a lot of data is available. In this case, enough implicit feedback can be gathered in the exact setting that is needed. Another limitation of implicit evaluation is the lack of negative feedback. This also applies to the field of music retrieval. It is much easier to capture feedback from users that indicates which songs are similar, compared to dissimilar songs. Implicit feedback is also inherently noisy because the true motives of the user are guessed. In addition, implicit feedback is often relative and does not include absolute numbers [18]. This makes implicit evaluation not always ideal for the scientific community, where reliable results are required. A more general methodology, that still reduces the costs of evaluation, would therefore be beneficial.

1.3. Low-cost evaluation model

Evaluation by test collection as performed by MIREX is costly. When aiming for a more efficient evaluation method this should not reduce validity and reliability. That is why this thesis will focus on the low-cost evaluation model[39]. This model uses the same test collection structure but does not require all judgment labels. Often it needs less than 5% of the judgment labels to evaluate which system is better. The model predicts the labels based on meta-data and the output of other systems. Three examples to illustrate how meta-data and the other systems can predict the similarity:

- Genre: two songs from the same genre are probably more similar than two random songs on average.
- Artist: the same holds as genre two songs are probably more similar when they are from the same artist.
- Retrieval results of other systems: Given an input song how many systems do agree on which songs are the most similar. The more systems retrieve the same song the higher the probability that this song is actually similar.

Based on all this additional information the model can make rough predictions on how similar song pairs are. To make the predictions more accurate some judgment labels can be added. The few judgments are then combined with meta-data and system output. To make more precise predictions. Take for example a query where one retrieved song is labeled. This song is from a classical genre. This song is labeled as perfectly similar. Two other retrieved songs are also of this genre. While the remaining songs are of the genre hip hop. The model can now use this information to make a more accurate prediction. This probably means that the two classical songs will be predicted more similarly compared to the hip hop songs.

The goal of the evaluation is measuring how well an AMS system is performing. This performance can be absolute or relative to another system. Sometimes when comparing systems it is enough to know which system is better, without quantifying the difference. For each specific use-case, the low-cost evaluation model proposes a method of calculation. Despite the differences in the calculation, the foundation is always the prediction of similarity labels. These predictions contain a degree of error. This error propagates to the final estimation of the performance score. These estimations do not only contain an expected value but the confidence of this prediction as well. The confidence of the prediction indicates how accurate this prediction is. Depending on the application, a different degree of accuracy is required. If the confidence is not high enough, judgments can be added to the model. In general, by judging more song pairs the model will predict with more certainty. As explained the judging of songs is costly. Once the correct confidence is obtained, no more time has to be spent on judging song pairs. Often a low number of judgments is already enough to obtain high confidence, this makes the model low cost.

Figure 1.1 shows a conceptual overview of the steps in the evaluation process. At the end of the evaluation pipeline, there are many different metrics available to calculate the performance score. Depending on the context and application, different metrics can be preferred. In a general evaluation model supporting these different metrics is beneficial. The low-cost evaluation model supports a wide range of different metrics.

1.3.1. Overconfidence

When predicting the rank of two systems with a confidence of 95%, the predicted order should be correct 95% of the time. However, the predicted confidence does not always match the observed accuracy. In case of a mismatch, the confidence is observed to be higher than the accuracy. This overconfidence is observed in the rank predictions of the original model. Table 1.1 contains the rank predictions which system is the best for all system pairs. The model makes all predictions with 0% of the judgments available. Predictions with similar confidence are grouped, the table contains 4 groups. The most confident group contains all predictions with at least a confidence of 0.99. The accuracy is defined as the ratio of correct predicted rankings. The bottom row indicates the total, this shows that for all the 341 predictions 90% of the time the model correctly predicted the system pairs. However, the expected accuracy is 92%, this is the average confidence of all predictions. The confidence of each group should match the obtained accuracy to reliably interpret the predictions. However, the overall predicted confidence is higher than the observed accuracy. This mismatch between confidence and accuracy is addressed in Chapter 3. The chapter does not exclusively focus on the ranked estimations. Overconfidence is also observed in the absolute and relative estimation of the performance score.

		CG@5		
Lower bound	Upper bound	Confidence	Accuracy	In bin
0.49	0.65	0.57	0.71	18 (5%)
0.65	0.90	0.80	0.69	28 (8%)
0.90	0.99	0.95	0.91	44 (13%)
0.99	1.00	1.00	1.00	251 (74%)
Total		0.92	0.90	341 (100%)

Table 1.1: Relative performance predictions grouped on confidence into 4 bins. The predictions are made with 0% of the judgments available. The lower and upper bound indicates the lowest and highest confidence of the bin. The confidence and accuracy columns are the average of the bin. The last column is the ratio of predictions that fall into this bin.

1.3.2. Overestimation

In Figure 1.2 different metrics to compute the absolute performance score are shown. In the perfect case all points would perfectly lay on the diagonal, this means that the estimated performance score is exactly equal to the actual score. As explained this method is estimating the scores and therefore contains some degree of error. This leads to the expectation that most points should lay around the diagonal line. This is partially observed, but there is a clear difference between metrics. For Cumulative Gain (CG) and Discounted Cumu-

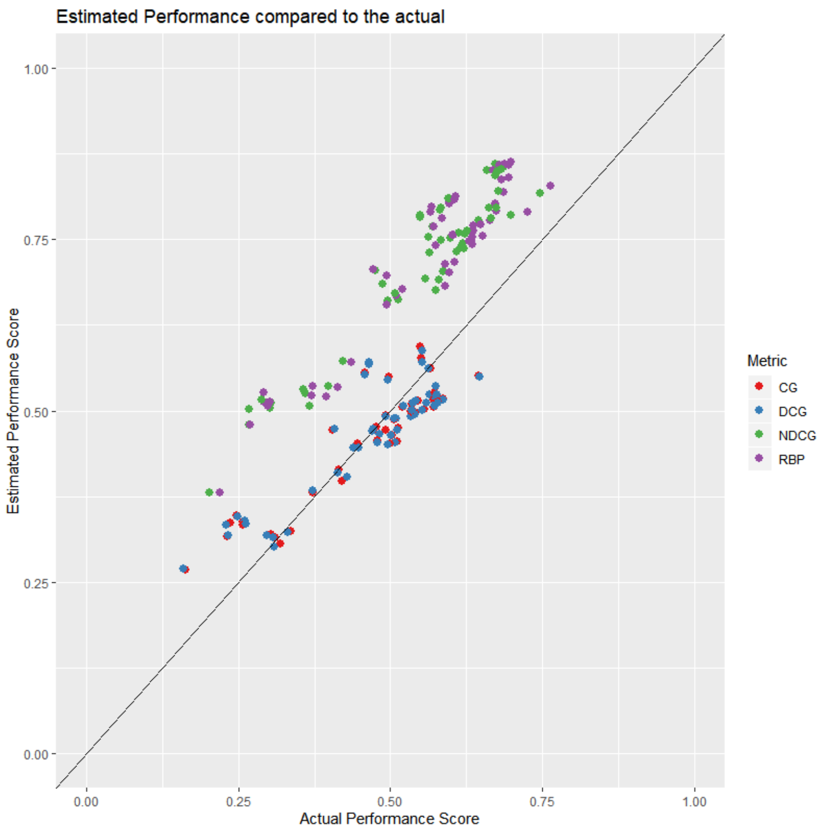

Figure 1.2: Predicted performance score plotted against the actual value. All points from one color represent all the different systems in the years 2007, 2009, 2010 and 2011. The predictions are made by the output-based model. A diagonal line is shown because all estimations that match the actual score lie on this line.

lative Gain (DCG) the points are indeed roughly located around the diagonal. In contrast to the Normalized Discounted Cumulative Gain (NDCG) and Ranked-Biased Precision (RBP) where almost all points are above the line. This means the estimated score is higher than the actual value. This systematical overestimating of the two metrics is a significant problem. In chapter 4 this problem is explained and solutions are proposed.

1.3.3. Contributions

In Chapter 3 the overconfidence is addressed. The similarity predictions are the starting point in breaking down potential causes. Error metrics are introduced to quantify how reliable the predictions are. This enables measuring to what extent the predicted probabilities actually match the observed values. Both output-based and judgment-based predictions do not perfectly match the observed values. To improve the reliability possible causes are investigated that could make the model unreliable. Three improvements are proposed:

1. A fundamental problem in the judgment-based predictions is found and solved. This is a mismatch between the features calculated to train the model and the features used when making predictions. The features used should always represent the same information. Independent of the song of interest or the dataset used. This is covered in section 3.2.1.

2. The number of values used in feature calculation affects how stable this features is. Taking this into account is beneficial for the reliability. This is covered in section 3.2.4.

3. Calculating features on a system level instead of on a song level increases reliability as well. This limits the maximal influence, that the output from the system itself has on the feature value. This is covered in section 3.2.3.

After these adjustments, the similarity predictions of the model become more accurate and more reliable. For the absolute estimation, this leads to a better match between predicted variance and obtained error. Similar consequences, but of lesser size, are reported for the estimation of the relative performance score.

After improving on the level of similarity predictions the overestimation for NDCG and RBP remains. This is addressed in Chapter 4. When estimating similarity labels of songs, values close to the border of the range have a bias. The lowest values are overestimated and the highest values are underestimated. This is due to the uncertainty in the predictions, the expected value will never reach the lowest or highest value. Therefore the expected value is biased towards the middle. In general the bias introduced by songs at the lowest value is compensated by songs with the highest value. As a result the average estimation of the system performance is not biased. However, when a system consistently retrieves songs scoring close to one of the two extremes, bias is observed. Normalized metrics such as RBP and NDCG require the optimal rank for each query. The optimal rank includes many songs with the highest similarity label. The bias towards the middle causes underestimation of the optimal rank. When using a systematically underestimating optimal score to normalize, the actual score becomes overestimated. Two solutions are proposed:

1. Rounding the expected value to the nearest similarity label. This simple approach mitigates the problem significantly but does not solve it entirely. This is covered in section 4.3.1.

2. Instead of using the expected value a sampling method can be used. A label will be randomly selected, based on the probabilistic prediction. This means that in a prediction, where label x has a 0.9 probability, the value x will be selected 90% of the time. This solution removes the overestimation entirely. This is covered in section 4.3.2.

2
State-of-the-art in AMS evaluation

This chapter covers the state-of-the-art of evaluation of Audio Music Similarity (AMS). By first diving into the evaluation of retrieval systems and in particular systems that predict AMS. Then in section 2.1.1 different performance metrics are explained and how they differ. In section 2.2 the low-cost evaluation model is explained. This model addresses the high cost of judgment. However, this model contains biases that limit the applicability. These problems are detailed in section 2.3.

2.1. Evaluation of information retrieval systems

Music Information Retrieval(MIR) is a specific field within the information retrieval field. Literature on information retrieval in general, is to some extent applicable to the field of MIR. When we look at the more general field of information retrieval, we see that evaluation methods are dating back decades [34]. Many different evaluation methods are proposed. Three important aspects worth taking into account for appropriate evaluation are: validity, reliability and efficiency[36]. Validity is the extent to which the outcome actually succeeds in measuring what is considered as the goal. Where reliability is the consistency when repeating the experiment. Lastly we have efficiency which indicates the number of resources usually time and money[37]. Results often can be made more valid or reliable by increasing the resources used [36]. The Cranfield experiments in the 1960s lead to the evaluation format now common on information retrieval conferences[8]. In Cranfield, researchers perform experiments on test collections to compare the effectiveness of different retrieval approaches. Decreasing the cost of retrieval experiments as compared to evaluating the user satisfaction system by system. Test collections allow evaluation of multiple systems and comparison between. This is crucial for scientific progress. The efficiency gained by the use of test collection is not directly at the expense of validity and reliability. The evaluation frameworks build on the concept of test collections started in the field of text retrieval, as done in the well known TREC datasets[44]. Translating the TREC evaluation paradigm to MIR is not a process without complications and issues[9]. The main difficulty that arises where this thesis is centered around is the high costs of judgment for AMS.

A test collection consists of documents, queries and relevance judgments. In the case of AMS the three components are:

- **Query document**: the song for which the user is searching similar songs.
- **Retrieved documents**: the songs that a system considers the best matches to the query song.
- **Judgment label**: A rating indicating how good every retrieved song matches the query song. This label is a ranked value from a rating scale.

This is best illustrated with the example in Figure 2.1. The process starts with a given query song. The AMS system searches for the best matching songs in the data set. Then the top matches are returned. In this example only the top 5 is used. Of these 5 songs the relevance judgment needs to be obtained. The remaining step is converting these judgment labels into a score. This score should tell how well the current system is performing for this query and then we do the same for many other queries. This quantification is called a performance score. The next section will explain performance scores in more detail.

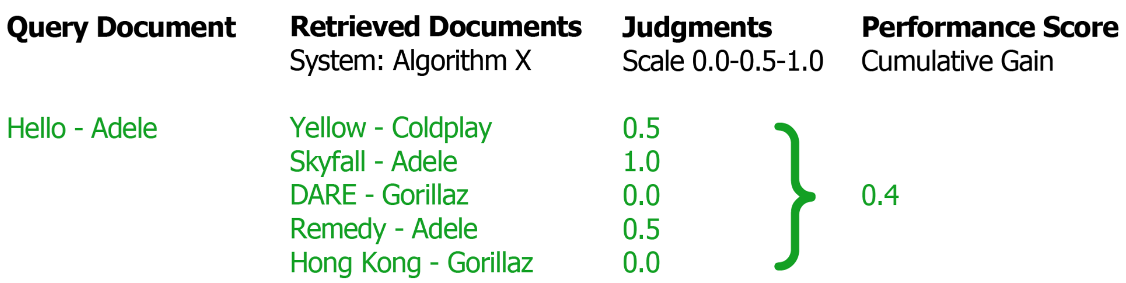

Figure 2.1: An example query for music retrieval: 1 song is the query document, the retrieval system retrieves 5 songs which it expects to be relevant. Then the labels indicate how similar they are to the query song. The 5 judgment labels are used to calculate the performance score. In this particular example it is simply taking the average label of the 5 retrieved documents.

2.1.1. Metrics

Different metrics are available to determine the performance of a system. One of the simplest types of performance scores is Cumulative Gain (CG) this is taking the average of the judgment labels. If the performance score for a certain query is 1, this means the system managed to retrieve only relevant songs. If the score is 0 this means none of the retrieved songs is similar to the query song. An example of this is illustrated in Figure 2.1. By taking the average judgment of the 5 retrieved songs a performance score of 0.4 is obtained. The goal of performance evaluation is to identify differences of user satisfaction between retrieval systems. Depending on the context different metrics may be applicable. To understand this better the following properties can be considered when selecting a performance metric:

- **Binary relevance scale or a graded relevance scale.** A binary scale simply illustrates if a document is relevant for the query or not. Where the graded scale indicates how relevant the document is. This type of evaluation can be important when there are many relevant documents. Then most systems will manage to retrieve mostly relevant documents, but the better systems retrieve highly relevant documents[42].
- **Discounted or ranked-based evaluation.** By applying a discount factor to the relevance scores the late-retrieved documents are devalued. This rewards the systems that manage to retrieve the best documents at the top[19]. This form of discounting makes a ranked-based evaluation.
- **Normalization of a metric.** Normalization is accounting for the ideal ranking, this gives the retrieval of the ideal ranking always the perfect score[19]. Even when there are not as many relevant documents as the number of documents that are retrieved. This normalization benefits cross comparisons between different datasets and queries.

Depending on context, different properties can be relevant which determines which metric is most suitable. While Cumulative Gain (CG) is fairly easy to calculate, depending on the context other metrics are often more suitable. Cumulative Gain is simply the sum of relevance of each retrieved documents. As illustrated with the example in Chapter 1 in Figure 2.1. In this example the CG is divided by the number of retrieved songs to obtain the average instead of the sum. By dividing the score by the maximal theoretical obtainable score the value is scaled to the range 0 to 1. For ease of comparison this is how all the scores will be calculated in this thesis. But first all original definitions are shown in this section to explain the starting point. When retrieving k documents CG is calculated with the following formula[19]:

$$CG@k = \sum_{i=1}^{k} r_{\mathrm{A}_i} \tag{2.1}$$

Where r_{A_i} is the judgment label of the retrieved song on position i of system A. When the rank needs to be taken into account a metric like Discounted Cumulative Gain (DCG) can be used. Which is formulate like this:

$$DCG@k = \sum_{i=1}^{k} \frac{r_{\mathsf{A}_i}}{d(i)} \tag{2.2}$$

In the original definition a logarithmic discount function is used [19]:

$$d(i) = \log_2(i+1) \tag{2.3}$$

If needed this metric can also be normalized and that is called Normalized Discounted Cumulative Gain (NDCG) [19]. NDCG is a well established metric for rank-based graded relevance[32]. NDCG can be calculated with the following formula:

$$nDCG@k = \frac{\sum_{i=1}^{k} r_{\mathsf{A}_i}/d(i)}{\sum_{i=1}^{k} r_{\mathsf{I}_i}/d(i)} \tag{2.4}$$

Here the optimal judgment label for this position is defined as r_{I_i}. But also arguments are made for a metric derived from a simple model of user behavior called Rank-Biased Precision (RBP)[26]. For the graded relevance scale this can be calculated with the following formula:

$$RBP = \frac{1-p}{n_{\mathcal{L}}-1} \sum_{i=1}^{n_{\mathcal{D}}} r_{\mathsf{A}_i} \cdot p^{i-1} \tag{2.5}$$

Simply put there is no one best metric for every setting and context. Depending on the requirements some metrics are a better fit. This substantiates the importance of supporting different evaluation metrics in evaluation models. In this work we are using the metrics CG, CG, NDCG and RBP. Because the measures CG and DCG are the most stable for relative effectiveness scores, and RBP and NDCG the most stable for absolute effectiveness scores[38].

2.2. The low-cost evaluation model

Judging an entire test collection is costly. Fortunately this is not always needed. Only judging retrieved documents is often done. Another approach to reduce the costs more is to judge only highly ranked songs by at least one system[43]. However the cranfield evaluation methodology is not robust to gross violations of the completeness assumption[5]. Meaning that leaving out too many judgments will make the results unreliable. In 2005 a method to evaluate music similarity based on meta-data was proposed by Pampalk et al[29]. This is based on the genre of the music songs. It is a cheaper alternative, because genre labels for artist are often easy accessible. The assumption is that songs with the same genre, also have similar audio. There are obvious downsides of this method. Two obvious issue are that many artists have a very individual mix of several styles and some genres are closely related to each other and less distinctive than others. Although this method does not give the most precise results it illustrates the feasibility of using meta-data to derive music similarity.

Urbano[38] proposed a more elaborate low-cost evaluation model. This takes more meta-data into account besides only genre. In addition it also takes the output of other AMS retrieval systems into account. Meta-data of the songs is widely available and is therefore not that costly. The same holds for running other algorithms to obtain sets of retrieved documents for each query. These systems only require the audio signal of the songs, which should be available in the first place. If needed some judgments can be added to the model to make the predictions more precise. Depending on the use case a different percentage of judgments can be needed to obtain the desired precision. Most often this is an order of magnitude less than judging all retrieved songs. This is what makes the model low-cost.

The first step of the method is predicting the judgment labels as explained in the next section. Depending on the type of evaluation the steps to follow differ. There are three distinct applications of the model.

1. **Estimating absolute performance**, what the performance score of a particular system is.
2. **Estimating relative difference in performance**, how much better performs a system compared to another system.
3. **Estimating the rank**, which system is better.

The proposed method predicts for case 1 and 2 an expected performance score with a confidence interval. For the 3th application besides the predicted rank and confidence is calculated that the rank is actually correct. This confidence in addition to the estimation, contributes to the interpretation of the results.

2.2.1. Predicting the judgment label

The method starts by training a model which can make probabilistic predictions of the judgment labels. This model uses several features as input to train and prediction. These features can be divided into the following two groups:

- Output-based: Meta-data and retrieved documents from systems.
- Judgment-based: Some judgments in addition to the meta-data and retrieved documents.

For both groups of features the model is trained individually. During training the model learns how the features correspond with the judgment labels. Now the trained model can be used to predict the judgment labels of other test collections. Which lacks these annotations but for which the features can be calculated. When there are no judgments available only the output-based features can be used to make a prediction. But when some judgments are available the judgment-based features can be calculated for some of the songs. In general the predictions based on judgment-based features are more precise, thus will be used when possible. Even more precise is the actual similarity label, logically when the judgment is available this will always be used instead of making the prediction. In the MIREX dataset that is used some songs have a label as artist named 'Various Artists' this means the song is collaboration of multiple artists. This complicates the feature calculation to prevent this, artist labels are discarded. If this song needs to be predicted not all features can be calculated. To prevent any unintended effects in this specific case an uniform distribution will be predicted. Giving all possible labels the same probability.

In the proposed work there are different features specified and explained. Not all are adding the same amount of information to the model. Some could be redundant when other features are added. Therefore a selection procedure is used. Starting with all features and iteratively removing the non-significant features, until a model remains with only the most significant features. From the following features the bold features are actually used in the proposed model. The output-based features:

- **sGEN** whether the genre of the retrieved document is similar to the query document.
- **fGEN** fraction of all retrieved documents of all systems that belong to the same genre as the query document.
- **fART** fraction of all retrieved documents of all systems that belong to the same artist as the query document.
- **fSYS** fraction of systems that retrieved this document.
- **OV** degree of overlap in unique retrieved documents between systems.
- sART whether the artist of the retrieved document is similar as the query document.

The judgment-based features:

- **aSYS** the average relevance of documents retrieved by the system.
- **aART** average relevance of all the documents retrieved for the query performed by the same artist as the document.
- aDOC the average relevance of all the other documents retrieved for this query.
- aGEN the average relevance of all the documents retrieved for a query that belong to the same genre as the document.

Year	Systems	Queries	Retrieved songs (per query)	Total Judgments
2007	12	100	5	6000
2009	15	100	5	7500
2010	8	100	5	4000
2011	18	100	5	8925

Table 2.1: The general properties of the 4 different MIREX datasets.

The model in the literature is trained on the MIREX datasets. The data of years 2007, 2009, 2010 and 2011 is used. Table 2.1 illustrates the difference between the years. All year have the same amount of queries and retrieved songs per query, but the number of systems per year differ. This automatically also changes the total amount of judgments that are needed. When the model is used to prediction the judgment labels for a particular year the other three years are used to train the model. An ordinal logistic regression is used to train and make the predictions. The similarity variable that is predicted has ordered values. Which the logistic regression can take into account. The MIREX datasets contain two judgment scales Broad and the Fine scale. In this thesis the scales are normalized. The Broad scale consists of three values 0.0, 0.5 and 1.0. Where 0 means no similarity and 1 the most similarity possible. The Fine scale is not consistent through all years. Some years scale contains 101 values, other years 11. To keep it consistent all years are converted to a scale of 11 points: 0.0, 0.1, 0.2 ... 1.0. Again the higher the value the more similar the songs are.

2.2.2. Performance estimation

The model predicts probabilities for each label indicating how likely this is the actual label. With the probabilities of a prediction the variance and expected value can be calculated. See equation 2.6 where R_d is a random variable representing the relevance level assigned to document d. Each query has multiple retrieved documents that need to be judged or estimated. This combined forms the performance score. The next section will go into the calculation of the different metrics.

$$\begin{aligned} E[R_d] &= \sum_{\ell \in L} P(R_d = \ell) \cdot \ell \\ Var[R_d] &= \sum_{\ell \in L} P(R_d = \ell) \cdot \ell^2 - E[R_d]^2 \end{aligned} \tag{2.6}$$

Absolute Performance Metrics

When considering relevance as a random variable, $CG@k$ becomes a random variable that equals the sum of independent random variables. The expectation and variance for $CG@k$ are:

$$\mathrm{E}[CG@k] = \frac{1}{k} \sum_{i=1}^{k} \mathrm{E}\left[R_{\mathrm{A}_i}\right] \tag{2.7}$$

$$\mathrm{Var}[CG@k] = \frac{1}{k^2} \sum_{i=1}^{k} \mathrm{Var}\left[R_{\mathrm{A}_i}\right] \tag{2.8}$$

For the metric $DCG@k$ the formulas are:

$$\mathrm{E}[DCG@k] = \frac{1}{\log_2(i+1)} \sum_{i=1}^{k} \frac{\mathrm{E}\left[R_{\mathrm{A}_i}\right]}{\log_2(i+1)} \tag{2.9}$$

$$\mathrm{Var}[DCG@k] = \frac{1}{\log_2(i+1)^2} \sum_{i=1}^{k} \frac{\mathrm{Var}\left[R_{\mathrm{A}_i}\right]}{\log_2(i+1)^2} \tag{2.10}$$

For $NDCG@k$ the calculation of the expected value and the variance is a bit different. By considering the numerator (NM) and denominator (DN) from formula 2.4 separately the formula can be worked out the same as in formula 2.9 and 2.10. This way the expected values $E[NM]$ and $E[DN]$ can be calculated, the same holds for the variance for $Var[NM]$ and $Var[DN]$. The expected value and variance for $NDCG@k$ can be approximated with the Delta Method using Taylor series expansion[7].

$$\mathrm{E}\left[\frac{NM}{DN}\right] \approx \frac{\mathrm{E}[NM]}{\mathrm{E}[DN]} \tag{2.11}$$

$$\mathrm{Var}\left[\frac{NM}{DN}\right] \approx \left(\frac{\mathrm{E}[NM]}{\mathrm{E}[DN]}\right)^2 \left(\frac{\mathrm{Var}[NM]}{\mathrm{E}[NM]^2} + \frac{\mathrm{Var}[DN]}{\mathrm{E}[DN]^2} - 2\frac{\mathrm{Cov}[NM, DN]}{\mathrm{E}[NM]\,\mathrm{E}[DN]}\right) \tag{2.12}$$

The numerator and the denominator are not independent. Some of the top k documents retrieved by system A may be in the ideal ranking. However, for simplicity independence is assumed so that covariance is zero in the equation. For the metric $RBP@k$ all steps are the same as for $NDCG@k$. Except the ranking discount is not $log_2(i+1)$, but based on a persistence parameter p.

Relative Performance Metrics

Although the relative performance formula is similar to the absolute performance there are some notable differences. Therefore it is important to go over this as well. When comparing two systems the expectation and variance for a $\Delta CG@k$ score are:

$$\mathrm{E}[\Delta CG@k] = \frac{1}{k} \sum_{d \in \mathcal{D}} \mathrm{E}[R_d]\,(l(\mathsf{A}_d^{-1} \le k) - l(\mathsf{B}_d^{-1} \le k)) \tag{2.13}$$

$$\mathrm{Var}[\Delta CG@k] = \frac{1}{k^2} \sum_{d \in \mathcal{D}} \mathrm{Var}[R_d]\,(l(\mathsf{A}_d^{-1} \le k) - l(\mathsf{B}_d^{-1} \le k)) \tag{2.14}$$

The most important difference is that a document should only contribute to the difference when exactly one of the two systems retrieves the document. When both retrieve the same song the similarity of this song will not influence the difference. This will not influence the expected difference nor will it change anything about the variance. Not removing the influence of these estimated documents does not influence the expected value. The expected value of this document on both sides will cancel out. However, for the variance removing these double retrieved songs is crucial. Using the variance of these songs in the calculation does change the outcome of the Var[$\Delta CG@k$]. However, if this double retrieved song is predicted with a low variance it does not make the estimation of the difference more precise. This change in variance is simply not justified, because the retrieved songs by the two systems cannot be considered independent. The relative performance calculation for the other metrics can be calculated the same way. There is only one important distinction to be made. A retrieved song on exactly the same rank for both systems should always be excluded. For metrics that take rank into account a double retrieved song on different ranks does influence the difference. Therefore, should be taken into account in the relative performance calculation. In contrast to $CG@k$ where the rank does not matter. For these metrics a song retrieved at rank k or higher by both systems should always be excluded. The exact implementation details can be found in the work from Urbano[38].

Calculating System Performance

Both the absolute and the relative performance metrics can be represented as random variables. This is the performance score λ_q of a single query$q$. However, in the end the goal is to estimate the average $\overline{\lambda}_{\mathcal{Q}}$ over a sample of queries $\mathcal{Q}$ in a test collection. This is how we estimate the system performance. For some arbitrary metric Λ, the expectation and variance of the average are computed by iterating all individual estimates:

$$\mathrm{E}\left[\overline{\Lambda}_{\mathcal{Q}}\right] = \frac{1}{n_{\mathcal{Q}}} \sum_{q \in \mathcal{Q}} \mathrm{E}\left[\Lambda_q\right] \tag{2.15}$$

$$\mathrm{Var}\left[\overline{\Lambda}_{\mathcal{Q}}\right] = \frac{1}{n_{\mathcal{Q}}^2} \sum_{q \in \mathcal{Q}} \mathrm{Var}\left[\Lambda_q\right] \tag{2.16}$$

This expected average is the final estimation of the model for the performance score. The variance indicates the expected uncertainty of the prediction. With the Central Limit Theorem the confidence interval can be calculated as well.

$$\mathrm{E}\left[\overline{\Lambda}_{\mathcal{Q}}\right] \pm t_{\alpha, n_{\mathcal{Q}}} \sqrt{\mathrm{Var}\left[\overline{\Lambda}_{\mathcal{Q}}\right]} \tag{2.17}$$

This is the confidence interval of the average system score over all the queries. This means that actual average system performance has a confidence of $100(1-2\alpha)\%$ to lay within these calculated bounds. Both for the

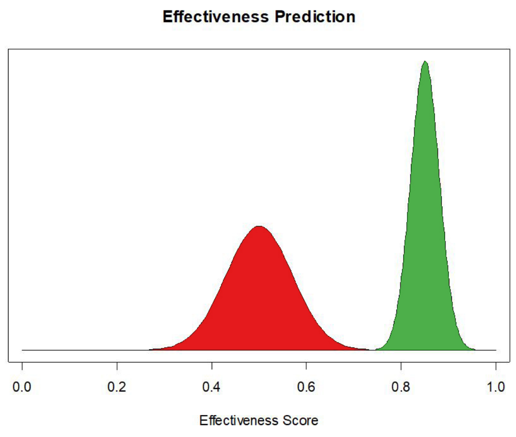

Figure 2.2: No overlap in possible performance score

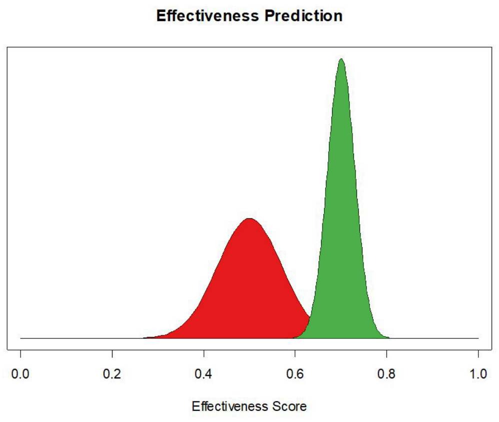

Figure 2.3: Overlap in possible performance score

estimated absolute and the relative performance score the confidence interval is defined. When only interested in the rank a confidence can also be defined. When all judgment labels are available the best system is the one with the highest performance score on average for all the queries. When having predictions instead of judgment labels a comparison can be made based on the probabilities. The formula for this is 2.13 when comparing 1 query and again the average can be calculated as in 2.15. When the expected value of the difference between two systems is non zero, this indicates which system is better. However, there will be some error in the estimate. When the difference between both scores is small, the chance is high of labeling the wrong system as the best of the two. Besides the difference in expected value the variance also plays a role in the confidence of the rank. To clearly illustrate this Figure 2.2 shows in red and green the distributions of possible performance score values without overlap. Where in Figure 2.3 systems do overlap this means there is uncertainty. The degree of overlap determines the final confidence in the rank.

We can use the t-distribution to approximate the probability that system A actually performs worse than system B and therefore the difference is negative:

$$P\left(\overline{\Delta\Lambda}_{\mathcal{Q},\mathsf{AB}} \leq 0\right) = F_t\left(\frac{\mathrm{E}\left[\overline{\Delta\Lambda}_{\mathcal{Q},\mathsf{AB}}\right]}{\sqrt{\mathrm{Var}\left[\overline{\Delta\Lambda}_{\mathcal{Q},\mathsf{AB}}\right]}}\right)$$

where F_t is the cumulative distribution function of the t-distribution with $n_{\mathcal{Q}} - 1$ degrees of freedom. If $P\left(\overline{\Delta\Lambda}_{\mathcal{Q},\mathsf{AB}} \leq 0\right)$ is below 0.5 the prediction indicates A is outperforming B. The more this probability approaches 0 the more confident the predictions are. The opposite holds for above 0.5, then the predictions is B outperforms A. When this probability approaches 1 the predictions become more confident. In general the confidence of the estimation can be calculate with the following formula:

$$C_{\mathcal{Q},\mathsf{AB}} = \max\left(P\left(\overline{\Delta\Lambda}_{\mathcal{Q},\mathsf{AB}} \leq 0\right), 1 - P\left(\overline{\Delta\Lambda}_{\mathcal{Q},\mathsf{AB}} \leq 0\right)\right) \tag{2.18}$$

2.2.3. Selection of songs to judge

For the judgment-features the model determines which documents to judge. Some documents will add more value compared to others. The documents are given a weight on how much value they add. The documents with the highest weight are selected to be judged. When comparing systems, documents retrieved by exactly 1 of the comparing systems is informative. The similarity label of songs retrieved by both systems will never influence the relative performance. When multiple system pairs are compared the most informative document is the one which is retrieved in the most pairs by exactly 1 system. When interested in the absolute score of multiple systems the weight is influenced by how much systems the document is retrieved. The weight is also depending on the selected performance metric. In case the metric contains a discount the higher retrieved documents are influencing the performance score more and should therefore be weighted more. In general the songs are selected which reduce the prediction variance the most. The exact implementation details can be found in the original work[38].

When to stop judging

Depending in the situation or use-case a certain accuracy will be required. It is feasible that without any judgments the model already makes accurate enough predictions. When this is not the case judgments can be added to increase the accuracy. Depending on the confidence of the predictions the amount of judgments needed can be determined. In case of ranked prediction, the confidence is calculate with formula 2.18. The correct amount of judgments can be obtained by adding judgments with the highest weight, as defined in section 2.2.3, until the calculated confidence is high enough. In case of the performance estimation the stopping condition can be defined as the amount of error that is acceptable. Based on the wanted confidence interval the required average variance can be calculated with formula 2.17. Then this serves as the stopping criteria, once exceed no more judgments need to be added.

2.3. Problems in the low-cost evaluation model

In Chapter 1 section 1.3 two biases of the model are named, overconfidence and overestimation. In the next section 2.3.1 the overconfidence is explained in more detail. In section 2.3.2 also more details are provided on this problem and it is linked to a similar observation in literature.

2.3.1. Overconfidence

As stated in the work of Urbano [38] the model is overconfident. This is measured in two different use-cases. In the estimation of the absolute performance score and the estimation of rank. The overconfidence in the absolute performance score is observed when setting a stopping criteria based on the variance. The goal of the stopping criteria is to determine the correct moment to stop judging more songs. By judging more songs the prediction variance and error is reduced. The stopping criteria indicates when to stop to obtain a certain expected error. However, when using this stopping criteria to stop at certain variance the obtained error is higher than expected. The stopping criteria can be improved by correcting the variance afterwards. However, this is a band-aid solution which does not address the root of the problem. It is a specific solution limited to the stopping criteria. Although not reported in the work it is likely that the estimation of difference in performance score also contains overconfidence. In the beginning of chapter 3 the overconfidence is quantified. Here the possible overconfidence in the estimation of difference in performance score will also be investigated.

2.3.2. Overestimation

Some absolute metrics are estimated with a clear bias. They are overestimated systematically. Figure 1.2 shows the predicted performance score against the actual performance score. If the prediction is unbiased and precise all points would lay close to the diagonal. Then the predictions match the actual values. In this figure no judgments are used thus the predictions are made with the output-based model. These predictions contain more variance than the judgment-based predictions. It is not expected that all values are close to the diagonal. For NDCG and RBP it is clearly visible that all points are above the diagonal. This illustrates a bias, namely overestimation. In the work of Urbano is shown that using the corrected stopping criteria to judge until an expected error of 0.05 does not resolve the overestimation. The values do move closer to the diagonal, which makes sense because more judgments make the estimations more precise. The reason for the overestimation could be the need to estimate the ideal ranking[38]. This overestimation is similarly observed in evaluation with imperfect judgments in text retrieval[45]. The possible causes and solutions are tested in Chapter 4.

3
Overconfidence in the performance scores

The predictions of the low-cost evaluation model contain a degree of uncertainty. Having a confidence alongside the prediction is therefore important for correct interpretation of results. When comparing systems you want to know which system is better. In most use-cases, there is a need for knowing the chance that the model predictions are actually correct. If the model predicts that system A is better than B with a certainty of 95% than in some use cases this prediction is probably accurate enough. Where a certainty of 55% is not much better than a coin flip and will not be as useful. However, the estimated confidence of the model is higher than the actual observed accuracy. This means that in some cases the results are less accurate than expected, this forms a problem for the applicability of the model. On the other hand if the confidence is lower than the actual observed accuracy, then more judgments would be used then actually needed to obtain the desired accuracy. This would unnecessarily increase the cost.

This chapter will begin in section 3.1 by introducing metrics an methodology to properly quantify the overconfidence. Then in section 3.2 the reliability of the features is improved, in order to reduce the overconfidence in the performance scores. In section 3.3 is discussed how the method of judgment selection plays a role in the overconfidence, but also the algorithm used in the model do influence the reliability of the predictions. In section 3.4 different algorithms are compared to find the most suitable. All found improvements are then combined and measured on how effective they are in reducing the overconfidence. These results are reported in section 3.5.

3.1. Quantifying overconfidence

Proper quantification of the problem is of importance in search for causes and solutions. The overconfidence is observed in the estimation of performance score and the rank. A quantification method of overconfidence in rank prediction is already shown in Table 1.1. This section will introduce a suitable quantification method for both the absolute and relative performance score.

When the relevance label predictions are overconfident, the absolute, relative and rank estimations will also become overconfident. Considering this could be the cause proper evaluation of the probabilistic predictions of the current model is crucial. The section 3.1.2 introduces an appropriate scoring method. The decomposition of this scoring method brings additional insights. Which is in particular beneficial when addressing the overconfidence.

3.1.1. Overconfidence in performance prediction

Overconfidence can be described more generally, as the mismatch between the variance of the prediction and the actual accuracy or error. The mean average error (MAE) can be compared to the variance of the stopping criteria as shown by Urbano[38]. However, this approach is indirect for addressing overconfidence. Another drawback is that the variance can already pass the criteria from the start. At that moment the expected MAE can only be described with an upper-bound. The prediction could be passing the criteria barely or ample. This scale cannot be taken into account. A more direct translation of variance to actual accuracy or error would therefore be more suitable. This can be achieved by calculating the confidence interval and counting how often the observed performance score actually falls in this interval. The confidence interval is defined by 2.17. The confidence of the interval represents the theoretical long-run frequency of confidence intervals that contain the true value of the unknown population parameter. Simply put 95% of confidence intervals contain the actual value when computed with a 95% confidence. Calculating the confidence interval for all system score estimation gives the opportunity to calculate how often the actual system score falls into this range. Although we do not have the long-run frequency of confidence intervals for one performance score. We do have many different system score estimations. This way a form of quantification of the possible overconfidence can be obtained. By comparing the observed ratio of scores within the interval to the confidence. If the ratio of observed scores is below the confidence level of the interval there is overconfidence.

Starting with the output-based prediction model for every system the performance score can be estimated. Along with the expected score a variance can be obtained. Based on the expected value and variance the confidence interval is calculated. This can be repeated for different levels of confidence. The next step is checking if the actual performance score falls within this interval. By repeating this for all systems an average can be calculated. This ratio indicates how often the values fall within the interval. The degree of mismatch between the obtained ratio and the confidence level quantifies the overconfidence. Table 3.1 shows this quantification of overconfidence. For four different confidence levels and for the four metrics. Left the output-based predictions are shown and on the right side the judgment-based prediction, when all judgment labels are available. Not all songs can be predicted by the judgment-based model. When an artist is retrieved exactly once for a query the aART feature can not be calculated. If this is the case normally the output-based model will predict this song. However, for this chapter we are analyzing both models in isolation to get better insight. This is accomplished by falling back to a simple uniform distribution. As expected the ratio of values within the interval is clearly less than the confidence interval for all levels and all metrics. The mismatch for the judgment-based model is slightly larger than the output-based model. For NDCG and RBP we see that none of the points are in the confidence intervals. Which can be explained by the structural overestimation that occurs for these metrics. This problem will be addressed in Chapter 4.

Table 3.2 is exactly the same table for the relative performance estimation. Again we observe a mismatch that indicates overconfidence in particular for the judgment-based predictions. This time the differences are smaller. Also the NDCG and RBP metric do not perform as bad. Which can easily be explained, because in the relative performance setting two systems are compared. Both systems are overestimated for these metrics, but this probably cancels each other out to some extent when only interested in the relative score.

3.1.2. Overconfidence in relevance prediction

To evaluate the probabilistic predictions a scoring method is needed. This score should not only calculate the error between the expected value and the actual value. It should also indicate how well the predicted probabilities match the actual values. Simply put a prediction with high probabilities should not often be wrong. The Probability Score also known as Brier Score is a scoring method that takes this into account. This score function measures the performance of probabilistic predictions[4]. In literature most often used to evaluate binary events. However, the original score can also be applied to multi-categorical events. The MIREX ratings are ordinal, when considering this as multi-categorical the rank would not be taken into account. To illustrate this with an example: Assume the relevance can be 1, 2 or 3. Model A predicts the probabilities respectively of (0.2, 0.6, 0.2). Model B predicts (0.6, 0.2, 0.2). The actual label is 3. In this case the Probability Score would give both models the same scoring value. Although model A actually makes a better prediction when looking at expected value. Therefore, it is more suitable to use the Ranked probability score (RPS)[13]. This scoring metric is similar to the Brier score, but takes the rank into account.

Reliability calculation

The RPS can be split into 2 components reliability and resolution [27]. The resolution indicates how precise the prediction is. In terms of resolution the worst prediction is uniform, this is not informative. The best predictions in terms of resolution contains 1.0 probability for 1 label and all other 0. The reliability indicates the consistency between the predicted probabilities and the subsequent observations[6]. If a model makes reliable predictions this means the predictions match the observations. More concrete when a perfectly reliable model makes 100 predictions for a label with 0.8 probability, the 100 observations should have 80 times the label. This decomposition makes it possible to compare different models solely on reliability. Reliability

CG@5

Output-based Model		Judgment-based Model	
Confidence interval	Within interval	Confidence interval	Within interval
0.60	0.17	0.60	0.13
0.90	0.35	0.90	0.27
0.95	0.38	0.95	0.29
0.99	0.46	0.99	0.38

DCG@5

Output-based Model		Judgment-based Model	
Confidence interval	Within interval	Confidence interval	Within interval
0.60	0.25	0.60	0.19
0.90	0.40	0.90	0.33
0.95	0.46	0.95	0.35
0.99	0.58	0.99	0.52

NDCG@5

Output-based Model		Judgment-based Model	
Confidence interval	Within interval	Confidence interval	Within interval
0.60	0.00	0.60	0.00
0.90	0.00	0.90	0.02
0.95	0.00	0.95	0.02
0.99	0.00	0.99	0.02

RBP@5

Output-based Model		Judgment-based Model	
Confidence interval	Within interval	Confidence interval	Within interval
0.60	0.00	0.60	0.02
0.90	0.00	0.90	0.02
0.95	0.00	0.95	0.02
0.99	0.00	0.99	0.02

Table 3.1: The ratio of actual performance scores that fall in a confidence interval of the model estimation. The model estimates the absolute performanceof all systems in the 4 years of the MIREX dataset. Both the output-based an judgment-based models are used.

can be used to evaluate how a change or addition to the model influences the overconfidence. This is how the reliability is calculated:

$$\sum_{k=1}^{K} \frac{n_k}{N} (\vec{y}_k - \vec{o}_k)^2 \tag{3.1}$$

The predictions are grouped into K groups with all equal prediction probabilities. For each group the reliability is calculated with $(\vec{y}_k - \vec{o}_k)^2$. Where $\vec{y}_k$ is the cumulative prediction vector. In the example from above for the prediction of model A this would be $(0.2, 0.8, 1.0)$. $\vec{o}_k$ is the vector of cumulative observations. Assume we have 30 predictions for label 1 and for label 3 and 40 predictions for label 2. This translates to $\vec{o}_k = (30/100, 70/100, 100/100) = (0.3, 0.7, 1.0)$. The reliability of a group predictions with the same probabilities is defined as the squared mismatch between the predicted probabilities and the observed values. Due to the cumulative component of the calculation rank is taken into account. This calculation is an empirical estimation of the reliability of the group. The total reliability is the sum of all groups weighted by n_k the amount of predictions in each group. This calculation of reliability ranges from 0 to 1, where 0 indicates that the predictions perfectly match the observed values. To make it clear that a lower value is better it will be called reliability error.

Inaccuracies in the reliability calculation

Calculating the reliability is not without any hurdles. Covering this in more detail will contribute to making this work reproducible and with the correct interpretation of the results. The number of predictions available will influence the accuracy of the calculated reliability. In general the reliability will tend to appear poorer when few predictions are available in the calculation[14]. When the empirical estimation is based on less

CG@5

Output-based Model		Judgment-based Model	
Confidence interval	Within interval	Confidence interval	Within interval
0.60	0.45	0.60	0.38
0.90	0.75	0.90	0.62
0.95	0.77	0.95	0.67
0.99	0.91	0.99	0.77

DCG@5

Output-based Model		Judgment-based Model	
Confidence interval	Within interval	Confidence interval	Within interval
0.60	0.57	0.60	0.46
0.90	0.81	0.90	0.72
0.95	0.90	0.95	0.77
0.99	0.98	0.99	0.90

NDCG@5

Output-based Model		Judgment-based Model	
Confidence interval	Within interval	Confidence interval	Within interval
0.60	0.28	0.60	0.17
0.90	0.46	0.90	0.31
0.95	0.55	0.95	0.35
0.99	0.66	0.99	0.42

RBP@5

Output-based Model		Judgment-based Model	
Confidence interval	Within interval	Confidence interval	Within interval
0.60	0.38	0.60	0.24
0.90	0.59	0.90	0.38
0.95	0.67	0.95	0.43
0.99	0.78	0.99	0.52

Table 3.2: The ratio of actual performance scores that fall in a confidence interval of the model estimation. The model estimates the relative performance between all systems pairs in the 4 years of the MIREX dataset. Both the output-based an judgment-based models are used.

data more noise is observed. Unless there are many predictions in each group, the calculated reliability will appear poorer than it actually is. In addition the calculated value contains some inaccuracy, because it is based on the empirical estimation. Again when based on less predictions the inaccuracy will be more significant.

The inaccuracy and overestimation depend on the number of predictions in each group. This means that the method used for grouping the predictions influences the calculated reliability. Instead of grouping identical predictions, similar predictions can be grouped[6]. This helps prevent small group sizes. Grouping similar predictions can be achieved by rounding the probabilities. Rounding the probabilities to a fixed interval is not ideal. Most probabilities are low in particular when using the Fine scale. For example when rounding to 1 decimal each similarity label can obtain 11 different predicted probabilities 0.0, 0.1, 0.2 ... 1.0. When using the fine scale we also have 11 different similarity labels that are predicted. In a prediction the sum of all predicted probabilities for the labels is 1.0. This means that most of the 11 labels get a low probability. Therefore are rounded to the few lowest values. When rounding the probabilities, smaller intervals for the lower probabilities should be used. Instead of using a fixed interval, quantiles can be used to calculate which values to round. This is based on the distribution of the probability values. When the probabilities are rounded to closer values the lose of information is decreased. For example when rounding to only 5 distinct values a scale such as 0.00, 0.05, 0.10, 0.50, 0.80 can be expected. An important side effect of rounding is the sum of a prediction will not always be exactly 1. This should be correct such that the total sum is always 1.

Dealing with the inaccuracies in the reliability calculation

Figure 3.2 shows the correlation between the number of distinct probability values and the group sizes. Obviously when rounding the probabilities to less distinct values more predictions are placed in the same group. Checking the convergence of the reliability estimates as the sample size increases is worthwhile[14]. Therefore Figure 3.1 plots the sample size against the estimated reliability. For each number of predictions multiple groups of this amount of predictions are sampled. The spread of these values give an indication how stable the calculation is. In general it is beneficial to have a stable calculation when comparing models on reliability error. It is clearly visible that the calculation becomes more stable when more predictions are available. As already stated in the literature the lack of enough data will cause reliability estimation to be worse [14]. This is clearly visible in the figure as well, all lines start with higher reliability error and then converge to lower value when more predictions are used. Depending on the use case the amount of predictions available to calculate the reliability will differ. Having an fast converges prevents overestimation of the reliability error for the cases that less predictions are available. Simply put, low variance and faster convergence is beneficial to obtain.

By rounding the predictions values more predictions will fall in the same group. Therefore the lack of data, in each group, to estimate the distribution will decrease. This makes the reliability converge faster. Figure 3.1 illustrates the different degrees of rounding and how this impacts the stability and overestimation. In blue the least rounding is used and the graph converges the slowest. In contrast to red, where all probabilities are rounded to only two values, here the reliability error converges the fastest. However, when only rounding to a few values a lot of information is lost. This leads to simplifying the probabilities too much this is at the expense of representing the details in the model. This can be problematic when comparing two similar models, if the difference between the predictions are lost due to rounding no difference can be determined.

An other possible way to improve the convergence is by removing small groups before calculation. A group with just a few predictions will not reliably reflect the actual distribution. Removing too much predictions could influence the final value, therefore not too many groups should be removed. Figure 3.3 shows the effect from removing the groups with a small size. Logically, the largest effect is on the calculations with few predictions. When having more total predictions the groups sizes will also be smaller. Therefore, more predictions are removed relative to the total amount of predictions. The higher the threshold the more predictions are required to have at least 1 group exceeding this. For the 2007 dataset the minimal group size of 100 requires more than 1000 predictions. As expected the reliability converges faster when the small groups are not used in the calculation. But the minimal group size also influences the value the reliability converges to. When trying to calculate reliability on a scarce dataset it can be beneficial to filter out the small groups. Those groups will contain much error because they often will not represent the actual distribution correctly. Figure 3.3 shows the influence of setting a minimal threshold on the group size. This contributes to early convergence.

Important to conclude is the need to take the inaccuracies of the reliability calculation into account. In this thesis when comparing models in terms of reliability multiple reliability errors will be calculated, of which the mean is used and the standard deviation is indicated. In particular in the setting where few predictions are available the variance is significant. When comparing models the standard deviation gives scale to differences. Small observed reliability differences should therefore be ignored. Also the lack of data will cause overestimation. Therefore, it is important when comparing two models the reliability always needs to be calculated on the same amount of predictions. Concrete this means in every figure all reliability errors are calculated on the same number of predictions. Comparing reliability between figures is only valid when the same configuration is used.

3.2. Reducing overconfidence via more reliable features

Reliable predictions are crucial. With the reliability error the predictions can be evaluated, with as a final goal increasing the reliability. As explained in the previous section the reliability error calculation is suitable for comparison. However, due to the limited size of the datasets available we expect overestimation of the reliability error. This means the goal of this section is not necessarily getting the reliability error close to zero. The focus will be on improving. Important starting directions are found based on observed difference in reliability. The predictions of the different years do not have the same reliability. Also the judgment ratio seems to influence the reliability of the predictions. This section will break down the differences. In an attempt to

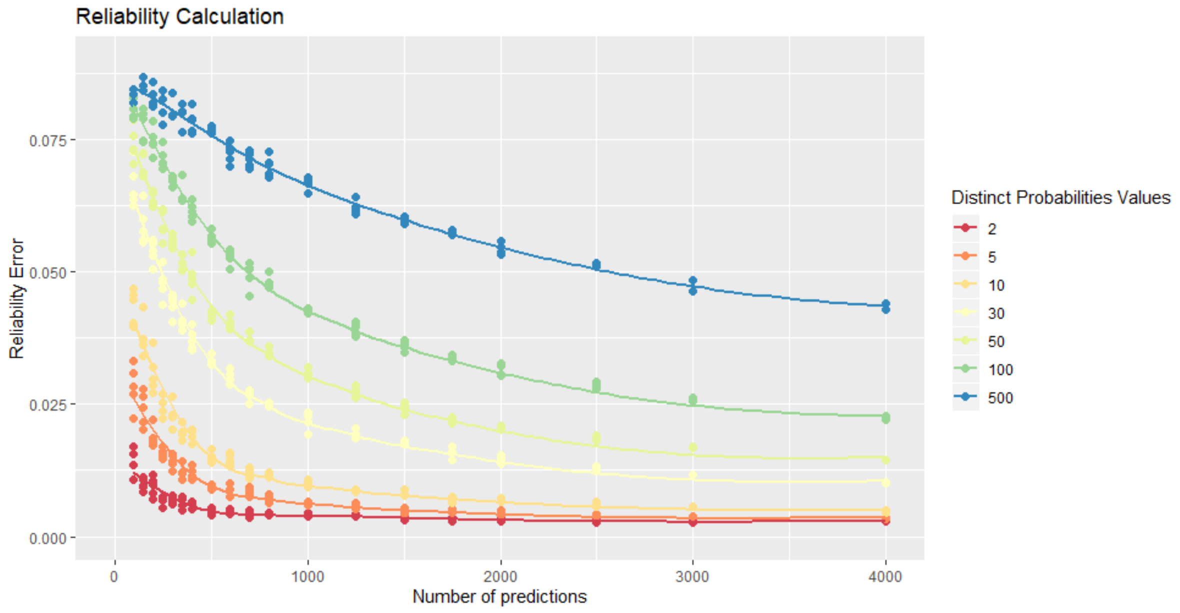

Figure 3.1: The reliability of the original judgment-based model. Calculated on different amount of predictions sampled from the 2007, 2009, 2010 and 2011 dataset. The different colors show distributions when the predictions are rounded to different amounts of distinct probability values.

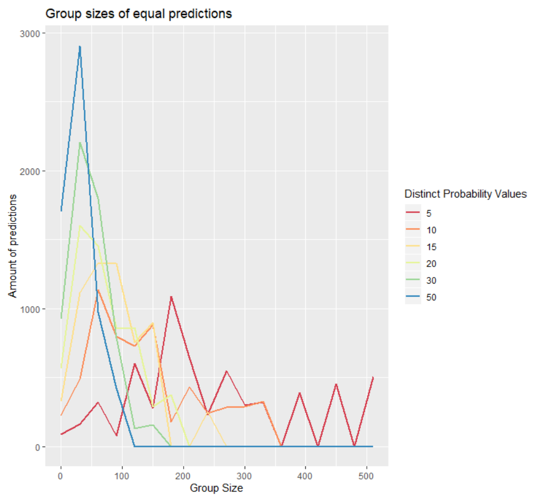

Figure 3.2: The distribution of group sizes for the predictions made for 2007 with the judgment-based model. The dataset 2007 has 5998 retrieved songs that can be predicted. The different colors show distributions when the predictions are rounded to different amount of distinct probability values.

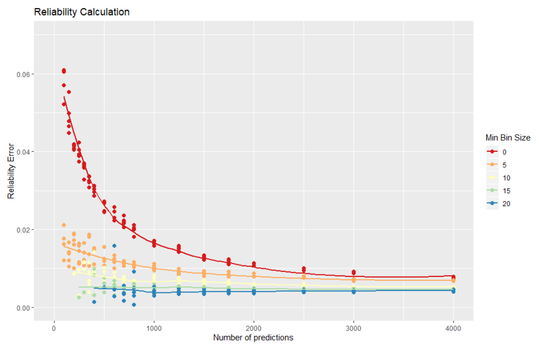

Figure 3.3: The reliability of the original judgment-based model. Calculated on different amount of predictions sampled from the 2007, 2009, 2010 and 2011 dataset. The predictions are rounded to 10 distinct values. The different colors show the minimal group size the groups are taken into account.

find the causes and propose possible solutions. Along the way an important problem in the judgment-based model is found. This is explained in the next section 3.2.1.

3.2.1. Inclusion of own judgment label

The features calculated in the training dataset should represent the same information as the features calculated on the prediction dataset. However, there is a mismatch in the calculation of some features. This only concerns features used in the judgment-based model. This is best explained with an example. Figure 3.4 is an example setting. Where the feature aART for a retrieved document d in query q is computed. Document d is written by 'Coldplay'. The aART feature is computed by taking the average of all retrieved documents for query q by any system that has the artist. In the example there are 3 songs from Coldplay retrieved. The average judgment is $(0.5+1.0+1.0)/3 = 0.83$. This is the aART value of the retrieved document *Clocks - Coldplay*. The problem with this method is that it uses the judgment label of the document for which the feature is calculated. The only application of the features is for predicting judgment labels. This only happens when the label is missing. Trivially, this means that the label of this song itself will never be used to calculate the feature. But the model is trained on features in which this value is incorporated. This is a mismatch between the training features and the features used for the prediction. This problem also happens when calculating aGEN and aSYS.

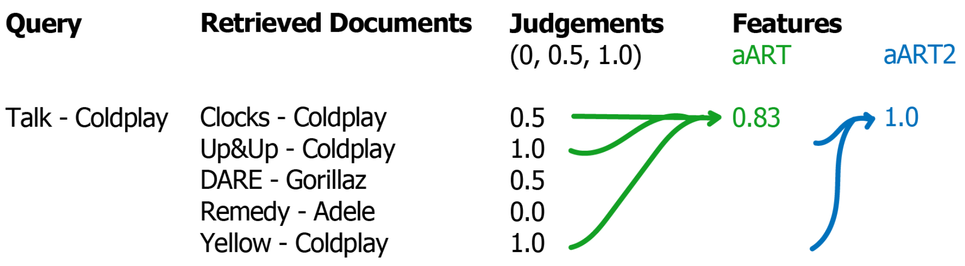

Figure 3.4: Average artist feature calculation. In green the original way of calculation. In blue the newly proposed method called Excluding Self.

In the example case when there are only 3 Coldplay songs the value is strongly influenced by including the judgment label of the retrieved document itself. When the number of judgments used in the feature calculation becomes higher this influence becomes less and less. This means that when there are few judgments available the impact of this mismatch is the most significant. The mismatch is easily prevented when always excluding the value in the feature calculation. To differentiate this change in the calculation the model with the features excluding the judgment label of the document itself will be called "Exclude Self". To measure the impact of this change Table 3.3 shows the difference in error. The original judgment-based model is compared to the model Excluding Self. This 10% is on purpose this low, as explained above in this setting the difference in feature values is the most significant. In addition is a low percentage also a common use case, simply because the model is designed to perform well with few judgments. The proposed change to the features decreases the reliability error for all four years in both low and high judgment setting. Although the differences are more significant in low judgment. Besides improving the reliability it also improves the RMSE, which means the expected value is in general closer to the actual value. This shows the improvement in reliability is not at the cost of accuracy. This adjustment is more removing a mistake in the calculation than an additional improvement. In terms of error and reliability there is no downside. This is why other proposed adjustments are also tested with taking this improvement into account.

3.2.2. Uniqueness of attribute

The maximal value of the features fART and fGEN varies between the different years. This is shown in Table 3.4. The value of a feature should represent the same information in terms of music similarity across the different years. Each dataset has different query songs and therefore also different feature values are expected. However, the difference between the maximal calculated fART value in 2010 and 2007 is notable. This creates

100% Jugments

Year:	2007		2009		2010		2011	
Model:	Original	Excluding	Original	Excluding	Original	Excluding	Original	Excluding
RMSE	0.160	**0.157**	0.176	**0.171**	0.190	**0.189**	0.182	**0.176**
RPS	0.084	**0.080**	0.093	**0.087**	0.107	**0.102**	0.102	**0.092**
Reliability	0.015	**0.013**	0.015	**0.012**	0.018	**0.014**	0.017	**0.012**

10% Jugments

Year:	2007		2009		2010		2011	
Model:	Original	Excluding	Original	Excluding	Original	Excluding	Original	Excluding
RMSE	0.198	**0.188**	0.196	**0.186**	0.211	**0.204**	0.204	**0.193**
RPS	0.110	**0.097**	0.109	**0.099**	0.116	**0.107**	0.115	**0.102**
Reliability	0.020	**0.013**	0.018	**0.012**	0.023	**0.014**	0.019	**0.014**

Table 3.3: The error of judgment-based predictions for the 4 years. Comparing the original model with the excluding self adjustment. The numbers that are bold are better in terms of error to clearly illustrated the difference.

	Year	Systems	fART	fGEN	fSYS	RPS	RMSE	Variance	Reliability
1	2007	12	0.37	0.85	0.67	0.136	0.250	0.061	0.0052
2	2009	15	0.39	0.70	0.60	0.129	0.235	0.064	0.0048
3	2010	8	0.57	0.86	0.88	0.154	0.269	0.056	0.0072
4	2011	18	0.45	0.92	0.56	0.143	0.256	0.060	0.0048

Table 3.4: Some properties of the 4 different datasets. The columns fART, fGEN and fSYS show the maximal value this feature in the dataset. The columns RPS, RMSE, Variance and Reliability are all calculated of the output-based predictions for that specific year.

the suspicion that there might be other reasons beside the different query songs that could cause this difference.

An possible explanation could come from the uniqueness of genres and artists. The features fGEN and sGEN do not make any distinction between specific genres. However, some genres may be more distinct than others. If a datasets does not balance the number of songs within each genre, this could be a problem. Also when the distribution of genres in the training set differs from the prediction dataset. For example when 50% of the songs are classical the fraction of retrieved songs of this genre will in general be higher compared to very unique genres. Exactly the same holds for artists. In the MIREX dataset there are hundreds of songs belonging to the same artist. The chance that multiple systems retrieved this artist for a query is much more likely than an artist which only has 5 songs in the dataset. To give the model the opportunity to account for this an extra feature can be added for both the genre and artist. Which indicates the ratio of songs that have this attribute in the total dataset. Taking this ratio into account for genre does not influence the model and predictions notably. The ratio for the artist attribute does influence the model. Figure 3.5 shows the difference in reliability between the original model(red) and adding this artist ratio to take into account how unique the attribute is(green). Their average reliability of 2007 and 2011 improves a bit compared to the original model. However, when looking at the errorbar of the reliability this slight improvement could also be by chance. The reliability calculation contains too much error to prove these additional features actually makes the model more reliable.

3.2.3. Number of systems

Not every year has the same amount of systems in the dataset in order 12, 15, 8 and 18. The number of systems determine the amount of retrieved songs. If a feature is calculated on just a few retrieved songs it could be less stable compared to a setting with many retrieval systems. When predicting for 2010 the model is trained on the other years, which all have more systems. The hypothesis is that the mismatch between the number of systems in the training data and the number of systems in the predicted dataset will make the reliability worse. Table 3.4 shows that the reliability error of 2010 is the highest. Also the average variance of predictions for 2010 is the lowest. This means the predictions of the model trained on all years except 2010 is the most precise. These numbers substantiate the hypothesis. A possible solution would be matching the number of systems in the training set with the dataset that is going to be predicted. However this means the model is trained only for a specific setting. Where a more general applicable model is desired. This also means information in the training or prediction data is left out. As already emphasized properly judged AMS data is rare.

A more general solution would make the model take the number of systems into account. More precise it should take the number of retrieved documents into account. In the four datasets the number of retrieved documents per query is always 5, but this is not necessarily the case. To make the model generally applicable the number of systems multiplied by the number of retrieved documents per query should be taken into account. By adding this number as an additional feature the model can learn the influence. If this number is high the features fGEN, fART and fSYS are expected to be more stable. The model should learn the influence from having a different number of values used in the feature calculation. However, the usage of only 4 datasets gives exactly 3 different counts in the training set. For every query in a given year the same amount of systems are included. When the year that the model needs to predict has a number of systems that differs a lot from the 3 years in the training then the model is not trained for this. To make sure the model can learn how the feature count influences the features the training data can be synthesized. In order to prevent the loss of information the training datasets can be copied. Then in the copied data the amount of systems per query is reduced randomly. This makes sure the number of retrieved songs in all the queries differs widely. Because the number of retrieved songs in a query is represented by the additional feature the model can now learn the influence.

Figure 3.5 shows in blue the reliability error of the model which also has the feature counts. When the hypothesis is correct and this solution would address this, the reliability error should improve and in particular 2010. Instead we see both 2010 and 2011 performing worse in terms of reliability, while 2007 and 2009 stay roughly the same. Therefore this is not an appropriate solution or the hypothesis is wrong. This could mean that the difference in number of systems between the four years does not impact how trustworthy the features are.

The influence of the system output itself

Another possible cause of higher reliability error could be the influence of the system output itself. For the calculation of fGEN and fART the retrieved songs of the system itself are taken into account. This could mean if a system is consistent totally off the fGEN and fART are not as reliable. For example when the query song is a heavy metal song. The system retrieves 5 classical songs, the fGEN will score relatively high because already 5 retrieved documents are from the same genre. This is also applicable for fART. Again the strength of this effect depends on the number of systems in the dataset. This influences the ratio of the retrieved songs which belong to the system itself. This could explain why the reliability error of 2010 is significantly higher.

By excluding the output of the system itself this possible influence is removed. As visible in Figure 3.5 in orange this improves the reliability. In particular for 2010 which as expected would be influenced most by this due to having the fewest systems. However, there is a downside of just excluding the system itself. In particular when predicting for songs which are not retrieved by any system. For example an artist which is only retrieved once for a query. When excluding the system itself the fART value of this document is 0. Which is also the case for every other not retrieved song with a never retrieved artist.

Calculating the fraction for fART and fGEN can also be done based on the number of systems instead of the number of retrieved documents. This is a possible approach to reduce the influence of the system itself without making the prediction for not retrieving documents worse. So concrete this means that fART will indicate the fraction of systems that also retrieved this artist and fGEN the number of systems that retrieved the genre. Figure 3.5 shows in yellow this system-based approach. Clearly the reliability improves in particular 2010. Interesting is that the reliability errors of all four years are now roughly equal. This makes it likely that the influence of a number of systems in a year for this dataset is not influencing the trustworthiness of features.

3.2.4. Number of judged documents

Depending on the required accuracy or confidence of the model the judgment ratio will vary. Figure 3.6 illustrates in red how the reliability of the original model increases when judgment ratio decreases. The same trend is also observed for suggested adjustment in blue Excluding Self. Not every feature is based on the same number of values. Take for example the feature aART which is the average judgment of all retrieved documents with the same artist. If many of the retrieved documents have the same artist this average is

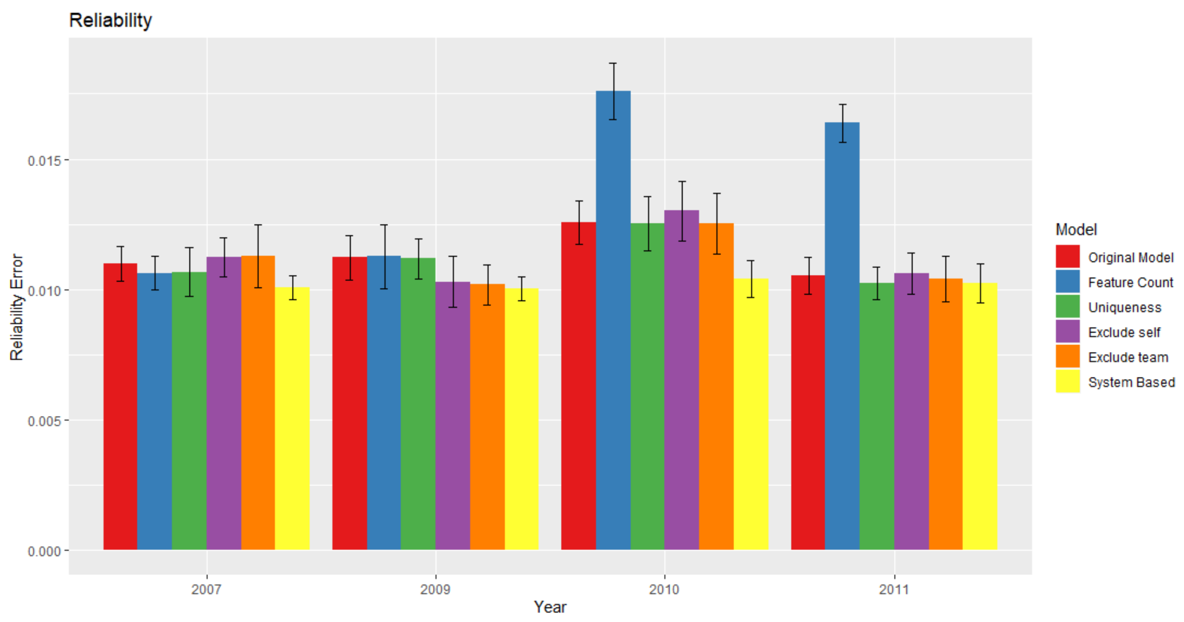

Figure 3.5: The reliability of the output-based predictions for the 4 different years. The colors indicate the different models used.

based on all those values, but when only one other document has the same artist the average is literally just that one value. Of the relevant retrieved documents not all will have the judgment available. When calculating features for a dataset with only a few percent of judgments. The calculated features are in general based upon less data points compared to the training set. In the proposed model the logistic regression is trained on multiple years of the MIREX dataset with all judgments available. This trained model will assume that the features of a new dataset are calculated the same. But this is not the case when only a few % of the data is judged. In general a judgment-based feature is now calculated based on less value compared to the training setting.

The same holds for the feature aSYS. When all judgments are available aSYS will be the exact average. When having only a few judgments for a system the aSYS value can be a bit off. In the MIREX dataset every system gets 100 queries and retrieves 5 documents, so there are 500 relevant documents for each system. Compared to the aART feature the average is in general based on more values. When there are only 5% of the judgments available aART is often based on just a few values where aSYS is based on roughly 25 values. However, when just 1% or 2% of the data is evaluated the same inaccuracy can happen. In addition as explained in section 2.2 in the original model which documents are judged is based on weight. This does not evenly distribute judgments over all systems. This makes it possible for a system to have barely any judgments after judging 10% or more.

Feature calculation count

Ideally the model should take into account the number of values every feature is based on and adjust the confidence correspondingly. This is similar to section 3.2.3. Only now the number of values the features are based on range from 1 to hundreds. Where for the output-based model a whole year is based on the same number of values. Again to make the model take this into account an additional feature with the count can be added. For the aART example (see Figure 3.4) this value would be 3. In case we exclude the judgment label of the song itself this value would be 2. For every feature this value can be calculated and added to the dataset. Adding this count for the feature aSYS is not changing the predictions notably. Where for aART there is a slight improvement in particular in the low judgment setting. Figure 3.6 shows in green in particular with 5% judgments how it is slightly more reliable.

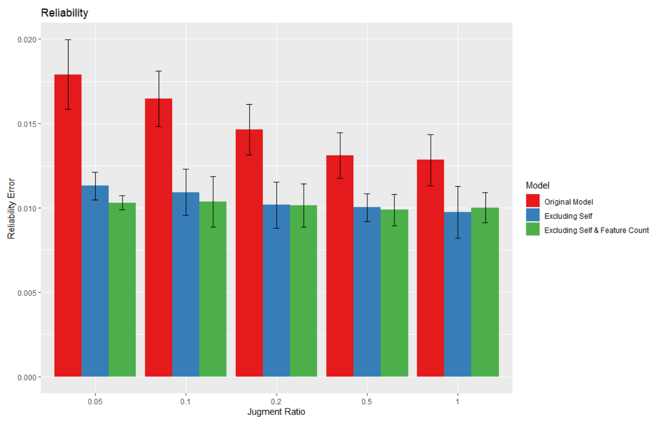

Figure 3.6: The influence of judgment ratio on reliability. Comparing the original model, exclude self and exclude self & feature count. Only judgment-based predictions are taken into account.

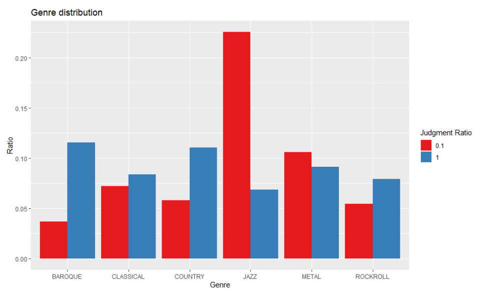

Figure 3.7: The distribution of songs over 6 genres. Blue shows the distribution of all 2007 songs that belong to those 6 genres. Red shows the distribution of the same genres, but for the first 10% judged songs.

3.3. Reducing overconfidence via the judgment selection method

When evaluating other components of the model random selection was used to make sure the selection method is not influencing other components. Instead of randomly judging songs Urbano[38] to use weighted selection based on the contribution of the song. This is explained in more detail in section 2.2. When estimating the absolute performance score the goal is to minimize the variance. In the context of comparing systems the variance in the comparison should be reduced. When comparing multiple systems reducing the most variance in all systems pairs is the goal. Intuitively, at all times we want to judge those documents that affect the most system pairs[38].

As explained in section 3.2.4 the lack of judgments can make features less stable. For the feature aSYS no significant solution is proposed in this work so far. Figure 3.8 shows the mismatch between calculating the system average on all judgments (500) and a subset of this. Clearly more judgments decrease the mismatch. In particular the first 50 judgments decrease the error the most. To obtain a sense of scale Figure 3.9 shows the distribution of the differences between system pairs in terms of average similarity. Naturally the maximal theoretical difference is 1, however this does not occur in the 4 years of data. It is rare that two systems differ more than 0.3 in average similarity. But when only a few judgments are used to approximate the average similarity an error of 0.3 can be expected. Therefore this number would not make a stable feature. The differences in predictions are small when using different amount of judgments to calculate aSYS. Which is probably partly due to that feature aSYS is contributing significantly less than aART in the judgment-based model. The reliability calculation contains too much noise to compare these small differences. Therefore the RPS is used, which measures both reliability and resolution. Figure 3.10 quantifies the influence of the number of judgments used to calculate aSYS. As expected using more values makes the predictions better. But having 500 instead of 50 values to calculate aSYS is not improving the model notably. This shows that judging at least 5-10 songs for every system to obtain a stable aSYS is beneficial for the model. In some cases the output-based model is already sufficiently accurate. when this is no the case and some judgments are required it is beneficial for the reliability of the results not to judge only a few songs for a system. This does not mean it is mandatory to directly obtain 10 or more judgments for all systems. When adding judgments to a number of systems it is beneficial to make sure they all have a fixed minimal amount. The remaining systems with no judgments are still predicted with the output-based model.

By deviating from random judgment selection unintended side effects can occur. When a ranked metric is used the proposed weights will prefer the first retrieved songs. Simply because this song is most influential when calculating the performance score. Estimating the average system similarity with only rank 1 songs, could also introduce bias. Most systems will try to retrieve the most similar documents first, therefore the estimation could overestimate the average song similarity. Another possible unintended side effect of weighted selection is that the selection does not reflect the overall dataset. For example a certain artist or genre can be more commonly retrieved. As explained, in general when a song is retrieved by more systems the weight goes up. When the highest weighted songs do not reflect the total datasets the results can be biased. Depending on

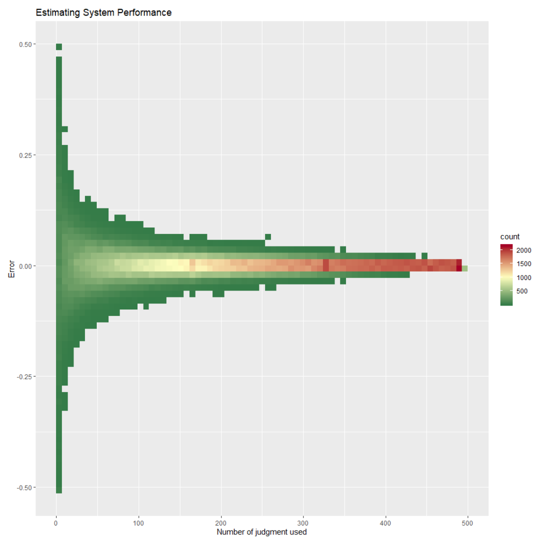

Figure 3.8: Estimation error of average system performance with a subset of the jugments.

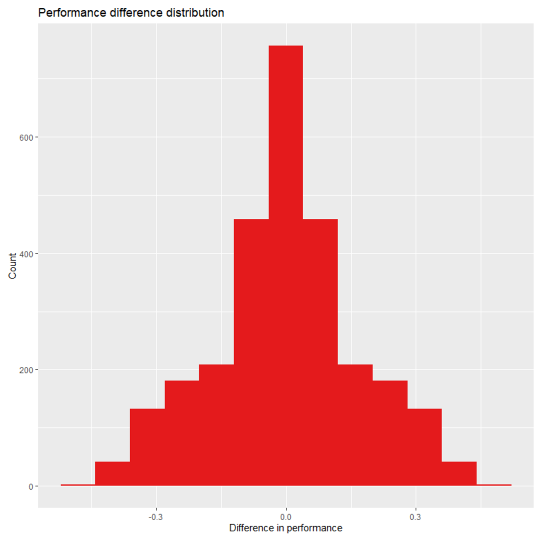

Figure 3.9: The distribution of difference in average performance between all system pairs.

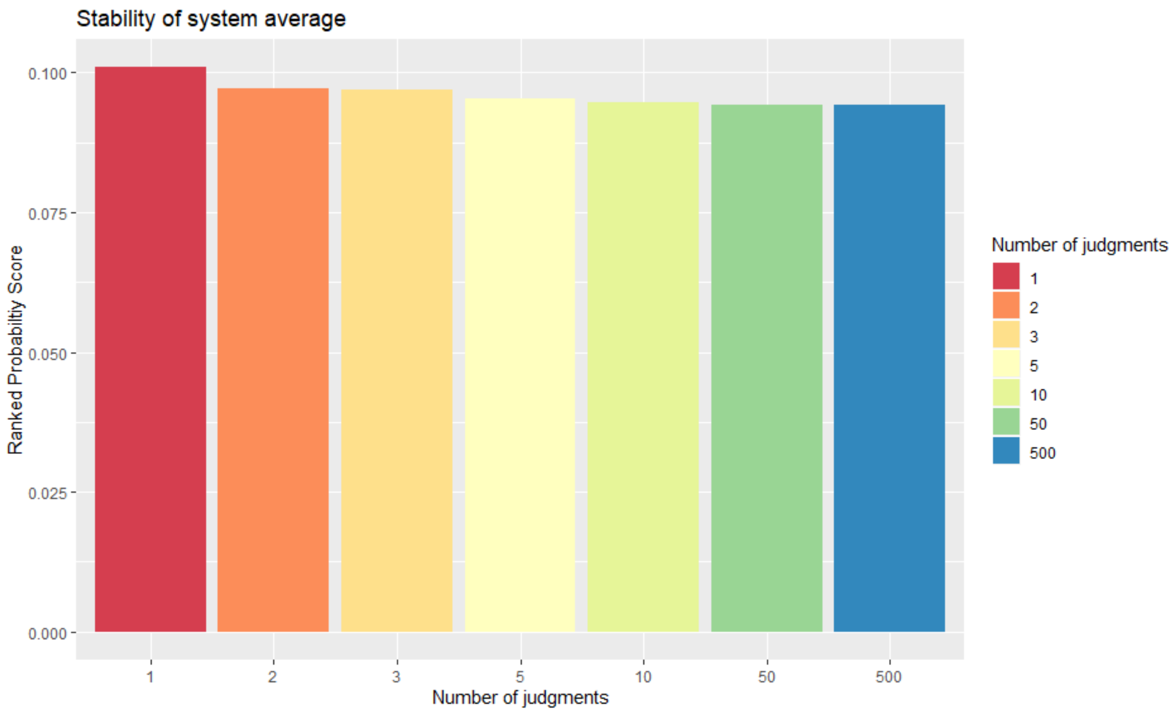

Figure 3.10: RPS error of the judgment-based predictions. The horizontal axis indicates the number of judgments used in the feature calculation for the average system. Note that the horizontal axis is not linear.

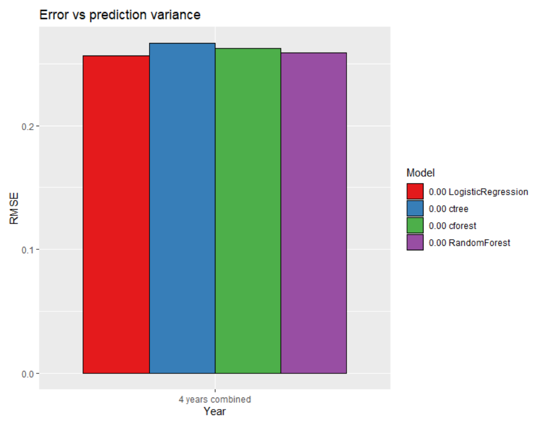

Figure 3.11: RMSE of the different prediction models. The predictions are made with the output-based features. The error is the average of the years 2007, 2009, 2010 and 2011.

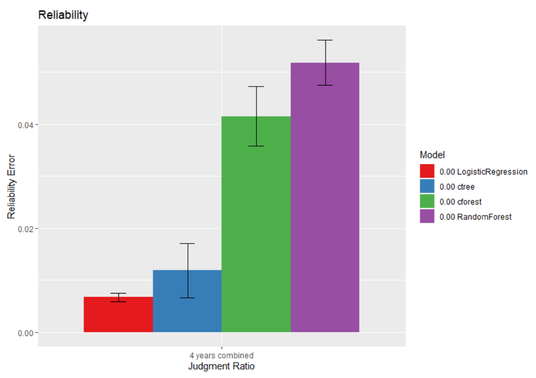

Figure 3.12: Reliability error of the different prediction models. The predictions are made with the output-based features. The error is the average of the years 2007, 2009, 2010 and 2011.

the year and the number of judgments the distribution of genres in the judged dataset varies widely. Figure 3.7 shows and example of six genres in the 2007 dataset. Where in red the distribution is shown of the 10% highest weighted songs. The weights are calculated for the cumulative gain metric. Clearly the jazz genre is over-represented in this selection. Perhaps one system is performing very well on jazz music. The performance of this system will be overestimated when 10% of the highest weighted judgments are used. These are two possible problems when using the weighted selection that can reduce the reliability of the model.

The goal of the weighted judgment selection is to minimize the variance in the prediction. The proposed method does only consider one form of influence. The reduction in variance due to having the judgment value instead of a prediction. However, there is at least one other way judgments can reduce variance. Judgments also create judgment-based features, which are used to make judgment-based predictions. In general the judgment-based predictions are more precise compared to the output-based judgments and therefore reduce the variance. To make the optimal selection this second variance reduction should be taken into account when selecting songs to judge. A song retrieved by two systems is not necessarily reducing variance more than a song retrieved only once. If this second song is of a very commonly retrieved artist, the aART feature can make the prediction of many other documents more precise out weighting the variance reduction of being retrieved by two systems. When the only goal is reducing the variance in the prediction this should also be taken into account.

The weighted selection method reduces the variance in the prediction faster then random selection. By incorporating a minimal number of judgments per system could improve the stability of the aART feature. However, there remain still multiple reasons why this type of selection could be biased and make the predictions less reliable. In addition the proposed calculation of weights is not optimal. The goal of this chapter is improving the reliability of the model. The most appropriate way of selection is therefore random. With one constraint that each system should have at least 10 judgments or none at all.

3.4. Reducing overconfidence via better prediction algorithms

There are different types of models that can be used to make the predictions based on the calculated features. In general a suitable prediction method should be accurate and precise, but also have some kind of estimation of the variance. This is to determine how confident the predictions are. Another important property is taking the ranking of the value into account. The similarity variable that is predicted has ordered values. As explained in section 2.2.1 the ordinal logistic regression is a good fit. However this prediction method is not proven to be the most suitable. Investigating if other predictions methods are better can be valuable. In section 3.2.4 the stability of a features is taken into account with the feature count. This improves reliability. The idea of this additional feature is to influence the confidence of the predictions made. Feature interaction like this could possibly be modeled better by other methods. This is something to take into account when

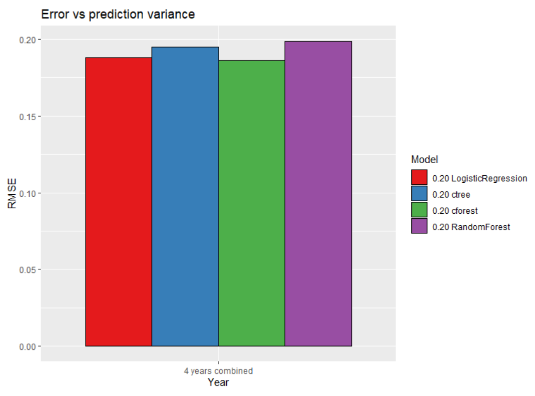

Figure 3.13: RMSE of the different prediction models. The predictions are made with the judgment-based features and all judgments available. The error is the average of the years 2007, 2009, 2010 and 2011.

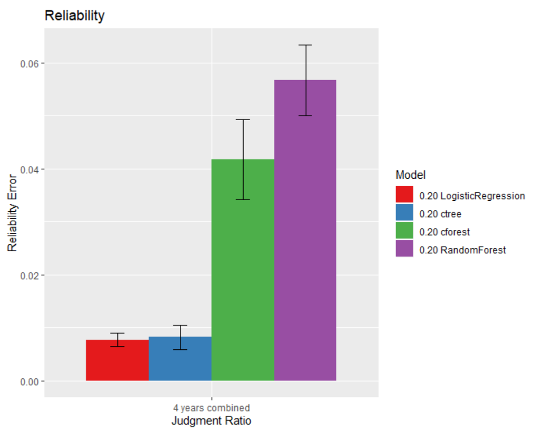

Figure 3.14: Reliability error of the different prediction models. The predictions are made with the judgment-based features and all judgments available. The error is the average of the years 2007, 2009, 2010 and 2011.

looking for other models. Decision trees are a flexible prediction method which can handle complex decision regions[31]. A step further is combining multiple classifiers, such as decision trees. Here an ensembles of classifiers is constructed. By voting a prediction can be made. This way of predicting gained recently more interest[23]. One method to group these classifiers is bagging. This method often produces better results then the single classifier and does not overfit easily[24]. A popular example of this is Random Forest[3]. Another method of grouping is boosting, which often outperforms bagging. The downside is that it is more susceptible for overfitting[24]. Which is a troubling property when the goal is reliability. The most widely used boosting method is AdaBoost[41]. However this method is not applicable for ordinal variables. As explained it is beneficial to use the ordinal nature of the similarity variable. The framework introduced by Hothorn et al. [16] is applicable. This framework introduces conditional inference trees which can predict ordinal variables(ctree)[17]. Applying this to random forest is suitable as well. When the ensemble of classifiers are conditional inference trees the model can predict ordinal variables as well(cforest).

3.4.1. Comparing performance of different prediction algorithms

As pointed out, modeling the influence of the feature count is an important aspect. Also the excluding self for the judgment-based model is considered as an important adjustment that is without doubt good to make. In general testing the newly proposed models in the improved setting is the most valuable. This means for the output-based model the system-based approach will be used. For the judgment-based predictions both the feature count for artists is added and the excluding self improvement is applied. Now when looking at the RMSE in Figure 3.11 the logistic regression model has the lowest error for the output-based predictions. For the judgment-based features it is a tie with cforest (Figure Figure 3.13). These error results do not give a reason to switch prediction method. The main goal of this chapter is increasing the reliability of the model. Again for both the output and judgment-based feature none of the other methods is outperforming logistic regression (Figure 3.12 and 3.14). Logistic regression actually makes a probabilistic prediction which reflects the probabilities from the training data. The probabilities outputted by random forest are based on the voting of the ensemble. Simply put when more trees predict a value it is probably more likely to be true, but this does not directly reflect the probabilities in the dataset. To address this mismatch probabilistic recalibration can be used[15]. Literature on this topic is mostly focused on calibration of binary predictions. A simple way to calibrate the predictions in this non-binary setting is by adjusting the variance of the prediction according to the prediction error. Figure 3.15 shows the variance in the prediction compared to the variance in the error. For Random Forest we see that the predictions are the most confident with a variance around 0.02, the error variance is higher. For logistic regression and ctree the variance in the prediction actually matches the prediction error on the training set. By raising predicted probabilities to a power less than 1.0 the prediction flatten out. In Figure 3.16 cforest is recalibrated to match variance. Random forest is just too much off to compensate for by exponent. Often the predictions contain to many zeros to deal with. Although there are most likely other ways to calibrate, the results do not indicate competitive performance to the logistic re-

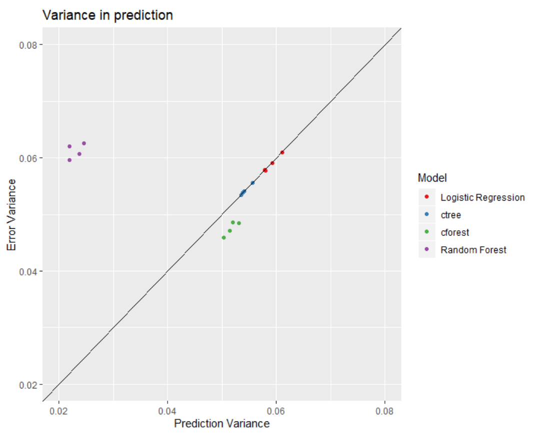

Figure 3.15: Prediction variance compared to variance in the error. For the different models and the output-based predictions.

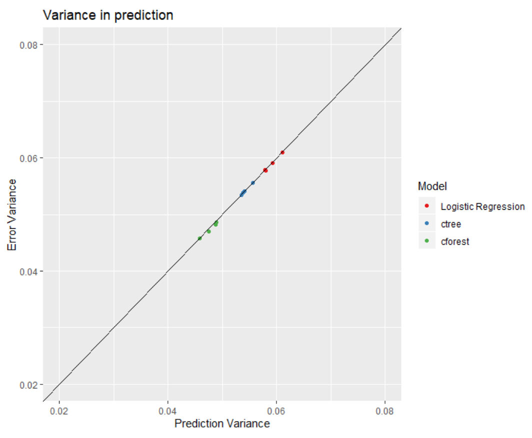

Figure 3.16: Prediction variance compared to variance in the error. For the different models and the output-based predictions. This time the predictions are calibrated based on variance.

gression. After calibration ctree and cforest are again compared against logistic regression. However, logistic regression is still the most reliable method. Only in terms of reliability the judgment-based predictions of ctree score almost as good as logistic regression.

Hyper parameter tuning

The final step to make a model better is by correctly tuning the hyper parameters. In literature ctree tuning is mostly focused on the min criterion and the max depth. Although the algorithm proves to be not very sensitive to the parameter tuning. Statistically improvements were observed in a quarter of the datasets[25]. To investigate if improvements can be made in this setting both min criterion and the max depth are adjusted in isolation. The min criterion is the value of the test statistic that must be exceeded in order to implement a split. If the value is close to 1 the tree will contain less splits and therefore contain less nodes. In other words the tree will become less complex. The upside of this is that it makes the model less susceptible for overfitting. The downside is it can incorporate less information from the training data into the model. Figure 3.18 shows that this simplifying of the tree will increase the prediction error on the training set. A simpler tree can model less of the relation between feature and prediction label. In terms of reliability Figure 3.17 shows that a less complex tree improves reliability. Remember that the reliability does not take into account how informative the prediction is a uniform prediction can still obtain perfect reliability. The max depth parameter is similar again it controls in an other way the complexity of the tree. Again we observe the same pattern where it is a trade-off of error in the prediction vs reliability. RPS is a stable metric to take both reliability and accuracy into account. By using 10 fold cross validation the best hyper parameters can be found. Turns out that for almost every year for both models the default values of 0.95 and Inf are the best. With these tuned parameters ctree is not outperforming the original model based on RPS. Not for the output-based model and not for the judgment-based model. As explained the judgment-ratio influences the predictions Figure 3.21 compares the performance with different ratios. Still the logistic regression outperforms ctree.

3.5. Results

The goal of this chapter is to prevent overconfidence. The main strategy to accomplish this is by improving the similarity predictions of the model. When the similarity prediction itself is overconfident this will manifest in all three applications. The estimation of the absolute performance score, the relative performance score, and the ranking. Therefore the suggested improvements are validated in these three applications in this section. Due to the lack of data, it is not wise to constantly test all possible adjustments during research on the final application. This increases the chances of observing less overconfidence just by luck, without actually improving the estimations outside this dataset. Therefore all suggested adjustments are first evaluated on similarity prediction level. Only significant improvements to the predictions are evaluated at the application level. This leveled approach of evaluation aims to improve the validity of the results.

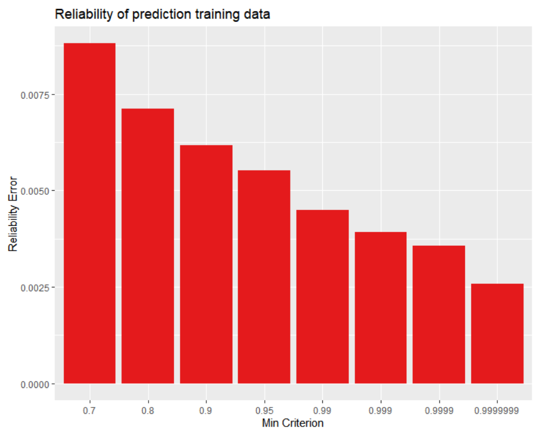

Figure 3.17: The reliability error of the ctree similarity prediction on the training set itself. Plotted against the Min Criterion value.

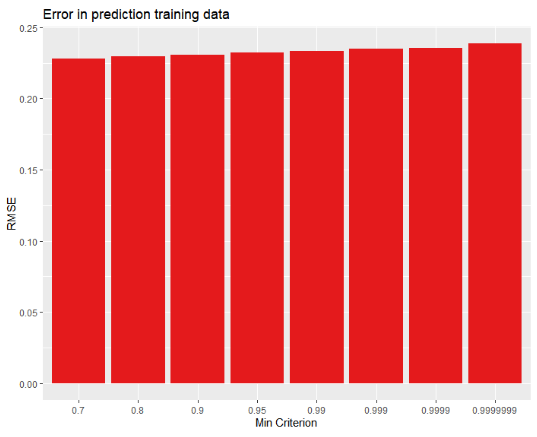

Figure 3.18: The RMSE error of the ctree similarity prediction on the training set itself. Plotted against the Min Criterion value.

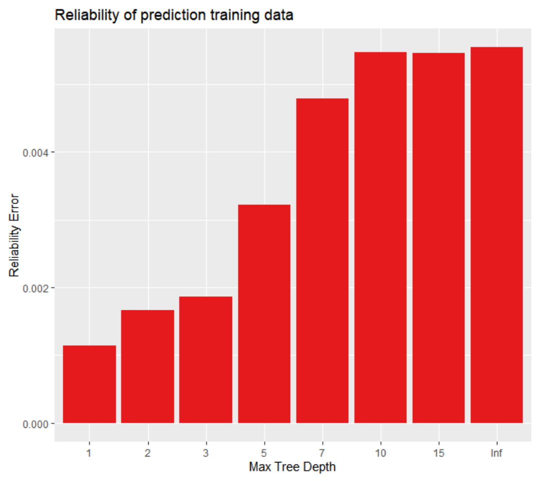

Figure 3.19: The reliability error of the ctree similarity prediction on the training set itself. Plotted against the Max Tree Depth.

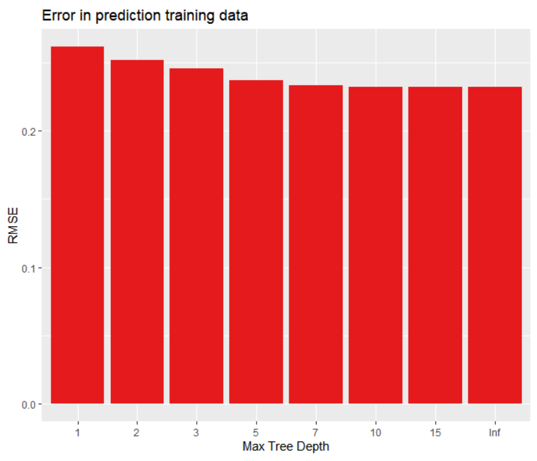

Figure 3.20: The RMSE error of the ctree similarity prediction on the training set itself. Plotted against the Max Tree Depth.

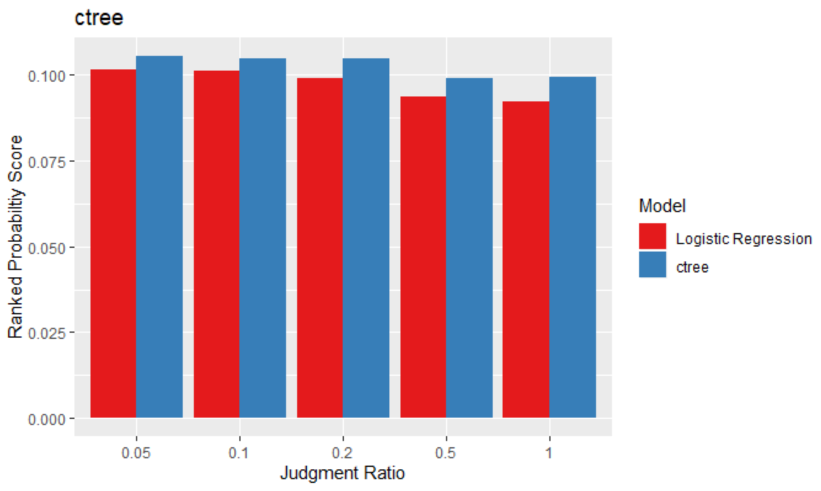

Figure 3.21: The RPS score of the judgment-based predictions comparing the logistic regression against ctree. The RPS score is the average of the 4 years combined. On the horizontal axis, the judgment ratio ranges from 0.05 to 1.

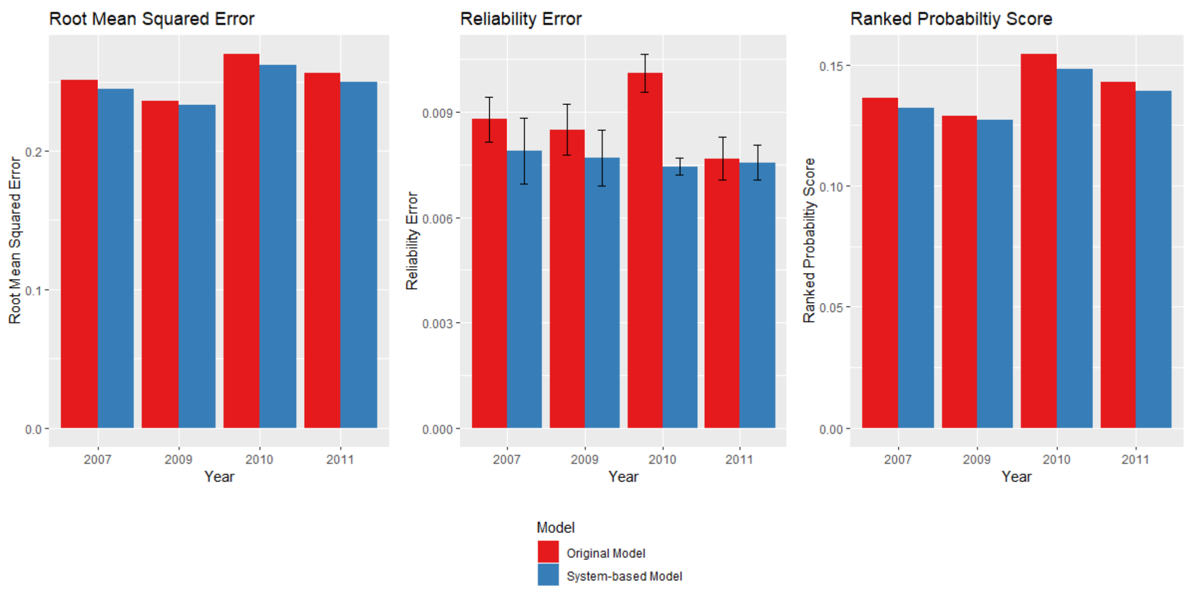

Figure 3.22: RMSE, Reliability and RPS error metrics. The output-based original model in red is compared against the proposed system-based model in blue.

3.5.1. Probabilistic predictions

To obtain a confidence that actually matches the accuracy of the estimations. The probabilistic predictions need to match the actual observed values. By decomposition, the reliability error can be obtained from RPS. This tells how well the probabilities match the observed values. For overconfident predictions there will be mismatch and therefore a higher reliability error. However, improving reliability at the cost of precision is trivial. Simply always predicting the prior distribution will obtain high reliability. This is at the expense of precision and makes the predictions less informative which is undesirable. Therefore, results will also show the change in RMSE and RPS alongside the reliability error.

Output-based Model

The only significant improvement made to the output-based features is making fART and fGEN system-based. In an attempt to reduce the influence of the retrieved songs by the system itself. Figure 3.22-left shows that the RMSE is improved in all four datasets. This means the expected value of the predictions is closer to the actual label. The middle figure shows the reliability improvement in particular for the year 2010. This year has the fewest retrieval systems in the dataset. This means that the ratio of retrieved songs belonging to the system itself is the highest. This could explain the difference compared to the other years. When both the RMSE and the reliability improve it is not surprising that the RPS as shown on the right side also improves. These results clearly illustrate the benefits of using the system-based approach instead of the original model.

Judgment-based Model

For the judgment-based model, the most significant improvement is the excluding self for the features aART and aSYS. This adjustment can be considered as fixing a mistake in the original model and you could even argue that independently of the changes in error it is the better way to calculate the features. Nevertheless, we see that the prediction error also decreases as visible in Figure 3.23-left. The middle figure shows it benefits reliability significantly. Which both again lead to an improvement in the RPS. The second suggested improvement for the judgment-based model is taking the feature count of aART into account. In Figure 3.23 in green both suggested improvements are applied. There is only some slight improvement in terms of RMSE and RPS on top of the excluding self fix (in blue). This improvement is only visible for the year 2010. Again this year is the year with the fewest retrieval systems and thus the most improvement is expected. In terms of reliability, the most improvement is observed in the low judgment ratio setting. As already shown in Figure 3.6. In Figure 3.23-middle no improvement in reliability can be observed. This is not surprising, because it is in a setting with all judgments available. Overall there is no clear downside of incorporating this feature count, but in a low judgment ratio setting there is an upside and a slight improvement of RMSE and RPS is observed.

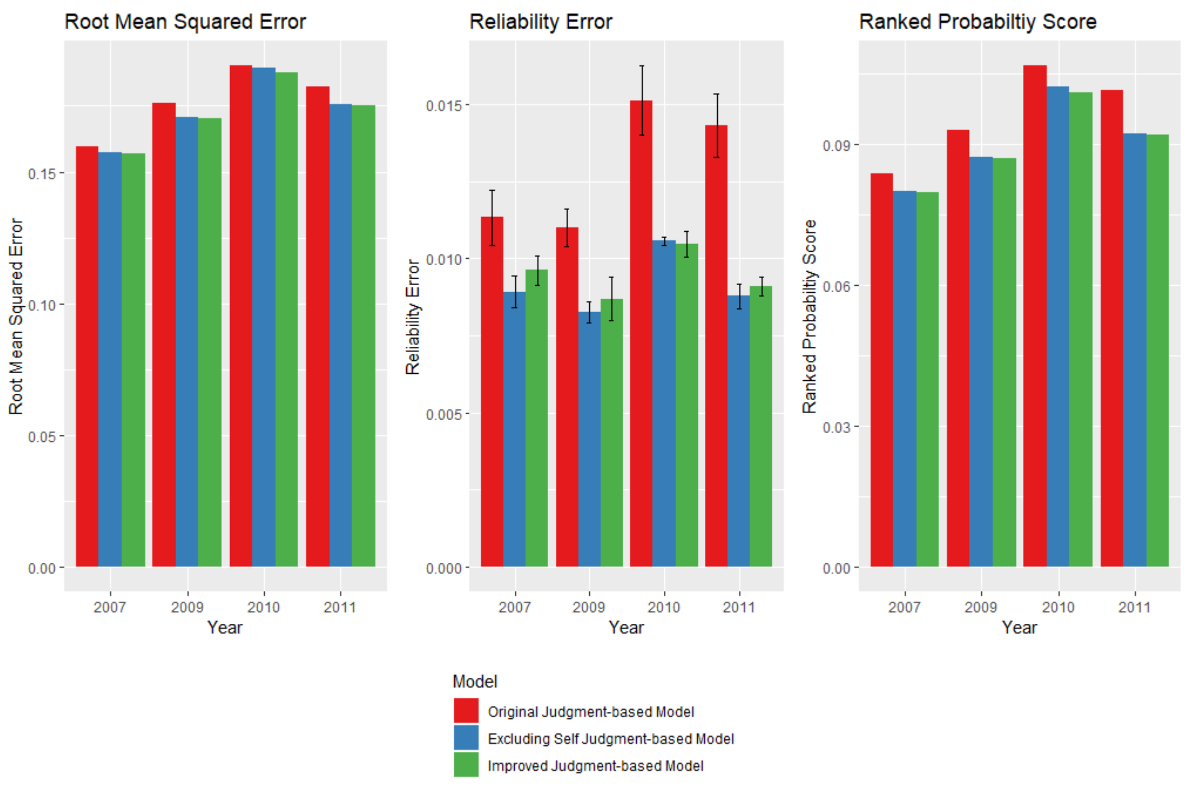

Figure 3.23: RMSE, Reliability and RPS error metrics of judgment-based predictions in a setting with all judgments available.

3.5.2. Absolute performance estimation

For both the output-based and judgment-based model improvements are suggested that increase the reliability of the similarity predictions. To evaluate the impact of this for the absolute performance estimation the same confidence intervals as in the beginning of this chapter are calculated. For ease of comparison, the values of the original model are placed side-by-side to the newly obtained values. Table 3.5 clearly shows improvement for both the new output-based and judgment-based models. For both CG and DCG the ratio scores within the interval are closer to the confidence level. This is observed for all 4 confidence levels. This illustrates that the proposed adjustments to both the output-based and the judgment-based model contribute to reducing the overconfidence of the absolute performance score estimation.

For completeness, NDCG and RBP are also displayed. However, as already pointed out the estimations are clearly off. This is not solved by making more reliable similarity predictions. For the original model almost no observed values fall in the calculated intervals. When applying the proposed adjustments this leads to no observed values at all within the intervals. It is not informative to try to explain this difference before the obvious overestimation is addressed.

3.5.3. Relative performance estimation

In Table 3.6 the suggested improvements are again compared to the original model, this time for the relative performance estimation. For the output-based model improvement is observed for CG and DCG. The improvement is small, which is not surprising considering that the original model is not much off the confidence level. In contrast, the judgment-based Model has a much larger mismatch. Here the improvement is also more significant. These results substantiate that the proposed adjustments improve the relative performance estimation by reducing overconfidence.

As explained the relative performance estimation is clearly less affected by the overestimation of NDCG and RBP. However, the overconfidence is still significantly higher then for CG and DCG. The table shows that the system-based approach slightly increases the mismatch for the output-based model. The improved judgment-based model decreases the mismatch a little. Still, both metrics are clearly overconfident also significantly more than CG and DCG. This makes it not valuable to analyze the observed difference. There is

CG@5

Confidence level	Within confidence interval			
	Output-based Model		**Judgment-based Model**	
	Original Model	System Based Model	Original Model	Improved Model
0.60	0.17	**0.19**	0.13	**0.21**
0.90	0.35	**0.42**	0.27	**0.35**
0.95	0.38	**0.48**	0.29	**0.40**
0.99	0.46	**0.63**	0.38	**0.48**

DCG@5

Confidence level	Within confidence interval			
	Output-based Model		**Judgment-based Model**	
	Original Model	System Based Model	Original Model	Improved Model
0.60	0.25	**0.29**	0.19	**0.27**
0.90	0.40	**0.48**	0.33	**0.40**
0.95	0.46	**0.60**	0.35	**0.46**
0.99	0.58	**0.75**	0.52	**0.56**

NDCG@5

Confidence level	Within confidence interval			
	Output-based Model		**Judgment-based Model**	
	Original Model	System Based Model	Original Model	Improved Model
0.60	0.00	0.00	0.00	0.00
0.90	0.00	0.00	**0.02**	0.00
0.95	0.00	0.00	**0.02**	0.00
0.99	0.00	0.00	**0.02**	0.00

RBP@5

Confidence level	Within confidence interval			
	Output-based Model		**Judgment-based Model**	
	Original Model	System Based Model	Original Model	Improved Model
0.60	0.00	0.00	**0.02**	0.00
0.90	0.00	0.00	**0.02**	0.00
0.95	0.00	0.00	**0.02**	0.00
0.99	0.00	0.00	**0.02**	0.00

Table 3.5: The ratio of actual performance scores that fall in a confidence interval of the model estimation. The model estimates the absolute performance of all systems in the 4 years of the MIREX dataset. Both the output-based and judgment-based models are used. The better numbers are marked bold.

clearly still another factor making these predictions less reliable.

3.5.4. Rank estimation

In Table 3.7 the system-based model is compared against the original output-based model. The results are very similar, for the system-based model we see that the most confident bin increases a bit in size. This indicates that overall the model estimations are a bit more confident. The confidence and accuracy of the metrics CG and DCG already match quite well in the original model. This is not changed significantly by the system-based model. In some bins the difference between accuracy and confidence is decreased wherein other bins it is increased. By looking at the total confidence and accuracy of the model the numbers are very similar to the original model. The metrics NDCG and RBP are clearly overconfident in the original model. However, this is not improved in the system-based model. Again it could be caused by the structural overestimation of these metrics. The same pattern is observed for the judgment-based model and the suggested improvement. As Table 3.8 shows again NDCG and RBP are overconfident. The suggested improvement does not significantly change the results. The CG and DCG metric of the judgment-based estimations are not overconfident in the original model nor in the improved model.

CG@5

Confidence level	Within confidence interval			
	Output-based Model		**Judgment-based Model**	
	Original Model	System Based Model	Original Model	Improved Model
0.60	0.45	**0.47**	0.38	**0.41**
0.90	0.75	**0.76**	0.62	**0.67**
0.95	0.77	**0.79**	0.67	**0.73**
0.99	0.91	**0.92**	0.77	**0.79**

DCG@5

Confidence level	Within confidence interval			
	Output-based Model		**Judgment-based Model**	
	Original Model	System Based Model	Original Model	Improved Model
0.60	0.58	**0.59**	0.46	**0.54**
0.90	0.81	**0.84**	0.72	**0.75**
0.95	0.90	**0.91**	0.77	**0.80**
0.99	0.98	0.98	0.90	**0.96**

NDCG@5

Confidence level	Within confidence interval			
	Output-based Model		**Judgment-based Model**	
	Original Model	System Based Model	Original Model	Improved Model
0.60	**0.28**	0.27	0.17	**0.22**
0.90	**0.46**	0.44	0.31	**0.37**
0.95	**0.55**	0.50	0.35	**0.41**
0.99	**0.66**	0.62	0.42	**0.50**

RBP@5

Confidence level	Within confidence interval			
	Output-based Model		**Judgment-based Model**	
	Original Model	System Based Model	Original Model	Improved Model
0.60	**0.38**	0.36	0.24	**0.30**
0.90	**0.59**	0.55	0.38	**0.44**
0.95	**0.67**	0.63	0.43	**0.50**
0.99	0.78	0.78	0.52	**0.63**

Table 3.6: The ratio of actual performance scores that fall in a confidence interval of the model estimation. The model estimates the relative performance between all systems pairs in the 4 years of the MIREX dataset. Both the output-based and judgment-based models are used. The better numbers are marked bold.

The judgment-based model does not show any indication of overconfidence when all judgments are available. However, as already found the original model is less reliable when fewer judgments are used to calculate the features. A graph is a more informative representation to see the influence of the judgment ratio. The confidence and accuracy of each bin are now plotted as a point. To make the graph less cluttered the two least confident bins are taken together. These two bins in general contain the least points. Figure 3.24 shows the influence of the judgment ratio in the original judgment-based model. When no judgment-based prediction can be made a uniform distribution is predicted. Surprisingly the opposite influence is found a low judgment ratio is the opposite of overconfident. This can be observed to what extent the points lie above the diagonal. A possible explanation for this is that performance scores of queries for the same system are dependent. This is not taken into account. The graph shows that the chance of predicting correctly which system is better based on only a few predictions is fairly high. When for example the model only predicts the songs for 1 query out of 100 queries with the judgment-based model. These predictions are in general pretty precise. Thus most of the time the model will predict correctly which system performs better for this 1 query. For all other queries, a uniform prediction is assumed, the expected scores are all equal for these 99 queries. This means the final prediction of which system scores is the best on average in all 100 queries solely depends on this 1 query.

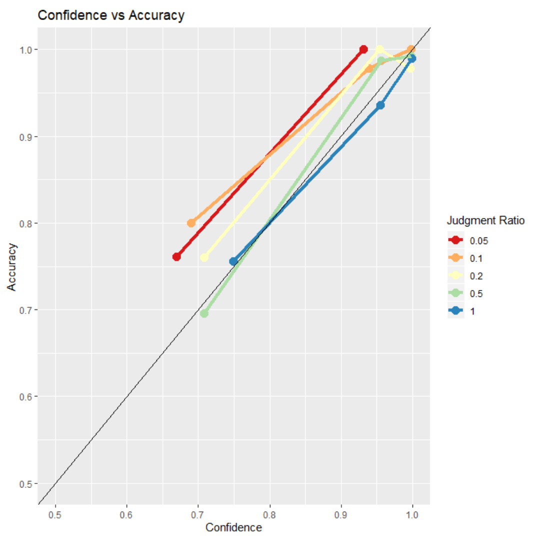

Figure 3.24: The accuracy and confidence of judgment-based predictions. In case the judgment-based features are not available a uniform distribution is assumed as prediction.

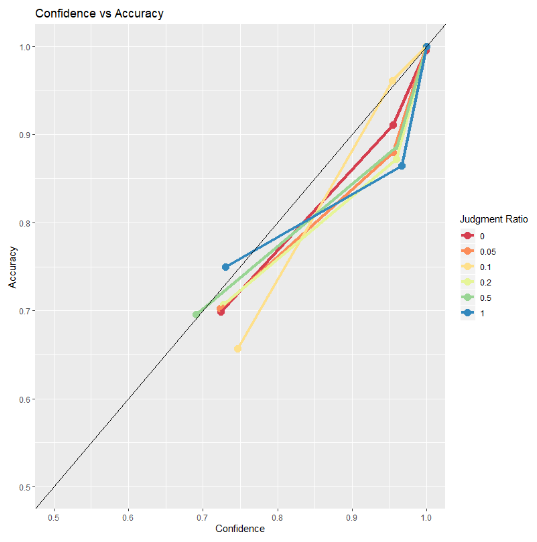

Figure 3.25: The accuracy and confidence of judgment-based predictions. In case the judgment-based features are not available the output-based model is used to predict.

The 100 query scores for the same system do correlate. If this system is good most queries will score high compared to a bad system. If the systems perform consistently and there is a clear difference in performance the best system will obtain the highest query score for most queries. To make it even more concrete assume that the better system outperforms in 80 of the queries. Then there is roughly a 80% where the model predicts the ranking correct with only knowledge of 1 query. However, the confidence of this estimation will be low due to the high variance in the other 99 queries. Eventually, both output-based and judgment-based models will be used in combination. Without any judgments, the model will have informative predictions with much less variance than a uniform prediction. Again the same graph is created this time falling back to the output-based predictions instead of the uniform predictions. Figure 3.25 shows that in this case, most points are below the diagonal. This indicates the estimations are somewhat overconfident. There is no clear influence or pattern observed. How the judgment-ratio would affect the overconfidence. Combine this with the information in Table 3.8 that shows that the judgment-based estimations are not overconfident. Based on this it is reasonable to conclude that when making the predictions with both the output-based and judgment-based features the overconfidence is mainly caused by the output-based features. In general, the ranked estimations are much less affected by the overconfidence in the underlying similarity predictions. Which is in stark contrast with the absolute performance estimation. The small observed overconfidence for the output-based predictions is in this case not removed by the system-based model.

3.6. Summary

The goal of this chapter is to address the overconfidence in the estimations of the model. In order to form trustworthy conclusions, the problem is broken down and quantified with the reliability error. The hypothesis is that addressing the overconfidence on the level of similarity prediction will decrease the overconfidence in the estimation of the performance scores. The features used in the output-based model are partially based on the retrieved song of the AMS system itself. Depending on the year the number of systems in the dataset differs. In the year with the least systems, the retrieved songs by the system self influence the features the most. This seems to negatively impact the reliability. Calculating features on system level instead of the original song level proves to be beneficial. This applies to the features that represent the ratio of retrieved songs with a particular artist or genre. Making the output-based model system-based, is a significant improvement. Increasing reliability and decreasing RMSE error. This means that the predicted probabilities better match the actual observed values and the expected value is in general closer to the actual value. For the judgment-based model two improvements are found. Excluding the similarity label of the song itself in the training set when calculating features is, without doubt, a modification that should be made. Again this increases the reliability and decreases the error. The second improvement is taking into account the number of values a

CG@5

Confidence Range		Original Model			System Based Model		
Lower bound	Upper bound	Confidence	Accuracy	In bin	Confidence	Accuracy	In bin
0.49	0.65	0.57	0.71	28 (8%)	0.58	0.62	26 (8%)
0.65	0.90	0.80	0.69	55 (16%)	**0.79**	**0.72**	50 (15%)
0.90	0.99	**0.95**	**0.91**	56 (16%)	0.95	0.85	53 (16%)
0.99	1.00	1.00	1.00	202 (59%)	1.00	1.00	212 (62%)
Total		**0.92**	**0.90**	341 (100%)	0.93	0.90	341 (100%)

DCG@5

Confidence Range		Original Model			System Based Model		
Lower bound	Upper bound	Confidence	Accuracy	In bin	Confidence	Accuracy	In bin
0.49	0.65	0.57	0.65	40 (12%)	0.56	0.64	36 (11%)
0.65	0.90	**0.79**	**0.74**	70 (21%)	0.80	0.71	72 (21%)
0.90	0.99	0.96	0.93	56 (16%)	**0.96**	**0.95**	40 (12%)
0.99	1.00	**1.00**	**1.00**	175 (51%)	1.00	0.99	193 (57%)
Total		0.90	0.89	341 (100%)	0.90	0.89	341 (100%)

NDCG@5

Confidence Range		Original Model			System Based Model		
Lower bound	Upper bound	Confidence	Accuracy	In bin	Confidence	Accuracy	In bin
0.49	0.65	0.57	0.62	8 (2%)	**0.56**	**0.60**	10 (3%)
0.65	0.90	**0.78**	**0.72**	25 (7%)	0.81	0.71	24 (7%)
0.90	0.99	0.96	0.56	18 (5%)	**0.96**	**0.82**	11 (3%)
0.99	1.00	**1.00**	**0.95**	290 (85%)	1.00	0.93	296 (87%)
Total		0.97	0.90	341 (100%)	0.97	0.90	341 (100%)

RBP@5

Confidence Range		Original Model			System Based Model		
Lower bound	Upper bound	Confidence	Accuracy	In bin	Confidence	Accuracy	In bin
0.49	0.65	0.57	0.47	15 (4%)	**0.58**	**0.64**	11 (3%)
0.65	0.90	**0.79**	**0.86**	21 (6%)	0.80	0.62	16 (5%)
0.90	0.99	**0.96**	**0.66**	29 (9%)	0.95	0.63	30 (9%)
0.99	1.00	1.00	0.96	276 (81%)	1.00	0.96	284 (83%)
Total		**0.96**	**0.91**	341 (100%)	0.97	0.91	341 (100%)

Table 3.7: Relative performance predictions grouped on confidence into 4 bins. The predictions are made with the output-based model. The judgment values themselves are not used in making the prediction. The lower and upper bound indicate the lowest and highest confidence of the bin. The confidence and accuracy columns are the average of the bin. The last column is the ratio of predictions that fall into this bin. The closer matching confidence and accuracy pairs are marked bold.

feature is based on. This seems to improve the reliability only in a setting with just a few judgments available. This adjustment does not seem to contribute to improving the accuracy.

The suggested improvements to the similarity prediction reduce overconfidence in both absolute and relative estimation. In particular the overconfidence in the absolute performance estimation benefits the most for both output-based and judgment-based predictions. The relative performance estimation suffers less from overconfidence to start with. Nevertheless, the proposed adjustments still decrease the overconfidence even more. In particular the judgment-based improves notably. The rank estimations are a different story. Here the judgment-based are not observed to be overconfident. The output-based estimations do indicate slight overconfidence, but this is not reduced with the system-based approach. Overall the proposed adjustments on feature level proof to reduce overconfidence in many of the applications notably. However, all the observed improvements are only for the CG and DCG performance metrics. The predictions made with the NDCG and RBP remain unreliable and require more investigation.

CG@5

Confidence Range		Original Model			Improved Model		
Lower bound	Upper bound	Confidence	Accuracy	In bin	Confidence	Accuracy	In bin
0.49	0.65	**0.57**	**0.52**	23 (7%)	0.58	0.68	25 (7%)
0.65	0.90	0.81	0.84	67 (20%)	0.80	0.83	63 (18%)
0.90	0.99	0.95	0.94	62 (18%)	**0.95**	**0.95**	64 (19%)
0.99	1.00	1.00	0.99	189 (55%)	1.00	0.99	189 (55%)
Total		0.93	0.92	341 (100%)	0.92	0.93	341 (100%)

DCG@5

Confidence Range		Original Model			Improved Model		
Lower bound	Upper bound	Confidence	Accuracy	In bin	Confidence	Accuracy	In bin
0.49	0.65	0.58	0.71	24 (7%)	**0.59**	**0.69**	35 (10%)
0.65	0.90	**0.79**	**0.83**	90 (26%)	0.80	0.88	80 (23%)
0.90	0.99	**0.96**	**0.97**	76 (22%)	0.95	0.97	66 (19%)
0.99	1.00	1.00	0.99	151 (44%)	1.00	0.99	160 (47%)
Total		0.90	0.93	341 (100%)	0.90	0.93	341 (100%)

NDCG@5

Confidence Range		Original Model			Improved Model		
Lower bound	Upper bound	Confidence	Accuracy	In bin	Confidence	Accuracy	In bin
0.49	0.65	0.55	0.33	3 (1%)	**0.55**	**0.75**	4 (1%)
0.65	0.90	**0.81**	**0.65**	17 (5%)	0.78	0.53	19 (6%)
0.90	0.99	0.96	0.79	28 (8%)	**0.95**	**0.88**	24 (7%)
0.99	1.00	1.00	0.96	293 (86%)	1.00	0.96	294 (86%)
Total		0.98	0.92	341 (100%)	**0.98**	**0.93**	341 (100%)

RBP@5

Confidence Range		Original Model			Improved Model		
Lower bound	Upper bound	Confidence	Accuracy	In bin	Confidence	Accuracy	In bin
0.49	0.65	**0.60**	**0.69**	13 (4%)	0.60	0.73	11 (3%)
0.65	0.90	0.83	0.79	24 (7%)	**0.79**	**0.76**	29 (9%)
0.90	0.99	0.95	0.79	38 (11%)	**0.97**	**0.88**	43 (13%)
0.99	1.00	1.00	0.96	266 (78%)	**1.00**	**0.97**	258 (76%)
Total		0.97	0.92	341 (100%)	**0.96**	**0.93**	341 (100%)

Table 3.8: Relative performance predictions grouped on confidence into 4 bins. The predictions are made with the judgment-based model and 100% of the judgments available. The judgment values themselves are not used in making the prediction. The lower and upper bound indicate the lowest and highest confidence of the bin. The confidence and accuracy columns are the average of the bin. The last column is the ratio of predictions that fall into this bin. The closer matching confidence and accuracy pairs are marked bold.

4 Overestimation of absolute performance metrics

In the previous chapter, the model is critically investigated and improved on three different fields: the features, the judgment selection method, and the prediction algorithm. All improvements do not help in removing the overestimation of the two performance metrics NDCG and RBP.

In this chapter that overestimation is addressed. Starting in section 4.1, here the overestimation is quantified. In section 4.2 the cause of the overestimation is explained. After understanding the cause, two solutions are proposed in section 4.3. In this section, both solutions are evaluated to what extent it solves the overestimation.

4.1. Quantifying the overestimation

As explained in Section 2.3.2 the expected performance is for some metrics systematically overestimated. In this chapter we are considering the metrics CG, CG, NDCG and RBP. Because the measures CG and DCG are the most stable for relative effectiveness scores, and RBP and NDCG the most stable for absolute effectiveness scores[38]. The first step is to verify to what extent it remains a problem after the improvements suggested in Chapter 3. The results of the absolute performance metric in terms of overconfidence for sure indicate that it is not solved.

Figure 4.1 plots the average performance score of each team in the 4 MIREX datasets against the estimated performance score. The actual score is calculated with the judgment values. The estimated performance score is calculated based on the expected value of the probabilistic predictions. These predictions are made with the improved models from the previous chapter. For the output-based model this means the system-based model is used instead of the original. On the left side the predictions are all made with the output-based model. The output-based model is still overestimating the NDCG and RBP. Which is not the case for the other 2 metrics. For the judgment-based model the excluding self features and the feature count of the aART are used. On the right side of the figure all judgments are used in the feature calculation and the predictions are made with the judgment-based model. Again we see that both NDCG and RBP are overestimated. Even though the model is making more precise predictions. The overconfidence is most prevalent for the output-based predictions, this is why most figures in the report show the output-based model to evaluate possible solutions. Nevertheless, the judgment-based predictions also needs to be evaluated and at the end of this chapter the relevant figures are shown to do this where the judgment-based prediction and the combination of both is taken into account.

4.2. Underestimation of optimal rank

Both overestimated metrics contain normalization. This normalization depends on the highest possible score a query can obtain. Perfectly listing the documents with the highest score in descending order. To predict the normalized metrics this perfect score needs to be estimated with the model. Estimating the values close to

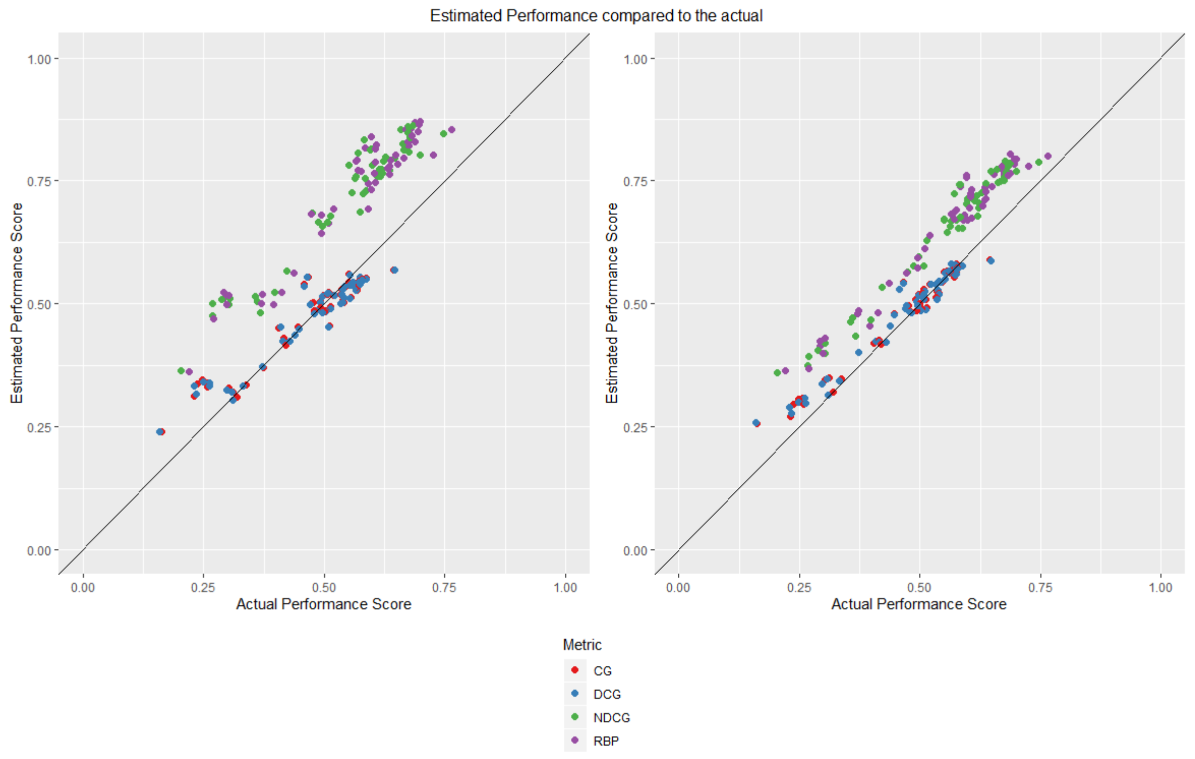

Figure 4.1: Estimated vs. actual performance scores of the teams in MIREX 2007, 2009, 2010 and 2011. On the left side the predictions are made with the improved output-based model. On the right side the predictions are made with the improved judgment-based model with all judgments available.

the limit of the rating range(highest and lowest values) with the expected value can introduce some bias. Due to the uncertainty in the prediction the expected value will always be a bit less than 1.0 or a bit more than 0.0. Only when the model makes a 100% certain prediction the expected value can be 0 or 1. Both the output-based and the judgment-based model have uncertainty in the predictions. In general, when estimating an average of many retrieved songs there will not be a clear bias. Biases towards the middle from both sides will cancel out. Only when most of these songs have a similarity label at the same edge of the range. Then the bias towards the middle is visible in the estimation of the group average. When looking at the CG and DCG points in Figure 4.1-Left a slight bias can be observed. The lowest scoring systems are a bit overestimated and the highest scoring systems are a bit underestimated. For the judgment-based predictions shown in Figure 4.1-Right this trend is less noticeable. Which can be explained by the lower variance in the predictions. The effect of bias towards the middle becomes more significant when estimating the maximal query score. Which is needed for the metrics that normalize the score. In this case most values will often be close to 1. But rarely close to 0. Thus the bias is clearly one sided, namely predictions are underestimating the actual values. Figure 4.2-Left shows a heatmap on how the optimal DCG query scores is estimated in the broad scale. This optimal query score is the denominator in the NDCG formula as formulate in formula 2.4. For simplicity the values optimal scores are scale to the range 0 to 1. By plotting the actual optimal query score against the estimated optimal query score in a heatmap overestimation or underestimation becomes clearly visible. The figure clearly shows underestimation on the left side for the broad scale and on the right side there is underestimation the fine scale. This underestimation of the optimal score means the denominator in the NDCG is estimated to low, which makes the estimated NDCG in total too high. Exactly the same is happening for the RBP score, which also has this normalization component that is underestimated.

4.3. Solutions to reduce the overestimation

As explained in the previous section when using the expected value of the predictions with uncertainty the value 1 can never be obtained. Consider the query where 50 songs are labeled in the broad scale (0.0, 0.5, 1.0). Of which 45 have the label 1.0 and 5 have label 0.5. They all have the probabilistic prediction of (0.0,

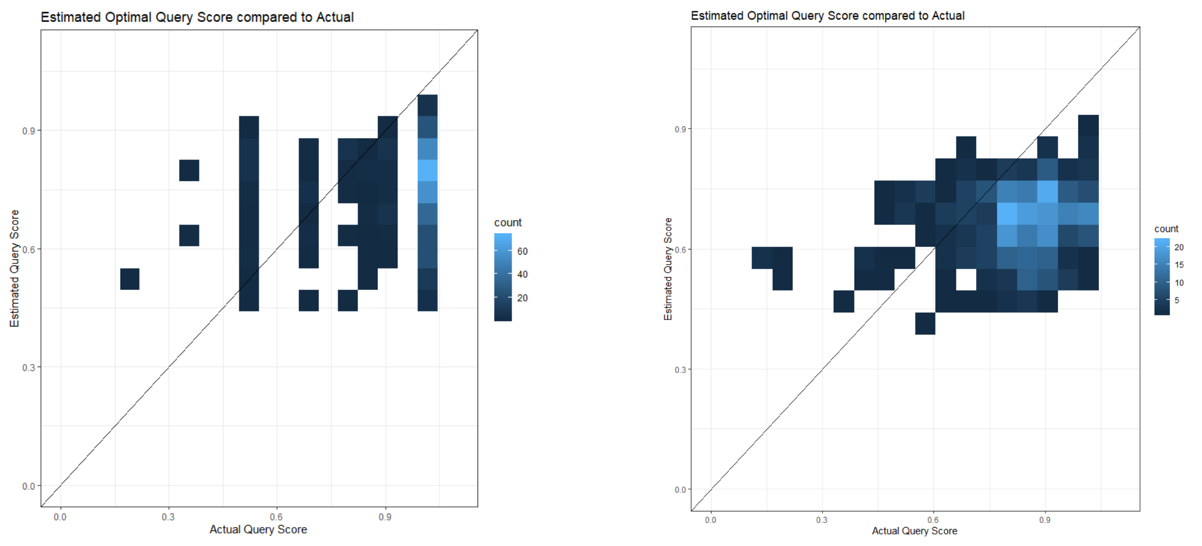

Figure 4.2: Estimated vs. actual performance scores in MIREX 2007, 2009, 2010 and 2011. Predictions are made with output-based model. Left Broad scale is used and on the right side the Fine scale.

0.1, 0.9). This prediction is perfectly reliable, because the probabilities reflect the observed data. Based on this prediction all songs will have the expected value of 0.95. In the datasets only the first 5 retrieved songs are used. Therefore the optimal ranking consists of 5 songs of all label 1.0. But when estimating this using the expected value 5 times a song of 0.95 similarity is obtained. The next two sections introduce a different method instead of using the expected value to address this problem.

4.3.1. Label of max probability

Instead of using the expected value, the label of the max probability can be used to reduce this bias. In the starting example of this section this would mean all 50 songs obtain the label 1.0, because this label has the highest predicted probability. This would give the correct optimal ranking for the example, namely 5 songs with label 1. As a starting point it is interesting to look at what kind of distribution is obtained when using the most probable label. Figure 4.3-Left compares the distribution of the actual judgment labels and the label of the max predicted probability label. Predicted by the output-based prediction model. For the Broad scale the max label distribution looks promising, because it is similar to actual label distributions. However, the same figure for the Fine scale shows more mismatch in the distributions (Figure 4.3-Right). This shows that the max probable label of all output-based predictions never obtains the value 1.0. Which raises the concern if this solution will indeed prevent the bias towards the middle. Another similar approach would be rounding the expected value to actual label values. If an expected value is than 0.95 in the fine scale this would be rounded to 1.00. However, the same calculations show that this looks less promising. The output-based predictions rarely reaches 0.8, in all the 4 years only 8 times are rounded to 0.9 and never reach 1.0.

Quantifying the improvement

The first step to quantify if this approach helps, is looking at the estimation of the optimal rank. Figure 4.4 shows again the heatmap that plots the estimated optimal query scores against the actual, but this time the predictions are made based on the label with the highest probability instead of the expected value. The clear underestimation is gone. On the left-side the broad scale shows no overestimation at al. On the right-side the fine scale still shows some overestimation, but compared to Figure 4.2 improvement can be observed. The next step is to calculate the performance scores for the metrics NDCG and RBP and using the max probability label instead of the expected value. Figure 4.5 both methods, for NDCG and RBP in the Broad and in the Fine scale. Using the max label (red) instead of the expected value (blue) proves to be beneficial for both metrics in both scales. The values lay closer to the diagonal and also both above and below, the clear overestimation is reduced. However most of the high performance scores are still overestimated and the lowest scores are slightly underestimated in particular in the fine scale. Nevertheless, overall a clear reduction of overestimation of both metrics.

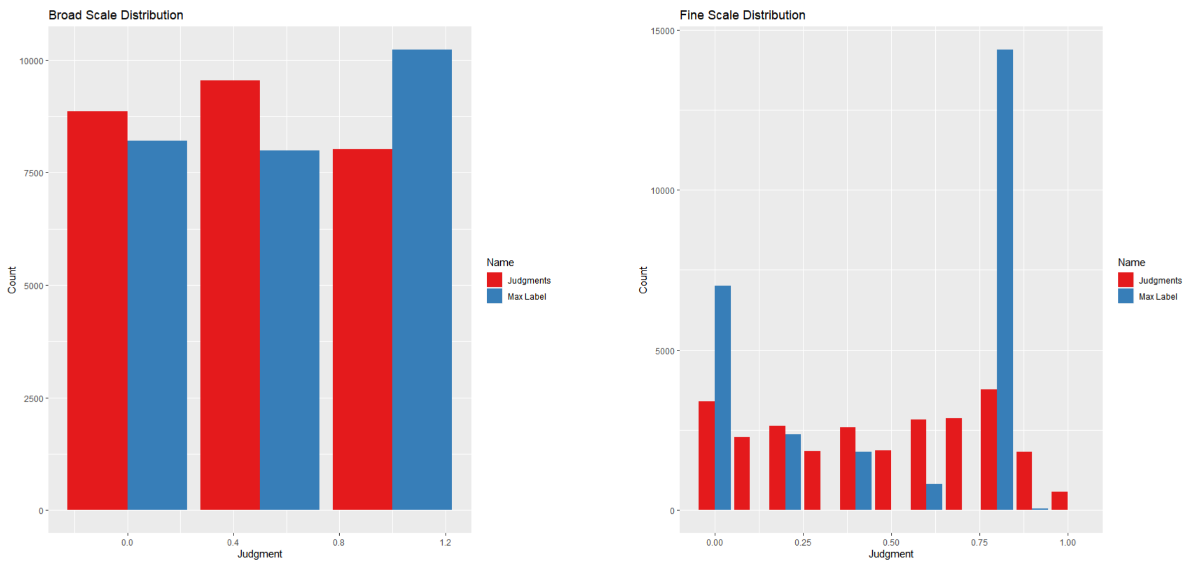

Figure 4.3: Red is the distribution of the actual judgment labels in the years 2007, 2009, 2010 and 2011 combined. On the left side the broad scale is used thus the possible values are 0, 0.5 and 1.0. On the right side the fine scale is used thus the possible values range from 0.0 to 1.0 with a step size of 0.1. The blue bars in both plots are again the 4 years combined, but instead of the judgment label the label with the maximum probability is used from the output-based predictions.

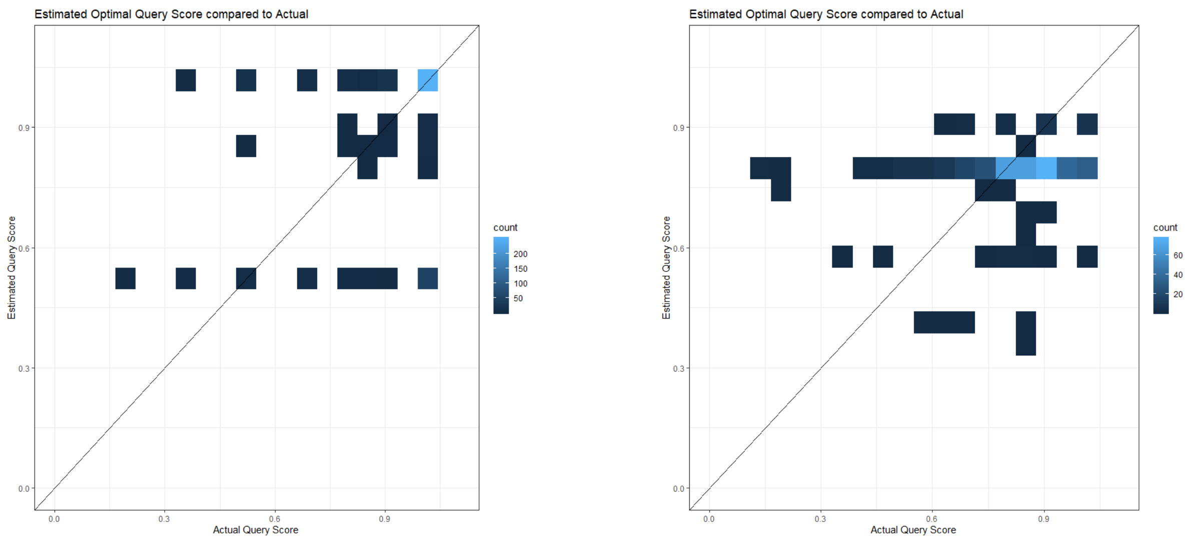

Figure 4.4: Estimated vs. actual optimal query scores in MIREX 2007, 2009, 2010 and 2011. Predictions are based on the max label of the output-based model predictions. Left the Broad scale is used, on the right the Fine scale is used.

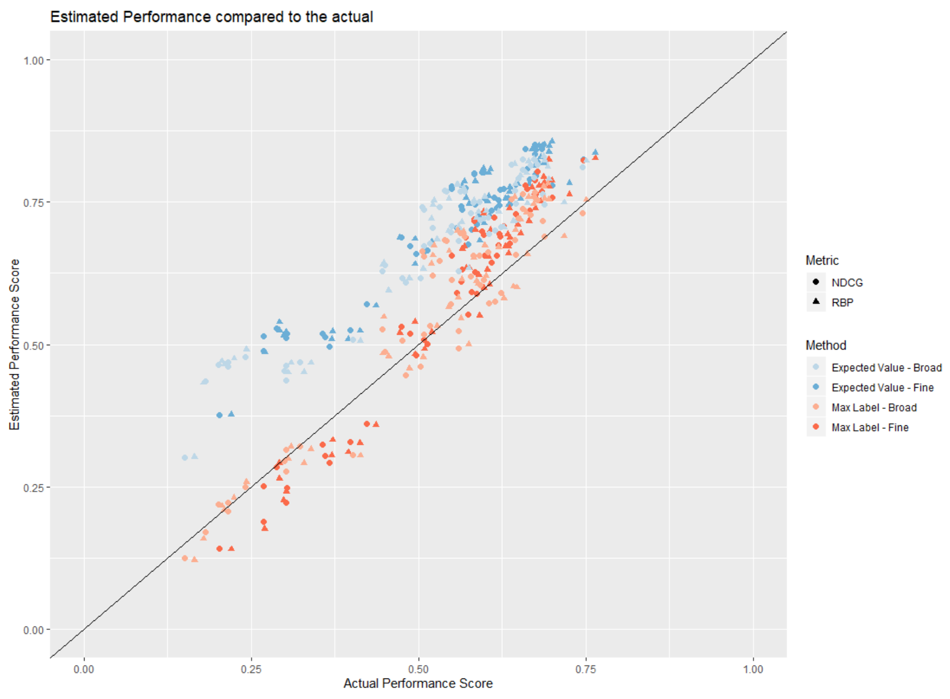

Figure 4.5: Estimated vs. actual optimal query scores in MIREX 2007, 2009, 2010 and 2011. Predictions made with the output-based model. For both NDCG and RBP indicates with different symbols. In blue the estimations are based on the expected value and in red they are based on the label with the highest probability. The saturation of the color indicates which scale is used.

4.3.2. Weighted sampling

Instead of using the expected value, sampling from the probabilistic prediction can be used. The chance of sampling a label is determined by the probability it has in the prediction. If a label is predicted with a chance of 0.9 then this is also the chance this label is sampled as 'expected' value for this prediction. This could help in reflecting the actual judgments more appropriately. To make it more concrete the sampling method can be applied to the starting example of this section by sampling from the (0.0, 0.1, 0.9) probabilities 50 times, 50 values are obtained. It is possible that we obtain around 45 labels of 1.0, 5 labels of 0.5 and no 0.0 labels. The last will always be true in this setting, because the probability for that label is 0. To determine the optimal score for the retrieval of 5 songs the best 5 songs are selected which in this case are all of label 1.0.

The sampling will introduce some additional noise in the prediction. But if this approach is unbiased then some decrease in precision is acceptable. To reduce the noise introduced the sampling and top 5 selection can be repeated multiple times and than averaged. To reduce noise further all small probabilities are removed before sampling. All probabilities below $1/n_{\mathcal{L}}$ are not sampled from. This also means that the uniform predictions are not contributing. This is on purpose, because the uniform predictions only occur when nothing can be used to predict the similarity. These predictions should not have influence on the outcome.

Quantifying the improvement

First step is again to look at the estimation of the optimal query score. Figure 4.6 shows that the optimal query score estimation is less biased for both the Broad and Fine scale. The most dense area lies exactly on the diagonal. Indicating that the estimated score matches the actual score. Clearly not all points lie around the diagonal this error is to some extent expected. Simply because all predictions of the model contain some variance. For the broad scale there is no clear bias in the error. Some points are above the diagonal and some are below. However, for the fine scale the lower scoring queries are overestimated. As indicated by the dark color in the heatmap there are just a few queries that have a query score below 0.6 where this overestimation occurs. Most scores are above this value and are estimated accurately and unbiased. The next to evaluate this sampled approach of estimation is by Figure 4.7 shows the sampled estimations of performance score for the different metrics in the Broad scale. Both NDCG and RBP are less biased, compared to using the expected

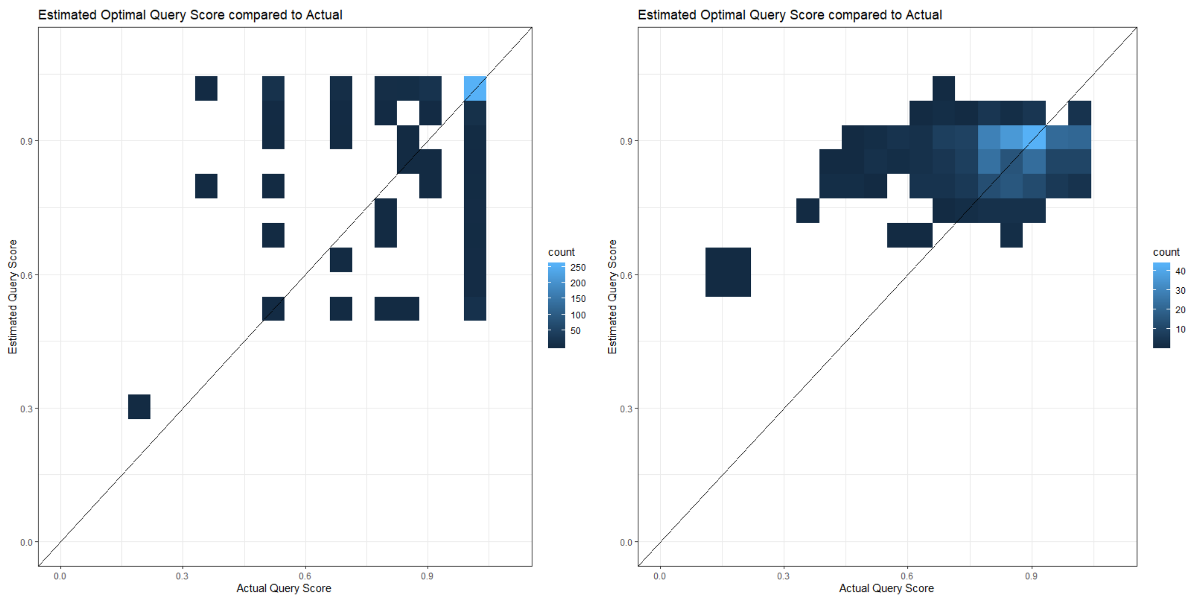

Figure 4.6: Estimated vs. actual optimal query scores in MIREX 2007, 2009, 2010 and 2011. Predictions are based on the proposed sampling method of the output-based model predictions. Left the Broad scale is used, on the right the Fine scale is used.

value. In contrast to the max label the improvement is as significant for the Fine scale as the Broad scale, see Figure 4.5. This makes this method more promising.

In practice the model will most likely make output- and judgment- based predictions when estimating the performance score of systems. Depending on the number of judgments available this ratio will differ. Figure 4.8-Left shows the influence of the different amount of judgments on the performance score estimation. The predictions are made with the expected value for the metrics NDCG and RBP. More judgment-based predictions makes the estimations closer to the actual values. The clear overestimation is again visible. By calculating the metrics with the proposed sampling method we obtain Figure 4.8-Right. Again the higher judgment ratio makes the estimations move closer to the actual value. However, because there is no obvious bias to start with there is no shift observed, but a nice convergence around the diagonal.

When the model is used, the obtained judgments will also be used to estimate system performance more precise. By adding these in the metric calculation the estimations will naturally converge faster to the actual value. This is shown in Figure 4.9. It is self-evident that with all judgments the predictions contain no error and are equal to the actual values. Besides that observation nothing significantly differs from what has been observed so far. There is no notable difference between the two metrics.

4.4. Summary

When estimating similarity labels of songs, values close to the border of the range have a bias. The lowest values are overestimated and the highest values are underestimated. Due to the uncertainty in the predictions, the expected value will not reach the lowest or highest value. Therefore the prediction is biased towards the middle. For most metrics this does not introduce significant problems. In general the bias introduced by songs at the lowest value are compensated by songs with the highest value. As a result the average estimation of the system performance is not biased. When a system consistently retrieves songs scoring close to one of the extremes bias is observed. In the datasets the systems range in performance scores from 0.15 to 0.64 for CG and DCG. In the predictions for the lowest scoring systems, some of this bias can be observed. But for metrics that use normalization, this bias becomes a significant problem. Normalized metrics such as RBP and NDCG require the optimal rank for each query. This automatically includes a not compensated amount of highest scoring songs this causes underestimation. When using a systematically underestimated optimal score to normalize the actual score becomes overestimated.

Instead of using the expected value the label with the maximum probably can be used. This is a form of

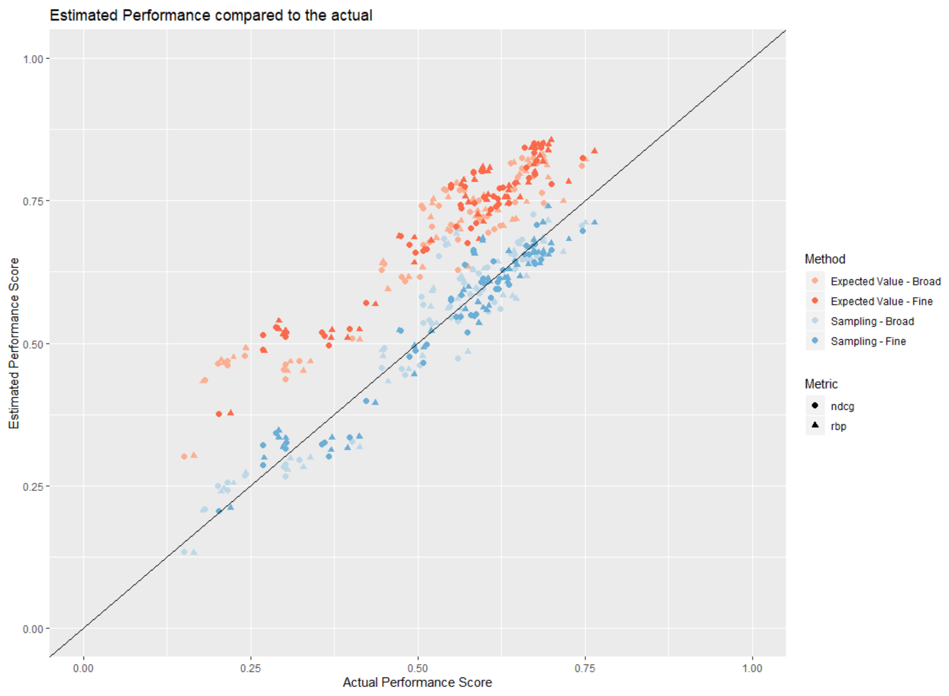

Figure 4.7: Estimated vs. actual query scores in MIREX 2007, 2009, 2010 and 2011. Predictions made with the output-based model. For both NDCG and RBP indicates with different symbols. In blue the estimations are based on the expected value and in red the proposed sampling method is used. The saturation of the color indicates which scale is used.

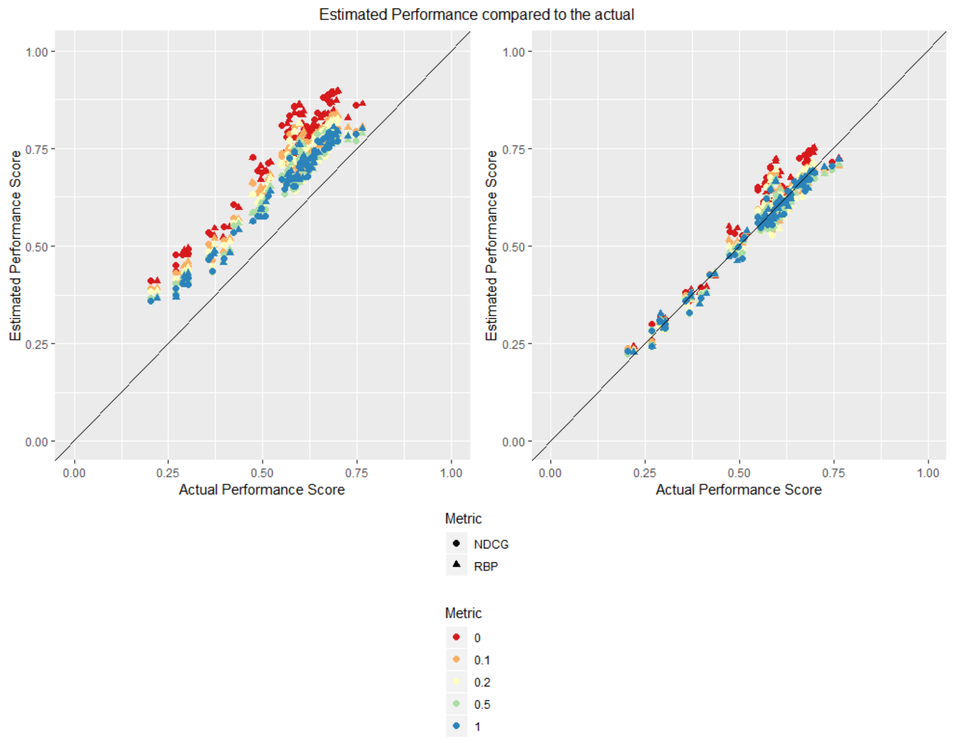

Figure 4.8: The estimation of the metrics NDCG and RBP compared to the actual values. On the left the expected value is used to estimate. On the right the proposed sampling method is used. The colors indicate the judgment ratio.

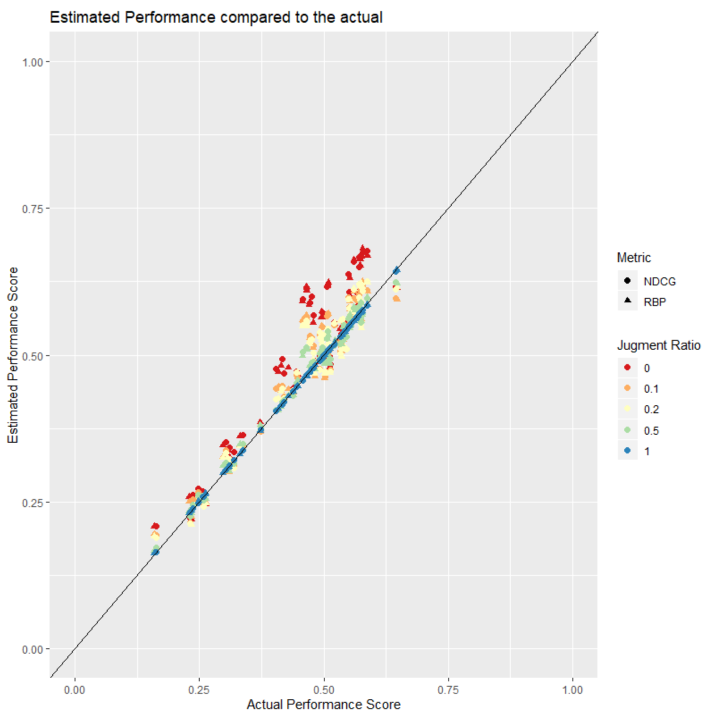

Figure 4.9: The estimation of the normalized metrics compared to the actual value. The colors indicate the judgment ratio used. The estimations are made with the available judgments, judgment-based predictions and output-based predictions.

rounding which seems to reduce the problem. This simple adjustment looks promising but does not resolve the bias entirely. A better approach is the use of sampling this is again instead of the expected value. Each label gets a chance to be selected as predicted value based on the corresponding predicted probability. The intuition behind this is that when the model predicts for 100 songs the highest label with a probability of 0.1, the expected value of none of the songs will actually be 1.0. Whereby sampling according to the probabilistic prediction around 10 of the songs will obtain this highest label. This approach removes the clear overestimation for normalized metrics and also improves the less obvious bias in estimating systems that have a score close to the border of the scoring range.

5 Conclusion and future work

The music information retrieval community has long recognized the need for a more comprehensive evaluation paradigm[9][40]. The low-cost model [39] is addressing the high cost of judgment in AMS. However, the model suffers from overconfidence and overestimation which limits the applicability of the model. In this work, both biases are addressed. The overconfidence is improved by critically going over three components: the feature calculation, the method of judgment selection and the prediction algorithm. The second and last problem that is addressed is the overestimation of normalized metrics.

5.1. Conclusion

The estimations for the absolute performance are clearly overconfident, this overconfidence is also present by the relative estimation only to a lesser extent. The predictions of the similarity label form the basis of all estimations, this is the first aspect addressed in this thesis. By decreasing the mismatch between the predicted probabilities and the actual observed values, which is called the reliability error, the overconfidence is reduced. This work introduces multiple adjustments to the calculated features that decrease the reliability error. For the output-based model calculating features on a system level instead of song level improves the reliability, but also the RMSE. This error changes from 0.256 to 0.250 on average over all 4 years of the MIREX dataset. For the judgment-based model two improvements are found. First, preventing mismatch in what information the training features contain compared to the feature used for prediction. Second, taking into account the number of values used in the feature calculation. When this amount is too low features become less stable. These adjustments make the predicted probabilities more reliable and the improvements also reduce the RMSE from 0.182 to 0.175. The second criticized step in the model is the weighted judgment selection. When finding the optimal songs to judge, the goal is to reduce the prediction variance as much as possible. Not only the judgment value itself will contribute, but also the additional judgment-based predictions that can be made are more precise than output-based predictions. This should be taken into account to make the weighted selection reduce the variance the most. Nevertheless, this work did not continue in this direction in search of the optimal weighted selection criteria. Any specific selection criteria can introduce unwanted biases. The selection should be a good representation of the entire dataset. The focus of this work is on removing biases, in line with this goal the judgment selection is made random. With one important restriction. Every system should have 0 judgments or at least a minimum of 10. Because the average judgment for a system is used as one of the features when making judgment-based predictions. When only a few judgments are available for a system this average is not stable. To make reliable predictions it is beneficial to set a minimum threshold. The third component is the algorithm used to make the predictions, here the already used ordinal logistic regression proves to be the most reliable. The suggested adjustments reduce the overconfidence in both the absolute and the relative performance estimation.

The proposed adjustments do not solve the overestimation of the metrics NDCG and RBP. Those metrics are still structurally overestimated by the model. This is due to the normalization component of these metrics. To normalize the optimal score needs to be estimated. However, this is structurally underestimated. By normalizing with a too low score the actual estimated score is too high. This is caused by the bias towards the middle. This affects scores that are at the boundary of the value range. The predictions of the model contain uncertainty, therefore the expected value is never exactly the minimal or maximal similarity label. This bias is present for all metrics, but when a system has an average score some of the estimated queries are underestimated where others are overestimated. The system score is an average of many queries and is not seriously affected by this. Only when a system scores outstanding high or low the bias can be observed. However, when estimating the optimal ranking most values will be at the upper border of the range. Due to the bias towards the middle, the optimal rank is underestimated. Instead of using the expected value a sampling method is proposed. This does not underestimate the high scores and thus eliminates the overestimation for the normalized metrics. Also, a slight reduction of the bias for the other metrics is observed. In the case of extreme scores, in particular the very low scoring systems that are present in the datasets.

The overconfidence in performance estimation is reduced by the proposed improvements on the similarity prediction level. Combined with the better estimations for normalized metrics make the results of the low-cost model better interpretable. This work also shows that the methodology of the low-cost evaluation model requires careful design. By going thoroughly over all the steps in the model many biases can be eliminated or mitigated. This offers a positive prospect for further development of trustworthy and applicable low-cost evaluation methods in AMS.

5.2. Future Work

Some of the covered topics in this work raise additional questions which are not answered. Simply because the answer would not directly be relevant for the problems that were addressed. Still, these questions could improve the low-cost evaluation model in the future. In particular for the weighted selection criteria, a better approach than random can be created as already shown. This would lead to obtaining more precise estimations within the same amount of judgments. Additional steps can be made by taking into account that the judgment-based predictions are more precise than the output-based predictions. In the perfect situation, the selection criteria are directly based on the reduction in variance due to model change and requiring fewer predictions. On top of this, the selection criteria should not introduce biases. Possibly more insight can also be obtained by using another dataset. Starting with how applicable this model is on other AMS datasets. This is an important next step to verify. The performance in less controlled datasets could vary. The MIREX datasets are well-constructed datasets created by a scientific community. The four datasets used in this thesis all have 5 retrieved songs per query. This number is on purpose not higher because all these songs need to be judged by humans which is costly. Also, the amount of systems per dataset is in the same order ranging from 8 to 18. What AMS systems are used to base features on could also influence the results. The submitted systems to MIREX are all designed in the same scientific setting with a common goal. These systems do not necessarily reflect all the different AMS systems available. Investigating the impact of changing retrieval systems could be valuable. To some extent, the model already incorporates some difference between datasets in terms of AMS systems. This is based on the degree of overlap between systems in each dataset. When the low-cost evaluation model does not generalize to datasets with those different properties additional steps can be taken. For example, evaluating or adjusting the predictions based on the few available judgments. The judgment-based model, can be retrained based on the few judgments available to steer the model to some extent if needed. Or the variance of the predictions can be adjusted where appropriate. To prevent any overconfidence.

Adding more systems and judging all the extra retrieved songs is costly for MIREX. However, for the model having more data when predicting is beneficial. Even when none of this data is judged. The model proves to predict already fairly confident and accurate without any judgments at all. There are three ways of adding more data, which could make the model predict the average system performance more precise. This data does not need to be labeled and thus does not introduce extra judgment cost:

1. **Increasing the number of queries.** For every query, the model makes and prediction of the performance score of the system. Averaging more of these predictions will decrease the variance of final average system performance.
2. **Increasing the number of systems in the dataset.** This could make most of the calculated features more stable as explained in section 3.2.3.
3. **Increasing the number of retrieved songs per query** This bases the calculation of features like fGEN and fART on more values. It is hard to hypothesis what would be the influence of this. Besides the

system-based approach of those features is possibly also influenced differently by the number of retrieved songs per query.

In particular the first, increasing the number of queries, could potentially contribute a lot to the precision of the performance score estimation. There is no clear stopping point where it is not beneficial for the estimation to add more queries. The second, increasing number of systems, is most likely more limited in added value. At some point the feature calculation is stable and adding more systems does not necessarily add value. Changing the dataset in this way does also impact the judgment-based predictions. The number of retrieved songs per query is influential. In general when there are more retrieved songs the number of retrieved songs with the same artist also increases. This already influences one of the judgment-based features. Investigating all this in future work is beneficial for more general adoption of the low-cost evaluation model. The usage of a larger dataset could also lead to a methodology that requires even fewer judgments.

Bibliography

[1] Adam Berenzweig, Beth Logan, Daniel PW Ellis, and Brian Whitman. A large-scale evaluation of acoustic and subjective music-similarity measures. *Computer Music Journal*, 28(2):63–76, 2004.

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

[3] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[4] Glenn W Brier. Verification of forecasts expressed in terms of probability. *Monthly weather review*, 78(1): 1–3, 1950.

[5] Chris Buckley and Ellen M Voorhees. Retrieval evaluation with incomplete information. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 25–32, 2004.

[6] Guillem Candille and Olivier Talagrand. Evaluation of probabilistic prediction systems for a scalar variable. *Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography*, 131(609):2131–2150, 2005.

[7] George Casella and Roger L Berger. *Statistical inference*, volume 2. Duxbury Pacific Grove, CA, 2002.

[8] Cyril W Cleverdon. The significance of the cranfield tests on index languages. In *Proceedings of the 14th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 3–12, 1991.

[9] J Stephen Downie. The scientific evaluation of music information retrieval systems: Foundations and future. *Computer Music Journal*, 28(2):12–23, 2004.

[10] J Stephen Downie. The music information retrieval evaluation exchange (2005–2007): A window into music information retrieval research. *Acoustical Science and Technology*, 29(4):247–255, 2008.

[11] J Stephen Downie and Michael Nelson. Evaluation of a simple and effective music information retrieval method. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 73–80, 2000.

[12] J Stephen Downie, Andreas F Ehmann, Mert Bay, and M Cameron Jones. The music information retrieval evaluation exchange: Some observations and insights. In *Advances in music information retrieval*, pages 93–115. Springer, 2010.

[13] Edward S Epstein. A scoring system for probability forecasts of ranked categories. *Journal of Applied Meteorology*, 8(6):985–987, 1969.

[14] Christopher AT Ferro and Thomas E Fricker. A bias-corrected decomposition of the brier score. *Quarterly Journal of the Royal Meteorological Society*, 138(668):1954–1960, 2012.

[15] Carlo Graziani, Robert Rosner, Jennifer M Adams, and Reason L Machete. Probabilistic recalibration of forecasts. *International Journal of Forecasting*, 2019.

[16] Torsten Hothorn, Kurt Hornik, and Achim Zeileis. Unbiased recursive partitioning: A conditional inference framework. *Journal of Computational and Graphical statistics*, 15(3):651–674, 2006.

[17] Torsten Hothorn, Kurt Hornik, and Achim Zeileis. ctree: Conditional inference trees. *The Comprehensive R Archive Network*, 8, 2015.

[18] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *2008 Eighth IEEE International Conference on Data Mining*, pages 263–272. Ieee, 2008.

[19] Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446, 2002.

[20] M Cameron Jones, J Stephen Downie, and Andreas F Ehmann. Human similarity judgments: Implications for the design of formal evaluations. In *ISMIR*, pages 539–542, 2007.

[21] Peter Knees, Tim Pohle, Markus Schedl, and Gerhard Widmer. Combining audio-based similarity with web-based data to accelerate automatic music playlist generation. In *Proceedings of the 8th ACM international workshop on Multimedia information retrieval*, pages 147–154, 2006.

[22] Beth Logan and Ariel Salomon. A music similarity function based on signal analysis. In *ICME*, pages 22–25, 2001.

[23] Wei-Yin Loh. Fifty years of classification and regression trees. *International Statistical Review*, 82(3): 329–348, 2014.

[24] Richard Maclin and David Opitz. An empirical evaluation of bagging and boosting. *AAAI/IAAI*, 1997: 546–551, 1997.

[25] Rafael Gomes Mantovani, Tomáš Horváth, Ricardo Cerri, Sylvio Barbon Junior, Joaquin Vanschoren, and André Carlos Ponce de Leon Ferreira de Carvalho. An empirical study on hyperparameter tuning of decision trees. *arXiv preprint arXiv:1812.02207*, 2018.

[26] Alistair Moffat and Justin Zobel. Rank-biased precision for measurement of retrieval effectiveness. *ACM Transactions on Information Systems (TOIS)*, 27(1):1–27, 2008.

[27] Allan H Murphy. Scalar and vector partitions of the ranked probability score. *Mon Weather Rev*, 100(10): 701–708, 1972.

[28] Douglas W Oard, Jinmook Kim, et al. Implicit feedback for recommender systems. In *Proceedings of the AAAI workshop on recommender systems*, volume 83. WoUongong, 1998.

[29] Elias Pampalk, Arthur Flexer, Gerhard Widmer, et al. Improvements of audio-based music similarity and genre classificaton. In *ISMIR*, volume 5, pages 634–637. London, UK, 2005.

[30] Geoffroy Peeters, Julián Urbano, and Gareth JF Jones. Notes from the ismir 2012 late-breaking session on evaluation in music information retrieval. 2012.

[31] S Rasoul Safavian and David Landgrebe. A survey of decision tree classifier methodology. *IEEE transactions on systems, man, and cybernetics*, 21(3):660–674, 1991.

[32] Tetsuya Sakai. On the reliability of information retrieval metrics based on graded relevance. *Information processing & management*, 43(2):531–548, 2007.

[33] Himanshu Sharma and Bernard J Jansen. Automated evaluation of search engine performance via implicit user feedback. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 649–650, 2005.

[34] Steve Smithson. Information retrieval evaluation in practice: A case study approach. *Information Processing & Management*, 30(2):205–221, 1994.

[35] Xiaoyuan Su and Taghi M Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in artificial intelligence*, 2009, 2009.

[36] Jean Tague-Sutcliffe. The pragmatics of information retrieval experimentation, revisited. *Information Processing & Management*, 28(4):467–490, 1992.

[37] William MK Trochim and James P Donnelly. *Research methods knowledge base*, volume 2. Atomic Dog Pub., 2001.

[38] Julián Urbano. *Evaluation in Audio Music Similarity*. PhD thesis, University Carlos III of Madrid, 2013.

[39] Julián Urbano and Markus Schedl. Minimal test collections for low-cost evaluation of audio music similarity and retrieval systems. *International Journal of Multimedia Information Retrieval*, 2(1):59–70, 2013.

[40] Julián Urbano, Markus Schedl, and Xavier Serra. Evaluation in music information retrieval. *Journal of Intelligent Information Systems*, 41(3):345–369, 2013.

[41] Alexander Vezhnevets and Vladimir Vezhnevets. Modest adaboost-teaching adaboost to generalize better. In *Graphicon*, volume 12, pages 987–997, 2005.

[42] Ellen M Voorhees. Evaluation by highly relevant documents. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 74–82, 2001.

[43] Ellen M Voorhees. The philosophy of information retrieval evaluation. In *Workshop of the cross-language evaluation forum for european languages*, pages 355–370. Springer, 2001.

[44] Ellen M Voorhees and Donna K. Harman. *TREC: Experiment and evaluation in information retrieval*, volume 63. MIT press Cambridge, MA, 2005.

[45] Emine Yilmaz and Javed A Aslam. Estimating average precision with incomplete and imperfect judgments. In *Proceedings of the 15th ACM international conference on Information and knowledge management*, pages 102–111, 2006.