M.Eng. Thesis Proposal

Alexander Nordin

Advisor: Kalyan Veeramachaneni

1 Abstract

Machine learning has seen a swell of activity in recent years. Many applications of machine learning involve industry-specific tasks where acquired data is leveraged to build classifiers that can better predict events. Today we have an abundance of data and a need for more machine learning experts to perform analysis and derive insights. We'll call data analysis the process of a machine learning expert analyzing data. One major impediment to the data analysis process is coming up with relevant prediction problems that can be answered with the data. We propose a system that can automatically generate prediction problems. Incorporating our technology into the data analysis process will greatly reduce the barrier to entry for understanding and advantageously using machine learning for data scientists and laymen.

2 Introduction

The introduction is split into three sections, Background, Machine Learning Workflow and Trane. The Background will address the current state of machine learning and it's applications. The Machine Learning Workflow details the major components of gaining insights from a dataset. The Trane section introduces our idea.

2.1 Background

Machine learning is a field of computer science that gives computers the ability to learn without being explicitly programmed [2]. Machine learning solves problems across many different fields. For example, financial trading is becoming littered with bots that outperform traders and work entirely by themselves. Healthcare uses machine-learned programs to predict many facets of health for patients. For instance, in one study, computers were able to spot 52% of the cancers as much as a year before the patients were officially diagnosed [4]. Machine learning is also used in marketing personalization, fraud detection, security, search, natural language processing and many more fields. Soon enough, machines will be getting behind the wheel and taking over driving. Clearly, machine learning has amply demonstrated its capability to generalize and problem-solve.

Machine learning is capable of solving multi-faceted problems across many domains of knowledge and expertise. Machine learning algorithms automate superfluous tasks by using generalizations across problem domains. In this spirit, we propose a system that can generate prediction problems automatically. We seek to automate and assist with part of the process of building a functional machine learning algorithm. Next, I show the typical workflow for deriving insights from a dataset.

2.2 Machine Learning Workflow

The following is a typical workflow for deriving insights from a dataset:

1. Organize and clean the relevant data
2. Determinine the question
3. Apply a series of operations to the data to generate labels based on the question you came up with
4. Engineer and calculate features
5. Select and apply a machine learning algorithm

Consider a hypothetical example. Owen, the owner of a website has data about customer activity. He wants to improve the customer experience, so he hires a data scientist, Dan. Dan spends a few days looking through the data to understand what it is and what he can do with it. Dan comes back to Owen and suggests that he predict customer churn or the next product a customer is likely to buy. Owen says, predicting churn two months in advance could be helpful. Dan spends a week building a pipeline to predict churn. He discovers that it's hard to predict 2 months in advance, but he can predict it 1 week in advance. That's not helpful for Owen because he needs the information 2 months ahead of time. Now that they've established predicting customer churn two months in advance is not feasible, they move on to solving another prediction problem. Owen asks Dan to predict the next product a customer will buy. The process goes on, typically taking weeks or months.

2.3 Trane

Kaggle, a platform for hosting data science competitions, recently conducted a massive survey of over 16000 machine learning experts. They found that the four largest barriers that data scientists face, in order, are dirty data, a lack of data science talent, a lack of management/financial support and a lack of a clear question to answer [5]. The short narrative above, based in reality on the Kaggle survey data, shows a clog in the pipeline of machine learning.

We propose a system, named Trane, which data scientists can use to greatly improve their efficiency. Trane is designed to use a dataset to automatically generate possible and relevant prediction problems that a data scientist can solve. Trane consists of three main parts: a language for defining prediction problems, a generator to invent prediction problems and a labeller to generate prediction problem labels. The language specifies the format and order in which different types of operations may be applied to the dataset in order to generate labels while simultaneously representing a prediction problem. The generator takes the dataset and generates prediction problems relevant to the data. The labeller uses prediction problem definitions defined in Trane and generated by the generator to come up with labels for data entities.

Both Owen and Dan could use Trane to better understand the kinds of problems they can solve with their data. Owen the website owner would use Trane to see the kinds of problems his data is capable of solving. In other words, a person with no machine learning background could use Trane to understand problems machine learning could solve. Dan, the data scientist, could use Trane to improve his ability to brainstorm prediction problems and even observe prediction problems he wouldn't have otherwise been able to invent. In other words, a machine learning expert can leverage Trane to accomplish his job more efficiently and effectively. Trane provides value to both the layman and machine learning experts. Essentially, Trane automates and helps with steps 2. and 3., shown above, of the typical machine learning pipeline.

One extremely helpful feature we intend to build onto Trane is a natural language processor. We seek a way to translate problems formally defined in Trane's proprietary language to plain English. Layman would thus be able to use Trane to not only generate prediction problems, but also read in English what these solvable problems are. Furthermore, we seek to build the reverse operation. In other words, we want a way to translate plain English into a formal prediction problem in Trane. A layman would be able to define prediction problems without any machine learning knowledge, thereby democratizing access to the power of machine learning.

3 Related Work

Kalyan Veeramachaneni and Benjamin Shreck, a research supervisor and past M.Eng student respectively, in the Data to AI lab within the Laboratory for Information and Decision Systems worked on a very similar problem. Shreck built a language, named Trane, which is used to express prediction problems and outlined the major ideas behind the system. His work also proved the validity of Trane's ability to represent prediction problems. Shreck selected 54 data science problems from the data science competition website *Kaggle*. These problems come from a wide array of applications from predicting sales for Walmart stores to predicting taxi trip durations. Trane was capable of representing all 54 problems. Furthermore, Shreck explored restricting Trane to a smaller sub-set of operations called LittleTrane to reduce the quantity of questions generated by the generator. Despite restricting Trane's expressiveness 51 of 54 problems could still be expressed. To be clear, Shreck's Trane is a language, the Trane proposed in this paper is a system to autonomously perform prediction engineering [1].

Feature Tools is an open source library that can perform deep feature synthesis [3]. Deep feature synthesis uses labels, training cutoff times and entities to construct features from the data. In the five steps mentioned in the introduction above, feature tools automates and assists with the fourth step, engineering and calculating features. Trane occupies steps 2 and 3 of the pipeline: determinig the question you seek to answer and applying the series of operations to the data to generate labels based on the question you came up with. Combining Trane and Feature Tools would allow for a massive efficiency boost in data science and possible full automation of the data science pipeline.

4 Implementation

My ideas will be influenced by Shreck's implementation and I will perform a ground-up implementation of the system. My system will be called Trane. Trane is focused on generating prediction problems for time series data. My implementation of Trane is broken into four important modules: the input parser, the operations, the generator and the labeller. I will elaborate on what each of these modules is responsible for. Trane aims to build prediction problems for predicting entities characteristics across time. For instance, it could be used to predict sales. To do so, Trane must segment the input data, allowing only part of the data relevant to an entity to be observed for training the classifier; otherwise the training data would contain all the answers. For instance, when trying to predict when a customer will leave the website, ourdataset for his interaction with the website will consists of actions he takes. If our training data includes the action where he leaves the page, our classifier will be able to predict when the customer leaves perfectly. That's bad because the classifier would not generalize to cases where it can not see the future. There's an obvious conundrum there, which is why the system needs cutoff times. While generating cutoff times automatically may be something to build into the system in future iterations, for now we will require the user to input cutoff times.

We seek to explore how to apply transfer learning to our domain. Transfer learning is the process of generalizing results from one problem and using them to solve another [6]. For instance, we could use Trane to generate relevant questions for a data set, then use user-annotated results to determine the most relevant questions. Finally, we could use the learned results in each of the new domains Trane is applied in. In other words, we could have the ability to succesfully determine the most relevant questions for unseen datasets.

The system will be built to allow for simple human interaction. There are two key efforts to making the system easily human-compatible. First, Parameter tweaking can be optimized by the user in order to more accurately cater to whatever prediction problem he has interest in. For example, if one prediction problem is defined as predict if sales will be over $2000 in two weeks, the user could tweak the specified value to a more realistic value for his use case. Second, the system will involve a way to translate prediction problems formulated in Trane into simple English. Parameters for various operations will be tweakable by the user. This would allow for laymen to use and easily understand Trane's outputs. Additionally, we seek to build a way to translate English into the formal language Trane. If we can accomplish this goal, our system would offer an extremely simple and accessible way for laymen to easily interact with and take advantage of machine learning.

4.1 Input Parser

The input parser is the input spout of the system. The input parser takes two pieces of input. The first input is a meta.json file containing all the meta information about the data in question. The second input is the data-set, either as a series of relational tables with a single primary key or a single table. The input parser flattens the relational table data into a single data via the primary key. The input parser passes two pieces of information on to the next stage of the pipeline: the meta.json data and a single table. The single table contains a mapping from each entity of the data, for instance taxi id or store id, to all the data relevant to that entity. There will be many data points for each entity as Trane deals with data that will be measured across time. An example is a dataset containing sales information for stores over a multi-month period.

4.2 Operations

Operations are functions that execute on the data in the table. Chaining these operations together in a specific way can create questions. These questions are precisely what our system, Trane, is meant to generate. We define 4 different types of operations: row, filter, transformation and aggregation operations. Each operation has a specification of its inputs and outputs. There are multiple functions available within each operation. All the functions that are a sub-class of the operation must conform to the specification of the operation.

4.3 Row Operation

A row operation has a variety of different functions that each conform to the specification of a row operation. A row operation takes in the entire dataset as input and returns the entire dataset. The operation must apply the function to each row of the data on the specific column of interest (otherwise referred to as the label generating column).

4.4 Filter Operation

Filter operations remove rows from the dataset. They take the dataset and a function as input. They iterate over the values in the column of interest. All filter operations return a boolean true or false. If a row's column of interest value returns false, the row is removed.

4.5 Transformation Operation

A transformation operation is a subset of the class of multi-row operations. Row operations and filter operations operate on a row by row basis. Multi-row operations operate on many rows at a time. A transformation operation takes in a dataset and returns a dataset with at least 1 fewer rows.

4.6 Aggregation Operation

An aggregation operation is also a subset of the class of multi-row operations. Aggregation operations only slightly differ from transformation operations. Aggregation operations take in a dataset and return only a single row.

4.7 Generator

The generator uses the metadata information from meta.json to chain together operations in specific ways. A generator looks to chain sequences together as follows, Filter Operation $\Rightarrow$ Row Operation $\Rightarrow$ Transformation Operation $\Rightarrow$ Aggregation Operation. The generator places functions from each operation type into the predefined order to generate prediction problems. The prediction problems are then passed on to the labeller.

4.8 Labeller

The labeller generates labels on the data based on prediction problem definitions. The labeller simply executes each of the prediction problems operations on each entity's data in order of the operations.

5 Expected Contributions

I anticpate contributing a system fully capable of performing prediction engineering to generate relevant prediction problems for a dataset. Furthermore, the system will be capable of translating questions from Trane into English. This system has the potential to revolutionize the way machine learning tasks are completed and greatly improve efficiency for data scientists and understanding for non-ML experts.

I will deliver an implementation of Trane built from the ground up. The code will be designed with aspects of modularity and readability in mind to facilitate easy adoption of the technology by experts.

Additionally, I will use transfer learning to deliver a system that can generalize knowledge about prediction problems across domains.

6 Conclusion

Trane is a system that takes a dataset and relevant meta information as input and creates two important outputs: a list of relevant prediction problems and the data required to train a machine learning algorithm. It's a general system that can be used on time varying datasets to obtain insights into the data. Trane has the potential to greatly improve data scientists productivity and layman's access to the powers of machine learning.

References

[1] Kalyan Veeramachaneni Benjamin Shreck. “What Would a Data Scientist Ask? Automatically Formulating and Solving Predictive Problems”. In: (2016).

[2] Wikipedia contributors. *Machine learning*. URL: `https://en.wikipedia.org/wiki/Machine_learning`.

[3] James Max Kanter. “Deep Feature Synthesis: Towards Automating Data Science Endeavors”. In: (2015).

[4] Bernard Marr. *The Top 10 AI And Machine Learning Use Cases Everyone Should Know About*. URL: `https://www.forbes.com/sites/bernardmarr/2016/09/30/what-are-the-top-10-use-cases-for-machine-learning-and-ai/#7362362794c9`.

[5] Mark McDonald. *Introducing Kaggle's State of Data Science & Machine Learning Report, 2017*. URL: `http://blog.kaggle.com/2017/10/30/introducing-kaggles-state-of-data-science-machine-learning-report-2017/`.

[6] Sebastian Ruder. *Transfer Learning - Machine Learning's Next Frontier*. URL: `http://ruder.io/transfer-learning/`.