Prediction of Call Arrival Times and Rates

Ateeb Ahmed, Bilal Hoda, Emad Bin Abid, Syeda Saleha Raza, Zeehasham Rasheed
Department of Computer Science
Habib University
Karachi, Pakistan

Abstract

In this paper, we plan to solve a pre-requisite problem for scheduling human resources in a call center industry. In a traditional call center, under-staffing leads to effecting services whereas over-staffing generates a high financial overhead. This problem can be solved/minimized with efficient prediction mechanisms which can be used to estimate call arrival times and rates. Precisely, in this project, we plan to solve the problems of arrival time for next call, determining the category of predicted call and predicting the count of calls expected to come in a given time-interval.
Keywords: call center, arrival time, category, count of calls.

Introduction

In an industry which requires precise allocation of resources, scheduling mechanisms play an integral role. Accurate scheduling offers benefits and satisfaction to businesses as the business leaders can safely control their service quality as well as the finances. Examples of such businesses include food delivery (such as Food Panda, Eat Mubarak, Home Chef, etc.), ride-hailing services (such as Careem, Uber, Bykea, Siayara, etc.), call centers, etc. All these industries can revolutionize customer experiences if provided with proper analytics which in turn allow them to perform scheduling operations effectively and precisely.

In this project, we plan to solve a pre-requisite problem for scheduling human resources in a call center industry. In a traditional call center, under-staffing leads to effecting services whereas over-staffing generates a high financial overhead. This problem can be solved/minimized with efficient prediction mechanisms which can be used to estimate call arrival times and rates.

We divide our problem set into three separate modules, stated below:

Module 1
When can we expect the next call to come in?

Module 2
Amongst well-defined possible categories, which type of call will it be?

Module 3
Within a defined interval, how many calls can we expect from a particular category of callers?

Literature Review

As discussed above, the project is divided into three modules in total; Call Time Predictor, Call Category Predictor and Call Count Predictor. Considering the fact that the nature of problems addressed in all three modules is different, the process of literature review for them is separate and independent.

Call Time Predictor

For the given problem of predicting the time for next call, there is no vast literature available to solve the problem. The work which we do adds to the novelty of this research problem.

A Poisson Process is a model for a series of discrete event where the average time between events is known, but the exact timing of events is random. The arrival of an event is independent of the event before. For example, suppose we own a website which our content delivery network (CDN) tells us goes down on average once per 60 days, but one failure doesn't affect the probability of the next. All we know is the average time between failures.

Poisson process modeling is one of the potential ways to explore in the domain call arrival process. [1] discuss a similar kind of an approach but based on the call arrival count rather than the call arrival rate itself. Another proposed approach for predicting the starting time of the next unobserved activity is to model activities occurring at a variable rate using a Log-Gaussian Cox Process (LGCP) and learn the rate function from the training data. Then the starting time is predicted using importance sampling algorithm [14]. [8] redefines the problem at a personalised level for intelligent phones instead of call centers and proposes probabalistic models based on the parameters of *caller's behavior* and *reciprocity*.

Soon after the recent popularity of Long Short-Term Memory networks, the domain of time-series prediction captured the highlights of its efficiency and started proposing time-series models based on LSTMs. In a recent paper published by Uber, they used LSTM network to solve the problem of predicting the time for next ride. With the use of LSTM model, they minimised the risk of the contribution of exogenous variables while training. Earlier, instead of using bayesian deep networks with LSTM cells, Uber used probabalistic time-series models which had its own demerits especially in cases of frequent training and eliminating exogenous variables [15].

Holt-Winters method introduces the concept of seasonality and trends in the time-series models to map the data according to its underlying trends. [10] discusses the use of Holt-Winters method as a call prediction model for frequent and periodic callers. Some papers discuss about solving a similar kind of a problem by using queuing theory. In order to apply queuing theory on such problems, it is necessary to decompose the problem into three fundamental concepts: arrival, customer patience and service duration [13].

Call Count Predictor

For the given problem of predicting the time for next call, there is significant literature available to solve the problem. The researchers in the domain have vastly contributed to solving the problem.

Bayesian forecasting of inhomogeneous poisson process contributes to solving the problem of call count with in a given interval of time using correlation and autoregressive techniques. [1]. In the presence of intraday and interday dependence in call arrival rates, standard time series models may be applied for forecasting call arrivals, for example autoregressive integrated moving average (ARIMA) models and exponential smoothing [2]. A doubly stochastic Poisson process can be viewed as a two-step randomization: a stochastic process (for the arrival rate) is used to generate another stochastic process (for the call arrival process) by representing its intensity [2]. [4] discusses Point Forecast and Distributional Forecast approaches of predicting the call count with in a given time interval. Moreover a bayesian gaussian model, multiplicative model and additive model can be effectively used for forecasting as well. A modulated Poisson process model [5] to describe and analyze arrival data to a call center. The attractive feature of this model is that it takes into account both covariate and time effects on the call volume intensity and in so doing enables us to assess the effectiveness of different advertising strategies along with predicting the arrival patterns. A Bayesian analysis of the model is developed and an extension of the model is presented to describe potential heterogeneity in arrival patterns. The proposed model and the methodology are implemented using real call center arrival data.

The use of nearest neighbour algorithm for forecasting call arrivals to call centers can also contribute to solving the question under discussion. The algorithm does not require an underlying model for the arrival rates and it can be applied to historical data without pre-processing it. The nearest neighbour algorithm with the Pearson correlation distance function is also able to take correlation structures, that are usually found in call center data, into account. [6].

In [11], the methods considered include seasonal Autoregressive Integrated Moving Average (ARIMA) and regression with ARIMA errors. Two models out of different candidate models are identified through statistical inferences of the model parameters, diagnostic checking and fit indices by using the training set for the original series and logged series. Forecasting performances of the models are evaluated with one-step ahead forecasts by using the test set and it is concluded that suggested models adequately predict hourly calls of the call center.

Moreover, the approaches discussed for module 1 can be altered and tweaked in a way that they contribute to solving this problem of call arrival count instead call arrival rate.

Experiments and Results

As discussed above in introduction, the whole process of predicting call arrival times and rates is divided into three modules. Following is a detailed explanation of our experiments and the results for each of the required modules.

Module 1: When can we expect the next call to come in?

Bootstrapping
Simulation methods in which the distribution to be sampled from is determined from data are called bootstrap methods. Bootstrap is a powerful, computer-based method for statistical inference without relying on too many assumption. The basic idea of bootstrap is make inference about a estimate(such as sample mean) for a population parameter - theta - (such as population mean) on sample data. It is a resampling method by independently sampling with replacement from an existing sample data with same sample size n, and performing inference among these resampled data.

Our bootstrapping function takes in the dataFrame as input along with day of week (otherwise it runs on the whole data), number of times it needs to resample. In our bootstrapping approach we make a list which has a length of 90 percent of data that is being fed in. Then we sample with replacement and calculate the mean. We repeat this process 10000 times. As per the central limit theorem the sample means would form a normal distribution. Our function outputs the list of sample means, the mean of the data being fed in, a list of seed values that were put into the resample function, the length of samples and the data that was inputted in the function.

Furthermore, we calculate the mean of sample means which we call the bootstrap mean. Then we calculate the difference between the mean of arrival differences of the data and the bootstrap mean to see how much the values differ.

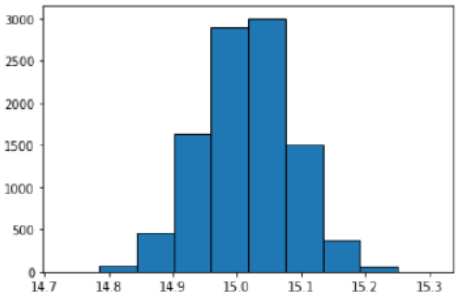
For each day of the week we calculate the mean of sample means which we call bootstrap mean. Then we calculate the difference between the simple mean (of arrival differences) and the bootstrap mean:

Labels	Monday	Tuesday
Sample Mean	15.015320520847958	17.04326839939892
Bootstrap Mean	15.015856365878586	17.043415994368996
Difference	-0.0005358450306278684	-0.00014759497007688083

Wednesday	Thursday	Friday
19.90720098119531	21.089445326852633	22.148069853519115
19.90689372352281	21.09044311952291	22.14717803630309
0.0003072576724996168	-0.0009977926702759987	0.0008918172160257143

Saturday	Sunday
39.80605238829787	50.03980967819629
39.80510127204907	50.0411711558025
0.0009511162487996216	-0.0013614776062098 6

Then we plot the sample means of a day, such as Monday, to determine that the central limit theorem is working.



Moving on, we calculate the sample means of the whole dataset i.e. containing all days of week. We then calculate the bootstrap mean and then finally we calculate the difference between the simple mean and bootstrap mean.

The purpose of calculating the mean of the whole dataset was to see if there is a difference between the means of individual days and the mean of the whole dataset.

Furthermore, by calculating the mean of the whole dataset we could use it to model the days which have same means as the mean of this whole dataset. So instead of finding seven parameters for the exponential random variables we can use lesser number of parameters when there is an overlap. In accordance with our calculations, the bootstrap mean of the whole dataset turns out to be: **22.347058536955526** which is approximately the same as the mean of the dataset: **22.347058536955526**.

We calculate the confidence intervals for our bootstrapping. We use the approach mentioned in the chapter 5 of the book. Statistics for Engineers and Scientists by William Nivadi. We sort the data and then calculate the mean of values from 250-260 for the lower con

dence interval and calculate the mean of 9750-9760 value for the upper confidence interval. Plotting the bootstrap Means of each day and the bootstrap Mean of the whole dataset along with confidence intervals. If there is an overlap between confidence intervals we can say that the difference between those two particular means is not statistically significant. From our plots we can see the mean of the week overlaps with Friday hence the difference is not statistically significant. However, since there is only one overlap so we still need seven parameters or seven expected values to make an exponential distribution for each day of the week. Hence, we do not perform any hypothesis testing.

Following are the seven bootstrap means or our expected values.

Stat	Mon	Tue	Wed	Thu	Fri	Sat	Sun	Week
B-Means	15.01	17.04	19.90	21.09	22.14	391.80	50.04	22.34

We can calculate the rate parameter by the following formula lambda = 1/E[x]:

Stat	Mon	Tue	Wed	Thu	Fri	Sat	Sun	Week
Lambdas	2.71	3.99	1.50	1.19	2.84	3.52	3.01	2.68

Survival Analysis

Survival Analysis is used to estimate the lifespan of a particular population under study. It is also called 'Time to Event' Analysis as the goal is to estimate the time for an individual or a group of individuals to experience an event of interest. This time estimate is the duration between birth and death events. Survival Analysis was originally developed and used by Medical Researchers and Data Analysts to measure the lifetimes of a certain population. But, over the years, it has been used in various other applications such as predicting churning customers/employees, estimation of the lifetime of a Machine, etc. The birth event can be thought of as the time of a customer starts their membership with a company, and the death event can be considered as the customer leaving the company. Next, we also need to talk about one of the fundamental objects in survival analysis, the **Survival Function**.

Let **T** be a (possibly infinite, but always non-negative) random lifetime taken from the population under study. For example, the amount of time a couple is married. Or the time it takes a user to enter a webpage (an infinite time if they never do).

The survival function - **S(t)** - of a population is defined as:

$$S(t) = P(T > t)$$

Simply, the survival function defines the probability the death event has not occurred yet at time t, or equivalently, the probability of surviving past time **t**.

Modeling the Call Problem

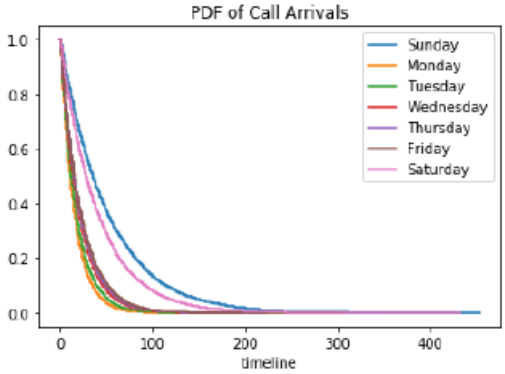
Now, we need to model our problem for Survival Analysis. First, we begin by deciding the how we represent our call in a life and death scenario. We decide that the death of a call occurs when a call is received. This means that all the calls are, initially, placed at a reference kine, also known as their point of birth, and these calls die after an interval, in other words, a call is received. Therefore, Survival Analysis estimates the survival or the death (aka call arrival) of the call.

Secondly, we decide upon the survival function that we will be using to estimate the probability of a call surviving with the passage of time. For that, we utilize Ka-plan Meir Estimator to form a Probability Density Functions, which is a conditional distribution. The Kaplan-Meier estimate is the simplest way of computing the survival over time in spite of all these difficulties associated with subjects or situations. The survival curve can be created assuming various situations. It involves computing of probabilities of occurrence of event at a certain point of time and multiplying these successive probabilities by any earlier computed probabilities to get the final estimate. Its advantage of KMS lies in its essence of being a non-parametric estimator. Parametric Estimators assume that the data follows a certain distribution and utilizies the parameters of the distribution to obtain results. KMS, instead, form a PDF by directly computing conditional probabilities from the data.

Through our exploratory data analysis, we know that there are 7 lambdas for each day, which is to say that there is an exponential distribution describing the intervals between our calls for each day. We also know that any Exponential distribution can be described using its parameter, lambda.

Survival Analysis allows us to learn the probability distributions describing the **probability of a call arriving during or after a specific interval for each day**, which in-turn can be **used to obtain other statistics such as lambdas** that describes the average interval between two successive call, or in

other words, our exponential distribution. Following are the PDFs obtained for each day:



Additionally, the mean survival time would be the area under the curve. Therefore:

Day	Mon	Tue	Wed	Thu	Fri	Sat	Sun
Mean	15.51	17.54	20.40	21.58	22.63	40.22	50.35

Module 2: Amongst well-defined categories, what type of call is it?

Generalized Linear Models

In these models, the response variable y_i is assumed to follow an exponential family distribution with mean μ_i which is assumed to be some (often non-linear) function of $x_i^T\beta$. The covariates affect the distribution of y_i only through the linear combination $x_i^T\beta$

. The generalized linear models (GLMs) are a broad class of models that include linear regression, ANOVA, Poisson regression, log-linear models etc. The table below provides a good summary of GLMs. **There are three components to any GLM**:

- Random Component refers to the probability distribution of the response variable (Y); e.g. normal distribution for Y in the linear regression, or binomial distribution for Y in the binary logistic regression. Also called a noise model or error model. How is random error added to the prediction that comes out of the link function?

- Systematic Component - specifies the explanatory variables $X_1, X_2, ..., X_k$ in the model, more specifically their linear combination in creating the so called linear predictor; e.g.,

$\beta_0 + \beta_1 x_1 + \beta_2 x_2$ as we have seen in a linear regression, or as we will see in a logistic regression in this lesson.

- Link Function, $\eta org(\mu)$ - specifies the link between random and systematic components. It says how the expected value of the response relates to the linear predictor of explanatory variables.

Assumptions:

- The data $Y_1, Y_2, ..., Y_n$ are independently distributed, i.e., cases are independent.

- The dependent variable Y_i does NOT need to be normally distributed, but it typically assumes a distribution from an exponential family (e.g. binomial, Poisson, multinomial, normal,...)

- GLM does NOT assume a linear relationship between the dependent variable and the independent variables, but it does assume linear relationship between the transformed response in terms of the link function and the explanatory variables. Independent (explanatory) variables can be even the power terms or some other nonlinear transformations of the original independent variables.

- The homogeneity of variance does NOT need to be satisfied. In fact, it is not even possible in many cases given the model structure, and overdispersion (when the observed variance is larger than what the model assumes) maybe present.

- Errors need to be independent but NOT normally distributed.

- It uses maximum likelihood estimation (MLE) rather than ordinary least squares (OLS) to estimate the parameters, and thus relies on large-sample approximations.

- Goodness-of-fit measures rely on sufficiently large samples, where a heuristic rule is that not more than 20 percent of the expected cells counts are less than 5.

Logistic Regression

Since our response variable has two categories we have used logistic regression. Binary Logistic Regression models how binary response variable Y depends on a set of k explanatory variables, $X = X1, X2, ..., X_k$. $logit(\pi) = log(\frac{\pi}{1-\pi}) = \beta_0 + \beta * x_i + ... + \beta_0 + \beta * x_k$ which models the log odds of probability of success as a function of explanatory variables.

Random component: The distribution of Y is assumed to be Binomial(n, η), where η is a probability of success.

Systematic component: X's are explanatory variables (can be continuous, discrete, or both) and are linear in the parameters, e.g., $\beta_0 + \beta * x_i ... + \beta_0 + \beta * x_k$. Again, transformation of the X's themselves are allowed like in linear regression; this holds for any GLM.

Link function:

- Logit link: $logit(\pi) = log(\frac{\pi}{1-\pi})$

- More generally, the logit link models the log odds of the mean, and the mean here is η. Binary logistic regression models are also known as logit models when the predictors are all categorical.

Note: We first fitted a GLM with all of the explanatory variables included. Then we used the dredge function to make models of all possible combinations of 2 from those features. Since we have 6 features then all possible combinations will be $2^6 = 64$. In order to simplify things a little, we also made another model which did not contain ArrivalDifferences since our problem specifically asks for predicting the call category of the next call. Then we used the dredge functions to make models from all possible combinations of 2 from those features. In this case since we removed Arrival Differences we have 5 features so all possible combinations will be $2^5 = 32$. Furthermore, it wasn't correlated with the response variable as per the Cramer's V test. In order to simplify things we also made model without ArrivalDifferences.

Bayes Naive and Developing a Bayesian Network

This section will be about obtaining a Bayesian network, given a set of sample data. Learning a Bayesian network can be split into two problems:

Parameter learning: Given a set of data samples and a DAG that captures the dependencies between the variables, estimate the (conditional) probability distributions of the individual variables.
Structure learning: Given a set of data samples, estimate a DAG that captures the dependencies between the variables which is further classified into:

- score-based structure learning
- constraint-based structure learning

We will be briefly discussing and applying the first approach on our problem.

Score-based Structure Learning

This approach construes model selection as an optimization task. It has two building blocks:

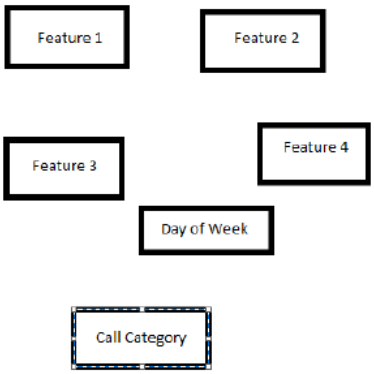
- A scoring function that maps models to a numerical score, based on how well they fit to a given data set.

- A search strategy to traverse the search space of possible models M and select a model with optimal score.

Scoring functions: Commonly used scores to measure the fit between model and data are Bayesian Dirichlet scores such as BDeu or K2 and the Bayesian Information Criterion (BIC, also called MDL).

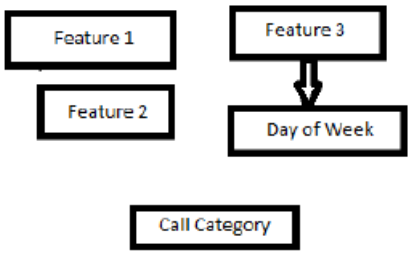
Search strategies: The search space of DAGs is super-exponential in the number of variables and the above scoring functions allow for local maxima. The first property makes exhaustive search intractable for all but very small networks, the second prohibits efficient local optimization algorithms to always find the optimal structure. Thus, identifying the ideal structure is often not tractable. Despite these bad news, heuristic search strategies often yields good results. Take for instance, **HillClimbSearch**. implements a greedy local search that starts from the DAG 'start', our reference node is a graph with all nodes but no edges or conditional dependencies, and proceeds by iteratively performing single-edge manipulations that maximally increase the score. The search terminates once a local maximum is found. By Single-Edge manipulations, we refer to legal operations that can be performed on the network. Operations such as addition, deletion or modification of an edge between two nodes.

For our problem, we used BicScore as our Scoring functions. The BIC/MDL score ("Bayesian Information Criterion", also "Minimal Descriptive Length") is a log-likelihood score with an additional penalty for network complexity, to avoid over-fitting.As for the search space strategy, we employed HillClimbSearch.



To start off with, above is the reference graph that is created from our features and call category. First, the Scoring function calculates the log probability of this Directed Acyclic Graph to to determine how good the graph is classifying Cal Categories. Then, as the next step, the search space function finds a better directed graph that represents our problem by performing legal operations on the graph such as the addition or deletion of an edge. As a result of either of the two actions, a new graph is formed whose fitness is again

determined through the Scoring functions. If the current graph is better than the old one, our algorithms chooses this as the most optimal graph up til now and repeats the process until it cannot find another directed graph that has a better score.
As a result, the optima directed graph determined by our algorithm comes out to be:



According to the above graph, our output - Call Category - is independent of any other features. In other words, it is either random or there are no patterns within the current features that affect our output label.
For the purpose of ensuring the reliability of the process and results, we also carried out a conditional independence test between the output label, call category, and our features. Independencies in the data can be identified using chi2 conditional independence tests. The chi-squared statistic is a single number that tells you how much difference exists between your observed counts and the counts you would expect if there were no relationship at all in the population.
The Chi-squared test confirms the results achieved through Score-Based Structure Learning. It states that there is no conditional relationship between the output label oand and any of the features. It also states that there are no conditional relationships between the feature themselves, with the exception of Feature 1 and Day of Week.

Parameter Learning

Having learnt the DAG that captures dependencies between the variables, now using the data samples we need to estimate the (conditional) probability distributions of the individual variables. We did this through Maximum Likelihood Estimation and Bayesian Parameter Estimation.

Maximum Likelihood Estimation A natural estimate is to simply use the *relative frequencies*, with which the variable states have occurred. Hence, let's assume a scenario where I have two types of calls: HR and Business based calls. I receive a total of 10 calls where 7 of these calls are HR based calls, then according to the concept of relative frequencies, MLE will over-fit the data, where it estimate that in any scenario 70 percent of the calls are HR based calls. In the long run, this might actually turn out to not be the case. This 70 percent could be because we didn't have enough data. For our problem, given our data, MLE estimated that:

Feature2(PostQ)	0.507065
Feature2(PreQ)	0.492935

Feature3(F)	0.291573
Feature3(I)	0.303369
Feature3(M)	0.405058

Feature4(SPR)	0.370019
Feature4(TMO)	0.44436
Feature4(VER)	0.185621

CallCategory(PostQ)	0.5
CallCategory(PreQ)	0.5

Feature1(A+)	0.142063
Feature1(A-)	0.285841
Feature1(AB+)	0.428357
Feature1(AB-)	0.143739

From the estimations, we can see that at any point in time when a call is received, there is a 50 percent probability that the call category can be either PostQ or PreQ. However, when estimating parameters for Bayesian networks, lack of data is a frequent problem. Even if the total sample size is very large, the fact that state counts are done conditionally for each parents configuration causes immense fragmentation. If a variable has 3 parents that can each take 10 states, then state counts will be done separately for $10^3 = 1000$ parents configurations. This makes MLE very fragile and unstable for learning Bayesian Network parameters. A way to mitigate MLE's over-fitting is *Bayesian Parameter Estimation*.

Bayesian Parameter Estimation

The Bayesian Parameter Estimator starts with already existing prior CPDs, that express our beliefs about the variables *before* the data was observed. Those "priors" are then updated, using the state counts from the observed data. One can think of the priors as consisting in pseudo state counts, that are added to the actual counts before normalization. A very simple prior is the so-called K2 prior, which simply adds 1 to the count of every single state.Another is BDeu (Bayesian Dirichlet equivalent uniform prior). For BDeu we need to specify an equivalent sample size N and then the pseudo-counts are the equivalent of having observed 'N' uniform samples of each variable.

We used BDeu as our prior for Bayesian Parameter Estimation and obtained the below results. Note, that the estimator learns of a conditional probability distribution between Feature 3 and Day of Week but as for the call category probabilities, due to the absence of any conditional dependencies, the estimations for it are the same as that estimated by the MLE. Again, this is because there are no conditional dependencies, and hence, in any case, regardless of what estimator we use, the probabilities of the Call Category will always be a result of the relative frequencies.

Conclusion:

Prediction of call arrival times and rates is indeed a difficult problem to target due to its probabalistic and statistical nature. In this paper, we tried to address this complex problem using various statistical methodologies discussed above. Our work contributes to the novelty of this problem which can be backed by our literature review as well. Our results are satisfactory and approved by the supporting organization from which we acquired the data and technical consultancy.

References:

1. Weinberg, J., Brown L.D. and Stroud J.R. *Bayesian Forecasting of an Inhomogeneous Poisson Process with Applications to Call Center Data*, 2006.

2. Ibrahim R., Ye H., LEcuyer P. and Shen H. *Modeling and forecasting call center arrivals: A literature survey and a case study*, 2016.

3. Ibrahim R., Ye H., LEcuyer P. and Shen H. *On the modeling and forecasting of call center arrivals*, 2015.

4. Shen H. and Huang J.Z. *Interday Forecasting and Intraday Updating of Call Center Arrivals.*

5. Soyer R. and Tarimcilar M.M. *Modeling and Analysis of Call Center Arrival Data: A Bayesian Approach.*

6. Bhulai S, W.H. Kan and Marchiori E. *Nearest Neighbour Algorithms for Forecasting Call Arrivals in Call Centers.*

7. Ibrahim R., LEcuyer P., Regnard N. and Shen H. *On The Modeling and Forecasting of Call Center Arrivals*, 2012.

8. Phithakkitnukoon S. and Dantu R. *Predicting Calls New Service for an Intelligent Phone.*

9. Koen van den Bergh. *Predicting Call Arrivals in Call Centers*, 2006.

10. Sennaroglu B. and Polat G. *Time Series Forecasting for a Call Centre*, 2017.

11. Noiman S.A., Feigin P.D. and Mandelbaum A. *Workload Forecasting For a Call Center: Methodolgy and a Case Study*, 2009.

12. Brown L., Gans N, Mandelbaum A., Sakov A., Shen H., Zeltyn S. and Linda Zhao. *Statistical Analysis of a Telephone Call Center*, 2013.

13. Mahmud T., Hasan M., Chakraborty A., and Chowdhury A.R. *A Poisson Process Model for Activity Forecasting*, 2016.

14. Zhu L. and Laptev N. *Deep and Confident Prediction for Time Series at Uber*, 2017.

15. Zhu L. and Laptev N. *Deep and Confident Prediction for Time Series at Uber*, 2017.