Causal analysis of Titanic disaster through Bayesian Networks

Grzegorz Meller, Gabriele Risino

1. Introduction

The sinking of the Titanic is one of the most infamous shipwrecks in history. On April 15, 1912, the widely considered “unsinkable” Titanic sank after colliding with an iceberg. Unfortunately, there weren’t enough lifeboats for everyone on board, resulting in the death of 1502 out of 2224 passengers and crew. The story of the Titanic is very popular in the academic community of machine learning, which started with the legendary Titanic ML competition. The competition is fairly simple, the task is to use machine learning to create a model that predicts which passengers survived the Titanic shipwreck.

In this project, we decided to analyze the Titanic data from a bit different angle. Our goal is to build the Bayesian Network, so to be able to drive the inference questions, and gain knowledge on how certain parameters (like ticket class, gender, age) were affecting the probability of survival.

2. Dataset

The dataset comes from the Kaggle platform, from the already mentioned competition. The training set contains 890 samples of passengers, each having the following feature information:

- Survived: Outcome of survival (0 = No; 1 = Yes)
- Pclass: Socio-economic class (1 = Upper class; 2 = Middle class; 3 = Lower class)
- Sex
- Age
- SibSp: Number of siblings and spouses of the passenger aboard
- Parch: Number of parents and children of the passenger aboard
- Fare
- Embarked: Port of embarkation of the passenger (C = Cherbourg; Q = Queenstown; S = Southampton)

The following features are not all of available in the dataset. During data pre-processing, we removed features that do not influence the survival variable (e.g. Name). What is more, during data pre-processing we converted all categorical data to dummy variables. When it comes to Age and Fare, we converted the values into three discrete ranges in order to group passengers of a similar age or fare.

3. Building Bayesian Network

To learn the structure of the network, we used the algorithm called Hill Climb search, which automatically constructs the Bayesian Network. The essence of this algorithm is to find a structure that maximizes the scoring function which in our case is the Bayesian Information Criterion (BIC). BIC is a log-likelihood score with an additional penalty for network complexity, to avoid overfitting and it measures how well a model is able to describe the given dataset. After applying this method we obtain the following Bayesian Network:

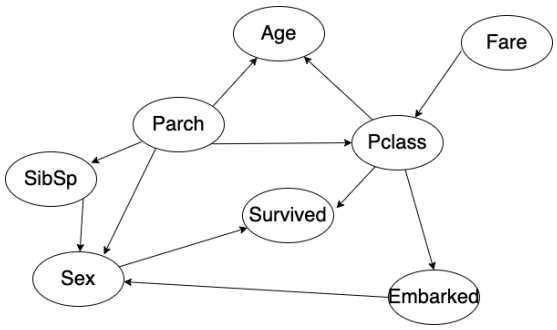

Fig 1. Graphical representation of the Bayesian Network

The automatically created network looks fairly reasonably. However, we decided to apply our domain knowledge and manually modify the arrow direction between Age and Pclass. We know that age is this

parameter that greatly defines a higher level of well-being and this way affects the passenger class. After applying this manual refinement the BIC score improved (from -5488 to -5686).

4. Running Exact Inferences

Having the Bayesian network constructed, we would like to extract knowledge from it and perform inference queries i.e. computing the posterior probability for a set of query variables, given some observed event in the form of evidence ($P(X \mid e)$). To perform it, we used the exact inference method for computing queries, using namely variable elimination algorithm. Questions that we asked our network and the results are the following:

- *Did women have a better chance of surviving the catastrophe than men?*

Sex	Probability of Surviving
Women	70.2%
Men	19.5%

- *Does the age influence the chance of survival?*

Age	Probability of Surviving
from ~0 to ~27	35.3%
from ~27 to ~53	36.8%
from ~53 to 80	47.9%

- *Did the purchase of a first-class ticket give a greater guarantee of security?*

Class	Probability of Surviving
1	58.5%
2	41.3%
3	26.1%

5. Running Approximate Inferences

The exact inference gave us a strong confirmation on our questions. However, performing inference this way is very time-consuming and can have exponential time and space complexity, which could significantly slow down the execution of analytical queries when dealing with larger networks than this one. The solution to this problem is performing the approximate inference, which consists of randomized sampling algorithms having an accuracy depending on the number of samples generated.

As an example, we run the approximate inference query (using the rejection sampling method (blue line) and likelihood weighting method (green line)) with a question on the probability of surviving, while having the ticket in the first class. From the exact inference, we remember that this probability is equal to almost 60%. Figure 2 presents the approximate inference results in relation to the number of samples generated.

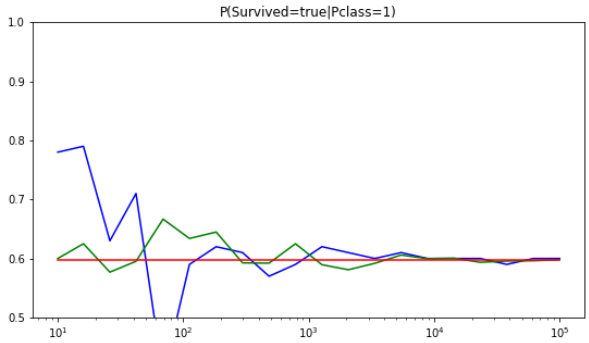

Fig. 2. Approximate inference