2016 IEEE 16th International Conference on Data Mining Workshops

Domain Driven Forecasting: Applying Theory Into Practice

Praveen Venkateswaran*, Vikrant Shimpi†, Maitreya Natu†, Vaishali Sadaphal†
*BITS-Pilani Hyderabad Campus
{f2011640}@hyderabad.bits-pilani.ac.in
†Tata Research Development and Design Centre
{vikrant.shimpi, maitreya.natu, vaishali.sadaphal}@tcs.com

***Abstract*—In this paper, we present our experiences in applying state-of-the-art forecasting solutions to meet the forecasting needs of various business domains. We present four real-world case-studies varying in the business objectives, forecasting needs, and domain properties. Each case-study presented unique challenges in translating theory into practice and translating forecasting observations into domain-specific recommendations. We summarize the lessons learnt while deploying across various case-studies, and demonstrate how the state-of-the-art solutions coupled with industry best practices can deliver powerful solutions to meet forecasting needs of any business domain.**

I. INTRODUCTION

The ability to forecast provides powerful capabilities to better understand and manage an enterprise. Business needs to understand the likely future behavior of the system for activities such as inventory planning, product strategy planning, creating sale pipelines, and developing marketing plans. With the wide adoption of digitization, a large volume of data gets collected, capturing various aspects of business and operations. The increasing availability of such rich data has made it easier to forecast future behavior.

However, the forecasting approaches across domains vary due to the following reasons:

1) *Business objectives*: Different domains use forecasting to meet different objectives. For instance, system architects use capacity forecasts to make plans for capacity augmentation or rationalization. On the other hand, support staff personnel use the forecasts of trouble tickets in the future in order to get more time to take corrective actions. Domains thus vary in the way they use forecasting output ranging from planning inventory to developing marketing plan to planning team size and building competency.
2) *Forecasting needs*: Different domains have different forecasting needs. For instance, some business problems (such as IT capacity planning) demand a coarse-grained forecast but for a longer duration in future. On the contrary, some domains (such as banking and financial organizations) are keen on fine-grained forecasts of near future. Similarly, domains vary in their needs to process peaks and troughs, noise, desired levels of accuracy, treatment of outliers, etc.
3) *Domain properties*: Domains also differ in the data properties observed in the time-series of its various metrics. For instance, supply chain management observes gradual trends reflecting the adoption of new products in the market. In the event of sales and promotions, they observe sudden changes in demands. Retail enterprises observe strong periodic signature with changing seasons. Similarly, domains differ in other time-series properties showing variations in trends, presence/absence of periodic patterns, type of periodic patterns, level of noise, and types of changes.

A large body of literature [5], [9] exists on forecasting algorithms which forecast future data as a function of past data. In business domains, the forecasts of a large number of univariate time-series are required. Under these circumstances, two popular types of forecasting algorithms are regression and smoothing based algorithms or ARIMA-models.

While forecasting as a science is fairly advanced, converting theory into practice is an interesting journey. We present our experience through four such journeys on how to leverage the state-of-the-art forecasting solutions to meet business objectives. While forecasting algorithms stop at accurately predicting the future values, we further demonstrate the process of translating the forecast observations into business-relevant insights and recommendations. While presenting these case-studies, we also highlight situations where we leveraged industry best practices and domain knowledge to obtain effective solutions.

We present the following four real-world case-studies:

- *Multi-tier transactional system*: We present the analysis of a set of servers serving a front-office insurance application. The objective of this case-study is to predict the utilization of compute and storage devices to better plan the system capacity. The predictions are used to derive recommendations for either capacity augmentation or capacity rationalization.
- *Supply chain management*: We next present a case-study of a retail organization that analyzes the demand of different products to predict future demand. The predictions are used to derive recommendations for either decommissioning of products with diminishing demand or stocking-up of products with rising demands.
- *Infrastructure support*: We present a case-study involving the analysis of tickets data resolved by the support teams. Predictions are used for generating notifications for future volume of tickets and the volume of specific types of issues. These insights provide the support teams more time to take corrective actions.
- *Back-office banking operations*: Finally, we present a case-study of a batch system driving the back-office operations of a banking organization. We analyze the execution time of batch jobs to generate notifications of future performance problems. These recommendations are used by the problem management teams to take timely corrective actions to prevent the problem from occurring in the first place.

Each of the above case-studies presents different challenges with respect to the availability and quality of data, the business challenges, and the domain properties. As a result, each demands different solutions to derive effective forecasts. Through these case-studies, we present the lessons learnt while deploying and also present various challenges and open issues in this space.

II. RELATED WORK

Given the plethora of available forecasting algorithms, many existing works aim to answer the question of when each algorithm should be used. Authors in [12] determined the accuracy of various

2375-9259/16 $31.00 © 2016 IEEE
DOI 10.1109/ICMDW.2016.148

forecasting methods in order to select the most appropriate one. Authors in [11] presented the first graphical classification for exponential smoothing methods. Authors in [1] evaluated several forecasting methods across several demand patterns such as constant, linear, trend, etc. The findings indicated that no single model is consistently better than the others, and that their performance depended on various factors such as demand patterns, randomness, etc.

In more recent work, authors in [7], [10] have discussed specific situations where a particular set of algorithms would perform better. While choosing the forecasting algorithm, we adhere to the overall consensus that when there are patterns present in the time-series, using ARIMA models [3] would provide good results and when the time-series consists of only trends without the presence of patterns, a regression-model [6] would be the choice of algorithm.

III. Case Study 1: Multi-Tier Transactional System

We first present a case-study of the transactional system of an insurance company in the US. The system consisted of an array of web, application, and database tiers hosted on a set of Linux servers.

A. Objective

The organization was planning an infrastructure upgrade and hence the objective of the case-study was to analyze the current and future capacity requirements and generate recommendations for capacity augmentation or capacity rationalization.

We analyzed the history of the performance of 17 servers. Specifically, we analyzed time-series data of CPU utilization, IOPs, and memory utilization of these 17 servers. The forecasting was performed on 51 time-series (3 time-series of each of the 17 servers). The data was collected for a duration of 3 months at an hourly granularity. Figures 1(a), (b) and (c), show the time-series of the CPU utilization, IOPs, and memory utilization respectively of one of the servers.

B. Domain properties

We observed the following properties in the domain which in-turn also reflect in the time-series of various metrics:

- *Patterns:*
  1) *Effect of business hours:* The system was serving an internal application that was operational mostly during the business hours. As a result, the application observed high workload during the business hours and very low workload otherwise. This property reflects in the form of temporal patterns in the underlying servers as well. Figure 1(d) shows a representative slice of 2 days of the time-series of the CPU utilization which shows high usage during business hours (8 AM to 8 PM) and shows low usage during non-business hours (8 PM to 8 AM). Similar periodic patterns are observed for the IOPs and memory utilization as well as shown in Figures 1(e) and (f).
  2) *Distinct periods:* The application had dedicated servers which were not shared across different applications. As a result, the patterns observed in the time-series of CPU, memory, and IOPs observe clean distinct patterns. The memory utilization generally shows more variation than both IOPs and CPU utilization.
- *Value ranges:* Both CPU and memory utilization have a limited range of possible values ranging from 0 to 100. IOPs values have no such limit.
- *Noise:* Being an internal application with consistent homogeneous workload, it observed minimal variations. The workload is well-defined with finite users and a finite set of operations. As a result, the time-series of the metrics observed very less noise.
- *Trends:* Some of these time-series displayed the presence of trends due to gradual changes in the business workload.

C. Forecasting approach

The presence of periodic patterns along with the lack of noise implies that the prediction of these time-series can be done very effectively by deriving ARIMA models. However, the effectiveness of these models depends upon the use of the right data and the right parameters. Below we discuss our approach for forecasting these time-series:

1) *Period*: ARIMA models require defining the period of the time-series. A common approach to detect the periods is by using a periodogram. Figures 1(g), (h) and (i) show the periodogram of the time-series of the three metrics. The x-axis represents the period value while the y-axis represents the power of the period. The usual approach is to identify the highest peak as the most dominant period. As shown in the figures, we obtain the top 2 peak values for each metric. However, sometimes there are multiple periods present in the time-series. For instance, in Figure 1(b), we see that the most dominant period is 13 hours followed by 90 hours. The 90 hour period reflects the weekly pattern which is dominated by the daily pattern of 13 hours. Hence, it is necessary to identify the longest relevant period.

At times, the period values obtained from the periodogram need correction to reflect the patterns generally observed in this domain. In transactional systems, the patterns follow certain fixed temporal dimensions such as Half-daily (12 Hours), Daily (24 Hours), Half-weekly (84 Hours), Weekly (168 Hours), etc. We correct the observed peaks (13 and 90 hours) of the CPU utilization (Figure 1(g)) to the corresponding domain specific periodic patterns (Half-Daily and Half-Weekly).

2) *ARIMA forecasting*: We then build the ARIMA model using the derived period in order to obtain the forecast values. Figures 1(j), (k), (l) show the forecast results of the three time-series for the next 3 weeks.

D. Accuracy of forecast

We compute the accuracy of the point forecasts in all the case-studies using the Mean Absolute Percentage Error (MAPE) computed as $M = \frac{1}{n}\sum_{t=1}^{n}|\frac{A_t - F_t}{A_t}|$. Figure 2(a) shows the observed distribution of accuracy for the CPU utilization forecast for all 17 servers. We see that 12 servers show an accuracy $>98\%$. This high accuracy is obtained due to the presence of clear periodic patterns.

E. Using forecasts to plan future infrastructure capacity

Figure 3(a) shows the heatmap of the CPU capacity utilization of 17 servers for both past observed values as well as future forecast values. The thresholds for low, medium and high utilizations were derived from industry standards. On the basis of the heatmap we provide two types of recommendations:

1) Capacity augmentation: We recommended capacity augmentation for the following servers in order for them to be able to handle the increase in CPU capacity utilization.

- We observed 3 servers (Servers 6, 7, and 8) with significant increasing trend.
- We observed 2 servers (Servers 9 and 12) with consistently high usage.
- There were 3 servers (Servers 2, 3, and 5) with a sharp change to high capacity.
- Figure 3(b) shows the example of the CPU utilization of a server that observed consistently high usage.

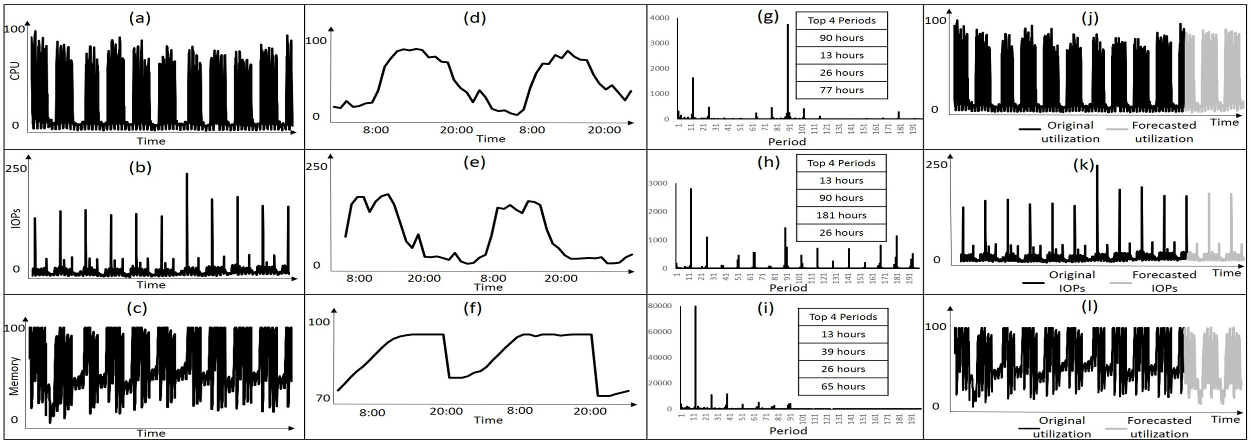

Fig. 1. Example of forecasting server metrics : (a, b, c) Time-series of CPU utilization, IOPs, Memory utilization, (d, e, f) Representative slice of CPU, IOPs, Memory, (g, h, i) Periodogram of CPU, IOPs, Memory, (j, k, l) Forecast of CPU, IOPs, Memory

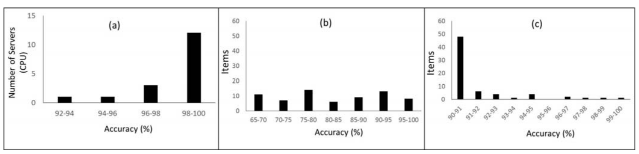

Fig. 2. Forecast accuracy distribution (a) CPU utilization (b) Demand point forecast (c) Demand confidence bands

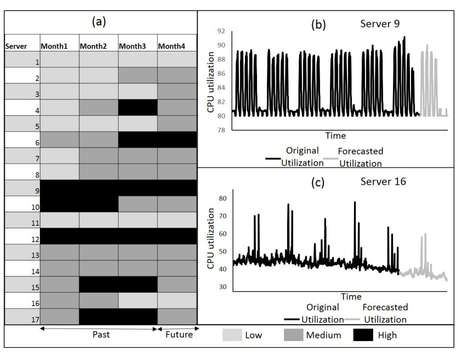

Fig. 3. (a) Heatmap of CPU capacity utilization, (b) Example of continuous high CPU utilization, (c) Example of decreasing CPU utilization

2) Capacity rationalization: We recommended capacity rationalization for the following servers in order to minimize wasted compute resources.

- There were 3 servers (Servers 10, 15, and 17) with significant decreasing trend.
- We observed 2 servers (Servers 1 and 11) with consistently low usage.
- We observed 1 server (Server 16) with a sharp change to low capacity.
- Figure 3(c) shows the example of a server with a decreasing trend in its CPU utilization.

IV. Case Study 2 : Supply Chain Management

We next present a case study from the domain of supply chain management of a retail customer in the US.

A. Objective

The customer was observing significant changes in the demands of various items and hence was struggling to plan the inventory. The objective of the case-study was to predict the future demand of various items. The predicted demand is then used to generate recommendations for either the stocking-up or decommissioning of items. The business objective thus did not require fine-grained forecasts but a broader range.

We analyzed the time-series data of the number of orders generated for 77 products. The data was collected over a duration of 4 months at a daily granularity. Figure 4(a) shows the time-series of the number of orders generated for one of the products.

B. Domain properties

The properties of the order time-series are observed to be significantly different from the previous case-study. Below are some of the key properties:

- *Patterns*:
  1) *Constant behavior*: The order time-series for some products were observed to be very consistent. This behavior is common in the items that have reached maturity in the market, thus leading to stable demand.
  2) *Absence of periodic patterns*: In most of the order time-series, we did not observe any dominant periodically occurring pattern.
- *Value ranges*: The values were observed to be ranging between 0 to 600 orders.
- *Noise*: The order time-series observed high noise levels showing high variations in the demand. Noise is caused by various factors. The retailers order in bulk to maintain stock, plan with a look-ahead, or negotiate a good deal. As a result, the time-series demonstrate irregular peaks and troughs in the demand values.
- *Trends*: The demand time-series of many items observe strong trends. Increasing trends are observed when the product is promoted by means of advertisements or word-of-mouth publicity. This results in a gradual increase in the demand of the product. Similarly, many products observe a gradual decrease in demand. This decreasing trend is generally due to the emergence of a competitor in the market.
- *Changes*: Abrupt changes are observed in the demand patterns. These changes are associated with a rapid change in demand

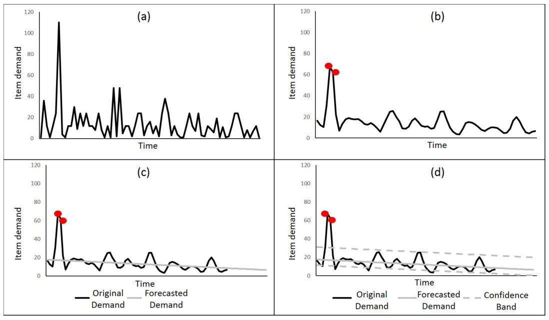

Fig. 4. (a) Example of time-series exhibiting noise, (b) Identifying outliers after smoothing, (c) Forecast using regression model, (d) Confidence bands

due to sales and discounts. Sudden increases are observed when the current product is on sale, while sudden decreases can be due to discounts on rival products.

C. Forecasting approach

Owing to these properties, the time-series of orders observe strong trends and high levels of noise. Unlike the previous case-study, ARIMA-models are not a good fit due to the lack of patterns. Instead, we propose to use regression-based solutions to predict future demand values. In addition, given the high levels of noise, forecasting precise values of future demand would be inaccurate. Instead, we propose to forecast a band of likely values. Below we discuss our approach for forecasting these time-series.

1) *Smoothing*: In order to minimize the effect of noise and better capture the trend of the demand, we smoothen the time-series by using a moving average window. Figure 4(b) shows the time-series after smoothing.

2) *Outlier removal*: Regression-based solutions are highly sensitive to outliers. The outliers in Figure 4(b) are denoted by dots. In the presence of outliers, the regression models fail to capture the trend correctly. We remove the outliers before computing the trend.

3) *Regression model*: We then fit regression models [6] to compute the relationship between item demand and time. We then extrapolate these trends to forecast likely future values. Figure 4(c) shows the derived regression model as well as the forecast demand values.

4) *Confidence bands*: In order to cater to the high noise volumes, we compute confidence bands along with the prediction. Some common approaches are to use the model fit error or to compute a percentage confidence interval. However, generating bands using the model fit error and the point-wise confidence results in symmetric bands which might not be accurate if there is a significant difference in the noise above and below the best-fit line. For instance, in Figure 4(c), the amount of variation is larger in the values above the regression line as compared to the values below the line. In this case, the lower band should be narrower than the upper band.

We compute the confidence bands as a function of the level of noise above and below the best-fit line. We first compute the time-series residuals. We then check whether it resembles noise using the Ljung-Box test [8]. We analyzed 77 demand time-series and were able to detect significant levels of noise in 72 of them. The remaining time-series had constant demand values. Figure 4(d) shows the demand prediction for the item. We see that there are wider bands above the best-fit line and narrower bands below.

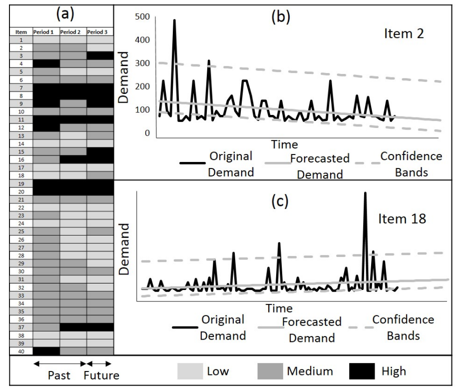

Fig. 5. (a) Heatmap of item demands, (b) Example of decrease trend, (c) Example of increasing trend

D. Accuracy of forecast

We first identified the point forecast accuracy for the items as shown in Figure 2(b). We see that there are 40 items that have an accuracy $<85\%$. The overall accuracy values range widely from 65% to 98%. This is primarily due to the noisy nature of the order time-series.

We then identified the accuracy using the confidence bands. The definition of accuracy used was the proportion of actual test values that were contained within the confidence bands. Figure 2(c) shows the distribution of accuracy. We see that the order time-series forecasts of 48 items display an accuracy of approximately 90%. There are 6 items that have an accuracy $>95\%$. The average width of the confidence bands was not large and observed to be 15% above or below the best-fit line.

E. Using demand forecasts for inventory planning

We used the demand forecasts in order to generate recommendations for inventory planning. Figure 5(a) shows the heatmap of the demands for 40 items for both past as well as future demands. On the basis of the heatmap we were able to provide the following recommendations:

- We recommended potential stocking up of inventory for 9 items (e.g. Items 16, 18, 37) due to strong increasing trends. Figure 5(c) shows an example of an item that exhibits increasing trend in its demand.
- We recommended the potential decommissioning of 12 items (e.g. Items 2, 5, 28) due to the presence of decreasing trends. Figure 5(b) shows an example of an item that exhibits decreasing trend in its demand.
- There were 19 items (e.g. Items 1, 10, 14) that exhibited constant trends leading to recommendations to retain the current demand policy.

V. Case Study 3: Infrastructure Support

We next present a case-study from the domain of IT infrastructure support (ITIS). ITIS is responsible for effective management and maintenance of IT infrastructure resources such as hardware, system software, and business application programs. Whenever any resource observes errors, faults, difficulties or special situations, a ticket is created in a ticketing system to bring it to the attention of experts in the ITIS function. Tickets are automatically created by monitoring tools that continuously monitor systems for anomalies. Alternatively,

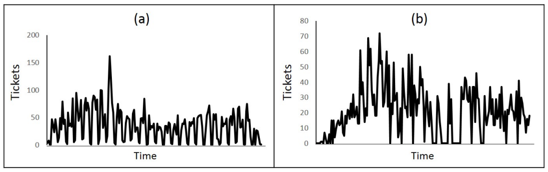

Fig. 6. Example of time-series with (a) Weekday/Weekend pattern, (b) Increasing trend followed by decreasing trend

users also generate tickets when observing system problems. This ticket is assigned to a resolver who obtains more information about the problem and then fixes it.

A. Objective

An incoming ticket needs to be resolved within a frame of time defined as a part of the Service Level Agreement (SLA). Resolution time for high severity alerts is usually in the order of a few minutes or hours. Consequently, resolvers do not get enough time to act on the problem resolution. The objective of this case study was to use the past history of tickets in order to predict future issues. By predicting future problems, sufficient time can be given to the resolvers to either (a) prevent the problem from occurring or (b) identify appropriate resolution.

We analyzed the time-series of number of tickets generated for 43 distinct types of issues. The data was collected for a duration of 6 months at a daily granularity. Figure 6(a) shows the time-series of the number of orders generated for one of the products.

B. Domain properties

The time-series of number of tickets demonstrated the following properties specific to the domain of ITIS:

- *Patterns*:
  1) *Weekday-Weekend pattern*: The tickets for various issues generally observed a weekly pattern in the volume of tickets, observing higher volumes over weekdays and lower volumes over weekends. This behavior gets explained by the usage pattern of the application. The application being analyzed was used heavily over the weekdays. Figure 6(a) shows the time-series of the issue referring to high execution time of SQL queries. This time-series shows higher volume of tickets on weekdays varying from 30 to 80 tickets per day and almost zero tickets on weekends.
- *Value ranges*: The values were observed to be ranging between 0 to 500 tickets per day.
- *Noise*: The time-series of tickets observed high noise as the arrival of tickets was dependent on the usage of the application, which is usually non-uniform and is driven by business needs.
- *Trends*: Many time-series observed dominant increasing or decreasing trends owing to different phases of application instability and hardening.
- *Changes*: In many time-series, we observed a change in trend. Every new application goes through a process of hardening. In the initial phases, we observed increasing trend in majority of the tickets, owing to various deployment issues and instability in the application. Over time, as the application stabilized, we observed a decreasing trend, eventually resulting in close to zero tickets. Consider the time-series of a performance issue observed at an application as shown in Figure 6(b). This issue observed an increasing trend after the deployment of a patch. The ticket volume of the issue starts observing a decreasing trend after multiple optimizations by the problem management teams.

C. Forecasting approach

Given the presence of strong periodic patterns, we used an ARIMA-based forecasting approach. However, given the fact that many time-series' observe a change in trend, it is necessary to first detect these changes [2] and then select the right history to predict the future. Failing to detect these change-points can significantly mislead the forecasts. We propose the following approach to predict the time-series of future ticket volume of different issues:

1) *Detect change in trend*: We first detect the presence of any change in trend in the time-series. Figure 7(a) shows a change-point wherein there was an increasing trend before the change-point and a decreasing trend after it. We identify the most recent steady state by identifying the last change-point.

2) *Periodogram*: The presence of periodic patterns calls for the usage of ARIMA models. As stated in the first case study, these models are dependent on the period of the time-series.

We use periodogram to detect the length of the periods. Figure 7(b) shows the periodogram of an issue where the x-axis shows the period length in days. We observed three dominant periods of 5, 7, and 2 days in decreasing order of strength. These periods indicate the presence of a weekly pattern where the periods of 7, 5, and 2 indicate day-of-week, weekdays, and weekends respectively.

3) *ARIMA forecasting*: Since the dominant periods all refer to the presence of a weekly pattern, we set a weekly period to the time-series and forecast the number of tickets of the issue using an ARIMA model. Figure 7(c) shows the forecasted values.

4) *Confidence bands*: The presence of noise implies that point forecasts would be inaccurate and hence confidence bands are required. However, unlike the previous case-study where we could derive the bands using the overall amount of noise, the presence of patterns calls for a different method. The level of noise varies across the different patterns. For instance, in Figure 7(a), the weekend values are consistent and show only a small amount of variation. On the other hand, weekday values observe a high variation. As a result, we compute different confidence bands for each pattern. Figure 7(d) shows the forecast result along with the confidence bands. From the figure, it can be seen that weekends have narrow bands while weekdays have broader bands.

D. Accuracy of forecast

We analyzed 43 unique issues of which 4 issues had extremely high noise and lack of any trend or pattern and hence were not predicted. The distribution of the point forecast accuracy is shown in Figure 8(a). We observe that 30 issues had an accuracy $>85\%$ with the overall range being from 65% to 97%. This large range was due to the presence of noisy time-series that could not be predicted accurately using point forecasts. We then identified the accuracy using confidence bands as in the previous case-study. Figure 8(b) shows the distribution of accuracy. It can be seen that 24 issues could be predicted with an accuracy $>95\%$. The overall accuracy range was from 90% to 100%.

E. Using ticket forecasts for timely resolution

Figure 9(a) shows the heatmap of the health of the issues for both past observed tickets as well as future forecasted tickets. Using the heatmap we were able to provide the following recommendations:

- 12 issues (e.g. Issues 3, 21, 28) observed increasing trend and are likely to appear in higher volumes. These issues primarily

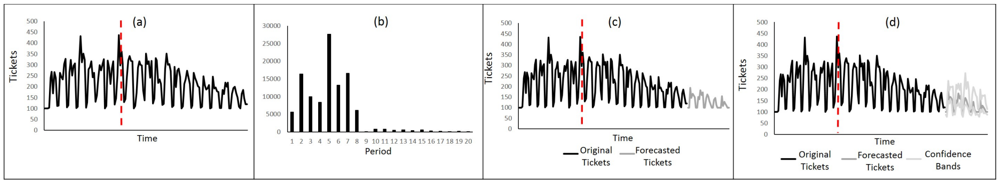

Fig. 7. (a) Time-series showing change in trend, (b) Periodogram with weekly pattern, (c) Point forecast (d) Confidence bands

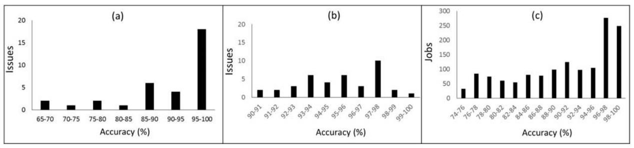

Fig. 8. Accuracy distribution of forecasts for (a) Ticket point forecast, (b) Ticket confidence bands, (c) Job runtime

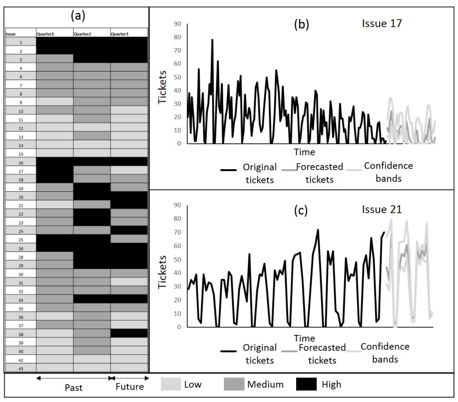

Fig. 9. (a) Heatmap of ticket count (b) Example of decreasing trend (c) Example of increasing trend

referred to the SQL performance issues. Proactive measures could be taken to fix these issues and prevent their increased volumes in the future. Alternatively measures can be taken to ensure timely resolution when they occur (Figure 9(c)).

- 13 issues (e.g. Issues 17, 25, 31) observed decreasing trend. These issues primarily referred to infrastructure issues such as file system getting full, or high CPU usage and were likely to further diminish (Figure 9(b)).

VI. Case Study 4: Back-Office Banking Operations

We present the case-study of the batch system of a banking firm in the US. We present analysis of one business unit that was undergoing expansion resulting in an increase in the number of transactions being processed by the jobs. An increased workload manifests in different ways in the performance of different jobs.

A. Objective

The objective of the case-study was to predict future behavior of the batch system and generate predictive-preventive alerts, and thus enable the batch administrators to eliminate the problem before it occurred.

We analyzed the time-series of the runtime (time to complete execution) of 1410 batch jobs. The data was collected for a duration of 6 months at an hourly granularity. Figure 10(a) shows the time-series of the runtime of one of the batch jobs.

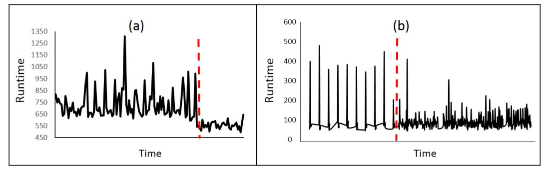

Fig. 10. Example of change in (a) mean, (b) period - increase in frequency

B. Domain properties

The time-series of the runtime of the batch jobs demonstrated properties that are very specific to batch systems. Below we discuss these properties:

- Patterns:
  1) *Well-defined execution patterns*: Batch jobs have a fixed execution sequence. Most jobs run once a day or have a cyclic pattern of executing every few hours.
  2) *Constant value*: Some jobs execute for a constant time and are unaffected by changing workload. As a result, the time-series of these jobs are constant.
  3) *Day of week patterns*: Some jobs behave differently across different days of week or days of month. For instance, there are maintenance jobs that perform complete backups on Mondays, and incremental backups on rest of the days of the week. Such jobs show high peaks on Monday and lower values on rest of weekdays.
- *Value ranges*: The runtime values were observed to be ranging between 1 to 1500 minutes.
- *Noise*: Given the off-line nature of workload and fixed execution patterns, the execution time of batch jobs do not observe high noise levels and demonstrate predictable runtimes.
- *Changes*:
  1) *Significant changes in mean*: Batch jobs are highly interdependent on one-another and as a result, any change in business or infrastructure reflects across many jobs. The time-series of batch jobs thus observe significant persistent changes. Figure 10(a) shows the effect of infrastructure upgrade in the form of reduction in the mean of the job runtimes.
  2) *Change in period due to business changes*: An interesting change that is quite often observed in batch systems is the change in periodicity. Execution of batch jobs is governed by a certain schedule. Any change in this schedule leads to a change in the periodic pattern. For instance, Figure 11(a) shows the example of a job that ran full health-checks every Sunday and partial health-checks every Tuesday. As a result, the time-series shows 2 weekly patterns. The full health-check resulted in high peaks on Sundays and the partial health-check resulted in smaller peaks on Tuesdays. Over time, due to a change

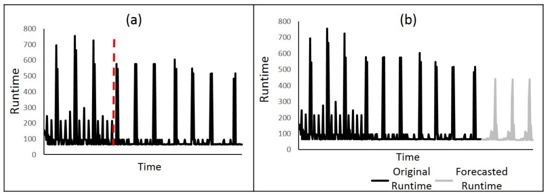

Fig. 11. (a) Example of change in period - multiple to single period (b) Forecasted runtime

in policy, the health-checks were done only on Sunday. This change is reflected in the time-series as a single weekly pattern causing high peaks only on Sundays. Consider another example shown in the Figure 10(b). The batch job in this figure collected performance measurements over predefined periodic interval of 6 hours. After a few months, this configuration was changed as shown by the dotted line, to collect the measurements every hour. This change reflects as a change in periodic pattern from 6 hours to 24 hours.

C. Forecasting approach

The batch jobs exhibit some very specific properties such as strong periods, low noise, and significant-persistent changes. We hence derive ARIMA models in order to generate the forecasts.

1) *Outliers*: Each observed temporal pattern of a job displays a normal range of values. Hence, the removal of outliers requires the identification of those points that display a significant deviation from the normal behavior of the temporal pattern. We used a local outlier definition by identifying the different temporal dimensions and defining each of their normal behaviors.

2) *Changes*: Batch jobs usually demonstrate two types of changes - changes in mean/standard deviation and changes in periodicity. We were interested in identifying the changes that were both significant and persistent. We detected these changes to arrive at the most recent steady state to use for forecasting.

3) *Period*: After extracting the latest steady state, we compute the periodic pattern exhibited by the job. The business domain of the batch systems ensures that there are fixed combinations of possible temporal dimensions. Jobs display patterns conforming to hour-of-day, day-of-week, day-of-month, week-of-month, etc. Instead of detecting periods using periodogram, using the domain knowledge leads to better derivation of periods. We first use Classification and Regression Trees (CARTs)[4] to find influential temporal dimensions. We then profile these temporal dimensions to identify the normal behavior of each dimension and then forecast the runtimes of the job.

4) *ARIMA forecasting*: We then build the ARIMA model using the derived period in order to obtain the forecast values. Figure 11(b) shows the forecast results of Figure 11(a).

D. Accuracy of forecast

We analyzed one business unit consisting of 1410 jobs. Figure 8(c) shows the accuracy distribution of the jobs. We were able to forecast the time-series of 1024 jobs with an accuracy >86%. This accuracy was due to the presence of strong patterns and low noise. The accuracies are also attributed to the correct detection of changes. We observed changes in 703 time-series. The jobs with lower forecast accuracy had high levels of noise, undetected changes, and unexpected behavior in the form of outliers in the test set.

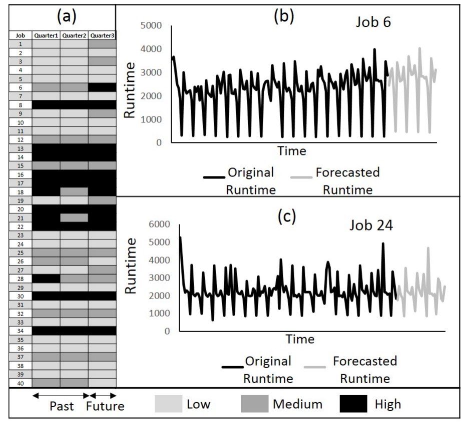

Fig. 12. (a) Heatmap of job runtimes, (b) Example of increasing trend, (c) Example of constant trend

E. Using forecasts to generate preventive alerts

Figure 12 shows the heatmap of the alert counts for 40 jobs for both the past as well as the forecasted duration. Using the heatmap we were able to generate the following recommendations:

- Alerts were generated for 20 jobs (e.g. Jobs 6, 9, 27) showing increasing trends in runtime. They need further analysis to prevent any potential disruption. Figure 12(b) shows an example of a job with an increasing trend.
- Alerts were generated for 6 jobs (e.g. Jobs 26, 28, 40) showing decreasing trends in runtime. Abnormally low runtimes could be an indication of the job terminating with exceptions if not receiving the right data.
- 14 jobs (e.g. Jobs 2, 24, 35) are healthy and observe constant behavior. We do not expect any potential problems with these jobs. Figure 12(c) shows an example of a job which has no significant trend.

VII. Lessons Learnt

In this section we describe some of the lessons learnt while forecasting various metrics across different domains that were described in the case-studies.

A. Data Preparation

1) Missing data handling: Different domains have different methods to capture data. For instance, in domains where the data is system-generated, it is captured at a fixed granularity and is uniform. The data can generally be used as-is for forecasting. In system-generated data, it is easy to detect the presence of missing data since the capturing granularity is fixed and known beforehand. Both case-study 1 and 4 collected data using monitoring tools and there was no missing data observed.

However, in domains where the generation of data is ad-hoc and dependent on the environment, it is non-uniform and cannot be directly used for forecasting. The data needs to be made uniform by either inserting the missing timestamps and/or aggregating at a larger granularity. When the data is environment-dependent, we cannot differentiate missing data from the absence of data. Case-studies 2 and 3 had holes because the nature of data was itself ad-hoc. Hence, the data in both these case studies does not follow a consistent structure with respect to time granularity. We aggregated the time-series in both these cases at a daily level to make it uniform.

There were no hourly patterns present and hence the daily aggregation ensured that no patterns were suppressed.

2) Smoothing: Smoothing of a time-series is needed mainly when noise is present. Smoothing the time-series was not necessary in case-studies 1 and 4. However, for case-studies 2 and 3, we had to implement different smoothing approaches. In case-study 2, due to the high levels of noise, we used an aggressive smoothing approach in order to minimize the effect of noise on the regression model. This approach could not be used in case-study 3 due to the presence of patterns. We, instead, used a more relaxed smoothing approach in order to ensure that the observed patterns were retained even after smoothing.

3) Outliers: In case-study 2, we used a global definition of outliers across the entire time-series. However, we could not use a global definition in the other case-studies due to the presence of patterns. This is because, the occurrences of large values may themselves define a periodic pattern and hence would not constitute outliers. For instance, consider the following scenario - the time-series exhibits larger values on every Monday as compared to the other days of the week. Using a global definition would result in all the data points on Mondays being classified as outliers which would be incorrect. We instead employed a local definition in case-studies 1, 3 and 4 by identifying the different temporal dimensions and defining their normal behavior. We then identified outliers within each dimension.

B. Dealing with Changes

One of the most common changes that we observed across all four case-studies is the change in *mean*. This behavior manifests prominently in IT infrastructure case-studies (case-studies 1 and 4). For instance, in case-study 1, we observed scenarios of infrastructure upgrade that results in a change in the mean of the time-series of CPU utilization.

Another change that significantly impacts prediction is the change in *trend*. These changes are commonly evident in the infrastructure support case-study (case-study 3). An application undergoes a life-cycle of gradual increase in number of issues observed in the initial phases of deployment, and then eventually a gradual decrease in number of issues as it stabilizes. This behavior manifests in the time-series of daily tickets as a change in trend from increasing to decreasing.

An interesting change that we observed specifically in the case-study of back-office operations (case-study 4) is the change in *periodicity*. For instance, a change in execution schedule of a batch job can lead to a change in the periodic patterns from a 12 hour period to a 1 hour period.

The definition of the significance and persistence of the change are dependent on the nature of the time-series as well as the domain. For instance, a 20% increase in orders over a week would constitute a significant and persistent change in the supply chain management domain (case-study 2). On the other hand, in the IT infrastructure domain (case-study 1 and 4), an increase in CPU utilization over a value of 80% and persisting over a month would reflect a significant and persistent change.

C. Detecting periods

Forecasting algorithms are dependent on the defined period of the time-series. We utilized periodograms to identify the period in case-study 1 and 3. However, the periods identified using the periodogram often needed small adjustments in order to reflect the right period as described in case-study 1. Also, in cases where multiple periods were observed, the largest peak often referred to a smaller period since its frequency of occurrence was higher than the larger periods. Hence, instead of going by the period with the largest peak, identifying the largest relevant period was more important. For instance, when a time-series has both a Day/Night pattern as well as a Weekday/Weekend pattern, the largest peak of the periodogram would refer to the Day/Night pattern due to its frequency which would result in the missing-out of the other pattern.

In case-study 4, the possible temporal dimensions of the domain were well-defined. Hence, instead of using a periodogram, we used domain knowledge of relevant temporal dimensions. We then identified the most influential dimensions and profiled them in order to derive the period.

D. Confidence bands

Confidence bands are needed when the time-series are noisy and can thus not be predicted accurately using point forecasts. We, hence, introduced confidence bands in case-studies 2 and 3. In case-study 2, there was a need to set different definitions of the width above and below the best-fit line to provide a more accurate range of possible values. The confidence bands were computed as a function of the noise above and below the best-fit line. We could not use the same approach in case-study 3 due to the presence of patterns. The data points belonging to different temporal patterns exhibit different levels of noise. We, hence, derived bands separately for each pattern.

VIII. Conclusion

While forecasting as a science is fairly advanced, converting theory into practice is an interesting journey. We present our experience through four such journeys on how to leverage the state-of-the-art forecasting solutions to meet business objectives. We present case-studies of the deployment in the domains of front-office transactional systems of an insurance company, supply-chain management, IT infrastructure support, and back-office batch system of banking operations. We explain how each case-study varied in the data and domain properties and had different forecasting needs and challenges. While presenting these case-studies, we demonstrate how the state-of-the-art solutions coupled with industry best practices can deliver powerful solutions to meet forecasting needs of any business domain.

References

[1] E. E. Adam. Individual item forecasting model evaluation. *Decision Sciences*, (4):458–470, 1973.

[2] M. Basseville and I. V. Nikiforov. *Detection of Abrupt Changes: Theory and Application*. Prentice Hall, Apr. 1993.

[3] G. Box and G. Jenkins. Time series analysis: Forecasting and control. *Holden-Day*, 1976.

[4] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Wadsworth Intl., Belmont, CA, 1984.

[5] C. Holt. Forecasting trends and seasonal by exponentially weighted averages. *International Journal of Forecasting*, 20(1):5–10, March. 2004.

[6] R. J. Hyndman and G. Athanasopoulos. *Forecasting: principles and practice*. OTexts, 2016.

[7] R. J. Hyndman and Y. Khandakar. Automatic time series forecasting: The forecast package for r. *Journal of Statistical Software*, 27(3), 2008.

[8] G. Ljung and G. Box. On a measure of lack of fit in time series models. *Biometrika*, 65(2):297–303, 1978.

[9] E. McKenzie. General exponential smoothing and the equivalent arma process. *Journal of Forecasting*, (3):333–344, 1984.

[10] R. Nau. Statistical forecasting: notes on regression and time series analysis,http://people.duke.edu/ rnau/whatuse.htm.

[11] C. C. Pegels. Exponential forecasting: Some new variations. *Management Science*, (15):311–315, 1969.

[12] C. Shah. Model selection in univariate time series forecasting using discriminant analysis. *International Journal of Forecasting*, (13):489–500, 1997.