Technical Perspective: A Relational Framework for Classifier Engineering

Wang-Chiew Tan
Recruit Institute of Technology
wangchiew@recruit.ai

A fundamental step in developing machine-learning solutions is that of *feature engineering*. Feature engineering refers to the process of generating a representation from data (called *features*) that can be fed as inputs to machine-learning models. The results of feature engineering thus have direct impact on the performance of machine-learning models. In developing machine-learning solutions, a large amount of time is typically devoted to feature engineering, which determines the right features to capture for improving the performance of the models.

In this paper, the authors describe a framework for feature engineering for programming machine-learning solutions over a database, assuming the model inputs numerical paramaters that may be tuned by fitting to training examples. The focus of the paper is on a widely used class of machine-learning models, called *classifiers*, which are used to predict an unknown category of a given entity based on the properties of that entity.

The running example of the paper considers the problem where a credit card company wishes to identify whether an incoming credit card transaction is a legitimate or fraudulent transaction (e.g., made with a stolen credit card). The credit card company may leverage historical transactions with both legitimate and fraudulent transactions as training data to train a classifier. The features that are extracted from the data may include properties that concern the state and country where a transaction was made compared to the state and country of billing address of the owner, the amount billed in the transaction, the history of transactions and so on.

Their framework assumes an underlying *entity schema*, which is a relation schema with a distinguished relation symbol. The distinguished relation represents the set real-world objects where the classifier makes predications upon. For the credit card example, since the classifier will ultimately be applied on transactions to determine the legitimacy of transactions, a natural candidate for the distinguished entity relation in the credit card example is the transaction relation which stores all transactions that occurred. The remaining relation schema will include additional information about the transaction, such as the country and state where each transaction took place, the card and amount involved, and information about the billing address of the credit card. Feature engineering is modeled as the process where an analyst specifies a sequence of *feature queries* in some language. For example, a feature query may select all transactions that took place in the same country and state of the owner's billing address, and another feature query may select all the ones that took place in the same country (but not necessarily the same state) of the owner. In their framework, a classifier is a function that maps a vector of numbers, where the numbers encode the features, into a boolean answer. To train a classifer, it is therefore necessary to convert the results of feature queries into numbers. This is done as follows. For every feature query and every entity in the set of entities (in this case, transactions), a vector of 1 or -1 can be obtained based on whether the feature query produces an answer for that entity. If an answer is produced (resp. not produced), then the feature query for that transaction is a positive example which will be given a label 1 (resp. negative example labeled -1). Based on the input vectors obtained this way, a classifier is learnt and can then be applied to future inputs.

After formalizing a relational framework for feature engineering, the authors further describe three fundamental problems. The *separabiity* problem essentially asks the question that for a given language of the framework and a given category of classifiers, whether it is sufficient to achieve good classification (i.e., separation of the positive from the negatives) based on the training data. More precisely, given a training instance over an entity schema, determine whether or not there exists a sequence of feature queries and a classifier that completely agrees with the training labels. Another problem, called the Vapnik-Chervonenkis dimensionality (*VC dimensionality*) problem, essentially asks what is the complexity of learnability. More precisely, the VC dimenion measures the complexity of a class of classifiers (e.g., linear or polynomial classifiers) for feature queries specified in some language. The last problem posed by the authors is the *identifiability* problem. This problem asks to decide, given a sequence of feature queries, whether there exists a training instance such that the resulting feature matrix is of full column dimension. In other words, the columns of the matrix are linearly independent.

The paper presents complexity results on these three problems with attention on linear classifiers and where feature queries are conjunctive queries. Perhaps more importantly, the paper presents a novel formal framework for classifier engineering, describes an initial set of results based on the framework, and leaves several open challenges for the interested reader.