Ordered Sets in Data Analysis

Big Homework - Neural FCA

Vladislav Pendishchuk

Data Science

Faculty of Computer Science
Higher School of Economics

December 9, 2024

Data

For this homework I used Kaggle's famous Titanic passengers dataset. The dataset contains data on 891 passengers of Titanic, with each row representing a passenger with the following features:

- PassengerId - the ID of the passenger.
- Survived - the target feature, 1 if survived, 0 if didn't survive.
- Pclass - passenger class, a proxy for socio-economic status (SES).
- Name, Sex, Age - self-explanatory.
- SibSp - the number of siblings and spouses aboard.
- Parch - the number of parents and children aboard.
- Fare - the passenger fare.
- Cabin - the cabin number.
- Embarked - the port of embarkation (C = Cherbourg, Q = Queenstown, S = Southampton).

Example of data (with names shortened in order to make the table fit):

PassengerId	**Survived**	**Pclass**	**Name**	**Sex**	**Age**	**SibSp**	**Parch**	**Ticket**	**Fare**	**Cabin**	**Embarked**
1	0	3	Braund, (...)	male	22.0	1	0	A/5 21171	7.2500	NaN	S
2	1	1	Cumings, (...)	female	38.0	1	0	PC 17599	71.2833	C85	C
3	1	3	Heikkinen, (...)	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
4	1	1	Futrelle, (...)	female	35.0	1	0	113803	53.1000	C123	S
5	0	3	Allen, (...)	male	35.0	0	0	373450	8.0500	NaN	S

Objective

The goal is to train a model that can predict whether a passenger survived based on their features. That is, the problem we're solving is that of binary classification, with the target feature being Survived.

Methodology

FCA-based approach

Feature preprocessing

From the start, several features were discarded:

- Name, PassengerId and Ticket are all essentially identifiers, and are thus of no use to us for the purpose of classification.
- Fare roughly corresponds to the passenger's class in the context of survival analysis, so we'll treat this feature as a duplicate of Pclass and drop it to make things easier to interpret.
- Cabin, while not being an identifier, is close to being unique for most passengers, isn't an ordinal feature, and, when treated as a categorical feature, would generate dozens of features upon one-hot encoding.

Age was binarized via ordinal $\geq$ encoding with thresholds 18, 40 and 60: Age $\geq$ 18, Age $\geq$ 40 and Age $\geq$ 60. These thresholds were chosen as rough boundaries for major age groups: children $[0; 18)$, young adults $[18; 40)$, middle-aged adults $[40; 60)$ and senior adults $[60; +\infty)$. The $\geq$ operator was chosen because the intervals for the aforementioned age groups are closed on the left.

Features Parch and SibSp were shown to be distributed mainly around thresholds 0 and 1, with most passengers bringing no relatives on board. Therefore, it was decided to apply ordinal scaling to these features with two thresholds: 1 and 2. The operator used for ordinal scaling in these two cases is still $\geq$, as the intention was to have "loners" (that is, most people in the dataset) to not have a "true" value in any of the resulting features, reducing the number of computations needed for FCA.

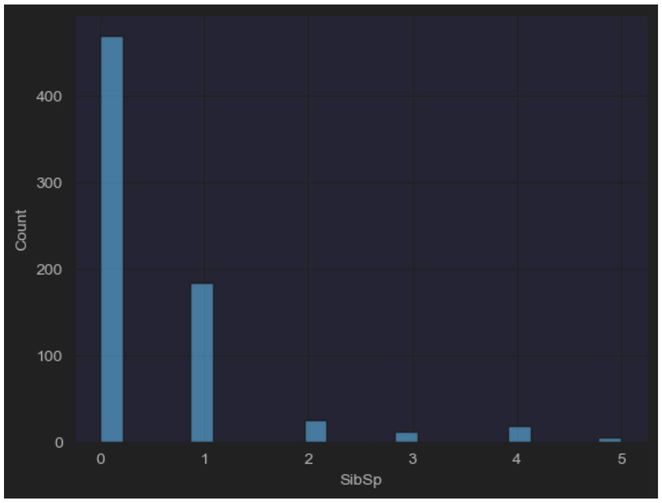

Figure 1: SibSp feature distribution

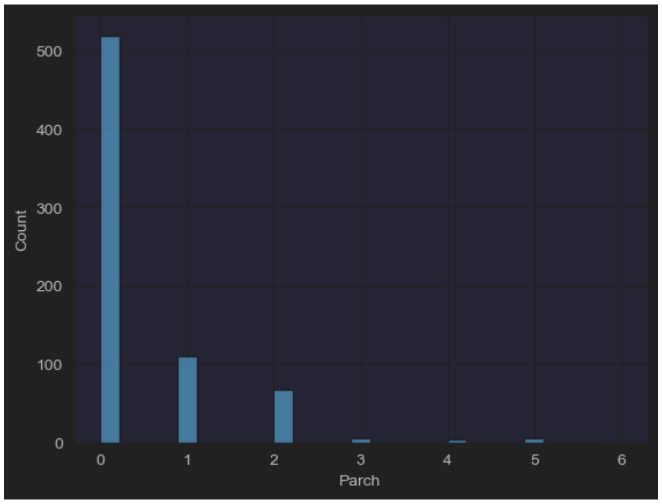

Figure 2: Parch feature distribution

Lastly, values in Pclass were mapped to corresponding ordinals in order to make the resulting features' names easier to read.

After preprocessing the numerical features and one-hot encoding the categorical features, there are 15 features in total: Age $\geq$ 18, Age $\geq$ 40, Age $\geq$ 60, SibSp $\geq$ 1, SibSp $\geq$ 2, Parch $\geq$ 1, Parch $\geq$ 2, Pclass_1st, Pclass_2nd, Pclass_3rd, Sex_female, Sex_male, Embarked_C, Embarked_Q, Embarked_S.

Concept network training and evaluation

To build concept networks the following process was used:

1. Build a formal context based on the given list of features.
2. Build a concept lattice.
3. Search for formal contexts with the highest F1 score.
4. Build the concept network based on the best concepts.

For step 2, the CbO algorithm has proven to be too slow to build a lattice from the given data in reasonable time. Thus, it was decided to use its faster polynomial time alternative - Sofia. Although this algorithm is less accurate, as it generates only a subset of the entire concept lattice, it does nonetheless make it feasible to solve the problem for large datasets with a large amount of objects and features.

For step 3, F1 was chosen as a metric that is most fitting for evaluating concepts for the purpose of binary classification. This step intentionally takes the formal context with the full feature set intent only as the last resort, no matter its actual F1 score ranking, as including it when building a concept network leads to unexplainable architecture with a sole neuron above the output layer.

At step 4 n best concepts were selected, where n is the minimum number of concepts sufficient to cover all train objects by their extents.

As 15 is a number of features that is pretty difficult to interpret in the context of FCA, I've decided to try building concept networks based on 4 different sets of features:

- all features;
- personal features - sex and age;
- passenger data - passenger class and port of embarkation;
- family-related data - SibSp and Parch.

However, when building the respective concept networks, it was discovered that it is impossible to build a concept network using only the family-related features. This is because after excluding the concept with the full feature set intent, we are left with concepts that are insufficient to cover all train objects. Therefore, this feature set has been removed.

The total number of obtained formal concepts for each feature set using Sofia is:

- all features - 99;
- personal features - 13;
- passenger data - 50.

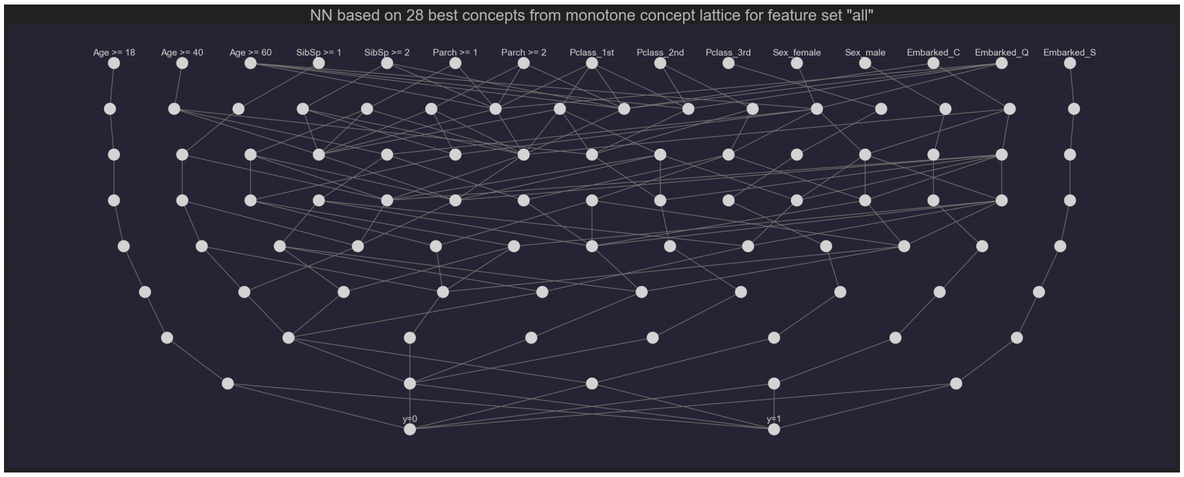

Figure 3: Architecture based on all features, 28 best concepts

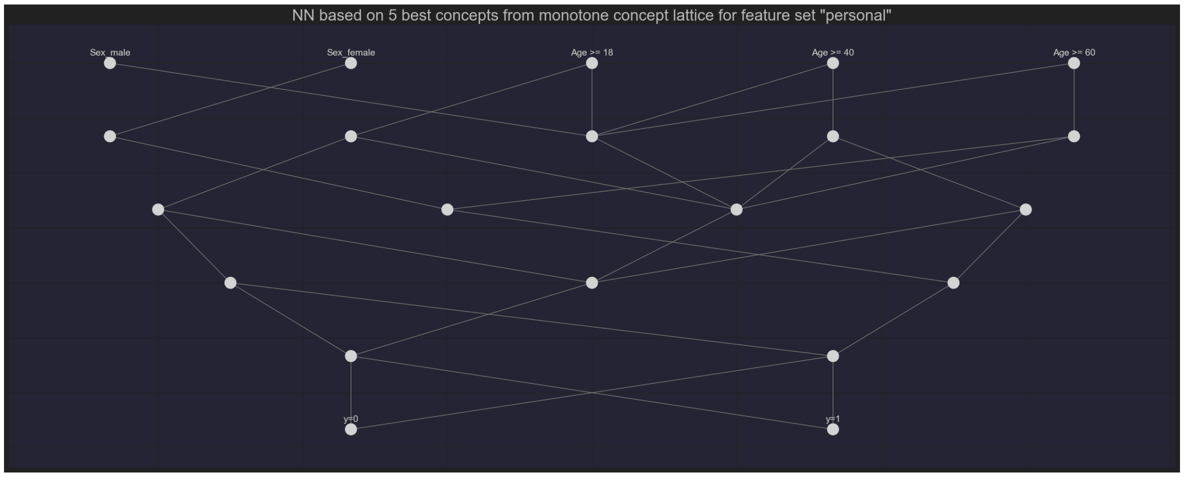
NN based on 5 best concepts from monotone concept lattice for feature set "personal"
Sex_male
Sex_female
Age >= 18
Age >= 40
Age >= 60
y=0
y=1

Figure 4: Architecture based on personal features, 5 best concepts

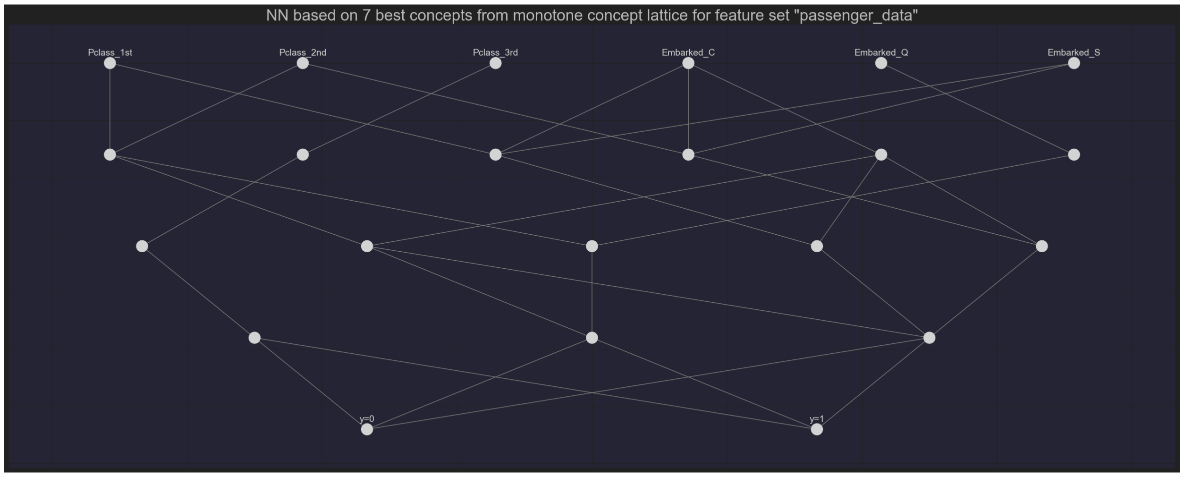
NN based on 7 best concepts from monotone concept lattice for feature set "passenger_data"
Pclass_1st
Pclass_2nd
Pclass_3rd
Embarked_C
Embarked_Q
Embarked_S
y=0
y=1

Figure 5: Architecture based on passenger data features, 7 best concepts

The resulting architectures are all of different degrees of complexity - in particular, the one based on all features is too complex to be explainable. The other two architectures feature less neurons and connections, but it is still quite difficult to explain the relations they are meant to capture.

For each architecture, I've tried fitting networks using different non-linearities, namely:

- GELU (Gaussian Error Linear Unit):

$$\text{GELU}(x) = x \times \Phi(x), \text{ where } \Phi(x) \text{ is the CDF for Gaussian distribution.}$$

- Leaky ReLU (Rectified Linear Unit):

$$\text{LeakyReLU}(x) = \max(0, x) + \eta * \min(0, x), \text{ where } \eta \text{ is the angle of the negative slope.}$$

- Hyperbolic tangent:

$$\tanh(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)}$$

The results are as follows:

	Macro F1		
Feature set	**GELU**	**LeakyReLU**	**Tanh**
all	0.721139	0.738858	0.727658
personal	0.760905	0.748826	0.760905
passenger_data	0.588988	0.588988	0.588988

The best macro F1 score was achieved at feature set "personal" with GELU non-linearity - macro F1: 0.7609, accuracy: 0.7850.

The following figure visualizes the fitted weights for each network. Overall, in my opinion, the resulting weights are difficult to interpret, providing little insight into how the features might be related to one another.

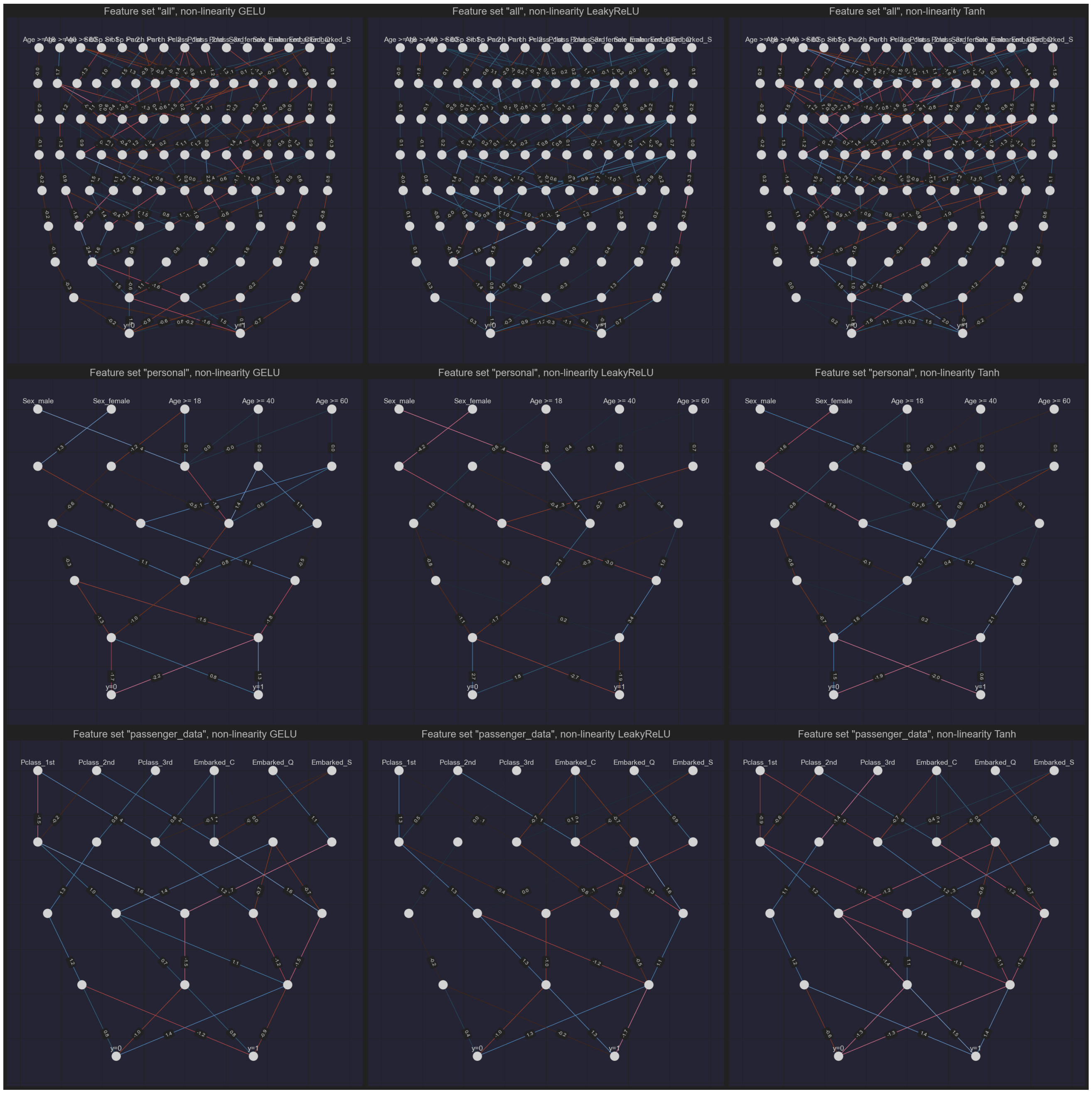
Feature set "all", non-linearity GELU
Feature set "all", non-linearity LeakyReLU
Feature set "all", non-linearity Tanh
Feature set "personal", non-linearity GELU
Feature set "personal", non-linearity LeakyReLU
Feature set "personal", non-linearity Tanh
Sex_male
Sex_female
Age >= 18
Age >= 40
Age >= 60
Feature set "passenger_data", non-linearity GELU
Feature set "passenger_data", non-linearity LeakyReLU
Feature set "passenger_data", non-linearity Tanh
Pclass_1st
Pclass_2nd
Pclass_3rd
Embarked_C
Embarked_Q
Embarked_S
y=0
y=1

Figure 6: Fitted weights visualization

Classic ML algorithms

In order to compare the performance of the best concept network with that of its alternatives from classic machine learning, the following algorithms were tested on the same dataset and problem statement:

- Gaussian Naive Bayes;
- KNN;

- Logistic Regression;
- Decision Tree Classifier;
- Random Tree Classifier;
- XGBoost.

Feature preprocessing

Just as before, the features Name, PassengerId, Ticket, Fare and Cabin were discarded.

The aforementioned ML algorithms do not require binarizing the features, and most of them are better suited to work with numerical data (some of them, such as KNN, practically only support numerical data). Because of this, for these algorithms the features were preprocessed in the following way:

1. All the categorical features were transformed into a number of numerical features via one-hot encoding.
2. All features were standardized.

Model training and evaluation

Cross-validation was used to tune each model's hyperparameters, and macro F1 was used as the scoring function for this purpose. The results of each model's training and evaluation are as follows (with the best obtained concept network added in the last row):

Model	**Accuracy**	**Macro F1**
KNN	0.804	0.774
LogReg	0.790	0.760
DecisionTree	0.804	0.773
RandomForest	0.836	0.807
XGBoost	0.781	0.755
Gaussian NB	0.683	0.654
Concept Network	0.785	0.761

Conclusion

While concept-based neural networks have the potential to provide insights into how different features influence model predictions, in this specific case, I must admit that its resulting architecture still seems too complex for me to easily understand the inner workings of the model. Even on a dataset as simple as the one used for this homework (the Titanic dataset is widely used as a "toy dataset" for learning purposes), the FCA-based approach shows its flaws, namely:

1. The requirement for all features to be binary increases the dimensionality of the data, making it more difficult for FCA-based algorithms to interpret it.
2. At the moment, the approach does not seem to be well suited for data with categorical features of high cardinality, for the same reasons as outlined above.

3. The resulting architectures can easily get too complex for explanation.

In terms of macro F1 score, the best performing concept network managed to outperform Gaussian Naive Bayes, Logistic Regression and XGBoost models. It is worth noting, however, that some of these classic algorithms may in fact outperform concept networks when trained on a dataset that is preprocessed in a way that best suits them.