Issues in Data Science

Ankush Varshneya
100853074

Carleton University

1 Report on Paper 1

[1]

In recent years business intelligence software has seen rapid growth and adoption from the enterprise industry. This is due to the fact that acquiring and storing large data from sources has become cheaper and easier to accomplish. Business intelligence software is a collection of decision support technologies, which given a large set of data allows knowledge workers to make faster and better decision. One key source of data in a typical business is a collection of operational databases. However, since the data may arrive from sources which may store data in different formats and quality, we must integrate, cleanse, and standardize data in preparation for BI tasks. We can use technologies such as Extract-Transformation-Loading tools (ETL) with a combination of Complex Event Processing engines (CEP) to effectively integrate data from different sources in real time. This data is then loaded into a repository, Data Warehouse (DW), which is usually a relational database management system (RDBMS).

After gathering the data in a DW, a set of mid-tier servers are used to provide various BI related services. Online analytical processing servers (OLAP) are used to analyse data in the DW. OLAPs allow the user to see a multidimensional view of the data, as well as common analytical functions. In recent years, as the main memory increases, in-memory engines are becoming more common. These in memory engines allow for dramatically improved performance of multidimensional queries. Reporting servers, allow for the rendering and execution of reports needed by knowledge workers. Enterprise search engine allow the user to find out all information based on particular keywords, such as a customer, based on data integrated in DW. Datamining engines extend OLAP and Reporting servers to allow for richer analytics on the DW in the form of predictive analysis. Text analytical engines are used to analyse large amounts of non-relational data to extract valuable information which may have been done manually in the past. After, the mid-tier component analysis of the data, we can build or use several existing front end applications to allow the user to call upon and make use of the data analysis in the mid-tier servers.

OLAP and other components of the mid-tier servers require fast query ability, so new data structures are introduced which are not found in traditional online transaction processing servers (OLTP). Index structures store all data related to a particular value in a column, so if a query asks for data where a column value is equal to something then it can use the index to returns the data as oppose to querying a larger base table. Further operations such as intersection and unions can be used on the indices to reduce or in some cases eliminate the query to a

base table. Reports usually require summarized and aggregated data; materialized views allow data to be precomputed and stored, as well as frequent results to be cached, which can help to speed up queries related to decision making and reports. However materialized views may only exist for specific reports, so we must combine index structures with materialized views to allow for something a bit more generalized. To allow for a greater degree of performance and manageability we can use partitions on the index to reduce the index into smaller partitions. We can also use column oriented storage in order to further increase performance. Data compression allows for faster queries since less scanning needs to be performed; reducing I/O costs, and allowing more data to be cached in memory. Null suppression allows fixed length values to be treated as variable length and dictionary compression allows repetitive values to be stored once in a key value map compressing space.

Query processing is another component, as discussed earlier OLAP allows queries on multi-dimensional data. Dimensions consist of a combination of operational data as well as aggregated and summarized data. As mentioned previously OLAP servers supports various operations on multidimensional view of data. OLAP servers may be implemented through either a multidimensional storage engine (MOLAP), a relational DBMS engine as the back end (ROLAP), or a hybrid combination of the two (HOLAP). MOLAP uses a multidimensional array abstraction to produce and support multidimensional view of the data; data is precomputed into large data cubes to speed up query processing. Using arrays allows us to leverage the indexing property on the arrays, however it is not so effective for space data; in case of space data, the data may be stored in another format. In contrast ROLAP maps the multidimensional model into relations which can use SQL to query the model. ROLAP relies on query optimization techniques, talked about before, to speed up query processing. In ROLAP, a schema in which dimensions are non-normalized is referred to as a star schema, where as a schema in which the dimensions are normalized is called a snowflake schema; majority of the ROLAP systems use the star schema to speed up query processing. A fact table with the coordination of the dimension tables are used to show what fact belongs to what coordinate in the dimensions. However to use ROLAP we need to extend SQL with additional functions such as richer aggregation functions. HOLAP combines the above two methods in various ways, such as splitting the detailed data to the ROLAP and the aggregated summarized data to the MOLAP, or by using ROLAP for older data and MOLAP for newer data.

RDBMS are used traditionally to support the data warehouses, and so we must ensure the RDBMS can support very large databases. Due to such large data we are faced with the challenge of effectively analyzing it in a timely manner; this is known as the Big Data Challenge. To ensure performance we have to make a query execution plan to effectively optimize the query, using the mentioned optimization above as well as some newer ones, to allow for better scalability. The execution plan must be chosen carefully as an incorrect plan may slow down processing. Using parallel RDBMS have been suggested which,

using the engines based on Map Reduce paradigm, allow data to be queried in parallel; hence reducing latency. Two approaches are used for the parallelism, shared disk where only disk is shared between nodes and shared nothing where nothing is shared and each node has its own data. The trade-offs are that in shared disk data does not need to be transferred; where as in shared nothing the bottle neck of using a common resource is reduced.

2 Report on Paper 2

[2]

Machine learning uses data to automatically learn about and make programs for various tasks. This makes it very attractive as appose to other manual methods. While several forms of machine learning are present, classification is one that is widely used. A classification system takes as input a vector of discrete or continuous values and outputs a single discrete value based on what was learned from the input. Learning for a classification system consists of 3 parts: representation, evaluation, and optimization. Representation has to do with the formal language a computer can handle which will be used to represent the classifier. In order for a classifier to be good it must be able to learn, for which it needs to be in a hypothesis space to do. Evaluation consist of evaluation (scoring) functions which with the help of an algorithms can distinguish good classifiers from the bad ones. Optimization searches all classifiers produced by evaluation to find the best one based on some criteria.

Generalization is a key goal of machine learning and is needed for a good classifier. A good approach for choosing data is to separate data into blocks based on some similarity, and then separate blocks into training and test data; we can choose one element from each of the groups as test data and the rest of the elements as training data. This way we will be able to train our machine without it being skewed and contaminated towards a particular set of data, and we will be able to test it with all elements for each of the groups. However, the data alone is not enough to make a good classifier. Even if we have a good sample of data we may still be missing crucial information. Luckily we can apply inductive and deductive reasoning to transform a small set of data input knowledge into a larger set of data output knowledge. Inductive output knowledge can be more easily generated from a smaller sample, so this is what is used by a learner.

Over fitting and under fitting are an issue that may arise in learning, which can be classified as bias and variance respectively. Bias is when the learner learns concise and consistent information which turns out to be false positive. Variance is when the learner learns random information due to noise which convolutes the overall result. In the case of over fitting and under fitting, while the training data will lead to 100% accuracy, the test data may have very low accuracy; this will lead to a bad classifier. Bias may be caused when a linear learner tries to learn about a non-linear data set and vice versa, as it will not be able to induce the correct result. High variance may be caused by a decision tree learner, as it may treat different data sets generated by the same phenomena or event as different when they should be treated identically. Techniques like cross-validation and the use of a regularization term may help combat over fitting and under fitting.

Another issue is that of the curse of higher dimensionality. In higher dimensions, as the number of features increase, generalizing gets exponentially harder and a fixed training set can only cover a tiny fraction of the input space. If we only need to learn about say two dimensions, its easy to do when there is a small amount of total dimensions; but increasing the total dimensions any larger say one hundred, then the noise created by the 98 extra dimensions distracts from the two dimensions of interest, the signal, and so any prediction is at best random. Even if the 98 extra dimension were of interest, in such a high dimensional space all inputs seem identical and so the result is meaningless. We can use the fact that in most application the sample is not spread uniformly and hence are

concentrated in a lower dimension. The learner can then map the sample to a lower dimension, and undo some effects encountered in higher dimensions.

Theoretical guarantees cannot always be relied upon, unless you are willing to settle for probabilistic guarantees. With deduction you are guaranteed that the conclusion is correct. With induction, things are not as clear, as historically speaking it was not a concretely guaranteed, but with recent advancements we can make a probabilistic guarantee. However, this guarantee has limitations in choosing for example a hypothesis space, since the bounds will be very loose. Asymptotic guarantees work on the condition that given infinite data they can provide the current classifier. But if we consider the bias-variance trade of, we can see that if learner A is better than learner B with infinite data then Learner B is the better with finite data. So we should not rely on the guarantees for practical decisions, but rather as a guide line for making and understanding a good learner.

Features must be chosen carefully as they are domain specific and they can make or break a project. Features that correlate well with a class are easy to learn from, but in contrast choosing features that are very complex can make learning very hard. Its very likely that the correct features will not be selected in the first try; which is why this is more of an iterative process of running the learner, analyzing the result, and then altering the data and/or the learner. Sometimes, even the best features may not have the desired results. We have two choices, choose a better algorithm or gather more data. Its a lot simpler in this case to gather more data, however as the data increases so does the time needed to process it as does the complexity of the classifier. If a simple learner does not get the desired result then we can use a clever algorithm, which makes the most of the resources and data, to effectively get the desired result; there is a complexity trade-off to this approach. Another approach to the above issue is to use a combination of learners; using a resampling technique, each learner is given training data to which the results are computed in parallel and combined using some sort of a voting technique.

Some common misconception are: simplicity implying accuracy, representation implying learnability, and correlation implying causation. Simplicity implying accuracy is not necessarily true, because the preferences are more accurate not that a hypothesis is simpler. There are several functions that may be represented but cannot be learned from; for example a standard decision tree cannot learn with more leaves if there is not enough training data. Given finite time, memory and data; standard learner can only learn about a subset of the data. Correlation is only based on current data whereas we try to use the causation part to make prediction of future event. Since the event spaces dont necessarily align, we can only use this as a guide line making the prediction rather limited; until the experiment is actually tried out.

3 Report on Paper 3

[3]

Data science has to do with focus on involving data and the systematic study of an entity. The data is increasingly heterogeneous and unstructured; the majority of this data is consumed by computers for automatic decision making. Big data in tandem with state-of-the-art analytics over the years had allowed for better knowledge creation and hence allowed decision making to be much more scalable. The traditional database model allows for summarization of data given a query; however for knowledge discovery we are interested in detecting interesting and robust patterns on data, not asking what data specifies a predefined pattern. We define interesting to be something unexpected and actionable, and robust to be a pattern likely to occur in the future. So, for a learned model to be considered good it must be predictive; predictive accuracy favours simpler and sparser models as they tend to be more robust of future data.

Since storage has become so cheap recently, we are inclined to keep a lot more data; large amount of data make decision making more practical. Ever since relational database technology has matured and business processes have begun to be automatized, data mining has increased and has become more used. This has allowed software tools to leverage transaction and behaviour data for explanation and prediction. In order for learning to work, methods should be able to detect subtle structures in data without having to make any strong assumptions of the distribution; the downside about this is that they will pick up noise in data and have no way to reduce it from what is useful data. With such large multi-dimensional data it becomes very hard for us to know a priori query and if its even useful, which is why we must rely on a learner to learn about patterns and then ask for a predictive query on the basis of the learned result; this is why the above method is so useful. Again, it is important to choose features very carefully, so the learner can discover patterns easily; this mean choosing simple features rather than very complex ones. With large data, the computer plays a key role in building the model through a generate and test processes. We also need to have context behind the data in order to entertain the causation of patterns to predictions.

Machine learning skills become necessary for data scientists, because they have to deal with plenty of data in order for a learner to be able to predict accurate data. Text processing and text mining is also important in various fields such as social networks, as this allows us to materialize more forms of data from non-relational sources. Knowledge of markup languages like xml and its applications are also essential, as this allows the computer to automatically interpret and easily learn knowledge about such data. Knowledge in context of machine learning can fall into 3 main classes: statistics (especially Bayesian statistics), computer science, correlation and causation. For statistics we need to know about probability distribution, hypothesis testing, and multivariate analysis for economics. Computer science deals with how the data is internally handled and manipulated by computers and can be done with courses such as data structure, algorithms and systems including distributed computing, databases parallel computing and fault tolerant computing. For small sized data, the scripting language like Python and Perl is required whereas for very large data sets the knowledge of RDBMS is necessary. The data scientist should know where to apply correlation and causality.

We are in a fast decision making era based on large data for many types of problems such as finance, online advertising, air traffic control, routing of package delivery and many types of planning tasks. Computers are better decision makers than humans in terms of cost, accuracy, and scalability. According to scientists, the models are based on human behaviour and let us say that they are 95% accurate but the rest of the 5% can affect the result in a positive or negative manner.

Common causes of errors in prediction are known to arise from three sources: misspecification of a model, small sample size, randomness. Misspecification of a model has to do with applying an incorrect model, for example if we apply linear model for non-linear phenomenon, it may generate error because linear model imposes a wrong bias on the problem as mentioned previously. Small sample sizes used for estimation will give a greater amount of bias in the models estimation as there is not enough data to get a meaningful result. Randomness in a sample can also contribute to a high variance as mentioned previously. However big data helps us to reduce the first two error as we can test the model properly, making less assumptions. But it restrict the theoretical limitations because big data may have been used in different conditions, as the conditions may never always be the same. It is like in health science treating few patients under same natural conditions, which we do not know may be different on other times of the same treatment. The power of machine learning is on the rise due to the availability of high quality human made data in Wikipedia.

4 Connections between the three papers

As was seen by the first paper, business intelligence consists of several components. The data source, Data movement, streaming engines, data ware house servers, mid-tier servers, and front end applications. The data mining and text analytics part of the mid-tier rely heavily on machine learning. In businesses intelligence, we also need to analyze the data for decision making a predictions this has to do with the third paper. Machine learning is talked about in the second paper and consists mainly of representation, evaluation, and optimization; with machine learn generalization is a key concern. These papers go hand in hand to archive what we need for business intelligence. Occams razor is another common topic among the papers; it states that entities should not be multiplied beyond necessity. In machine learning, this is often taken to mean that, given two classifiers with the same training error, the simpler of the two will likely have the lowest test error.

References

[1] Surajit Chaudhuri, Umeshwar Dayal, Vivek R. Narasayya. An Overview of Business Intelligence Technology. *Communications of the ACM*, 2011, 54(8):88-98.

[2] Pedro Domingos. A Few Useful Things to Know About Machine Learning. *Communications of the ACM*, 2012, 55(10):78-87.

[3] Vasant Dhar. Data Science and Prediction. *Communications of the ACM*, 2013, 56(12):64-73.