Correlations in Artworks to Popularity on Pinterest

Tom Eijkelenkamp s1889338

March 15, 2024

1 Introduction

This project is a lecture based around estimating the causal relationship between categorical variables. The motivation behind this subject is that I want to study how to compute a best fitting model in a more complicated scenario involving variables of mixed types. This I can use in my own studies later. I believe I will do good with further deepening my understanding in how to score graphs in order to later to move on to a more complicated scenario.

My initial idea was to create a model that explains what causes an artwork to gain likes on social media. There might be many parameters involved that influence this outcome. Over the time there had been many influential artists and different movements with each their own theme. Certain types of paintings might be liked more then others due to the content, what kind of paint is used, who was the artist, what is the purpose. Some artist might have spend their whole life painting, while others just do it as a hobby.

Figure 1: Art works made with oil[1], watercolor[2], AI[3], arcyl[4] respectively.

[01] https://nl.pinterest.com/pin/10766486601688996/
[02] https://nl.pinterest.com/pin/646618459026514298/
[03] https://nl.pinterest.com/pin/1061160730938301330/
[04] https://nl.pinterest.com/pin/68117013107123172/

To use these variables in a model we have to format them using some abstract formulation. We could say what kind of paint is used is a categorical variable that can take a number of values, for example acryl, oil, watercolor. On the other hand we have the experience an artist has. This is difficult to obtain, since it is not known how many hours someone has been painting. Also there might be talent involved, which makes someone to be better while he has spend less hours practicing. We could use the number of paintings he created, which is a value that might be obtained somewhere. Another thing is that we could look at the number of likes someone gets on social media on his or her paintings. Now these last two variables are integer values, different from categorical. Using these kinds of variables in the model creates an overall mixed graphical model.

This project I will build up to how to find the best estimate of a graph that describes the causal relationship between categorical variables. I will give examples in the theme of the influences in the artistic process. I will first discuss what directed acyclical graphs are. Then move on to explain the dirichlet distribution. Followed by explaining graph equivalents classes. I will discuss how to score a categorical graph using the Bayesian Dirichlet likelihood equivalent score. Later I will show how this can be used in the mcmc algorithm to obtain the posterior distribution over all possible graphs. All these concepts are used in experiments on a acquired dataset with an implementation in R. We estimate several influential diagrams for certain characteristics in art.

2 Directed acyclical graphs

In order to describe causal relationships between variables we can use directed acyclical graphs. These consist of nodes, which are the variables, and directed edges which describe the causal influences between the variables. These variables can be of any kind. The graph can be constructed with binary variables, categorical or floating point numbers. Variables that have no edge between them are considered conditionally independent.

In the example of art pieces, you can have that a person is particularly good at painting with watercolor paint. Now when looking at the variables, we have the artist, the paint used, his skill level with a particular paint and the quality of the painting. The artist chooses the content of the painting, so there is a causal relationship between the artist and the content. The artist has some skill level at painting particular content so there also is a causal relationship between artist and his skill level. The content that is being painted has an effect on what the skill level is going to be. You can be better at painting certain content and worse at other. Also I believe there is an reverse relationship that the skill level at painting a certain theme influences what you are going to paint, although bidirected edges are not allowed in DAG's. Someone might be more drawn towards what he is good at. The skill level at what he is creating has an effect on the end quality. Also the content itself has a qualitative aspect, since people would consider certain types of paintings better then others. Maybe someone is more drawn towards landscape paintings and someone else likes a more abstract style. There is probably a causal relationship between the artist and the quality of the painting, since some people might be influenced by seeing the name of an artist. A DAG visualising this relationship is shown in Figure 2.

3 Dirichlet distribution

This distribution describes a probability of probabilities. In particular one that involves multiple categories. It's an extension to the Beta distribution, which involves only two 'categories'.

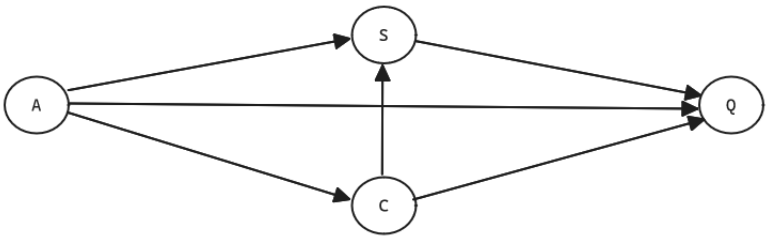

Figure 2: Example of a directed acyclical graph

Take for example the scenario where you want to know the chance that someone creates a painting showing something in a particular season. You can look at how many paintings he made showing the season summer, spring, winter and fall. Then from these values you can estimate the probability that his next painting will be of some season.

You could compute a point estimate $\hat{P}_c$ by just dividing the number of samples c that where of a particular category divided by the total number of samples n.

$$\hat{P}_c = \frac{c}{n} \tag{1}$$

This estimate has a certain error $\sigma(\hat{P}_c)$:

$$\sigma(\hat{P}_c) = \sqrt{\frac{\hat{P}_c(1-\hat{P}_c)}{n}} \tag{2}$$

This shows that when you obtain more data points, the error on your estimate is smaller.

Now you could make a better estimate by computing the full distribution for probabilities of the probabilities. The Dirichlet is this distribution.

The probabilities $p_1, \cdots, p_k$ can be computed given the values $\alpha_1, \cdots, \alpha_k$ which are the number of occurrences in previous observations plus one. This one is added since you start your guess when having observed no data points yet by giving equal probability to all categories, and so each category gets value one. So when looking at the painting example α_1 can be the number of times that the artist made a painting depicting something in summer, plus one. The equation is shown below:

$$f(p_1, \cdots, p_k; \alpha_1, \cdots, \alpha_k) = \frac{1}{B(\alpha_1, \cdots, \alpha_k)} \prod_{i=1}^{k} p_i^{\alpha_i - 1} \tag{3}$$

$$B(\alpha_1, \cdots, \alpha_k) = \frac{\prod_{i=1}^{k} \Gamma(\alpha_i)}{\Gamma(\sum_{i=1}^{k} \alpha_i)} \tag{4}$$

The gamma function is a factorial function for real numbers. Since the arguments to the function here are just integers it can be treated as a factorial function. It is used to calculate all the options you can order data points. This is similar to when you want to compute the chance of taking three blue and two red marbles from a bag full of colored marbles, you need to take into account all the orders you can end up with three blue and two red.

4 Graph equivalents classes

Graphs that have a different topology might have the same joined distribution. These are considered to be of one equivalents class. For example we can look at the following four graph networks:

$$A \to B \to C \tag{5}$$

$$\mathbb{P}(A,B,C) = \mathbb{P}(C|B)\mathbb{P}(B|A)\mathbb{P}(A) = \frac{\mathbb{P}(C,B)}{\mathbb{P}(B)}\frac{\mathbb{P}(B,A)}{\mathbb{P}(A)}\mathbb{P}(A) \tag{6}$$

$$A \leftarrow B \to C \tag{7}$$

$$\mathbb{P}(A,B,C) = \mathbb{P}(C|B)\mathbb{P}(A|B)\mathbb{P}(B) = \frac{\mathbb{P}(C,B)}{\mathbb{P}(B)}\frac{\mathbb{P}(A,B)}{\mathbb{P}(B)}\mathbb{P}(B) \tag{8}$$

$$A \to B \leftarrow C \tag{9}$$

$$\mathbb{P}(A,B,C) = \mathbb{P}(B|A,C)\mathbb{P}(A)\mathbb{P}(C) = \frac{\mathbb{P}(A,B,C)}{\mathbb{P}(A,C)}\mathbb{P}(A)\mathbb{P}(C) \tag{10}$$

$$A \leftarrow B \leftarrow C \tag{11}$$

$$\mathbb{P}(A,B,C) = \mathbb{P}(A|B)\mathbb{P}(B|C)\mathbb{P}(C) = \frac{\mathbb{P}(A,B)}{\mathbb{P}(B)}\frac{\mathbb{P}(B,C)}{\mathbb{P}(C)}\mathbb{P}(C) \tag{12}$$

For the first, second and last graphs the joined probability is equal to:

$$\mathbb{P}(A,B,C) = \mathbb{P}(C|B)\mathbb{P}(B|A)\mathbb{P}(A) = \frac{\mathbb{P}(C,B)}{\mathbb{P}(B)}\mathbb{P}(B,A) \tag{13}$$

Graphs that factorize as the same joined distribution are considered to be of the same equivalents class.

5 Bayesian dirichlet likelihood equivalent score

We want to compute how well a graph fits observed data. With such a score we can find the best graph given the data and propose this as an estimate of causal relationships between the variables. We can sample from reality many data points in order to use for a research. The goal of the research would be to find what influences the outcome in a process in order to change the behavior for better outcomes.

To translate this to the painting example, we want to find all that causes an artwork be well accepted by it's audience. We can find all the variables that we suppose have influence and measure these in the data. Now we can estimate the causal relationships by computing the graph with the highest score. After finding the causal relationships we can use them to identity what people like in art. You could then use these elements in your artwork.

We focus on data where each variable can take a number of categories as a realization. Here one method of scoring a Bayesian network is the Bayesian Dirichlet likelihood equivalent score. This is based around the earlier explained Dirichlet distribution and graph equivalent classes.

This function computes the marginal likelihood of a graph. So how likely is the data given that we consider a certain structure for the graph.

$$\mathbb{P}(\mathcal{D} \mid \mathcal{G}) \tag{14}$$

Here the dataset is an n by m matrix, where n is the number of variables and m is the number of observations.

$$D = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \tag{15}$$

Above this is visualized as an example where every row is an observation of n variables. An example graph is visualized in Figure 3 where n is four.

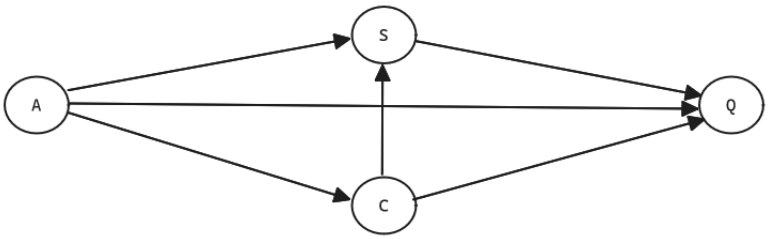

Figure 3: An example of a graph with four nodes.

All the nodes in the network correspond to the variables in the dataset. We are going to calculate the score for a particular graph and so we know the structure. Every node has a list of parents, which can be empty. The graph states that the parents have a causal influence on it's children. We are going to test how well such an assumption holds. A visualization of a parent set of a node in the example graph is shown in Figure 4.

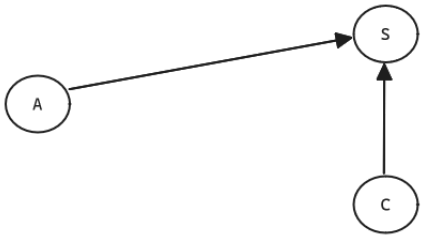

Figure 4: The parent set of node s in the previous example.

The parents of a node combined have a number of different realisations. For each one of those

we look at how the corresponding child realisations are distributed. We can find parameters for these child realisations, since they are dirichlet distributed.

$$\mathbb{P}(\theta_{ij1}, \cdots, \theta_{ijr_i}) = \frac{\Gamma(\sum_{k=1}^{r_i} \alpha_{ijk})}{\prod_{k=1}^{r_i} \Gamma(\alpha_{ijk})} \prod_{k=1}^{r_i} \theta_{ijk}^{\alpha_{ijk}-1} \tag{16}$$

Here $\theta_{ij1}, \cdots, \theta_{ijr_i}$ are the parameters for a child node X_i for the j-th realisation of the parent set. The child node can take a range of r_i realisations. The alphas are the arguments of the dirichlet distribution and so α_{ijk} is the number of times the child took the k-th realisation, while the parent set was in the j-th realisation. One is added to this number to obtain the dirichlet argument as explained in the previous chapter.

When these parameters are obtained we can use them to estimate the chance on the data given the graph with it's parameters. The multinomial distribution describes the chance on a certain sequence of outcomes for a categorical variable. This distribution is defined by the chance for each category.

$$\mathbb{P}(m_{ij1}, \cdots, m_{ijk} \mid m_{ij}, \theta_{ij1}, \cdots, \theta_{ijr_i}) = \frac{m_{ij}!}{\prod_{k=1}^{r_i} m_{ijk}!} \prod_{k=1}^{r_i} (\theta_{ijk})^{m_{ijk}} \tag{17}$$

Here m_{ij} is the number of observations where the parents set of the variable i took the j-th realisation. For this particular case we look at the number of observations in each category of the child variable. We can calculate the chance on this particular outcome. So $m_{ij1}, \cdots, m_{ijk}$ is the number of observations in each category of the child variable, given that the parent took the j-/th realisation.

Combining the two will give us the chance on the data, given the graph:

$$\mathbb{P}(\mathcal{D} \mid \mathcal{G}) = \int \mathbb{P}(\mathcal{D} \mid q, \mathcal{G}) \mathbb{P}(q \mid \mathcal{G}) \tag{18}$$

The parameters are represented as q. We integrate over all possible values of the parameters. With the Dirichlet distibution you compute the complete probability distribution for all possible parameter sets. This means that the integral over all possible parameters in this equation can be substituted as follows:

$$\int \cdots \int \prod_{k=1}^{r_i} \theta_{ijk}^{m_{ijk}+\alpha_{ijk}-1} d\theta_{ij1} \cdots d\theta_{ijr_i} = \frac{\prod_{k'=1}^{r_i} \Gamma(\alpha_{ijk'} + m_{ijk'})}{\Gamma(\sum_{k'=1}^{r_i} (\alpha_{ijk'} + m_{ijk'}))} \tag{19}$$

This results in an equation without an integral:

$$\mathbb{P}(\mathcal{D} \mid \mathcal{G}) = \prod_{i=1}^{n} \prod_{j=1}^{s_i} \left(\frac{\Gamma(\alpha_{ij})(m_{ij})!}{\Gamma(\alpha_{ij} + m_{ij})} \right) \prod_{k=1}^{r_i} \left(\frac{\Gamma(\alpha_{ijk} + m_{ijk})}{m_{ijk}! \Gamma(\alpha_{ijk})} \right) \tag{20}$$

$$\propto \prod_{i=1}^{n} \prod_{j=1}^{s_i} \left(\frac{\Gamma(\alpha_{ij})}{\Gamma(\alpha_{ij} + m_{ij})} \right) \prod_{k=1}^{r_i} \left(\frac{\Gamma(\alpha_{ijk} + m_{ijk})}{\Gamma(\alpha_{ijk})} \right) \tag{21}$$

Here $m_{ij} = \sum_{k'=1}^{r_i} m_{ijk'}$ and $\alpha_{ij} = \sum_{k'=1}^{r_i} \alpha_{ijk'}$.

6 Markov Chain Monte Carlo

Now we know how to score each graph we can use this information to find the best estimate for the causal relationships within a given dataset of categorical variables. We could simply try every possible graph structure and compute it's score, but in many cases this costs too much compute. A way of computing the posterior probability of graphs given the data we can use Markov Chain Monte Carlo.

We want to compute the posterior probability:

$$\mathbb{P}(\mathcal{G}|\mathcal{D}) = \frac{\mathbb{P}(\mathcal{D}|\mathcal{G})\mathbb{P}(\mathcal{G})}{\mathbb{P}(\mathcal{D})} \propto \mathbb{P}(\mathcal{D}|\mathcal{G})\mathbb{P}(\mathcal{G}) \tag{22}$$

We start at some graph structure. Then we iterate to neighbor graphs. These are graphs that only differ one edge operation (delete, change direction or add). We propose this graph with probability $Q(\mathcal{G}, \mathcal{G}_*)$:

$$Q(\mathcal{G}, \mathcal{G}_*) = \mathbb{1}(\mathcal{G}_* \in \mathcal{N}(\mathcal{G})) \frac{1}{|\mathcal{N}(\mathcal{G})|} \tag{23}$$

Here $\mathcal{N}(\mathcal{G})$ is the set of neighbors of graph $\mathcal{G}$. We the either accept or reject this graph with the probability:

$$A(\mathcal{G}, \mathcal{G}_*) = \min\left\{1, \frac{\mathbb{P}(\mathcal{G}_* \mid \mathcal{D})}{\mathbb{P}(\mathcal{G} \mid \mathcal{D})} \frac{\mathcal{Q}(\mathcal{G}_* \mid \mathcal{G})}{\mathcal{Q}(\mathcal{G} \mid \mathcal{G}_*)}\right\} \tag{24}$$

We can use the BDe score to compute $\mathbb{P}(\mathcal{G} \mid \mathcal{D})$. Then we can iterate this algorithm for many number of iterations depending on the complexity of the problem. This gives us the posterior distribution $\mathbb{P}(\mathcal{G} \mid \mathcal{D})$. Also we can compute the probability that an edge is present in the graph using the simulation:

$$\frac{1}{T}\sum_{t=1}^{T} I(\mathcal{G}_t) \to \sum_{\mathcal{G}\in\mathcal{S}} \mathbb{P}(\mathcal{G} \mid \mathcal{D}) I(\mathcal{G}) \tag{25}$$

7 Data

We want to research what is the causal diagram of getting likes on social media when creating art. We need data that shows two aspects: Categorical annotations for artworks, the number of likes each work gets on social media.

Therefor I collected a sample from the media platform Pinterest. I selectively filtered for landscape artworks when I logged into the platform. Pinterest personalizes search results according to previous usages. From these results I picked the first about 500 results. I discarded advertisements and duplicates.

In order to get the annotations I used an AI tool LLaVA [2][1]. This is a large language and vision assistant. It is one of the best open source models in it's class. It requires about 26gb memory GPU and so I ran it on Habrok to create categorical notes on each of the selected artworks.

I could have looked into any kind of patterns within the artworks. I chose to look the following: What kind of paint is used to make it, what kind of colors are used in terms of temperature, saturation and spectrum variety. Furthermore I annotated the dataset for what is the season and time of day shown in the painting.

All of these paintings are posted by accounts which are followed by a certain number of people. The data on this I added for my research.

Then I added the number of likes and amount of comments it received on the platform.

In the appendix I've added histograms showing what kind of data I collected.

Since not all variables are categorical, the numerical ones were translated into categorical values. The number of followers, comments and likes each were sliced into four equal sized bins.

8 Method

8.1 Why Bayesian networks

The idea behind this research is that an obtained graph about causal relationships between aspects within paintings, popularity of accounts and number of likes on single works gives us more insight into what makes an artwork good. You could create a simple linear regression model, but this gives a more abstract idea of what the underlying reality looks like.

For example we could find a correlation between what kind of paint is used in artworks and how this relates to attraction. You could find a very clear correlation and come to conclude that waterpaint has the best change on scoring high. What actually could be happening is that the artist that use this kind of paint are actually better and so it is not the paint that makes it good, but the artists itself.

Another example could be that you only look at the artist and their score on social media. Then you see that some are scoring better then others. Actually some could be only posting content that has a high scoring percentage. So it is not really that those artists are better, it could be that certain types of styles are liked much more then others and those artists chose to post these styles selectively.

A complete graph on all the correlations will give a better view on what is going on.

8.2 Implementation

To find an estimate of what the causal relations might be between the variables, an implementation of Markov Chain Monty Carlo is made using the BDe score. The package BNLearn has implemented the score function and graph structures so this was used as a basis for the code. The MCMC algorithm is implemented as described in the previous chapter. The code can be found in the appendix.

I run a small trial experiment simulating dependencies between variables and see if we can find these with the implemented algorithm.

```
# Number of samples
n <- 1000

# Generate data for A
A <- rbinom(n, 1, 0.5)  # A is a simple binary variable

# Generate data for B based on A
B <- ifelse(A == 0, rbinom(n, 1, 0.3), rbinom(n, 1, 0.7))  # B depends on A

# Generate data for C based on B
C <- ifelse(B == 0,
```

```
12              sample(1:3, n, replace = TRUE, prob = c(0.1, 0.6, 0.3)),  # Distribution
    when B is 0
13              sample(1:3, n, replace = TRUE, prob = c(0.4, 0.4, 0.2)))  # Distribution
    when B is 1
```

The following results were found and so the dependencies. The first table shows the graph with the highest BDe score and the second table contains scores for each of the edges.

	A	B	C
A	0	1	0
B	0	0	1
C	0	0	0

	A	B	C
A	0.00000	0.29929	0.00022
B	0.70068	0.00000	0.71749
C	0.00009	0.28227	0.00000

8.3 Experiments

I did three experiments on the dataset. The dataset consists of many variables. I also did experiments with fewer variables. One with only the categories 'account', 'paint', 'likes'. Another one with the categories 'account', 'followers', 'comments', 'likes'. The last experiment contains all the variables.

9 Results

First experiment

	account	paint	likes
account	0	0	0
paint	0	0	0
likes	0	0	0

	account	paint	likes
account	0	0	0e+00
paint	0	0	4e-05
likes	0	0	0e+00

From this MCMC simulation we find that for the sample we took we estimated a very small causal relation between the paint used and the number of likes. There seem to be no such relation between what account posted the art and the number of likes. This suggests that you can influence the attraction of a painting, by using a certain kind of paint.

Second experiment

	account	followers	comments	likes
account	0	0	0	0
followers	1	0	0	0
comments	0	0	0	0
likes	0	0	0	0

	account	followers	comments	likes
account	0.00000	0.49986	0.00000	0.00000
followers	0.49983	0.00000	0.09977	0.00106
comments	0.00000	0.03343	0.00000	0.01250
likes	0.00000	0.00000	0.01906	0.00000

Here in the second experiment we find that it is mainly the number of followers that an account has that make the artworks get likes and comments, not the account itself.

Third experiment

	account	paint	temperature	saturation	palette	season	time	followers	comments	likes
account	0	0	0	0	0	0	0	0	0	0
paint	0	0	0	0	0	0	1	0	0	0
temperature	0	0	0	1	1	0	0	0	0	0
saturation	0	0	1	0	1	1	0	0	0	0
palette	0	0	0	0	0	0	0	0	0	0
season	0	0	0	0	0	0	0	0	0	0
time	0	0	1	0	0	0	0	0	0	0
followers	1	0	0	0	0	0	0	0	0	0
comments	0	0	0	0	0	0	0	0	0	0
likes	0	0	0	0	0	0	0	0	0	0

	account	paint	temperature	saturation	palette	season	time	followers	comments	likes
account	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.39510	0.00000	0.00000
paint	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.55925	0.01365	0.00000	0.00000
temperature	0.00000	0.00000	0.00000	0.87400	0.82394	0.34077	0.21314	0.00099	0.03942	0.00000
saturation	0.00000	0.00000	0.12219	0.00000	0.50068	0.82263	0.00000	0.00000	0.11989	0.01872
palette	0.00000	0.00000	0.03030	0.47419	0.00000	0.00000	0.00000	0.00000	0.07113	0.12787
season	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.06197	0.00000
time	0.00000	0.44005	0.64342	0.00000	0.00000	0.00000	0.00000	0.00000	0.40883	0.00000
followers	0.50736	0.15591	0.02552	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
comments	0.00000	0.00000	0.00000	0.00037	0.01104	0.00000	0.00647	0.00000	0.00000	0.02879
likes	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000

In the last experiment we find the highest influence to likes being the variation in the color palette. When looking at how these values are distributed within the data in Figure 7 from the appendix, we can see that there is very little data on other categories than a well varied color palette. Therefor I would discard this finding.

The second aspect that seem to influence the number of likes is the amount of comments. Also this variable the distribution of categories is super skewed, almost all the data points fall in the category receiving zero comments, seen in Figure 9. When we would have had better distributed data on this variable and we found that actually there is this influence of the number of comments to likes, we could have looked what influences the number of comments. Then you could focus on the categories, and find out what category does make the number of comments go up or down, depending on what makes the likes go up (I suspect that it is more comments makes the art get more likes).

The last aspect that influences the number of likes is the use of saturation. In Figure 7 we see we have much data on the categories 'Vibrant' and 'Muted'. Further in the table we find that the saturation is heavily influenced by the color temperature and palette variety. From this I would say that all these choices influence the number of likes in the end. We do not see this influence from what account posted the art or the number of followers it has.

10 Conclusion

The main aspects that are involved when estimating a Baysian network for categorical variables have been discussed. These concepts were put to the test in various experiments on a constructed dataset containing annotations, artworks and account data. We found several small influences to the number of likes on social media.

11 Future work

These experiments can be enhanced using a scoring metric that deals with mixed data. This will make the findings more accurate since then the data can be evaluated as or close to it's original state. More and better distributed data should be found to more confidently say something about all variables. More in depth research can be done on how to get data for various variables that describe the content of a painting.

References

[1] H. Liu, C. Li, Y. Li, and Y. J. Lee. Improved baselines with visual instruction tuning, 2023.

[2] H. Liu, C. Li, Q. Wu, and Y. J. Lee. Visual instruction tuning. In *NeurIPS*, 2023.

Appendix

11.1 Data

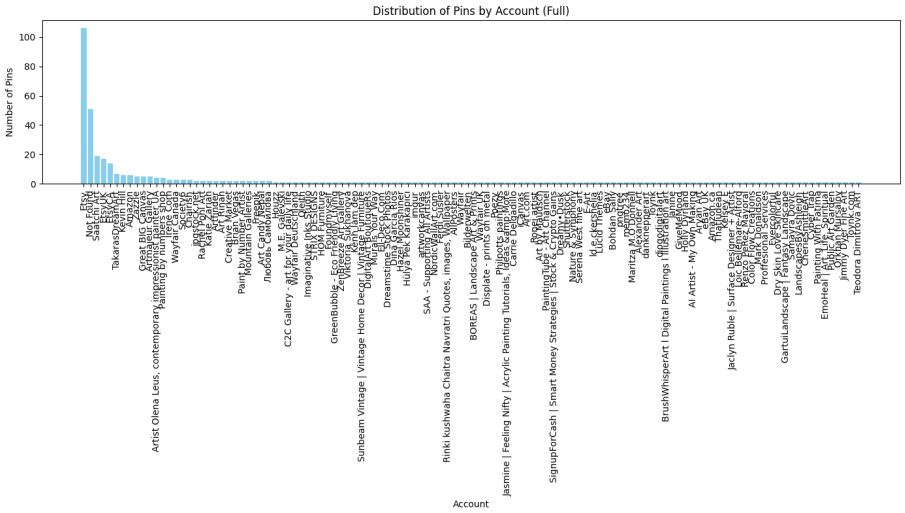

Figure 5: Distribution of accounts that have posted artworks used in this research.

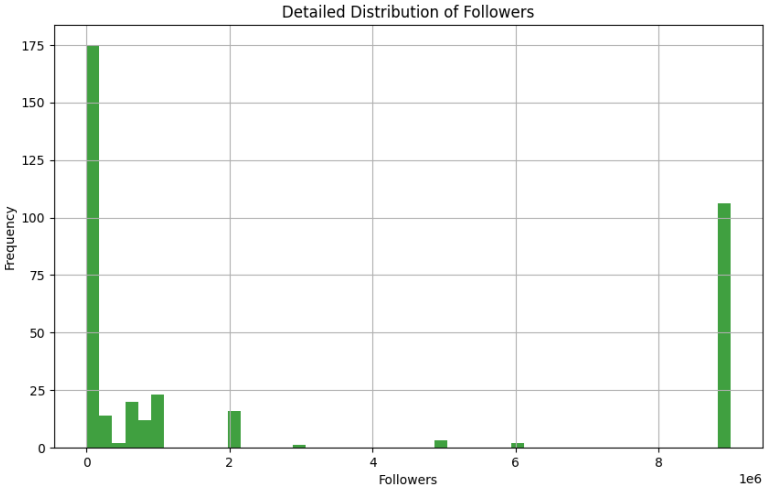

Figure 6: Distribution of number of followers over the data.

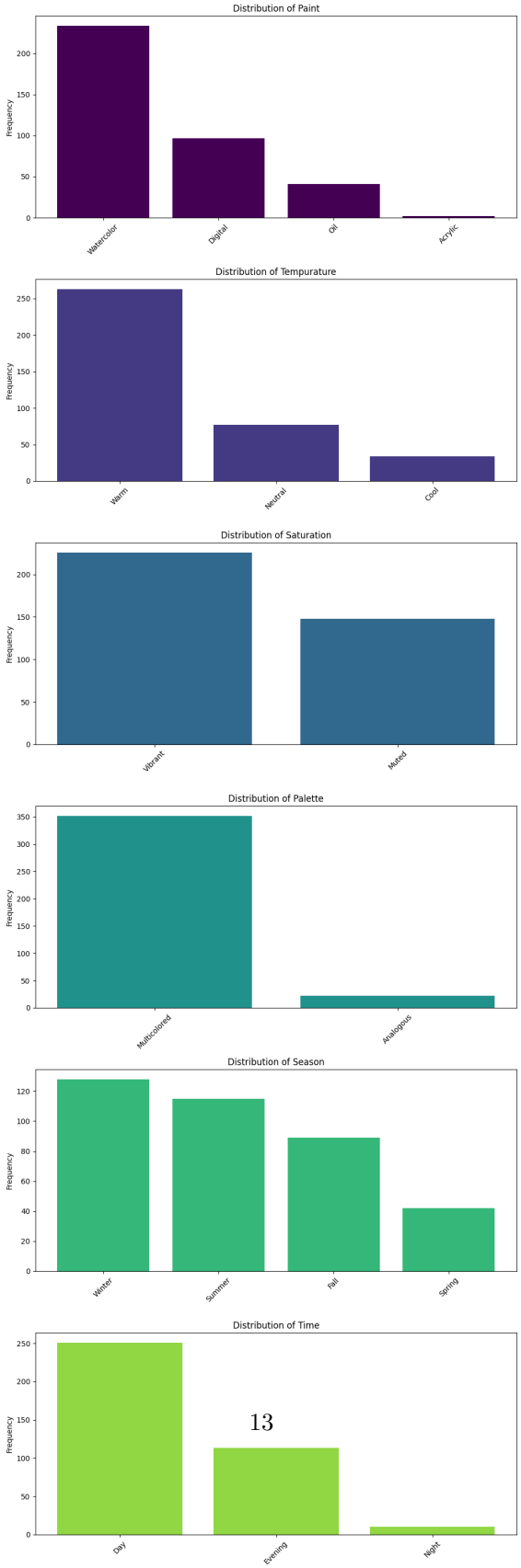

Figure 7: Distribution of all the categories annotated by the LLaVA model.

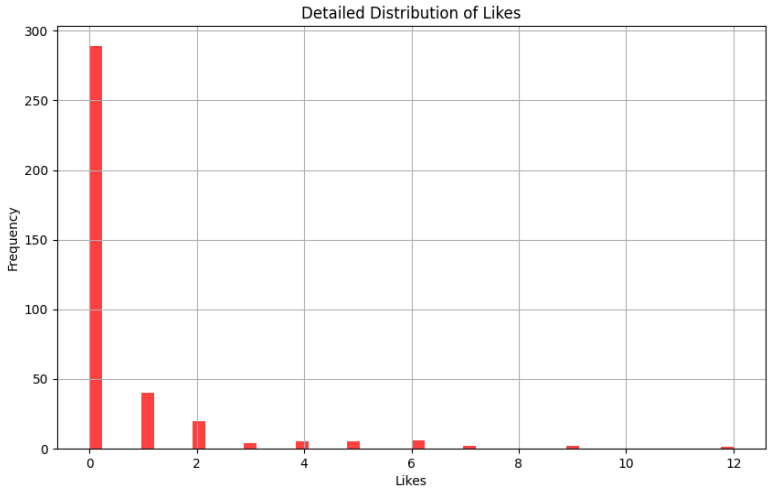

Figure 8: Histogram of likes.

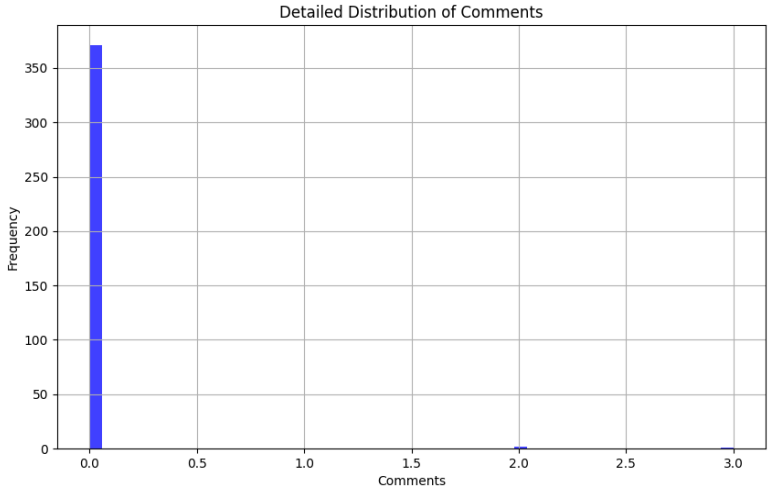

Figure 9: Histogram of comments.

11.2 Code

```r
library(bnlearn)

# Function to perform a simple MCMC over Bayesian Network structures
mcmc_bayesian_network <- function(data, num_iterations) {
  # Start time
  start_time <- Sys.time()

  # Generate a random initial DAG
  nodes <- colnames(data)
  initial_dag <- random.graph(nodes = nodes, prob = 0.1)

  if (!acyclic(initial_dag)) {
    stop("Generated graph is not acyclic")
  }

  # Initial score calculation
  # initial_fit <- bn.fit(initial_dag, data, method = "mle")
  current_score <- score(initial_dag, data, "bde")
  current_dag <- initial_dag

  # To store graphs and their scores at each iteration
  iteration_details <- list()

  for (i in 1:num_iterations) {
    current_neighbors_dags = generate_neighbor_dags(current_dag)

    current_neightbors_count = length(current_neighbors_dags)

    if (runif(1) < 1/current_neightbors_count) {
      # Generate a random neighbor
      proposed_index <- sample(seq_along(current_neighbors_dags), size = 1)
      proposed_dag <- current_neighbors_dags[[proposed_index]]  # Correctly access
    the selected DAG

      # Now use the selected DAG
      #proposed_dag <- nodes(proposed_dag)

      proposed_neighbors_dags = generate_neighbor_dags(proposed_dag)
      proposed_neightbors_count = length(current_neighbors_dags)

      # Fit and score the proposed DAG
      proposed_score <- score(proposed_dag, data, "bde")

      # Calculate acceptance probability
      likelihood_ratio <- exp(proposed_score - current_score)
      hastings_ratio = current_neightbors_count / proposed_neightbors_count
      acceptance_probability <- min(1, likelihood_ratio * hastings_ratio)

      # Decide to accept the new graph or not
      if (runif(1) < acceptance_probability) {
        current_dag <- proposed_dag
        current_score <- proposed_score
      }
    }
    # Record graph and score
    iteration_details[[i]] <- list(graph = amat(current_dag), score = current_score)
```

```
    if (i %% 100 == 0) {
      cat("Iteration: ", i, "\n")
    }
  }

  # End time
  end_time <- Sys.time()

  # Calculate duration
  duration <- end_time - start_time

  # Format and print the duration in a suitable unit, e.g., "minutes"
  cat("Execution Time: ", format(duration, unit="mins"), "minutes\n")

  return(iteration_details)
}


# Function to generate all neighbor DAGs of a given DAG
generate_neighbor_dags <- function(dag) {
  nodes <- nodes(dag)
  num_nodes <- length(nodes)
  am <- amat(dag)  # Get adjacency matrix

  neighbors <- list()  # Store neighbor DAGs

  # Check acyclicity of an adjacency matrix
  check_acyclic <- function(am) {
    temp_dag <- empty.graph(nodes = nodes)
    if (!tryCatch({
      amat(temp_dag) <- am
      acyclic(temp_dag)
    }, error = function(e) { FALSE })) {
      return(FALSE)
    }
    return(TRUE)
  }

  # Try adding arcs
  for (i in 1:(num_nodes - 1)) {
    for (j in (i + 1):num_nodes) {
      if (am[i, j] == 0 && am[j, i] == 0) {
        # Try adding the arc from i to j
        am[i, j] <- 1
        if (check_acyclic(am)) {
          temp_dag <- empty.graph(nodes = nodes)
          amat(temp_dag) <- am
          neighbors[[length(neighbors) + 1]] <- temp_dag
        }
        am[i, j] <- 0  # Reset

        # Try adding the arc from j to i
        am[j, i] <- 1
        if (check_acyclic(am)) {
          temp_dag <- empty.graph(nodes = nodes)
          amat(temp_dag) <- am
          neighbors[[length(neighbors) + 1]] <- temp_dag
```

```
        }
        am[j, i] <- 0  # Reset
      }
    }
  }

  # Try deleting arcs
  for (i in 1:num_nodes) {
    for (j in 1:num_nodes) {
      if (am[i, j] == 1) {
        # Try deleting the arc from i to j
        am[i, j] <- 0
        temp_dag <- empty.graph(nodes = nodes)
        amat(temp_dag) <- am
        neighbors[[length(neighbors) + 1]] <- temp_dag
        am[i, j] <- 1  # Reset
      }
    }
  }

  # Try reversing arcs
  for (i in 1:num_nodes) {
    for (j in 1:num_nodes) {
      if (am[i, j] == 1) {
        # Try reversing the arc from i to j
        am[i, j] <- 0
        am[j, i] <- 1
        if (check_acyclic(am)) {
          temp_dag <- empty.graph(nodes = nodes)
          amat(temp_dag) <- am
          neighbors[[length(neighbors) + 1]] <- temp_dag
        }
        # Reset the matrix
        am[i, j] <- 1
        am[j, i] <- 0
      }
    }
  }

  return(neighbors)
}

# Function to calculate edge probabilities from MCMC output
calculate_edge_probabilities <- function(iteration_details) {
  num_iterations <- length(iteration_details)
  if (num_iterations == 0) {
    stop("No iterations data available")
  }

  # Assume all graphs have the same size
  num_nodes <- nrow(iteration_details[[1]]$graph)

  # Initialize a matrix to count edge occurrences
  edge_count_matrix <- matrix(0, nrow = num_nodes, ncol = num_nodes,
                              dimnames = list(colnames(iteration_details[[1]]$graph)
    ,
                                              colnames(iteration_details[[1]]$graph)
    ))
```

```
170
171   # Accumulate edge presence
172   for (iteration in iteration_details) {
173     edge_count_matrix <- edge_count_matrix + iteration$graph
174   }
175
176   # Calculate probabilities
177   edge_probability_matrix <- edge_count_matrix / num_iterations
178
179   return(edge_probability_matrix)
180 }
181
182
183 # Function to find and return the best scoring graph from MCMC iterations
184 find_best_scoring_graph <- function(iteration_details) {
185   if (length(iteration_details) == 0) {
186     stop("No iteration data available")
187   }
188
189   best_score <- -Inf  # Initialize to very low to ensure any score is higher
190   best_graph <- NULL  # Placeholder for the best graph
191
192   # Loop through all iterations to find the highest score
193   for (iteration in iteration_details) {
194     if (iteration$score > best_score) {
195       best_score <- iteration$score
196       best_graph <- iteration$graph
197     }
198   }
199
200   # Check if a best graph was found
201   if (is.null(best_graph)) {
202     stop("No valid graph was found in the iterations")
203   }
204
205   return(list(graph = best_graph, score = best_score))
206 }
207
208
209 # Function to load and preprocess data
210 load_data <- function() {
211   # Hardcoded file path
212   file_path <- "C:/Users/tomei/OneDrive/Documenten/Genomics/data.csv"
213
214   # Load the data
215   data <- read.csv(file_path)
216
217   # Convert categorical columns to factors
218   categorical_columns <- c("account", "paint", "tempurature", "saturation", "palette
      ", "season", "time",
219                            "followers", "comments", "likes")
220   data[categorical_columns] <- lapply(data[categorical_columns], factor)
221
222   # Print the data structure to confirm conversions (Optional, can be commented out
      in production)
223   print(str(data))
224
225   # Return the learned network structures
```

```
226   return(data)
227 }
```