United Nations Millennium Development Goals

Predictive Analytics
Special Topics Section - 011
Fall 2016
Professor Anasse Bari, Ph.D

Pranit Arora <pa1230@nyu.edu>
Anand Bhave <asb761@nyu.edu>
Leslie Manrique <ljm391@nyu.edu>

Abstract

In the year 2000, the member states of the United Nations agreed to a set of goals to measure the progress of global development. These are known as the Millennium Development Goals. As a part, the UN uses a number of indicators collected by the World Bank. In this project, we use historical data for some of these indicators in order to predict future values of indicators which can be used to measure the project goals.

For an in-depth analysis, we split our project into two main into two subparts:
First, to identify some patterns in our dataset, we implement clustering. This is done so that we can create groupings of similar countries based on a specific socio-economic indicator. For this part, we use as much data as possible, for which at least some data is available.

Then, we proceed to our second part regarding the Millennium Development Goals and how the features pertaining to an indicator for a particular country change one to five years from now? For this part, we use a prepared dataset of indicators having yearly data from 1972 to 2007 in order to predict values of specific indicators for 2008 and 2012.

Also some other questions we explore are,
1. How effective is the previous year's data to predict the future values?
2. What is the optimal length of window in the time series that gives effective predictions?

We try multiple algorithms including a modified drift, exponential smoothing, and ARIMA and compare their performance based on RMSE. The best results are obtained with the modified drift algorithm, with an RMSE of 0.0515, which is the 12th best submission on the competition hosted on https://www.drivendata.org/ out of a total of over 1000 competitors.

Table of Contents

Introduction

In the year 2000, the member states of the United Nations agreed to a set of goals to measure the progress of global development. The aim of these goals was to increase standards of living around the world by **emphasizing human capital, infrastructure, and human rights**.
The eight goals are:

1. To eradicate extreme poverty and hunger
2. To achieve universal primary education
3. To promote gender equality and empower women
4. To reduce child mortality
5. To improve maternal health
6. To combat HIV/AIDS, malaria, and other diseases
7. To ensure environmental sustainability
8. To develop a global partnership for development

1
Eradicate Extreme Poverty and Hunger
2
Achieve Universal Primary Education
3
Promote Gender Equality and Empower Women
4
Reduce Child Mortality
5
Improve Maternal Health
6
Combat HIV/Aids, Malaria, and Other Diseases
7
Ensure Environmental Sustainability
8
Global Partnership for Development

The UN measures progress towards these goals using indicators such as percent of the population making over one dollar per day. **Your task is to predict the change in these indicators one year and five years into the future**. Predicting future progress will help us to understand how we achieve these goals by uncovering complex relations between these goals and other economic indicators. The UN set 2015 as the target for measurable progress. Given the data from 1972 - 2007, you need to predict a specific indicator for each of these goals in 2008 and 2012.

Driven Data is currently hosting a competition featuring Data from the UN pertaining to these goals. The challenge is to find the indicating features in our data, relevant to the Millennium Development Goals and to develop a system that can predict how these features will change one to five years from now.

Problem statement

We divide the problem statement into 2 parts:

1) In the year 2000, the member states of the United Nations agreed to a set of goals to measure the progress of global development. The UN measures progress towards these goals using indicators such as percent of the population making over one dollar per day. Our aim is to predict the change in these indicators one year and five years into the future. Predicting future progress will help us to understand how we achieve these goals by uncovering complex relations between these goals and other economic indicators. The UN set 2015 as the target for measurable progress. Given the data from 1972 - 2007, we wish to predict a specific indicator for each of these goals in 2008 and 2012.

2) Based on certain indicators, we wish to generate clusters of similar countries in order to better understand how different countries are progressing or performing on the various indicators and to understand which countries are behaving similar to which countries. This can help identify which countries are leading the way, and which countries are lagging behind in crucial areas of development.

Identifying Data Sources

For most part, we have 2 major data sources for this project. The first is the one provided as part of the competition on drivendata.com, titled “United Nations Millennium Development Goals”. The second one was World Bank website.

Since its founding in 1944, the World Bank has been gathering data to help it alleviate poverty by focusing on foreign investment, international trade, and capital investment. We are interested in predicting the timeseries that are relevant to the Millennium Development Goals. There are a set of indicators from the World Bank dataset that represent our progress towards these goals. The competition website had created the subset of these indicators for which data from 1972-2007 was available. This was available as a download on the competition site.

The World Bank also provides these data to the public through their data portal. We used this portal to download further data we required on some common indicators, which we used for the clustering part of our project.

The Data

The dataset provided gives us yearly data on over 1200 macroeconomic indicators for 214 countries around the world. Each row has the time series data, Country name, series code, and series name. Note that not all indicators are available for all countries.
We are provided with 195,402 rows of this dataset. Given this data, we wish to predict certain indicators that the UN is using to monitor their Millennium Development Goals *[3]*. These are a total of 60 indicators that have been classified under one of the eight goals mentioned above. To assist with the prediction, we are provided with a separate file indicating the label numbers corresponding to these indicators in our dataset.

Data pre-processing algorithms

1) Exploratory data analysis

The first course of action on obtaining the data was to perform initial data exploration. We studied the data, and understood what kind of indicators were present in the data and what kind of indicators we were supposed to predict.

The training dataset consisted of row wise comma separated values, for which conversion into time series format was important. So an equivalent file was created which was the transpose of the original dataset and consisted of the column wise values for the target indicators for the period from 1972 to 2007. After this, we generated plots for the data to help better understand the trends in the data.

One of our first observations was that there was quite a lot of missing data in the file. This was not ideal, and meant that handling the missing data would be a major part of the project, especially in order to obtain improved accuracy. We decided to try out multiple ways to handle the missing data, which are mentioned in subsection 3. Another observation was that most of the indicators seemed to be autocorrelated, and seemed to have trends in the data. However, they seemed to lack seasonality in the data. This information was verified, as mentioned in the next subsection.

2) Convert Time Series to Stationary Time Series

A number of time series algorithms are applicable only to stationary data. A stationary time series is one whose statistical properties such as mean, variance, autocorrelation, etc. are all constant over time. So it is a common procedure to convert a time series into an approximately stationary series, i.e., the time series is stationarized. A stationarized series is relatively easy to predict: you simply predict that its statistical properties will be the same in the future as they have been in the past.

Analysis of our data revealed that our data does not contain any seasonality. However, it does contain trends and is autocorrelated. So, depending on the model, we would need to convert our data into stationary data before we can apply our algorithms. This can be done using the 'diff' function in R. The process can be reversed using the 'diffinv' function after application of the algorithm. Fortunately, a number of the packages provide function which take care this within the function. 'auto.arima' is one such function which first converts its input to a stationarized series before applying the arima algorithms.

3) Handling Missing Values

Our data contains a large number of missing values. As a result, methods of replacing missing data, such as replacing all missing data by attribute minima or maxima will not work across all the individual time series. After researching on common methods used to handle missing values in Time Series, we selected and tried the following for use in our models:

A) Na.interpolation (Using the imputeTS function)

Na.interpolation is a function in the imputeTS package of R that tries to impute missing values of a series using interpolation. It makes use of either linear, spline or stineman interpolation to replace missing values. We have made use of the linear option for our code. For any missing value, the function draws a linear function between the nearest known point on the left and the nearest known point on the right. This linear line is assumed to be representative of the missing function in that region. If the missing value is at an extreme, i.e., if there is no known point on either the left or the right of it, then the function is assumed to be flat, or constant, after the last known value.

B) Kalman Filtering

In R, the stats package includes two methods, KalmanRun & KalmanSmooth which make use of this. Both of them use Kalman Filtering to find the (Gaussian) log-likelihood, or for forecasting or smoothing. Kalman filtering is an algorithm that uses a series of measurements observed over time, containing statistical noise and other inaccuracies, and produces estimates of unknown variables that tend to be more precise than those based on a single measurement alone, by using Bayesian inference and estimating a joint probability distribution over the variables for each timeframe. The two methods KalmanRun & Kalman Smooth are slightly different implementations of kalman filtering. Both were used to develop functions and prediction models.

C) Modified Na.interpolation

This is a self implemented modification to the na.interpolation function. In the middle part, it performs the same job as na.interpolation, but in case of missing values in the extremes, instead of using a constant, or flat, function, it using a linear function with the slope being the same as it is just ahead of the missing values. For example, if the first 5 values are missing, and the sixth and the seventh values are 0.24 and 0.27, then the slope is calculated to be 0.03 and hence the fifth value is assumed to be 0.21, the fourth to be 0.18 and so on. In the original na.interpolation, all the first five values would have been equal to the sixth, i.e., equal to 0.24.

D) Na.locf [Last Observation Carried Forward]

Generic function for replacing each NA with the most recent non-NA prior to it. Part of the 'zoo' package in R. The result is an object in which each NA in the input object is replaced by the most recent non- NA prior to it. If there are no earlier non-NA's then the NA is omitted (if na.rm = TRUE) or it is not replaced (if na.rm = FALSE).

4) FS for correlation

In order to implement the correlation model, a lot of pre-processing & Feature Selection specific to this model was required. For this analysis, we divided the data according to country. This is done because of two reasons. Firstly, it doesn't make much sense for one country's progress to affect another, especially for a different type of indicator. Secondly, with the processing capabilities of R, calculating the correlation matrix of a 1000* 1000 matrix was taking between 30 & 60 seconds. To calculate the correlation matrix for the entire data, or a 195402*195402 matrix could have taken days.

So the initial analysis was performed on a small subset of countries(targeting those with a high number of target indicators). First, we separated the countries data for the 36 years into column vector format and stored them in the file. Then, we created a correlation matrix between all indicators of this country(averaging almost 1000 indicators per country). However, as we wish to use one indicator to predict the other in the future, we 'lag' one of the series by 1 year before we calculate the correlation matrix.

Once the matrix has been calculated, for each target indicator, we select the indicator with the highest magnitude of correlation. This indicator, along with its data and the correlation are used in the model. Further explanation is provided in the Modelling section.

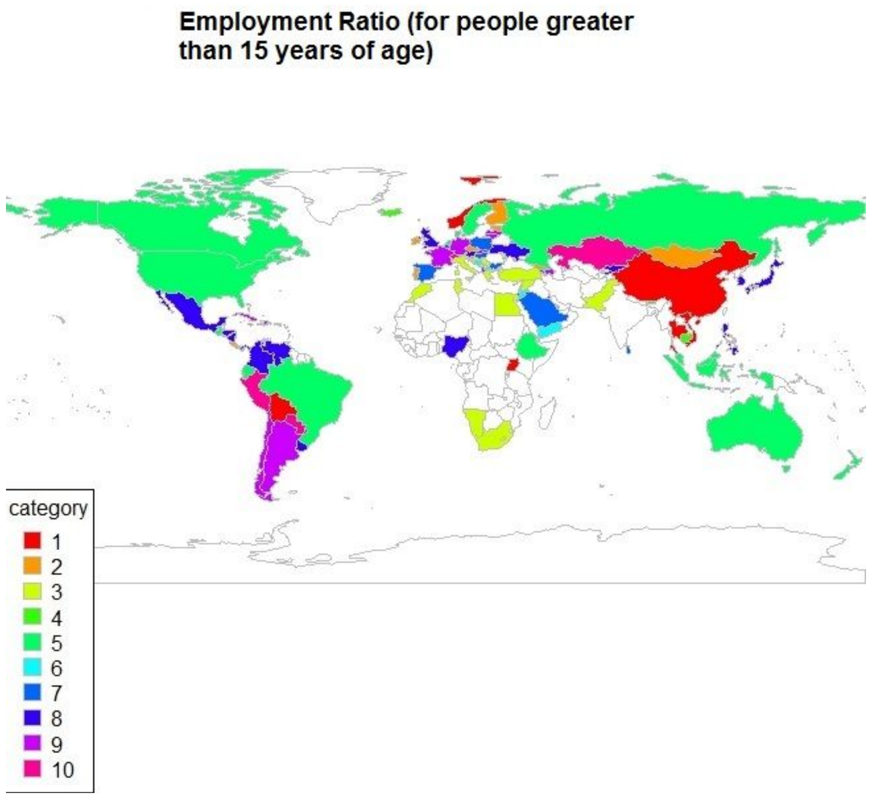
5) Clustering

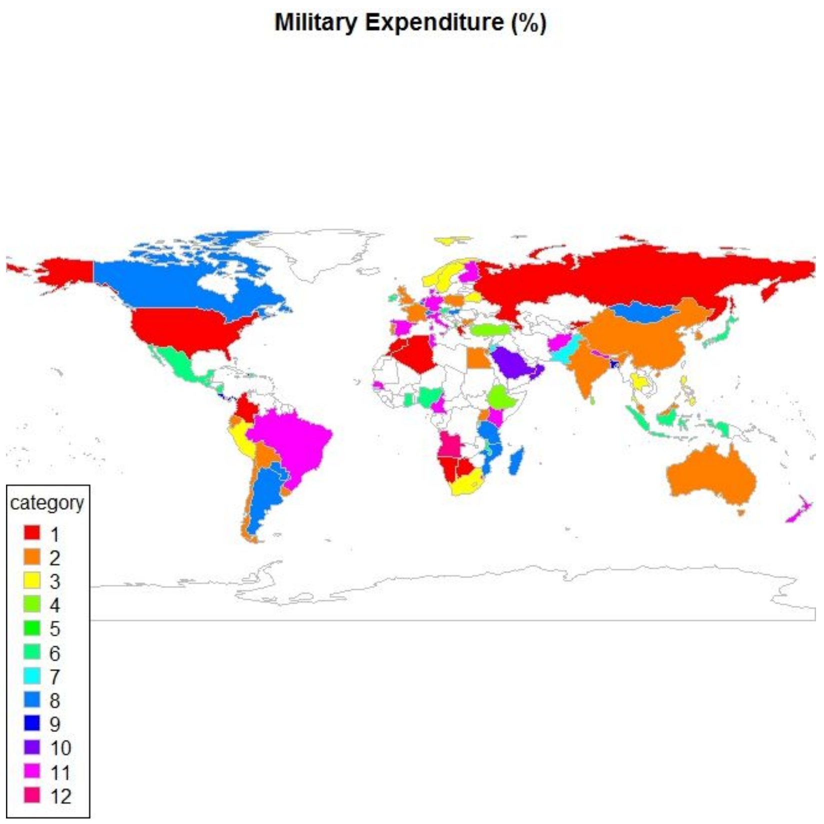
For identifying patterns in our dataset, like for example the groupings of countries which are the most similar in terms of a particular indicator, clustering techniques like hierarchical and K-means were implemented.

This gave us an insight into the dataset regarding which countries can be mapped to a particular set for an indicator.

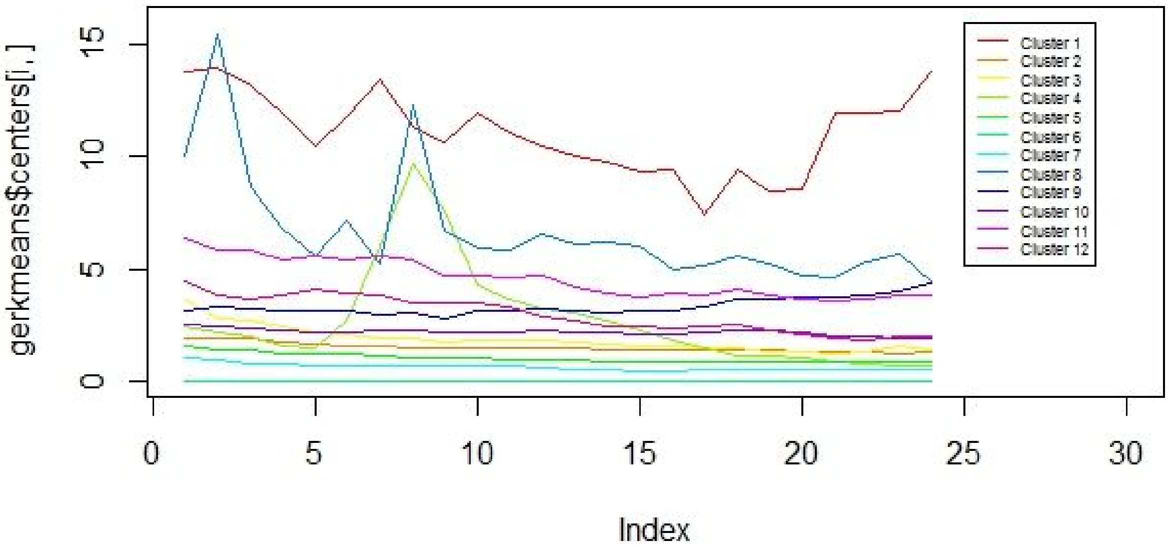
For an in-depth analysis, we first plotted the cluster centroids as representatives of the particular cluster with other cluster centroids using the plot() function.
Then using the rworldmap package in R, we plotted the corresponding countries on the world map, with each country assigned the color of its cluster to denote similarity.

The following figures give an insight into the dataset and the how the countries perform over a period of time with respect to other countries.

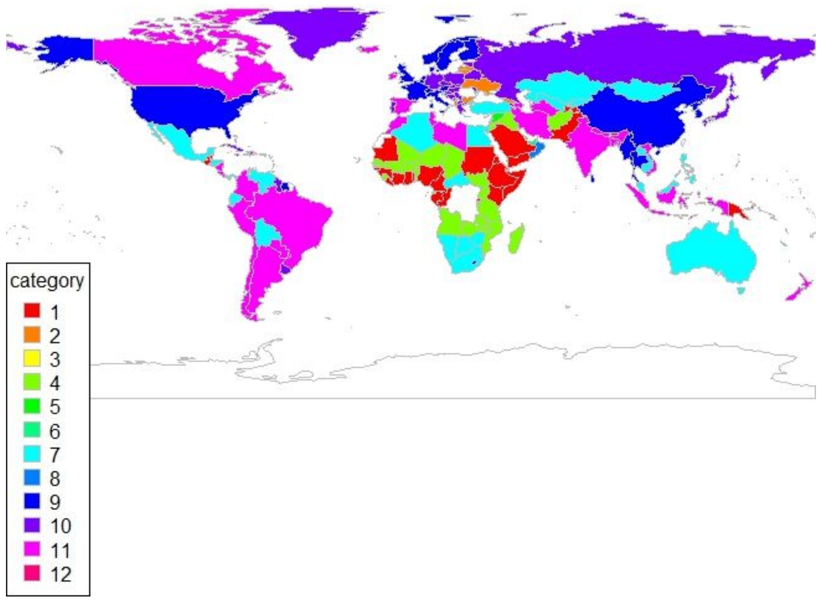




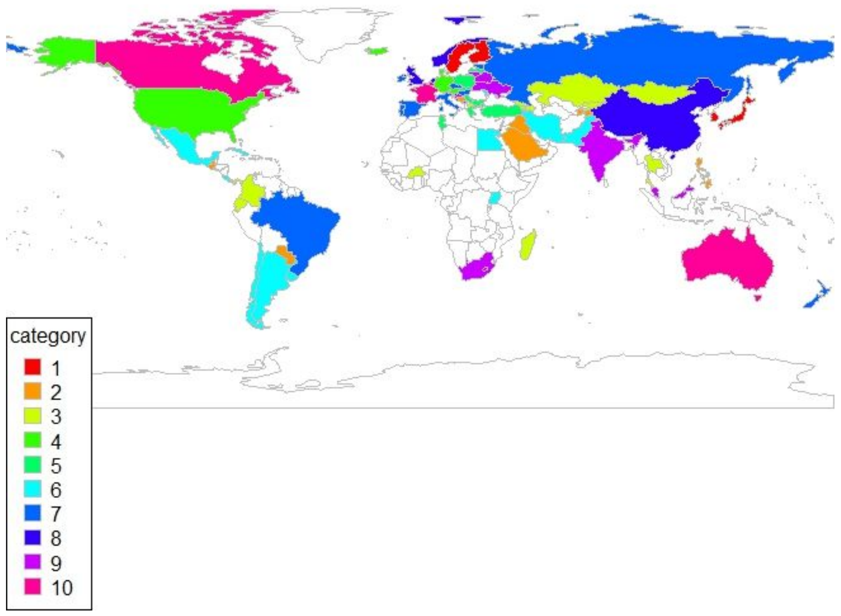
This time series plot for Military Expenditure shows the trends of different countries clustered, with the cluster centroid acting as a representative candidate in the plot.



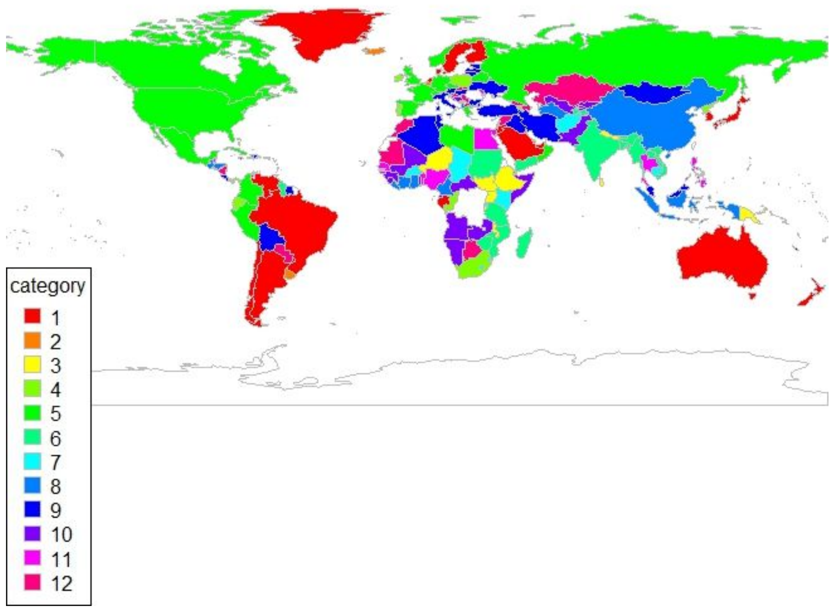
Population Growth

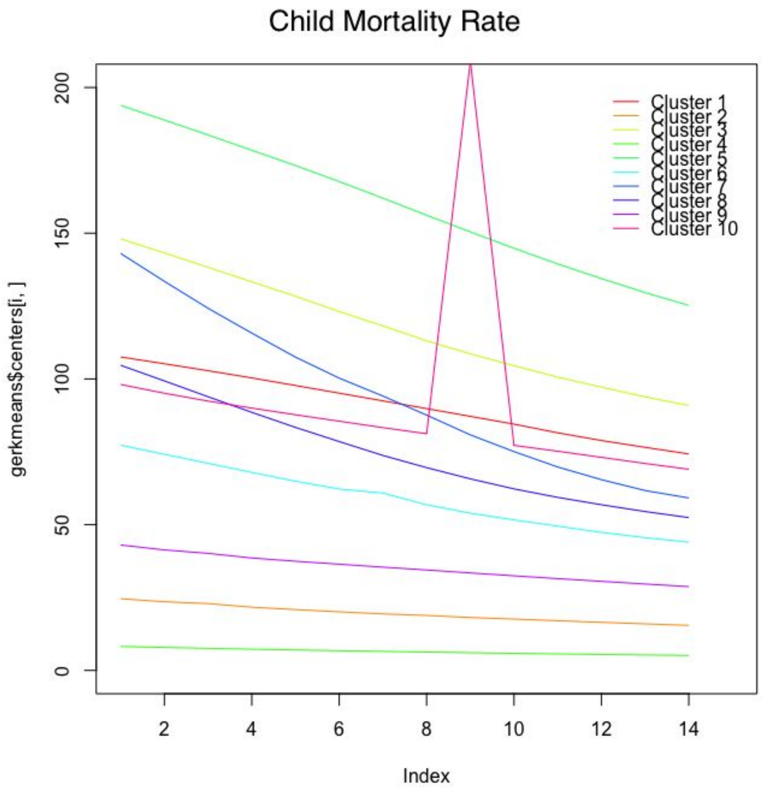


Research and Development Expenditure (%)



Percentage of urban population





Programming and applying models

1) Baseline

The competition page including a link to a page with the idea for a trivial baseline model, along with code in python. The model is the simplest model imaginable: For all indicators, calculate the slope of the data using just the last 2 years data (2006 and 2007), and assuming a linear model for the years from 2007 to 2012. If the data for 2006 is not available, then use the value of 2007 as the prediction of 2008 and 2012. There is also an alternative naive implementation that always uses the 2007 values as the predictions.

Due to the relatively low prediction range of 1 year and 5 years, as well as the relatively smooth nature of most of the indicators, this model performs decently well, giving a RMSE of 0.0678. The naive method, a.k.a the status quo method, gives a RMSE of 0.0734

2) Drift method

A variation on the naïve method is to allow the forecasts to increase or decrease over time, where the amount of change over time (called the drift) is set to be the average change seen in the historical data. So the forecast for time (T+h) is given by

$$y_T + \frac{h}{T-1}\sum_{t=2}^{T}(y_t - y_{t-1}) = y_T + h\left(\frac{y_T - y_1}{T-1}\right).$$

As we can see, this is equivalent to drawing a line between the first and last observation, and extrapolating it into the future.

3) Holt's Linear Method

Holt (1957) extended simple exponential smoothing to allow forecasting of data with a trend. This method involves a forecast equation and two smoothing equations (one for the level and one for the trend):

$$\begin{aligned} \text{Forecast equation} \quad & \hat{y}_{t+h|t} = \ell_t + hb_t \\ \text{Level equation} \quad & \ell_t = \alpha y_t + (1-\alpha)(\ell_{t-1} + b_{t-1}) \\ \text{Trend equation} \quad & b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1} \end{aligned}$$

In simple exponential smoothing, Forecasts are calculated using weighted averages where the weights decrease exponentially as observations come from further in the past --- the smallest weights are associated with the oldest observations.

As with simple exponential smoothing, the level equation here shows that l*t* is a weighted average of observation *yt* and the within-sample one-step-ahead forecast for time *t*, here given by ℓ*t*−1+*bt*−1. The trend equation shows that *bt* is a weighted average of the estimated trend at time *t* based on ℓt − ℓt−1 and *bt*−1, the previous estimate of the trend.
The forecast function is no longer flat but trending. The *h*-step-ahead forecast is equal to the last estimated level plus *h* times the last estimated trend value. Hence the forecasts are a linear function of *h*.

4) Linear with correlation

One idea mentioned in the starter code link was to explore if other time series were correlated to the ones we wish to predict, and if they could be used in the future. We need to use lagged time series of other indicators, as they should predict the values of our indicators in future years.
The idea certainly seems plausible, and initial explorations seemed promising. For example, In Kenya, the progress of 'Achieve universal primary education' is most correlated with a change in 'Gross national expenditure (current LCU)' during the year before. Also, ‘Reduce child mortality’ is highly correlated to the series ‘Mortality rate, infant (per 1,000 live births)’.

As mentioned before, a lot of data pre-processing specific to this model was required. Even doing the analysis country by country was a tedious exercise too. So, before doing analysis for all the countries, we decided to pick a small sample space of countries(targeting those with a high number of target indicators), we performed the analysis incorporating only a small number of indicators (22). First, we obtained the target indicator and its most correlated ‘lagged’ indicator. Then, using the slopes of the target indicator, the correlation coefficient, and the change in slope ratio of the lagged indicator in the last 2 years, an adjusted slope index was calculated and used for prediction of the target indicator. However, it was observed that the RMSE error obtained was 0.0679, which was not any better than the baseline model.

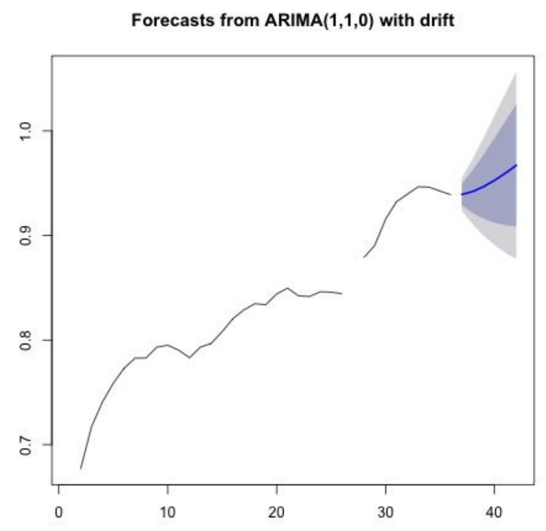
5) ARIMA

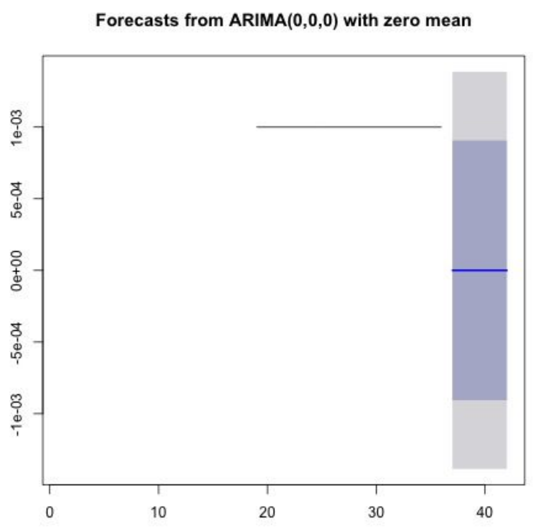
ARIMA is an acronym for AutoRegressive Integrated Moving Average. It is a generalization of the autoregressive moving average (ARMA) model. It is commonly fitted to time series data either to better understand the data or to predict future points in the series (forecasting).

There are actually a range of models which are classified as ARIMA models. Non-seasonal ARIMA models are generally denoted ARIMA(p,d,q) where parameters p, d, and q are non-negative integers, p is the order (number of time lags) of the autoregressive model, d is the degree of differencing (the number of times the data have had past values subtracted), and q is the order of the moving-average model. All these models could also be used with drift.

R includes a ‘forecast’ library, which has a auto.arima function, which returns best ARIMA model according to either AIC, AICc or BIC value(which are information criterion used for statistical model selection). The function conducts a search over possible model within the order constraints provided.

Auto.arima also has the nice property that it can handle missing data, which arima models cannot. However, using the arima model, the RMSE obtained was 0.1307. By plotting the data, it was observed that arima, though it gave good smooth functions for most indicators, it gave terrible predictions for functions where the ARIMA(0,0,0) function was selected. These were mostly for indicators with zero variance or with high number of missing values. For indicators where more than 30 out of the 36 values were missing, we replaced arima prediction with the simple slope prediction, but the RMSE only reduced to 0.1214





6) Arima With Kalman Filtering

Arima models can also be used to calculate missing values, as mentioned before. The stats package includes two methods, KalmanRun & KalmanSmooth which make use of this. Both of them use Kalman Filtering to find the (Gaussian) log-likelihood, or for forecasting or smoothing. Kalman filtering is an algorithm that uses a series of measurements observed over time, containing statistical noise and other inaccuracies, and produces estimates of unknown variables that tend to be more precise than those based on a single measurement alone, by using Bayesian inference and estimating a joint probability distribution over the variables for each timeframe. The two methods KalmanRun & Kalman Smooth are slightly different implementations of kalman filtering. Both were used to develop functions and prediction models.

KalmanRun gave RMSE of 0.1066, while KalmanSmooth gave RMSE of 0.0972. On observing the plots of the data, it is most likely due to the high percentage of missing values, especially in the earlier years, that do not facilitate good fits of the underlying function and lead to poor prediction accuracy.

7) Modifications to ARIMA

As mentioned above, a modified type of interpolation was also used with the data. This resulted in improved accuracy, mainly because the high amount of missing data. The RMSE score of the ARIMA model improved drastically to 0.0607.

We also tried to remove all missing values prior to the first known values in hope of them not affecting the predictions. However, this probably led to too less data to fit the models, as the RMSE error increased to 0.0644.

8) Discounted Models

One observation that was observed during the data exploration, especially during the clustering phase was that there was a slight discontinuity in some indicators between 2007 and 2008. This can be attributed to many factor, the most significant probably being the 2008 Financial Crisis that affected a large number of country between 2008 and 2011 and prevented progress in a number of areas. There were early indications of the crisis in 2007 as well. This motivated us to discount our predictions, that is, multiply them with a factor c in our final predictions. This led to a large improvement in accuracies in multiple models. For the baseline linear model, RMSE reduced to 0.0566 while ARIMA's accuracy improved to 0.0537, which is our best obtained accuracy. Both used the discount factors between 0.6 and 0.65.

9) Modified Drift method

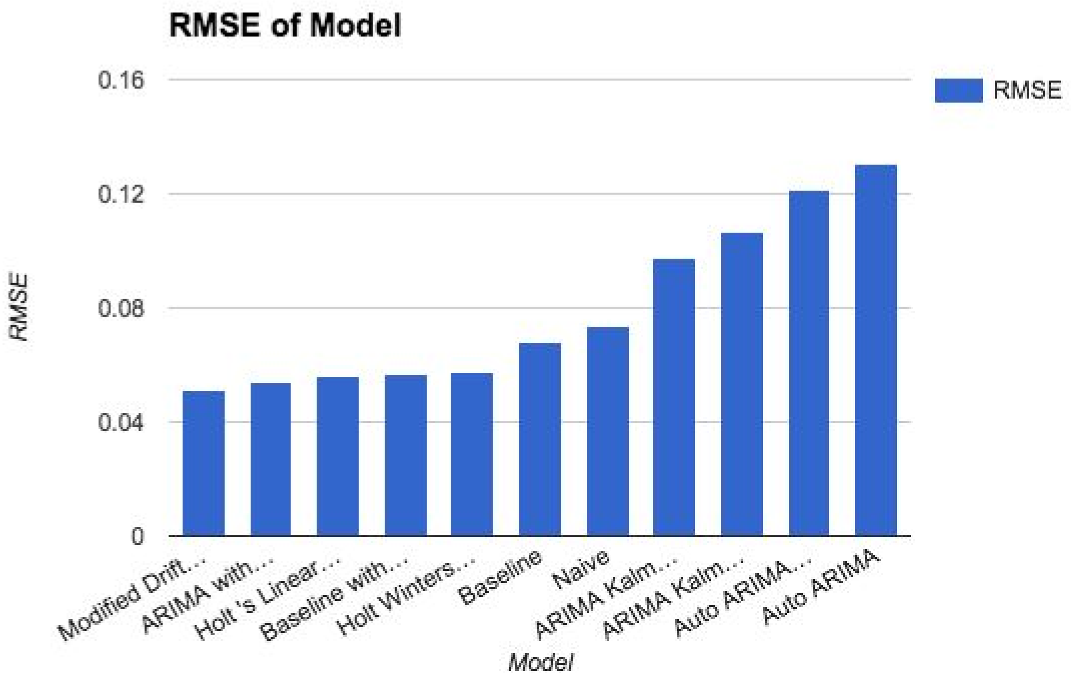
The first extension to Drift method was the introduction of Discounted Baseline Models to the Drift method when the number of NA's were greater than a specified amount. Thus for any given time series column, based on the Data quality, the function selected either the Discounted Model or Drift method.
The next extension to the Drift method was made where the window of the time period considered was modified to minimize the RMSE, in addition to implementing the Discounted Model.

Measuring Model Accuracy

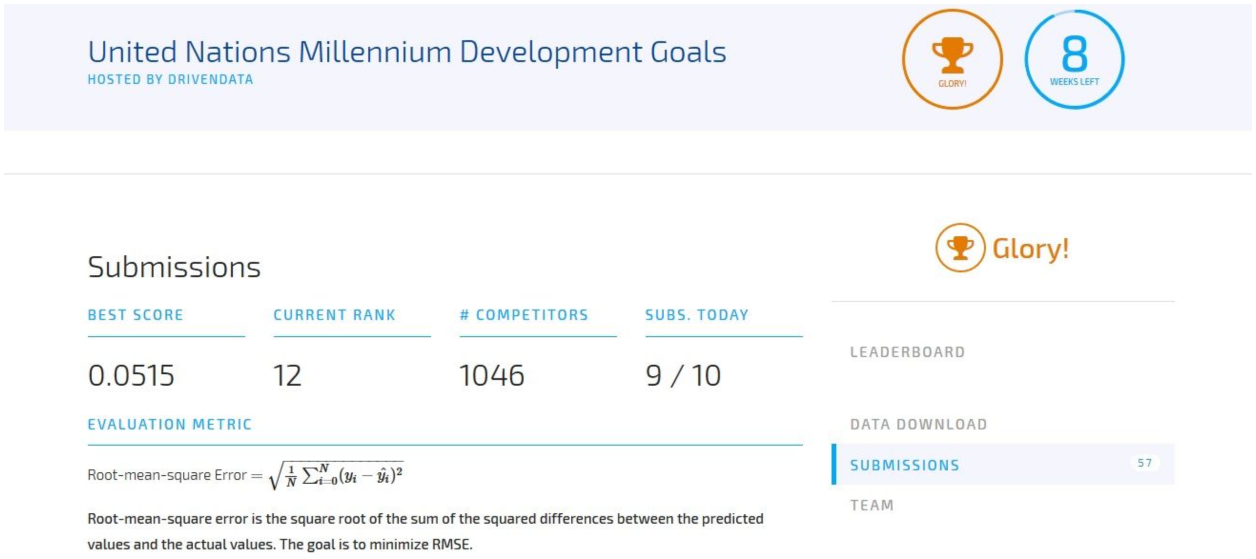
For accuracy of our models, we used the competition website, which contains the true values of the 2008 and 2012 data for our target indicators and calculates the Root Mean Square Error(RMSE) for the values we provide. For this, we needed to store our predictions in a csv file with the appropriate column headings and format. A sample submission file, which could be easily edited, was provided to assist in this.

$$\text{Root-mean-square Error} = \sqrt{\frac{1}{N}\sum_{i=0}^{N}(y_i - \hat{y}_i)^2}$$ where y_i is the predicted value and $\widehat{y}_i$ is the actual value

The objective of our models is to minimize the RMSE error. The performance of various models can be summarised in the chart below.



The results from the models were submitted to the drivendata competition website as a csv file. The website calculates the RMSE and returns the value. Hence the actual values of 2008 and 2012 are hidden from us and we cannot influence our model as per the exact values. The competition has had submissions from over 1000 competitors, and the results modified drift method, with a RMSE of 0.515, is the 12th best submission as off 16th December 2016.


United Nations Millennium Development Goals
HOSTED BY DRIVENDATA

GLORY!
8 WEEKS LEFT

Submissions

BEST SCORE	CURRENT RANK	# COMPETITORS	SUBS. TODAY
0.0515	12	1046	9 / 10

EVALUATION METRIC

Root-mean-square Error $= \sqrt{\frac{1}{N}\sum_{i=0}^{N}(y_i - \hat{y}_i)^2}$

Root-mean-square error is the square root of the sum of the squared differences between the predicted values and the actual values. The goal is to minimize RMSE.

Figure: Screenshot from the drivendata competition website

Framework and Tools used

For this project, the data was stored in .csv format. Initial data manipulation was performed using RapidMiner, but majority of data manipulation & modelling was performed in R. Rstudio was used as the user interface when using R. For visualization, the plotting features in R and tableau were used.

References

[1]
http://www.unfoundation.org/what-we-do/issues/mdgs.html?referrer=https://www.google.com/

[2]
https://www.drivendata.org/competitions/1/data/

[3]
http://unstats.un.org/unsd/mdg/Host.aspx?Content=Indicators/OfficialList.htm