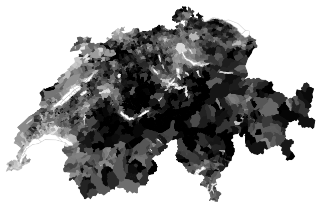
Online Collaborative Prediction of Regional Vote Results

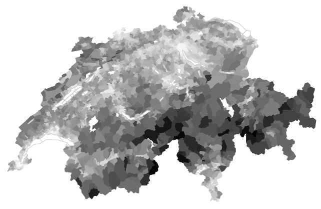
Vincent Etter*, Mohammad Emtiyaz Khan†, Matthias Grossglauser‡, Patrick Thiran‡
École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland
vincent@etter.io †emtiyaz.khan@epfl.ch ‡firstname.lastname@epfl.ch

***Abstract*—We consider online predictions of vote results, where regions across a country vote on an issue under discussion. Such online predictions before and during the day of the vote are useful to media agencies, polling institutes, and political parties, e.g., to identify regions that are crucial in determining the national outcome of a vote. We analyze a unique dataset from Switzerland. The dataset contains 281 votes from 2352 regions over a period of 34 years. We make several contributions towards improving online predictions. First, we show that these votes exhibit a bi-clustering of the vote results, i.e., regions that are spatially close tend to vote similarly, and issues that discuss similar topics show similar global voting patterns. Second, we develop models that can exploit this bi-clustering, as well as the features associated with the votes and regions. Third, we show that, when combining vote results and features together, Bayesian methods are essential to obtaining good performance. Our results show that Bayesian methods give better estimates of the hyperparameters than non-Bayesian methods such as cross-validation. The resulting models generalize well to many different tasks, produce robust predictions, and are easily interpretable.**

***Keywords*-vote prediction; political data mining; regression; matrix factorization; gaussian process; bayesian methods**



(a) Example of a vote result



(b) Population density

Figure 1: We show in (a) the outcome of one vote across all regions in grayscale, from regions that completely rejected the issue in black (0 % of "yes") to those that completely accepted it in white (100 % of "yes"), with everything else in between. In addition to spatial correlations, the similarities between the results could be partially explained by some of the characteristics of the regions, e.g., their population densities, which are shown in (b) where darker colors indicate lightly populated regions and lighter colors indicate densely populated regions.

I. Introduction

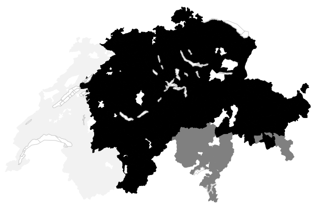
In order to promote transparency and accountability, as well as to stimulate citizen awareness, an increasing number of governments across the globe are adopting *open government* directives [7]. These result in the release of massive amounts of structured data about multiple aspects of state affairs, politics, and governmental agencies in various countries. As of 2016, the website Data Portals[1] references more than 500 such local, regional and national datasets.

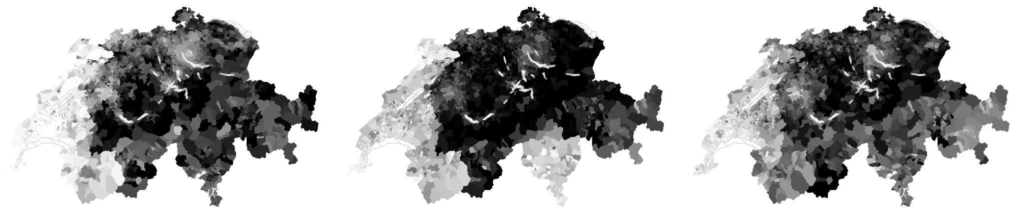
Among these datasets, the detailed outcomes of issue votes are published in many countries. Such votes are direct expressions of the opinion of the people, on various issues such as education, economy, and even ethics. In some cases, the detailed results are released at a fine geographical level, along with the national outcome of these votes. This newly available data gives an unprecedented view into the political landscape of a country. It enables us to gain a deeper understanding of the different voting behaviors across regions, and to investigate what makes regions similar, or dissimilar.

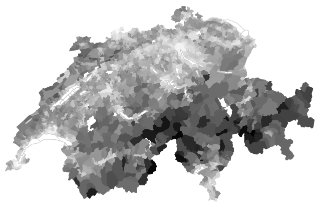
Of course, political parties are very interested in this information. Being able to identify patterns in vote results, based on characteristics of the vote or the regions, would enable them to better focus their campaigning efforts. The media also spend much of their resources trying to predict the outcome of votes, both before and during the day of the vote. Knowing whether a vote will be a narrow or clear win, and being able to identify regions that are crucial in determining the national outcome, would enable media and polling agencies to better focus their attention.

In this paper, we analyze a unique dataset of vote outcomes from Switzerland, where administrative regions across the country cast their vote to either accept or reject an issue under discussion. We show in Figure 1(a) an example of such a vote where we color each region according to its proportion of "yes" obtained for this vote; regions completely rejecting the issue (0 % of "yes") are shown in black and those completely accepting it (100 % of "yes") are shown in white, with everything else in between. We clearly see a structure in the pattern of results: some of the similarities between regions can be partially explained by their characteristics, such as their spatial proximity, their demographic attributes, and their political orientations. For example, the results of the votes, shown in Figure 1(a), seem to be correlated with population densities of regions, shown

[1]http://dataportals.org


(a) Linguistic regions


(b) "Röstigraben": French-speaking versus others


(c) Population density

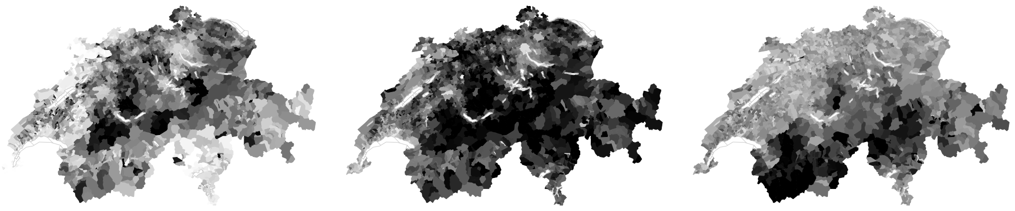

(d) Densely-populated versus rural areas

Figure 2: This figure shows a few examples of vote results to demonstrate that the results for votes and regions are correlated, and that the correlations depend on features of both the regions and the votes. Figure (a) shows regions that speak French (in light gray), Italian (in dark gray), and German (in black), and Figure (b) shows the results of three votes where the French-speaking regions, and sometimes the Italian-speaking regions, vote in opposition to the rest of the country (lighter colors indicate a higher percentage of "yes"). Similarly, Figure (c) shows the population density of regions across Switzerland, with lighter colors indicating a higher density of population, and Figure (d) shows votes where densely-populated regions vote in opposition to the rural areas.

in Figure 1(b) with darker colors indicating lightly populated regions and lighter colors indicating densely populated regions.

In addition to correlations between regions, we observe that the outcomes of different votes are correlated as well, and that the correlations depend on features of both the regions and the votes. We illustrate this in Figure 2. In Figure 2(a), we color regions that speak French in light gray, those that speak Italian in dark gray, and those speaking German in black. Figure 2(b) shows the results (again, lighter colors indicate higher proportions of "yes") of three votes that share similar patterns of results, linked to the linguistic regions: the French-speaking regions, and sometimes the Italian-speaking regions, vote in opposition to the rest of the country. Similarly, Figure 2(c) shows the population density of regions across Switzerland, with lighter colors indicating a higher density of population, and Figure 2(d) shows votes where densely-populated regions have voted in opposition to the rural areas. Other patterns, such as unanimous results, are also very common. We propose models to exploit the correlations across regions and votes, as well as their features.

As with many other works on voting-data analysis, we are interested in predicting the outcome of a new vote at the national level [9, 8, 4], but, unlike them, we would also like to predict the outcome in an *online* manner, i.e., to revise our prediction as more and more results are released region by region on the day of the vote. In addition, we would like to predict outcomes not only at the national level, but across regions as well [2, 3]. Such predictions are useful, e.g., for the media to focus their resources on a few key regions that might play a crucial role in deciding the outcome. This is a difficult task because, depending on the nature of the issue, the outcomes can differ from each other significantly, and the amount of data for a region alone may not be sufficient for an accurate prediction of its outcome. It is thus important to exploit the correlations across regions and votes.

A. Outline and Contributions

To achieve our goals, we make several contributions in this paper by using many successful machine-learning methods. First, to take into account the similarities between votes and between regions, we formulate the problem as a *collaborative-filtering* problem [15], where all regions express their opinion towards all votes. We use a latent-factor model to capture the *bi-clustering* of regions and votes [6]. Second, to accurately predict outcomes before the date of the vote, and when only a few regional results are available, we incorporate several features about the regions and the votes, which addresses the *cold-start* problem. These features not only improve our online predictions early on, but also enable us to interpret and understand voting behaviors. Third, we make extensive use of uncertainty by using a Bayesian approach. We show that such an approach gives stable predictions and helps us generalize well on different tasks. Fourth, we combine a Gaussian process model with the latent factor model and show that such a combination gives results better than using each of the models individually. Our Bayesian framework plays an important role here, enabling us to combine the two models using an automatic hyperparameter selection.

We show a variety of results in support of our analysis and discuss how such methodologies provide a powerful framework for analyzing general spatio-temporal data.

II. Dataset

We consider the case of Switzerland because it has a very active political system with easily available data. Swiss citizens vote on average eight times per year, on various issues regarding military, finances, transportation, culture, integration of foreigners, public health, etc. The results (i.e., the proportions of "yes") are publicly available[2] for each Swiss municipality[3]. In December 2014, there were 2352 municipalities in Switzerland. Our dataset[4] consists of the outcomes of the federal (i.e., nationwide) issue votes in each municipality between January 1981 and December 2014. There were 281 such votes.

In addition to the vote results, we gather side information about both votes and municipalities. For each vote, political parties publish voting recommendations, such as "in favor", "against", or "no recommendation". We gather these recommendations from the 13 main political parties in Switzerland[5]. For each municipality, we gather 25 features about its location, population, and electoral profile[6].

A. Preprocessing

Administrative regions change over time. It is common to have fusions and divisions of municipalities in Switzerland, and the total number of municipalities has been reduced by more than 10 % since 2000. This means that some of the current municipalities did not exist at some point in the past, and thus have no explicit results for some past votes. To make sure that all regions have a result for all votes, we could simply discard all municipalities that did not exist at some point in time during the whole 34 years that our dataset spans. However, this would result in about 7 % of discarded regions. To have as much data as possible, and to be able to make predictions about all regions that exist today, we chose to interpolate missing results instead. We also standardize the features of votes and regions, such that they have zero mean and unit variance.

[2]http://www.bfs.admin.ch/bfs/portal/fr/index/themen/17/03/blank/data/01.html

[3]Municipalities are the smallest administrative regions in Switzerland.

[4]Our dataset is available at http://vincent.etter.io/dsaa16.

[5]The voting recommendation are available on request at the Swiss Federal Statistics Office (http://www.bfs.admin.ch/bfs/portal/en/index.html).

[6]All the features describing Swiss municipalities are available online (http://www.bfs.admin.ch/bfs/portal/en/index/regionen/02/daten.html).

Table I: Summary of the notation and the dataset sizes.

Variable	Description
D	Number of regions ($= 2352$), indexed by d
N	Number of votes ($= 281$), indexed by n
y_{dn}	Outcome of the nth vote in the dth region
$\mathbf{y}_n$ ($\mathbf{Y}$)	Vector (or $D \times N$ matrix) of the outcomes of the nth vote (or all votes) in all the regions
$\overline{y}_n$ ($\overline{\mathbf{y}}$)	National outcome of the nth vote (or vector of the national outcome of all votes)
$\mathbf{x}_d$ ($\mathbf{X}$)	Vector (or $25 \times D$ matrix) of the features of the dth region (or all regions)
$\mathbf{w}_n$ ($\mathbf{W}$)	Vector (or $13 \times N$ matrix) of the features of the nth vote (or all votes)

III. Notation and Goals

We denote by y_{dn} the outcome of the nth vote in the dth region. We have $D = 2352$ regions and $N = 281$ votes. We gather the outcomes of the nth vote into a D-dimensional vector $\mathbf{y}_n$ and all outcomes into the $D \times N$ matrix $\mathbf{Y}$. For each vote n, the national outcome $\overline{y}_n$ is the weighted average of the regional results, with the weight equal to the number of ballots in a region (i.e., a weighted average of the elements of $\mathbf{y}_n$). We gather all national results into the N-dimensional vector $\overline{\mathbf{y}}$. We denote the 25 features of the dth region by $\mathbf{x}_d$ and the 13 features of the nth vote by $\mathbf{w}_n$. Finally, we gather the features $\mathbf{x}_d$ of all regions into the $25 \times D$ matrix $\mathbf{X}$ and the features $\mathbf{w}_n$ of all votes into the $13 \times N$ matrix $\mathbf{W}$. Our dataset is thus $\mathcal{D} = \{\mathbf{Y}, \overline{\mathbf{y}}, \mathbf{X}, \mathbf{W}\}$. Table I summarizes the notation and the dataset sizes.

We are interested in predicting the outcome of a new vote, with a feature vector denoted by $\mathbf{w}_\star$, in all regions. Moreover, we would like to make these predictions in an online manner, i.e., to refine the predictions as more regional results are made available.

Suppose that, at a certain time t on the day of the vote, we have observed the vote results in $D_t < D$ regions. Denote the set of observed regions by $\mathcal{O}_t = \{d_1, d_2, \dots, d_{D_t}\}$ (the ith observed outcome was that of region d_i). Denote the corresponding D_t-dimensional vector of outcomes by $\mathbf{y}_{\mathcal{O}_t,\star}$.

Given $\mathcal{D}$, $\mathbf{w}_\star$, and $\mathbf{y}_{\mathcal{O}_t,\star}$, our goal is to make the following two predictions at all future times greater than t:

1) predict the outcome $y_{d\star}$ in all regions $d \notin \mathcal{O}_t$,
2) predict the national outcome $\overline{y}_\star$.

In addition, we are interested in explaining and interpreting the reasons behind the predictions.

IV. Model

A popular approach to modelling collaborative data such as ours is to use matrix factorization, where we assume that $\mathbf{Y}$ can be predicted using a low-rank matrix. As mentioned above, this low-rank model can be combined with a regression model to address the cold-start problem [1]. We take a

Table II: Summary of the models we compare, along with their learning and inference methods. We give a short description of all models and list their hyperparameters. LS stands for *least-squares*, ALS for *alternating least-squares*, CV for *cross-validation*, Bayes for *Bayesian inference*, and ARD for *automatic relevance determination*.

Name	Description	Inference	Hyperparameters	Learning
BIAS	Bias term only	LS	-	-
LIN(r)	Linear regression using region features	LS	λ_β	CV
LIN(v)	Linear regression using vote features	LS	λ_γ	CV
LIN(r) + LIN(v)	Linear regression using region and vote features	ALS	$\{\lambda_\beta, \lambda_\gamma\}$	CV
GP(r)	GP regression using region features	Bayes	$\{\sigma_o^2, \sigma_s^2, \boldsymbol{l}\}$	ARD
MF	Matrix factorization	ALS	$\{\lambda_u, \lambda_v\}$	CV
MF + LIN(r)	Matrix factorization with linear regression using region features	ALS	$\{\lambda_u, \lambda_v, \lambda_\beta\}$	CV
MF + GP(r)	Matrix factorization with GP regression using region features	Bayes	$\{\sigma_o^2, \sigma_s^2, \mathbf{l}\}$	ARD
MF + GP(r) + LIN(v)	Matrix factorization with GP regression using region features and linear regression using vote features	Bayes	$\{\sigma_o^2, \sigma_s^2, \mathbf{l}, \lambda_\gamma\}$	ARD

similar approach. We model the "preferences" z_{dn} of the dth region for the nth vote using an additive model with four components (we describe each component in detail below):

$$z_{dn} = \underbrace{\mu_n}_{\text{bias}} + \underbrace{f_n(\mathbf{x}_d)}_{\substack{\text{regression using}\\ \text{region features}}} + \underbrace{f_d(\mathbf{w}_n)}_{\substack{\text{regression using}\\ \text{vote features}}} + \underbrace{\mathbf{v}_d^T\mathbf{u}_n,}_{\substack{\text{matrix}\\ \text{factorization}}} \tag{1}$$

where $\mu_n \in \mathbb{R}$ is a bias, $f_n : \mathbb{R}^{25} \to \mathbb{R}$ and $f_d : \mathbb{R}^{13} \to \mathbb{R}$ are regression functions, and $\mathbf{v}_d \in \mathbb{R}^L$ and $\mathbf{u}_n \in \mathbb{R}^L$ are latent factors, and L is the dimensionality of latent factors.

A benefit of such a model is that we can obtain many specialized models by adding/removing components. For our analysis, this proves to be useful as it enables us to establish the significance of individual components. Below, we first describe each component and then list the combinations we use in our experiments.

The first component μ_n is a bias term for each vote n. We could also add a bias term μ_d for each region d and a global bias μ, although in our experiments these do not make any difference.

The second component $f_n(\mathbf{x}_d)$ is a regression term that uses region features. We use two types of regression models. The first type is a linear regression model as shown in Equation 2, where $\boldsymbol{\beta}_n$ is a 25-dimensional weight vector associated with each vote n. We assume that each $\boldsymbol{\beta}_n$ is sampled independently from a normal distribution with precision parameter $\lambda_\beta > 0$ as shown in Equation 3.

$$\text{Linear: } f_n(\mathbf{x}_d) := \boldsymbol{\beta}_n^T \mathbf{x}_d \tag{2}$$

$$\boldsymbol{\beta}_n \sim \mathcal{N}(\boldsymbol{\beta}_n \mid 0, \lambda_\beta^{-1}\mathbf{I}) \tag{3}$$

The second type of model is based on the Gaussian Process (GP) regression [12] as shown in Equation 4. Here, each f_n is sampled independently from a GP with mean function $m(\cdot)$ and covariance function $k(\cdot,\cdot)$. Throughout the paper, we use a zero mean function and a squared-exponential (SE) covariance function with two hyperparameters $\sigma_s \in \mathbb{R}$ and $\mathbf{l} \in \mathbb{R}^{25}$, as shown in Equation 5.

$$\text{GP: } f_n(\mathbf{x}_d) \sim \text{GP}(\, m(\mathbf{x}_d),\, k(\mathbf{x}_d, \mathbf{x}_{d'})\,) \tag{4}$$

$$k(\mathbf{x}, \mathbf{x}') = \sigma_s^2 e^{-\frac{1}{2}(\mathbf{x}-\mathbf{x}')^T \text{diag}(\boldsymbol{l}^2)^{-1}(\mathbf{x}-\mathbf{x}')} \tag{5}$$

The third component $f_d(\mathbf{w}_n)$ of Equation 1 is a regression term that uses vote features and is defined similarly to the model described in Equation 2. The precision parameter associated with the prior on the weights of the vote features is denoted by λ_γ.

$$\text{Linear: } f_d(\mathbf{w}_n) := \boldsymbol{\gamma}_d^T \mathbf{w}_n$$
$$\boldsymbol{\gamma}_d \sim \mathcal{N}(\boldsymbol{\gamma}_d \mid 0, \lambda_\gamma^{-1}\mathbf{I})$$

We do not include a GP version of this component, because it gives similar results in our experiments but significantly increases computations in the combined model.

The fourth and last component of Equation 1 is a matrix factorization model. The latent features $\mathbf{v}_d$ are associated with the dth region and $\mathbf{u}_n$ are those associated with the nth vote. These latent features are vectors of length L to which we associate Gaussian priors, with precision parameter $\lambda_v > 0$ for $\mathbf{v}_d$ and $\lambda_u > 0$ for $\mathbf{u}_n$:

$$\mathbf{v}_d \sim \mathcal{N}(\mathbf{v}_d \mid 0, \lambda_v^{-1}\mathbf{I}), \quad \mathbf{u}_n \sim \mathcal{N}(\mathbf{u}_n \mid 0, \lambda_u^{-1}\mathbf{I}).$$

Finally, given the preference z_{dn} of the dth region for the nth vote, we model the corresponding outcome as

$$y_{dn} = z_{dn} + \epsilon_{dn},$$

where ϵ_{dn} is the observation noise. For each vote and each region, the noise is drawn i.i.d. from a Gaussian prior $\mathcal{N}(\epsilon_{dn} \mid 0, \sigma_o^2)$, where σ_o^2 is the noise variance.

This is not an ideal choice, as y_{dn} lies in the interval $[0, 1]$, however this choice does lead to simple inference algorithms. It is straightforward to use a different likelihood function by using more sophisticated inference methods, such expectation-propagation or sampling-based methods [12].

A. *Models and Inference Methods*

We now summarize the models we use in the remainder of this paper. Our models can be categorized into two categories: Bayesian and non-Bayesian. Our goal is to show that Bayesian methods give more accurate predictions than non-Bayesian ones. We design our experiments to clearly understand the reasons behind the better performance of Bayesian approaches. The complete list of the models we use in our experiments is given in Table II. We describe each model in details below.

The first model BIAS is our baseline, and it consists of the bias term only. For the Gaussian likelihood, μ_n is simply the sample mean of the vector $\mathbf{y}_n$. All of our models include the bias term.

The next three models are linear regression models involving a combination of $f_n(\mathbf{x}_d)$ using region features and $f_d(\mathbf{w}_n)$ using vote features. We fit LIN(r) and LIN(v) using least-squares (LS) and we use alternating least-squares (ALS) [17] for LIN(r) $+$ LIN(v). We outline the ALS algorithm in the appendix. Intuitively, we expect LIN(r)$+$LIN(v) to perform better than either LIN(r) or LIN(v). However, we will show that, if we use cross-validation to set the hyperparameters, the combination does always perform better than the individual models.

The next model GP(r) is a GP regression model that uses region features. We use the standard Bayesian method for inference with GPs. We learn the hyperparameters by maximizing the marginal likelihood [12, Section 5.4].

The next four models combine regression models with the matrix-factorization model. The first model MF does not have any regression terms, and is expected to give worse predictions for new votes. We learn the latent features of the MF model with an ALS algorithm, similar to the one used for LIN(r) $+$ LIN(v). We outline this procedure in the appendix.

The second model MF $+$ LIN(r) is an extension of MF, obtained by adding LIN(r). We train it using a combination of the two ALS methods presented above, where we append the explicit features of regions to their latent features. This model is expected to perform better than MF, but again we will show that CV leads to a sub-optimal performance.

In the third model, the component LIN(r) is replaced with GP(r) in order to enable the use of non-linear kernels. The resulting model bears similarities to a socialized Gaussian Process [13, 14]. For inference, we use a Bayesian method that works directly on z_{dn} and show that this model does not suffer from the problem of hyperparameters setting that affects MF $+$ LIN(r). This is made possible by adapting the EM algorithm of Khan et al. [5] to select the hyperparameters using the automatic relevance determination (ARD). Most importantly, this method automatically chooses a good value for the hyperparameter σ_s (shown in Equation 5) in order to combine the MF and GP(r) terms appropriately. The automatic selection is done by maximizing the marginal likelihood. We give the outline of this method in the appendix.

The fourth model adds the component LIN(v) to address the cold-start problem of new votes. However, we cannot easily integrate this component into the EM algorithm presented above. Thus, we first remove its contribution from $\mathbf{Y}$ and then apply the EM algorithm on the remainder. As we will show, this model obtains the best performance of all the models, while remaining computationally simple.

Overall, the purpose of our comparison is to demonstrate that, for online prediction of vote results, Bayesian methods work better than non-Bayesian methods. In our experiments, we show that this happens mainly for the following two reasons. First, during offline learning, a Bayesian approach enables us to find good hyperparameter values by using the automatic relevance determination (ARD) framework [12, Section 5.1]. This approach, as we will show, generalizes much better than non-Bayesian methods such as cross-validation (CV). Second, during online predictions, a Bayesian method uses uncertainty in the estimates to appropriately combine the components of Equation 1. Specifically, our Bayesian inference algorithm automatically computes the scale hyperparameter σ_s^2 of the GP prior, shown in Equation 5, by maximizing the marginal likelihood. Note that this hyperparameter controls the weight given to the features compared to the matrix-factorization component. We show that, in our experiments, this results in a better accuracy compared to a non-Bayesian method that uses a fixed value of the hyperparameter.

B. *Hyperparameter Learning*

We keep the last 50 votes of our dataset as the test set and train our models on the first 231 votes. We implement all our models using Matlab[7]. For the non-Bayesian models, we use 10-fold cross validation to set the hyperparameters and we monitor the validation root-mean-square error (RMSE) to test the convergence. For each fold, we randomly select 10 % of the outcomes as our validation data and use the rest to train the model. This means that, on average, we observe the results of 90 % of the regions for each vote of the training set, and we predict the outcome of the remaining 10 % of regions. Therefore, the models are trained for the "weak" generalization, as opposed to a "strong" generalization (see Section 3.3 of [10]). This is not optimal, as it means that the hyperparameters selected by this procedure will give the best results when we observe many outcomes for a vote, but not necessarily with only a few observations.

We could repeat this procedure for several percentages of observed results, e.g., with only 5 % of observed result for each vote. Our goal however is to have a single model that is able to make predictions with any number of observed results. We thus choose to train the ALS models with

[7]Our code is available at http://vincent.etter.io/dsaa16.

Table III: Hyperparameters selected for each of the non-Bayesian models, using 10-fold cross validation. We also show the hyperparameters of the hand-tuned version of MF + LIN(r) used in Figure 3(b), that are simply the hyperparameters found by CV for MF and LIN(r) individually.

Model	λ_u	λ_v	λ_β	λ_γ
LIN(r)	—	—	34	—
LIN(v)	—	—	—	32
LIN(r) + LIN(v)	—	—	36	80
MF	0.03	31	—	—
MF + LIN(r) (CV)	0.08	28	100	—
MF + LIN(r) (hand)	0.03	31	34	—

almost all the results observed, so that they have enough data to learn the global patterns of results. We show in Table III the hyperparameter values selected using 10-fold cross-validation for the non-Bayesian methods.

For the Bayesian models, we select the hyperparameters by maximizing the marginal likelihood [12, Section 5.4]. We can thus use all of the 231 train votes and do not need to use cross-validation[8]. We rely on the GPML toolbox [11] for the GP computations.

We use $L = 25$ as the dimension of the latent features of all models that have a MF component, as we empirically found that higher values do not increase the performances but result in longer training and evaluation times.

V. Results for Online Predictions

Our goal is to estimate the accuracy of the online predictions. We thus proceed as follows. First, we pick a random "reveal" order, which specifies the order in which we observe the results of 90 % of the regions (a total number of 2116 regions). We use the remaining last 10 % of the regions (a total number of 236 regions) as the test regions on which we will evaluate the error. We report the RMSE on the last 10 % of the regions, averaged over the 50 test votes and 500 random reveal-orders. The errors over the 50 test votes and 500 orders do not vary significantly, therefore we do not show error bars on the figures.

We first show in Figure 3(a) the results of the baseline, i.e., the BIAS model, and the models that only use linear regression, i.e., LIN(r), LIN(v), and LIN(r) + LIN(v). The model LIN(v) gets the best early performances, which clearly shows that the vote features are useful when only a few observations are available. However, with many observed regions, LIN(v) quickly reaches its limit. Next, the model LIN(r) starts with performances similar to those of BIAS and worse than those of LIN(v), but quickly outperforms both of them. The combined model LIN(r) + LIN(v) gets a slight advantage over LIN(r) by taking into account the voting recommendations, but is not able to reach the same early ”drop” in the error as LIN(v). As we will see next, this is because the hyperparameters selected by CV tend to overpenalize the LIN(v) component.

The same problem occurs with MF + LIN(r), as shown in Figure 3(b). First, we see in this figure that MF needs a few hundred observed regions before it can get performances better than LIN(r). This is because the MF model does not perform well for the "cold start" problem, and it needs a sufficient number of observations before it can achieve a reasonable performance. By combining the MF and LIN(r) components, we obtain performances better than those obtained by the MF model only, but now this gives a slightly worse performance compared to the LIN(r) model sometimes, e.g., when we have between 10 and 100 observations. The reason is the same – the hyperparameters obtained by CV do not result in performances matching those of the individual models. We can fix this by using hand-tuned hyperparameters, i.e., by using the parameter values shown in the last row of Table III. We can see in Figure 3(b) that MF+LIN(r) (hand) matches the early results of LIN(r) and gets performances better than MF. This clearly demonstrates that the hyperparameters obtained using CV are inferior.

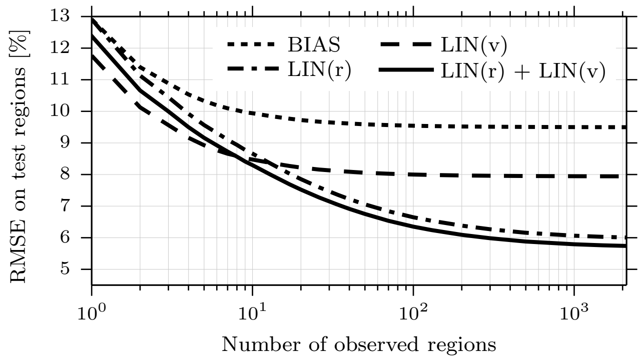
We show in Figure 3(c) that the solution to this problem is to use a Bayesian model. Indeed, the MF + GP(r) model is able to beat both the cross-validated and the hand-tuned versions of MF + LIN(r). To make sure that the improved performance does not come from the non-linearity of the SE kernel of the GP(r) component, we also show a variant of the MF + GP(r) that uses a linear isotropic kernel, which also beats the two non-Bayesian models.

Finally, we show in Figure 3(d) that adding the LIN(v) component to MF+GP(r) enables us to obtain the same early performances as those of LIN(v), completing the model. The final combination MF+GP(r)+LIN(v) thus obtains the best overall performances.

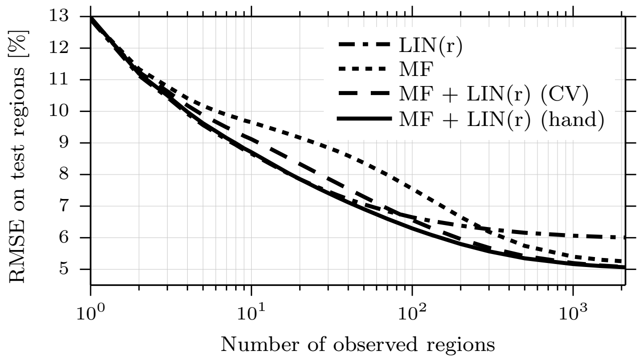
VI. National Results Prediction

We use the models presented above to predict the national result of a vote. To do so, we first predict the result in all unobserved regions, using the result of those observed. Then, we simply compute the average of the results of all regions (observed and predicted), weighted by their population. To achieve the most accurate predictions of the national result, we should use the turnout in each region as the weights, instead of their population. However, we do not have access to this information on the day of the vote, hence we cannot use it.

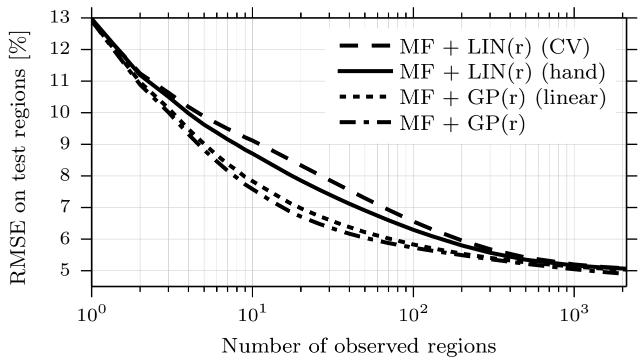
[8]As the hyperparameter of the LIN(v) component of MF+GP(r)+LIN(v) is not automatically set by the EM algorithm, we set it to $\lambda_\gamma = 200$. We empirically found that this value is small enough to take advantage of the vote features for early predictions, while not damaging the end performances.



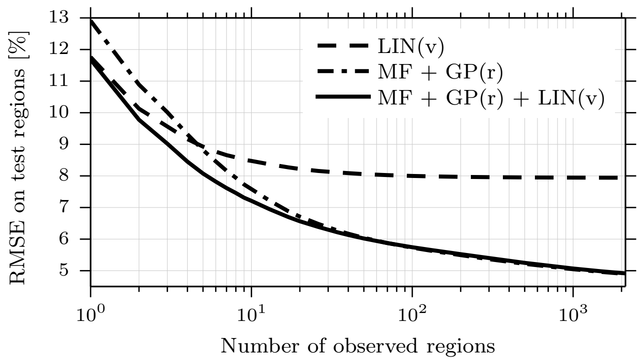
(a) While LIN(r) + LIN(v) achieves the best performance with many observations, its early performances do not match those of LIN(v).



(b) The hyperparameters selected by CV for MF+LIN(r) do not result in performances always matching those of the individual models. By hand-tuning the hyperparameters, however, we are able to obtain a model that has good performances all along, suggesting that the hyperparmeters obtained by CV are inferior.



(c) The Bayesian model MF + GP(r) gets better performances with few observations than both the cross-validated and hand-tuned MF + LIN(r) models, even with a simple linear kernel.



(d) MF + GP(r) + LIN(v) is able to properly combine the LIN(v) component with MF + GP(r) to obtain both good early performances and good results with many observed regions.

Figure 3: Comparison of the performance of the different models. We show the RMSE on the predicted result of the last 10 % of regions, averaged over 500 random reveal orders and 50 test votes.

Similarly to the results presented above, we show in Figure 4 the RMSE on the national result, averaged over 50 test votes and 500 random reveal orders. The difference in performance between MF+GP(r) and MF+GP(r)+LIN(v) is smaller than when predicting the result of individual regions. They can both predict the national outcome of a vote with an error smaller than 1 %, after having observed the result of only 50 regions.

As we show in Figure 5, the national outcome of the votes in our dataset are very diverse, spanning nearly the entire interval of possible results. To investigate whether votes whose outcome is close to 50 % have a larger error than others, we show in Figure 6 the relationship between the true outcome of votes and the error made by the MF + GP(r) + LIN(v) model. We first group the 50 test votes by their national result into eight bins, each 10 % wide. We then consider these bins separately, and show—for each bin—the distribution of the RMSE on the national result with 50 observed regions, over 500 random reveal orders. We see

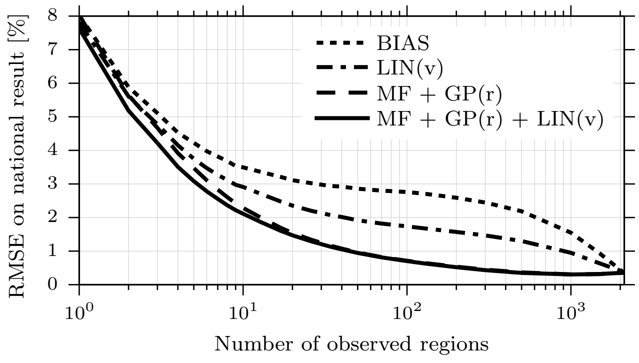


Figure 4: RMSE of the predicted national result, averaged over 500 random reveal orders and 50 test votes. Both MF + GP(r) and MF + GP(r) + LIN(v) are able to predict the national outcome of a vote within 1 % using the results of only 50 municipalities.

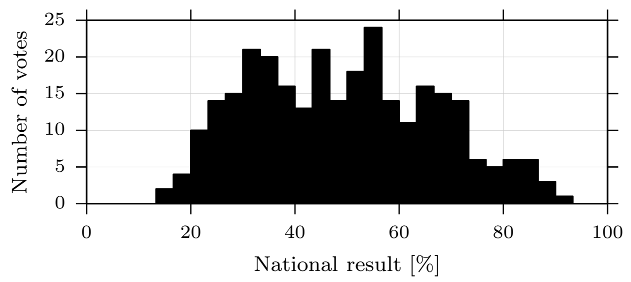


Figure 5: Distribution of the national results of the 281 votes in our dataset. The national results span nearly the whole range of possible results and are not biased towards the extremes.

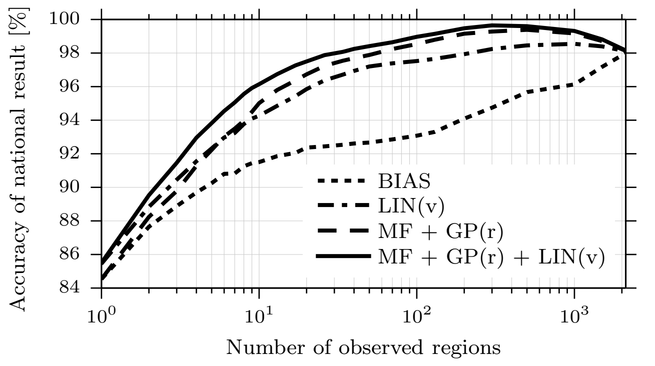


Figure 7: Accuracy of the national binary predictions, averaged over 500 random reveal orders and 50 test votes. MF + GP(r) + LIN(v) obtains an accuracy of 99 % with just 100 observed regions.

that there is no systematic relationship between the outcome of a vote and the error made by MF + GP(r) + LIN(v), with very similar distribution of errors over all bins. Therefore, our model is not biased towards any particular type of vote and can predict all votes equally well.

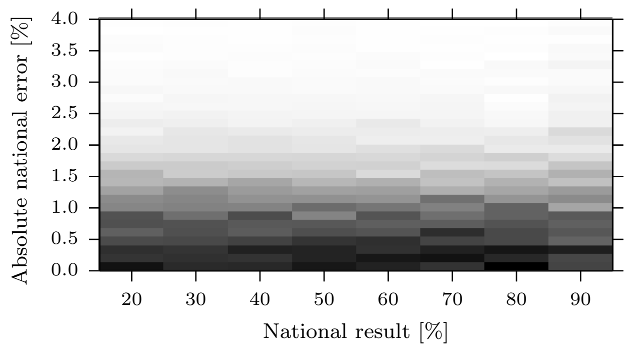


Figure 6: Distribution of the RMSE of the national predictions with respect to the true national outcome of the vote. We group the 50 test votes by their national result into eight bins. There is no systematic relationship between the national result and the errors made by the model.

Finally, we compare the accuracy of the models when predicting the binary outcome of a vote, i.e., whether it is accepted or not. To do so, we simply predict the national result as explained above, and then convert this predicted result to a binary outcome (a predicted national result greater than or equal to 50 % means that the issue is accepted, otherwise it is rejected). We show in Figure 7 the accuracy of these predictions for several models. With binary predictions, we see that the models with the LIN(v) component obtain a slight advantage when observing only a few regions, similar to what we observed in Figure 3(d). For example, MF + GP(r) + LIN(v) obtains an accuracy of 99 % with just 100 observed regions. We see a drop in accuracy with many observed regions, which could result from using the population of regions instead of the true turnout when computing the national result.

VII. Model Interpretation

As we already mentioned, one of the advantages of the models presented above is that they are easily interpretable. This means that political scientists, for example, could use such models to study the voting behavior of a country and verify the effect of some characteristics of the regions.

To illustrate the interpretability of these models, we show in Figure 8 the relative importance of the features of the regions, as learned by the MF + GP(r) + LIN(v) model. These weights can be directly obtained from **l**, one of the hyperparameters of the GP(r) component. We see that this component mostly explains the correlation between the results of two regions using their geographical proximity, and then their election results, i.e., their political orientations.

To explore further the correlation between regions, we show in Figure 9 a map of Switzerland with its cantons outlined. On this map, we draw a line between two municipalities if their correlation according to the MF + GP(r) + LIN(v) model is higher than 0.8. Again, such a map could lead to interesting interpretations. For example, we see that the Zürich area, in the North, is heavily clustered. We also clearly see a separation between the French-speaking and the German-speaking parts of the canton of Valais.

VIII. Conclusion

In this paper, we have introduced a novel dataset of political data. It is composed of the outcome of 281 national votes in 2352 administrative regions of Switzerland, along with 25 region features and 13 vote features. We introduced the problem of jointly predicting the outcome of a vote across all the regions, given the results in some observed regions. We showed that combining latent features with regression terms on the features of both regions and votes enabled

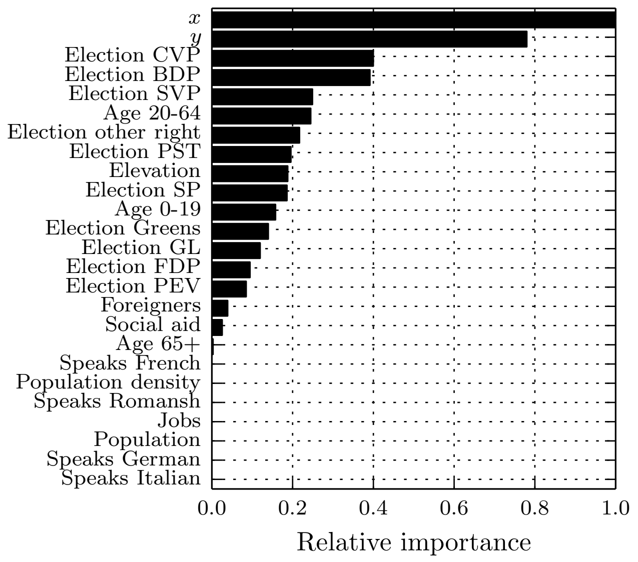


Figure 8: Relative importance of the region features learned by the GP(r) component of the MF + GP(r) + LIN(v) model. We see that most of the correlation between two municipalities is explained by their geographical proximity, with more weight given to the X-axis as it also partially encodes the language dimension (a large distance on the X axis usually implies different languages).

us to obtain accurate predictions, regardless of the number of observed results. Moreover, we have demonstrated that taking a Bayesian approach is key to finding the proper combination of these terms, with proper hyperparameter setting and results better than with non-Bayesian methods. The resulting model are easily interpretable and also give accurate predictions of the national outcome of the votes.

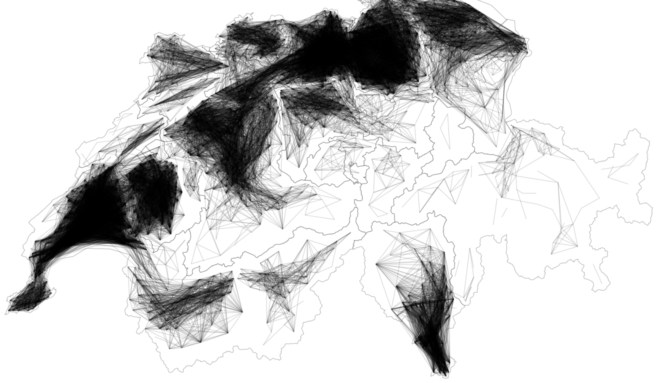


Figure 9: Visualization of the correlation between the results of municipalities captured by the MF + GP(r) + LIN(v) model. We show a link between two regions if their correlation is larger than 0.8, and add the boundaries of the cantons to have a frame of reference. Interestingly, we can clearly see the separation between the French-speaking and the German-speaking parts of the canton of Valais, in the southwestern part of the country.

REFERENCES

[1] D Agarwal and B.-C. Chen. Regression-based latent factor models. In *KDD'09*, pages 19–28. ACM, 2009.

[2] J. E. Campbell. Forecasting the presidential vote in the states. *American Journal of Political Science*, 36 (2):386–407, 1992.

[3] J. Gao and P. Revesz. Voting prediction using new spatiotemporal interpolation methods. In *dg.o'06*, pages 293–300, 2006.

[4] R. J. Jones Jr. The state of presidential election forecasting: The 2004 experience. *International Journal of Forecasting*, 24(2):310–321, 2008.

[5] M. E. Khan, Y. J. Ko, and M. Seeger. Scalable collaborative bayesian preference learning. In *AISTATS'14*, volume 33, pages 475–483, 2014.

[6] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, (8):30–37, 2009.

[7] D. Lathrop and L. Ruma. *Open government: Collaboration, transparency, and participation in practice*. O'Reilly Media, Inc., 2010.

[8] A. Leigh and J. Wolfers. Competing approaches to forecasting elections: Economic models, opinion polling and prediction markets. *Economic Record*, 82 (258):325–340, 2006.

[9] M. S. Lewis-Beck. Election forecasting: Principles and practice. *The British Journal of Politics & International Relations*, 7(2):145–164, 2005.

[10] B. Marlin. *Collaborative filtering: A machine learning perspective*. PhD thesis, University of Toronto, 2004.

[11] C. E. Rasmussen and H. Nickisch. Gaussian processes for machine learning (GPML) toolbox. *JMLR*, 11: 3011–3015, 2010.

[12] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

[13] Y. Shen, R. Jin, D. Dou, et al. Socialized gaussian process model for human behavior prediction in a health social network. In *ICDM'12*, pages 1110–1115. IEEE, 2012.

[14] Y. Shen, N. Phan, X. Xiao, et al. Dynamic socialized gaussian process models for human behavior prediction in a health social network. *KAIS*, pages 1–25, 2015.

[15] X. Su and T. M. Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence*, 2009, January 2009.

[16] M. E. Tipping and C. M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society*, 61(3):611–622, 1999.

[17] Y. Zhou, D. Wilkinson, R. Schreiber, and R. Pan. Large-scale parallel collaborative filtering for the netflix prize. In *Algorithmic Aspects in Information and Management*, volume 5034, pages 337–348. 2008.

Appendix

The LIN(r) + LIN(v) model simply combines regression terms on vote and on region features:

$$z_{nd} = \mu_n + \boldsymbol{\beta}_n^T \mathbf{x}_d + \boldsymbol{\gamma}_d^T \mathbf{w}_n.$$

We denote the $25 \times N$ vote weights matrix by $\boldsymbol{B}$ and the $13 \times D$ region weights matrix by $\boldsymbol{\Gamma}$. Algorithm 1 summarizes the resulting alternating least-squares algorithm for finding $\boldsymbol{B}$ and $\boldsymbol{\Gamma}$, the parameters of LIN(r) + LIN(v).

Algorithm 1: Alternating least-squares algorithm for the LIN(r) + LIN(v) model

```
Input: Vote outcomes $\mathbf{Y}$, region features $\mathbf{X}$, vote features $\mathbf{W}$, weight precision parameters $\lambda_\beta$ and $\lambda_\gamma$
Output: Weight matrices $\boldsymbol{B}$ and $\boldsymbol{\Gamma}$

Initialize $\boldsymbol{\Gamma}$
while convergence criterion is not met do
    $\mathbf{B} = \left(\mathbf{X}\mathbf{X}^T + \lambda_\beta \mathbf{I}\right)^{-1} \left(\mathbf{X}\left(\mathbf{Y} - \boldsymbol{\Gamma}^T \mathbf{W}\right)\right)$
    $\boldsymbol{\Gamma} = \left(\mathbf{W}\mathbf{W}^T + \lambda_\gamma \mathbf{I}\right)^{-1} \left(\mathbf{W}\left(\mathbf{Y}^T - \mathbf{B}^T \mathbf{X}\right)\right)$
```

The MF model expresses the outcome of a vote as the product of two latent factors:

$$z_{nd} = \mu_n + \mathbf{v}_d^T \mathbf{u}_n.$$

We denote by $\mathbf{V}$ the $L \times D$ matrix of latent factors associated with regions and by $\mathbf{U}$ the $L \times N$ matrix of latent factors associated with votes. Algorithm 2 summarizes the resulting alternating least-squares algorithm for finding $\mathbf{U}$ and $\mathbf{V}$, the parameters of MF.

Algorithm 2: Alternating least-squares algorithm for the MF model

```
Input: Vote outcomes $\mathbf{Y}$, latent features precision parameters $\lambda_u$ and $\lambda_v$
Output: Latent feature matrices $\mathbf{V}$ and $\mathbf{U}$

Initialize $\mathbf{V}$
while convergence criterion is not met do
    $\mathbf{U} = \left(\mathbf{V}\mathbf{V}^T + \lambda_u \mathbf{I}\right)^{-1} \mathbf{V}\mathbf{Y}$
    $\mathbf{V} = \left(\mathbf{U}\mathbf{U}^T + \lambda_v \mathbf{I}\right)^{-1} \mathbf{U}\mathbf{Y}^T$
```

The MF + GP(r) model is adapted from the work of Khan et al. [5]. It combines a matrix-factorization term with a GP regression term:

$$z_{nd} = \mu_n + \mathbf{v}_d^T \mathbf{u}_n + f_n(\mathbf{x}_d),$$

where $\mathbf{u}_n \sim \mathcal{N}(0, \mathbf{I})$, $\mathbf{v}_d$ is a parameter that will be estimated, and each $f_n \sim GP(0, \sigma_s^2 \mathbf{K})$. The covariance function $\mathbf{K}$ is a matrix whose (i, j)th component is $k(\mathbf{x}_n, \mathbf{x}_m)$, where k is a squared-exponential function defined as follows:

$$k(\mathbf{x}, \mathbf{x}') = e^{-\frac{1}{2}(\mathbf{x}-\mathbf{x}')^T \mathrm{diag}(\boldsymbol{l}^2)^{-1}(\mathbf{x}-\mathbf{x}')} \tag{6}$$

The parameters σ_s and $\mathbf{V}$ can be estimated using an EM algorithm, which is summarized in Algorithm 3. The key idea of the algorithm is to estimate the posterior distribution of $\mathbf{z}_n$ rather than $\mathbf{u}_n$, $\mathbf{v}_d$, and f_n. Since all the subcomponents are Gaussian, $\mathbf{z}_n$ is also a Gaussian with mean μ_n and covariance $\boldsymbol{\Sigma} = \mathbf{V}\mathbf{V}^T + \sigma_s^2 \mathbf{K}$. We can compute the mean $\mathbb{E}(\mathbf{z}_n)$ and covariance $\mathrm{Cov}(\mathbf{z}_n)$ in closed form, because the likelihood is also a Gaussian. This is shown in the E-step of Algorithm 3.

Given the sufficient statistics, we can then compute the marginal likelihood, which is given as follows:

$$\min_{V, \sigma_s} \sum_{n=1}^{N} \mathbb{E}[-\log \mathcal{N}(\mathbf{z}_n | \mu_n, \boldsymbol{\Sigma})] = \frac{N}{2}[\log |2\pi\boldsymbol{\Sigma}| + \mathrm{Tr}(\boldsymbol{\Sigma}^{-1}\mathbf{C})],$$

$$\mathbf{C} := \frac{1}{N} \sum_{n=1}^{N} [\mathrm{Cov}(\mathbf{z}_n) + (\boldsymbol{\mu} - \mathbb{E}(\mathbf{z}_n))^T (\boldsymbol{\mu} - \mathbb{E}(\mathbf{z}_n))].$$

The above optimization problem has a closed-form solution which can be obtained by using the probabilistic PCA model [16]. The hyperparameter σ_s^2 is equal to the variance unexplained by the principal components and can be computed given the eigenvalues of the $D - L$ eigenvectors. For details, see Khan et al. [5]. A significant benefit of this algorithm is that it sets σ_s^2, thus automatically finding the proper combination of the MF and GP(r) terms.

Algorithm 3: EM algorithm for the MF + GP(r) model

```
Input: Vote outcomes $\mathbf{Y}$, region features $\mathbf{X}$
Output: Latent feature matrix $\mathbf{V}$

Initialize $\mathbf{V}, \sigma_s^2, \boldsymbol{\theta} = \{\sigma_o^2, \mathbf{l}\}$
while convergence criterion is not met do
    Compute the covariance matrix $\mathbf{K}$ using $\mathbf{X}$ and $\boldsymbol{\theta}$
    Compute its Cholesky $\mathbf{L} = \mathrm{chol}(\mathbf{K})$
    Let $\boldsymbol{\Sigma} = \mathbf{V}^T\mathbf{V} + \sigma_s^2\mathbf{K}$
    Initialize $\mathbf{C} = \mathbf{0}$

    // E step
    for each vote $n$ do
        $E(\mathbf{z}_n) = \boldsymbol{\Sigma}(\boldsymbol{\Sigma} + \sigma_o^2\mathbf{I})^{-1}\mathbf{y}_n$
        $\mathrm{cov}(\mathbf{z}_n) = \boldsymbol{\Sigma} - \boldsymbol{\Sigma}(\boldsymbol{\Sigma} + \sigma_o^2\mathbf{I})^{-1}\boldsymbol{\Sigma}$
        $\mathbf{C} = \mathbf{C} + \frac{1}{N}\left(\mathrm{cov}(\mathbf{z}_n) + E(\mathbf{z}_n)E(\mathbf{z}_n)^T\right)$
    Let $\tilde{\mathbf{C}} = \mathbf{L}^{-1}\mathbf{C}\mathbf{L}^{-T}$
    Compute $\mathbf{R}$, the matrix of eigenvectors of $\tilde{\mathbf{C}}$, and $\boldsymbol{\Lambda}$, the corresponding diagonal matrix of eigenvalues

    // M step
    $\sigma_s^2 = \frac{\mathrm{Tr}(\mathbf{R}) - \mathrm{Tr}(\boldsymbol{\Lambda})}{D-L}$
    $\mathbf{V} = \mathbf{L}\mathbf{R}(\boldsymbol{\Lambda} - \sigma_s^2\mathbf{I})^{\frac{1}{2}}$
    Update $\boldsymbol{\Sigma} = \mathbf{V}^T\mathbf{V} + \sigma_s^2\mathbf{K}$
    Find $\boldsymbol{\theta}$ maximizing the likelihood of
    $\mathbf{y}_n \sim GP(0, \boldsymbol{\Sigma} + \sigma_o^2\mathbf{I})$ for all $n$
```