

Machine Learning Engineer Nanodegree

Capstone Project:

Forecasting time series web traffic for wiki articles

Akhilesh Jain

December 20th, 2018

I. Definition

Project Overview

Companies such as Google are interested in having models which can predict web traffic to certain websites potentially for better resource distribution and ad placements. This project is based on a [competition](#) floated by Google on [Kaggle.com](#) about a year ago. The objective is to explore the web traffic project and create a time-series model which can predict web traffic for hundreds of wiki articles.

Forecasting future events is a valuable many institutions such as stock markets, online sellers and advertisers. Forecasting web traffic is one of the most challenging problems the data has not just strong temporal effects but also dependencies on current events which are difficult to capture.

Problem Statement

The goal of the project is to predict the web traffic on the Wikipedia based on previous views for hundreds of wiki articles. Exploratory analysis will be performed on the dataset to find relevant information. A dataset will also be split and merged to find total daily views for each language. There will be two types of predictions made:

1. Prediction for daily views for every page; and,
2. Prediction for daily views by language.

For forecasting, the dataset will be divided into a training set and testing set and validation set. The final prediction will be made for about a month (50 days). The quality of the predictions made by the model will be evaluated using Root Mean Square Error (RMSE).

Metrics

Root Mean Square Error (RMSE) is used for evaluating the predictions made by the model. The mean squared error (MSE) is the average of the squared prediction errors. Squaring the error values gives a positive number and also puts more weight on large errors. RMSE is the root of MSE which has the same unit as the observations.

II. Analysis

Data Exploration

The dataset consists of approximately 1,45,000 time series. Each of these time series represent a number of daily views of a different Wikipedia article, starting from July, 1st, 2015 up until December 31st, 2016 (Fig 1).

Page	2015-07-01	2015-07-02	2015-07-03	2015-07-04	2015-07-05	2015-07-06	2015-07-07	2015-07-08	...	2016-12-27	2016-12-28	2016-12-29	2016-12-30	2016-12-31
2NE1_zh.wikipedia.org_all-access_spider	18.0	11.0	5.0	13.0	14.0	9.0	9.0	22.0	...	20.0	22.0	19.0	18.0	20.0
2PM_zh.wikipedia.org_all-access_spider	11.0	14.0	15.0	18.0	11.0	13.0	22.0	11.0	...	30.0	52.0	45.0	26.0	20.0
3C_zh.wikipedia.org_all-access_spider	1.0	0.0	1.0	1.0	0.0	4.0	0.0	3.0	...	4.0	6.0	3.0	4.0	17.0
4minute_zh.wikipedia.org_all-access_spider	35.0	13.0	10.0	94.0	4.0	26.0	14.0	9.0	...	11.0	17.0	19.0	10.0	11.0
52_Hz_I_Love_You_zh.wikipedia.org_all-access_s...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	11.0	27.0	13.0	36.0	10.0

Fig 1. Dataset showing time series data of wiki pages with daily views

For each time series, there is a name of the article and the type of traffic that the time series represents (all, mobile, desktop, spider).

Note that the data source for this dataset does not distinguish between traffic values of zero and missing values. A missing value may mean the traffic was zero or that the data is not available for that day. There were no missing values found in the data set however there were many pages with zero views.

There is one column for page name called "Page" and the other columns are for dates with number of views as values.

The format for column "Page" is '<name>_<project>_<access>_<agent>' (e.g. 'AKB48_zh.wikipedia.org_all-access_spider'). Each of these are explained below:

- **name** is the name of the wikipedia page;
- **project** is the Wikipedia project (e.g. en.wikipedia.org). The different projects are:
 - wikipedia
 - wikimedia
 - mediawiki

The project also contains the language of the Wikipedia page which can be extracted.
The languages are:

- English (en)
 - Spanish (es)
 - Japanese (ja)
 - Russian (ru)
 - Dutch (de)
 - French (fr)
 - Mandarin (zh)
 - www/commons – multi language pages;
- **access** is the type of access : desktop, mobile or all-access
 - **agent** is the type of agent: all-agent, spider

Train_1.csv - contains traffic data. This a csv file where each row corresponds to a particular article and each column correspond to a particular date. Some entries are missing data.

processed.csv – contains data from train_1.csv with extra features created from the column "Page".

Exploratory Visualization

Data by Category

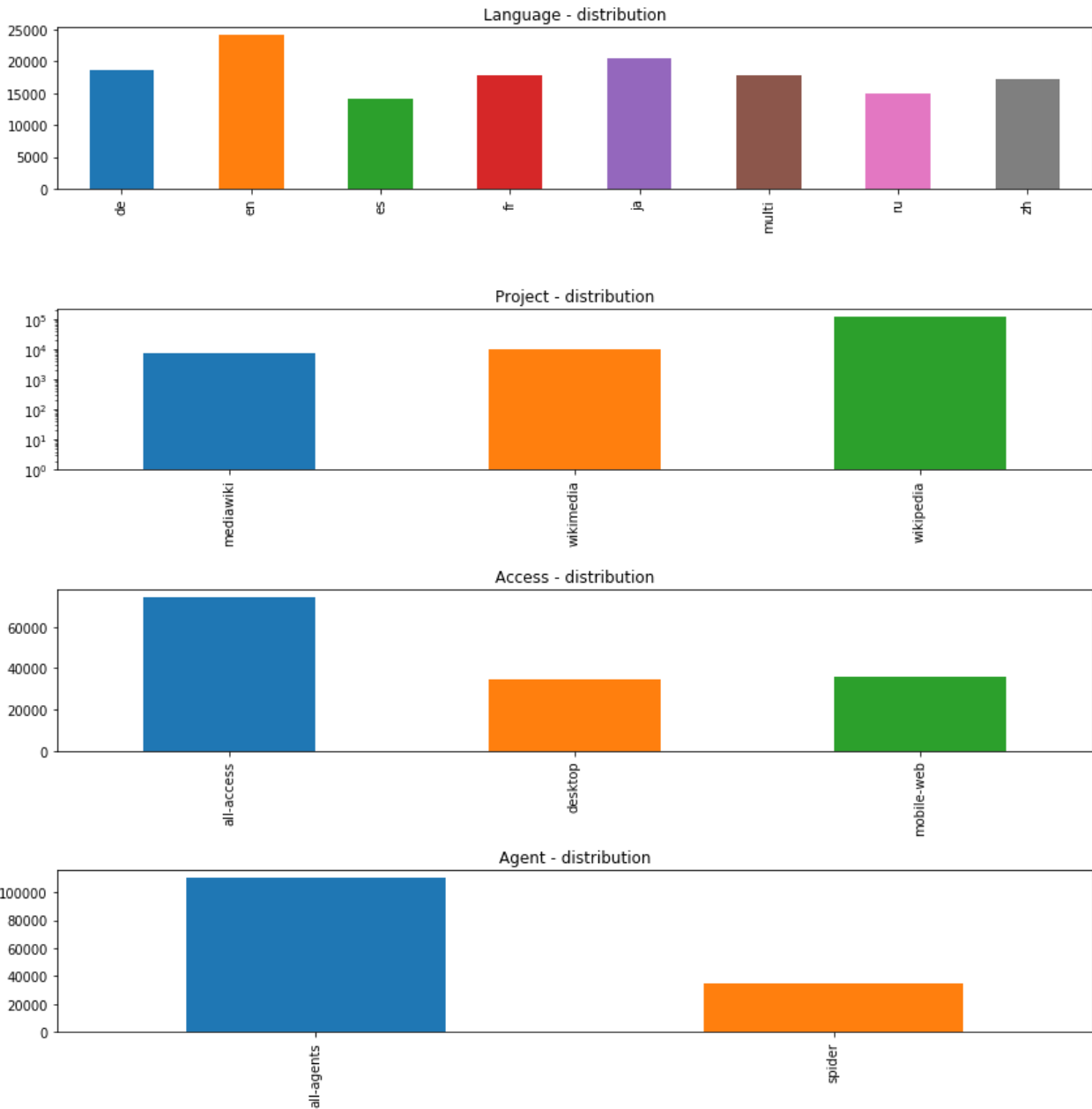
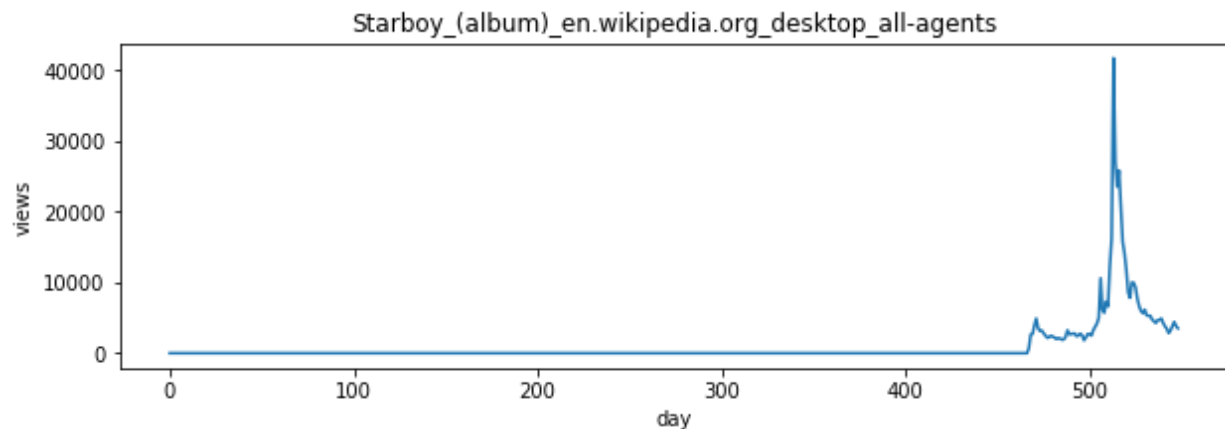
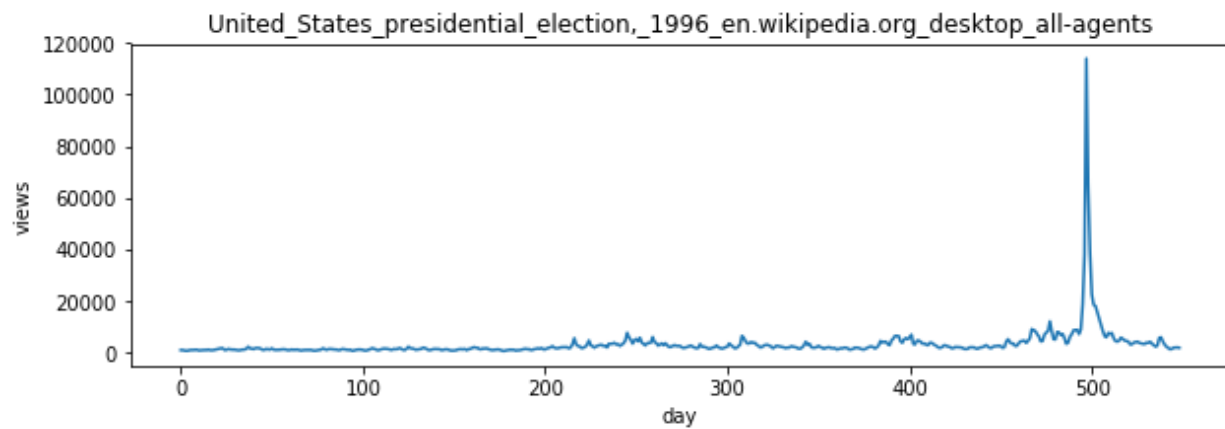


Fig 2. Count of data points by different categories

The above figure shows the count of data points by language, project, access and agent using bar plot. This is based count (y-axis) of data points and not actual views. The x-axis is different categories for each feature. Some of the key points to note are:

- There are more pages in English, Dutch and Japanese than other languages.
- For project, the bar plot is log scaled in y axis. There are 10 times for wikipedia than mediawiki and wikimedia.
- There are more data points under "all-access" but contribution from "desktop" and "mobile-web" is significant.
- There are about 3 times more data points under "all-agents" than "spider".

Page views by time



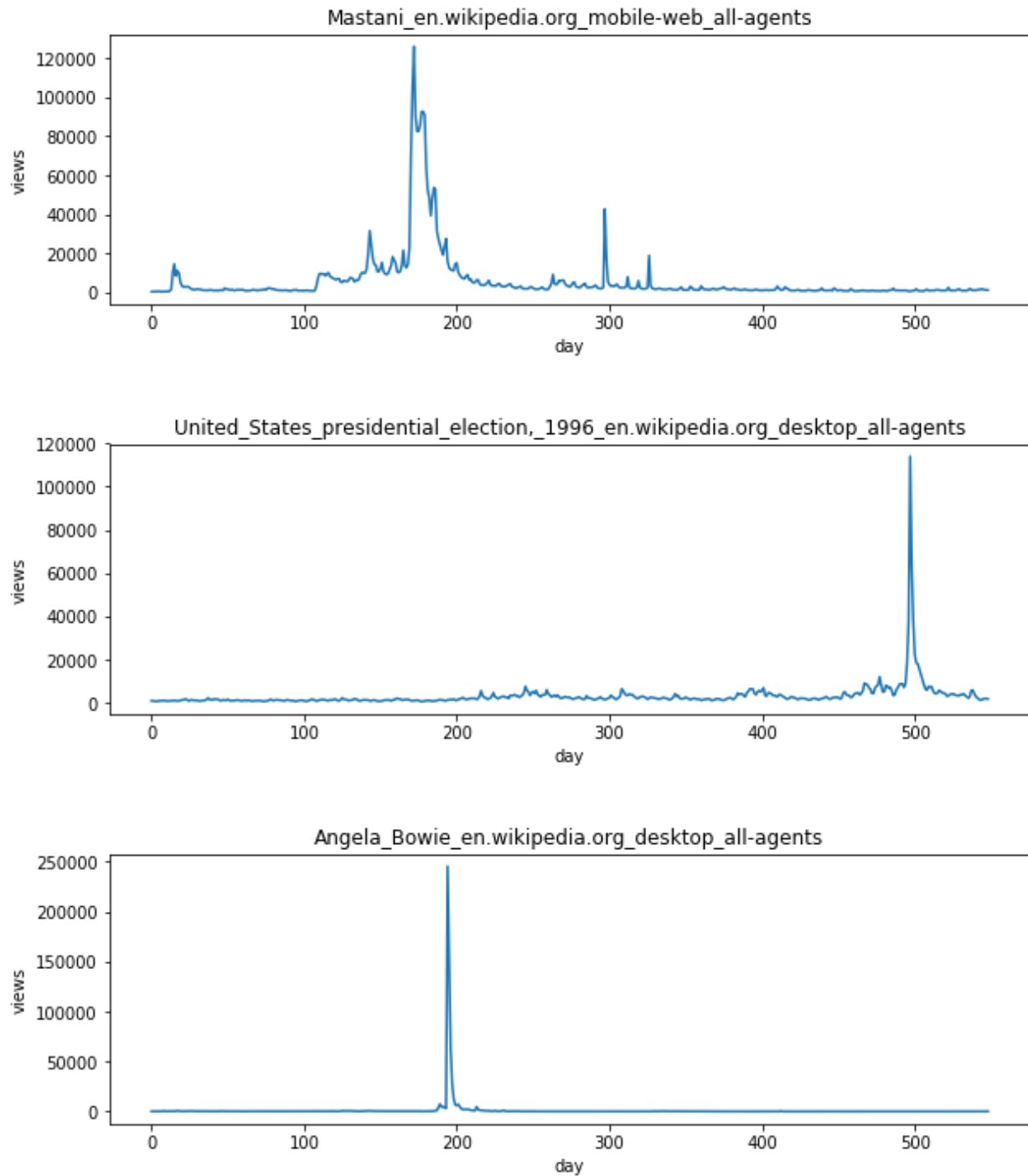


Fig 3. Page views for some English wiki pages

In fig 3, the views on wiki pages seem very distinct without any distinguishing pattern. Some things to note are as follows:

- Most pages have multiple spikes in views indicating sudden interest due to current events.

- Some pages have zero views in the beginning, indicating that they might have been created within that range of dates (for e.g. Starboy).
- Max views for some pages is >100,000 (e.g. US presidential election 1996) while others <1000.
- There are some pages (e.g. Angela Bowie) which has >100,000 for one day while almost zero views for most other days.

The main takeaway from these plots is that the views on each page is different with unique pattern in spikes and a simple technique such as predicting the mean of views for the provided time period may not be a good strategy to forecast views.

Average views per day

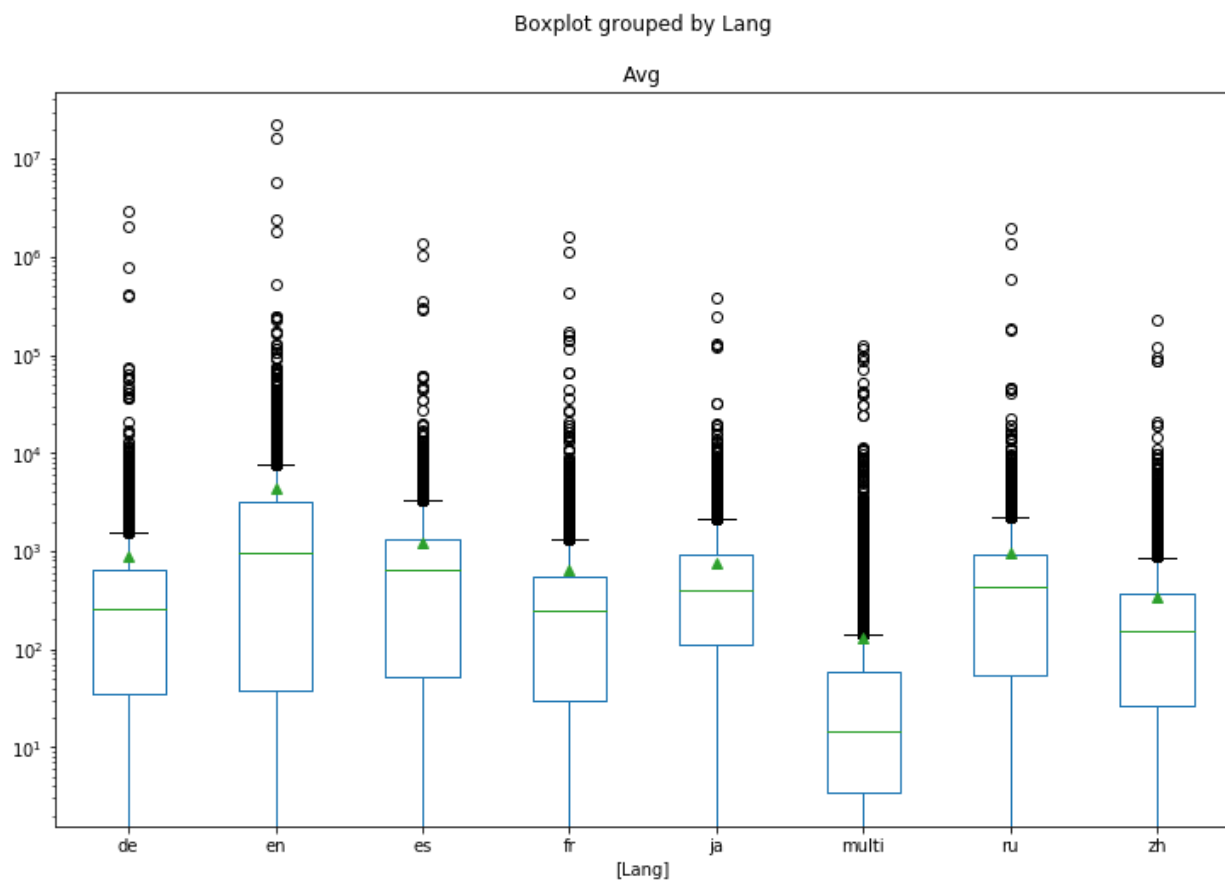


Fig 4. Boxplot showing average number of views for each language and its distribution (the y-axis is logarithmic)

Fig 4 shows a boxplot with the average number of views for each language and its distribution (logarithmic y-scale). The following are some key observations from the boxplot in Fig 4:

- English wiki pages have the maximum average view (per day)
- Wiki pages with mutple languages have the lowest average view
- There seem to be many outliers for each language, for forecasting it might be be prudent to pick web pages which have high average views to avoid pages which are not that popular.

Views by day and language

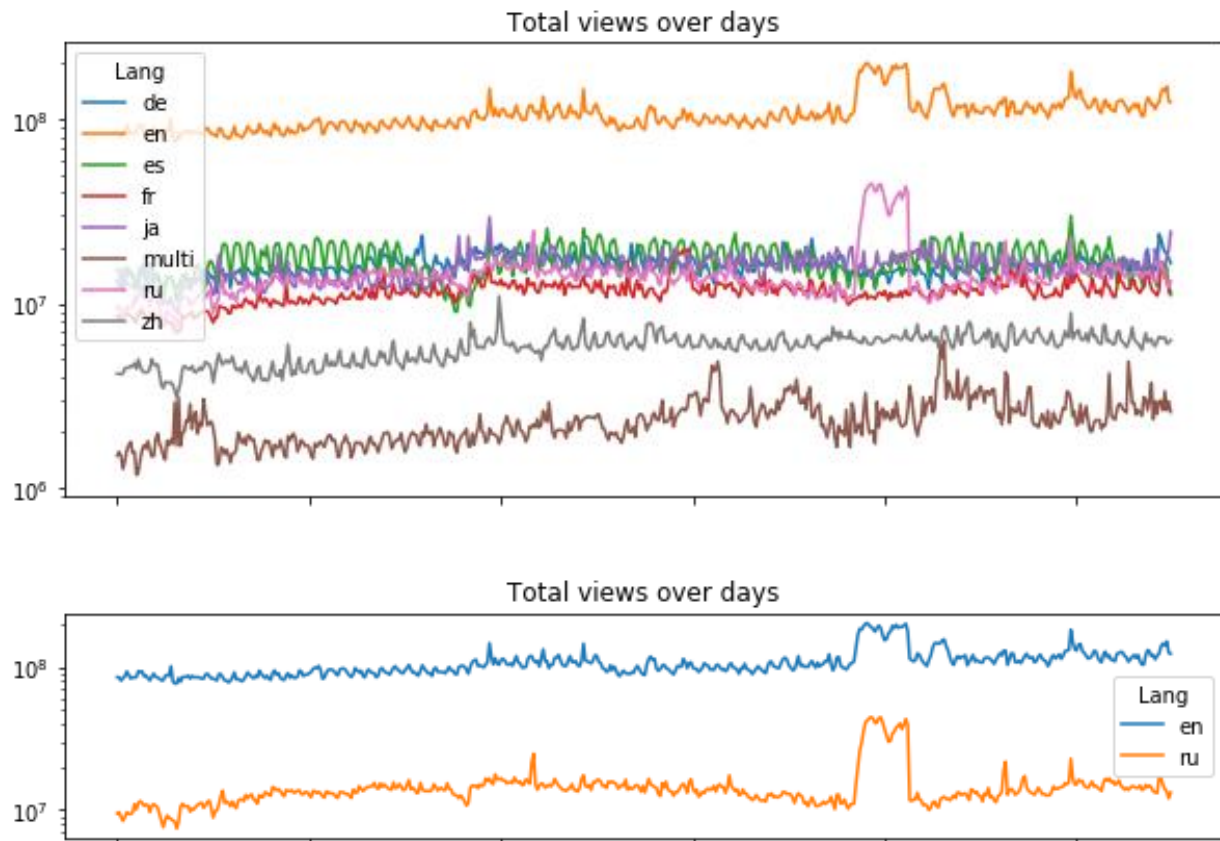


Fig 6. Graph showing web traffic by day and language (logarithmic y-axis)

Fig 6 shows that the english wiki pages get more than 10 times views than other languages. It seems that the russian and english websites have an increase in views at the same.

Algorithms and Techniques

Components of Time series data

There are two types of components in time series data: systematic and non-systematic. Systematic data is consistent, recurrent and can be described and model while non-systematic data cannot be modeled directly.

The components of time series data are as follows:

1. **Level:** Average value in the series
2. **Trend:** Increasing and decreasing value in the series
3. **Seasonality:** repeating short-term cycle in the series
4. Noise: random variation in the series.

While level, trend and seasonality are the systematic components. Noise is a non-systematic component of time series data.

There are three types of models available to decompose time series data:

1. **Additive model:** $y(t) = \text{Level} + \text{Trend} + \text{Seasonality} + \text{Noise}$
2. **Multiplicative model:** $y(t) = \text{Level} * \text{Trend} * \text{Seasonality} * \text{Noise}$
3. **Pseudo-Additive model:** $y(t) = \text{Trend} * (\text{Seasonality} + \text{Noise} - 1)$

The trend can also be "damped" or "exponential" as shown in Fig 7 below.






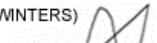






	Nonseasonal	Additive Seasonal	Multiplicative Seasonal
Constant Level (SIMPLE)	 NN	 NA	 NM
Linear Trend (HOLT)	 LN	 LA	 LM
Damped Trend (0.95)	 DN	 DA	 DM
Exponential Trend (1.05)	 EN	 EA	 EM

Fig 7. Models for decomposing time-series data

Here we use the "seasonal_decompose" function from the statsmodel library and the additive model to visualize total daily views of spanish (left) and english (right) wiki pages:

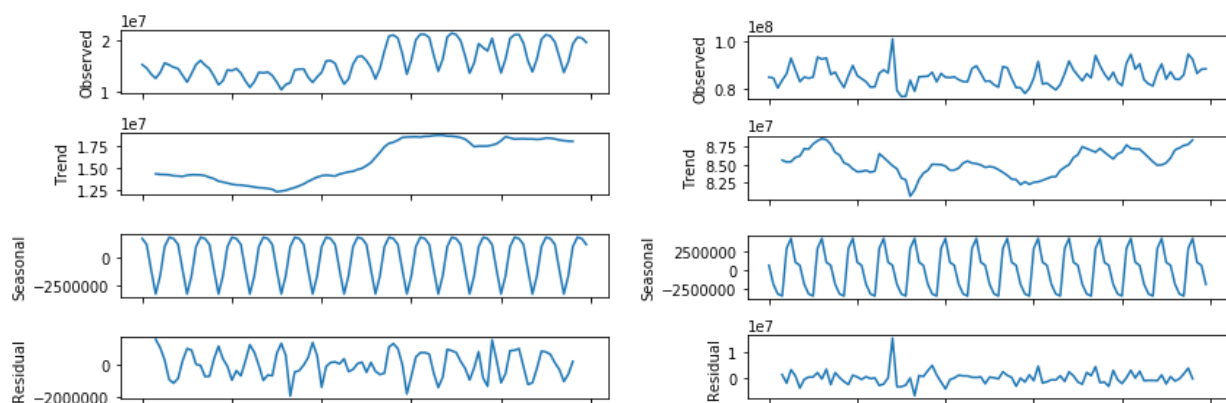


Fig 8. Graph showing seasonality and trend in web traffic for Spanish (left) and English (right) wiki pages.

The above plot show that the seasonality of the data has a frequency of 7 days. Note that frequency is an input for the decompose function. A frequency of 7 was picked because it seem to provide the smoothest curve for the trend. This could be because of lower/higher views during weekend.

Exponential Smoothing

Exponential Smoothing methods predict future values based on the weighted average of current value and previous values:

[Single exponential smoothing](#) [1] used the weighted average of current value and previous weighted values:

$$\hat{y}_{t+1} = \alpha y_t + (1-\alpha) \hat{y}_{t-1}$$

where α is the learning rate. This is a simple method but only predicts one value at a time. It is also not good at capturing trend and seasonality.

[Double exponential smoothing](#) takes into account the level as well as trend by introducing another equation which takes into account the trend in data. Double exponential smoothing however does not take into account seasonality. Seasonality is taken into account in a similar fashion in [Triple exponential smoothing](#) or Holt Winter's method by introducing another equation.

The basic equations for this method are given by:

$$S_t = \alpha \frac{y_t}{I_{t-L}} + (1 - \alpha)(S_{t-1} + b_{t-1}) \quad \text{OVERALL SMOOTHING}$$

$$b_t = \gamma(S_t - S_{t-1}) + (1 - \gamma)b_{t-1} \quad \text{TREND SMOOTHING}$$

$$I_t = \beta \frac{y_t}{S_t} + (1 - \beta)I_{t-L} \quad \text{SEASONAL SMOOTHING}$$

$$F_{t+m} = (S_t + mb_t)I_{t-L+m} \quad \text{FORECAST,}$$

where, y is the observation; S is the smoothed observation; b is the trend factor; I is the seasonal index; F is the forecast at m periods ahead and t is an index denoting a time period.

Benchmark

A simple model for prediction can be using the views in last n days as the prediction for next n days. For e.g. if a page has had 5, 10, 20 views on one each of the last 3 days, predict that it will get 5, 10, 20 views in the next 3 days. This simple model will be used

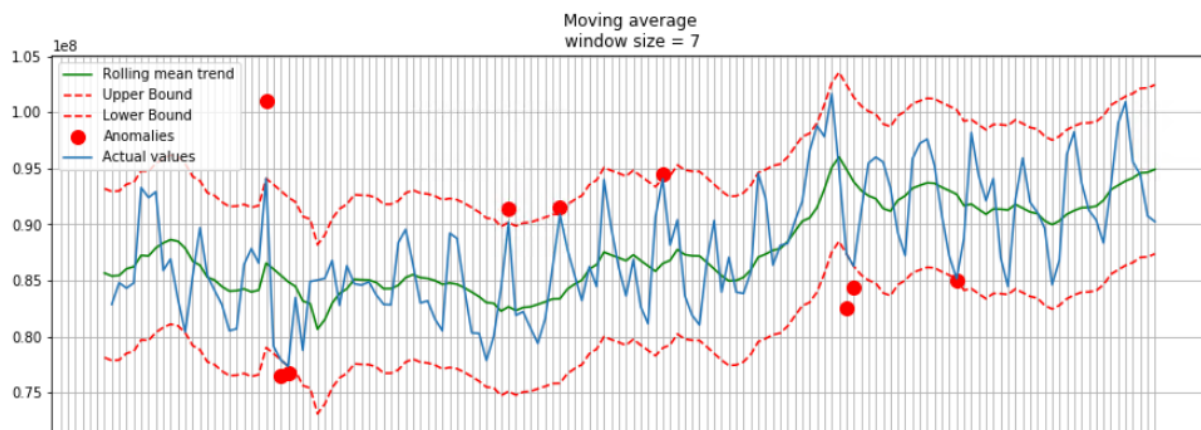
for benchmarking and the prediction from this “simple” model will be compared with the prediction from this project.

This may be a crude model on the lines of today will be just like yesterday, but it is very effective in determining what value our model is bringing. If our model is an improvement on the “simple” model by only a small margin, our model does not provide as much value.

III. Methodology

Data Preprocessing

The dataset does not have any missing values. However the time series has many outliers that may need to be replaced with more nominal values. We need to remove outliers from the time series because they result in artificial trends in data which in our case is usually because of some current events causing huge number of people to view a wiki page. For example, while exploring the dataset I found a wiki article that had <100 views on all days except one when it had 100,000+ hits. On further investigation, I found that the wiki page was that of an obscure pop artist, who had been caught in a drug related police investigation on that day resulting in huge traffic to his wiki page. It's unrealistic to expect the model to predict such anomalies as they may not repeat on a periodic basis. It is best to remove such data points in order to get more prediction for web traffic.



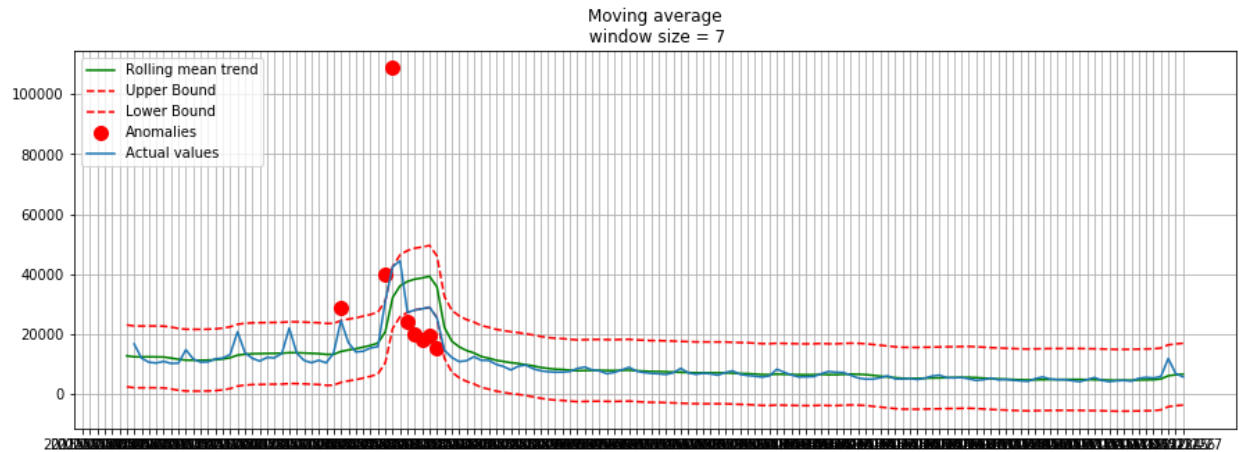


Fig 9. Time series plot shows total daily views of all the English wiki pages (top) and “Pretty Little Liars” (bottom) with outliers replaced by upper/lower bounds.

In order to determine the outliers, an upper and lower bound is defined and any value outside that band is considered an outlier. The bounds are a combination of mean absolute error and deviation of the data, both of which are calculated around the rolling mean with frequency 7 days. The outliers are replaced by the upper or lower bound.

Each time series is also normalized by rescaling the values between 0 and 1. The following figure shows data with outliers removed for total daily views of English wiki pages. The red dots are the outliers, the green line is the rolling mean average, the broken lines are upper and lower bounds while the blue line is the data with outliers removed.

Implementation

The preprocessing is carried out by passing a single time series into a function called “RemoveOutliers”. The function takes the timeseries, the window for rolling mean, scale as the main inputs. The output is a plot which shows the outliers and a time series with the outliers removed. It also shows the upper and lower bound. The function also rescales the data between 0 and 1.

The Holt-Winters’ method applied by creating another function called “HoltWinters_Prediction_function”. The function uses ExponentialSmoothing function from statsmodels for forecasting.

All three exponential smoothing methods (single, double and triple) can be tried with [holtwinters.ExponentialSmoothing](#) function in statsmodels library. However since the data has seasonality with a period of 7, we use triple exponential smoothing directly since

single and double exponential smoothing doesn't capture seasonality. The parameters for the functions are "trend", "damped", "seasonal" and "seasonal periods".

The following are the possible values of the parameters:

trend	<i>"add", "mul", "additive", "multiplicative", None</i>
damped	<i>True, False</i>
seasonal	<i>"add", "mul", "additive", "multiplicative", None</i>

There are four exponential smoothing models which are tried:

1. trend='add', seasonal='add'
2. trend='add', seasonal='mul'
3. trend='add', seasonal='add', damped=True
4. trend='add', seasonal='mul', damped=True

This function also outputs the prediction from benchmark model called the "simple model". The function also outputs a plot with the original values and the prediction from the five models.

The function takes time series data as input and breaks it into training and testing sets. The ratio of training to testing set is 3:1 i.e. it takes trains on 100 days of data and makes prediction on 50 days. The function then creates models by fitting the training data using the five methods mentioned above and a specified seasonal period (7 in our case). The function then makes predictions for 50 days using the model. An [example](#) from the statsmodels library is used to create this function.

The evaluation metrics were calculated and plotted by creating a function called HoltWinters_RMSE_Plot. The function takes as input the actual observations and predictions (one dataframe for five predictions) and outputs a matrix with RMSE for every week of prediction. It also a plots the RMSE for different methods over the weeks to provide a visualization of how the methods performed.

Refinement

Unfortunately this function does not allow for further fine tuning of the hyperparameters. However, the trend parameter was damped using "damped" = True in the original function for calculating predictions as means to refine the method.

IV. Results

Predictions for total daily views (by language)

Predictions were made for overall daily views of English, Spanish and Japanese wikipages.

a. English

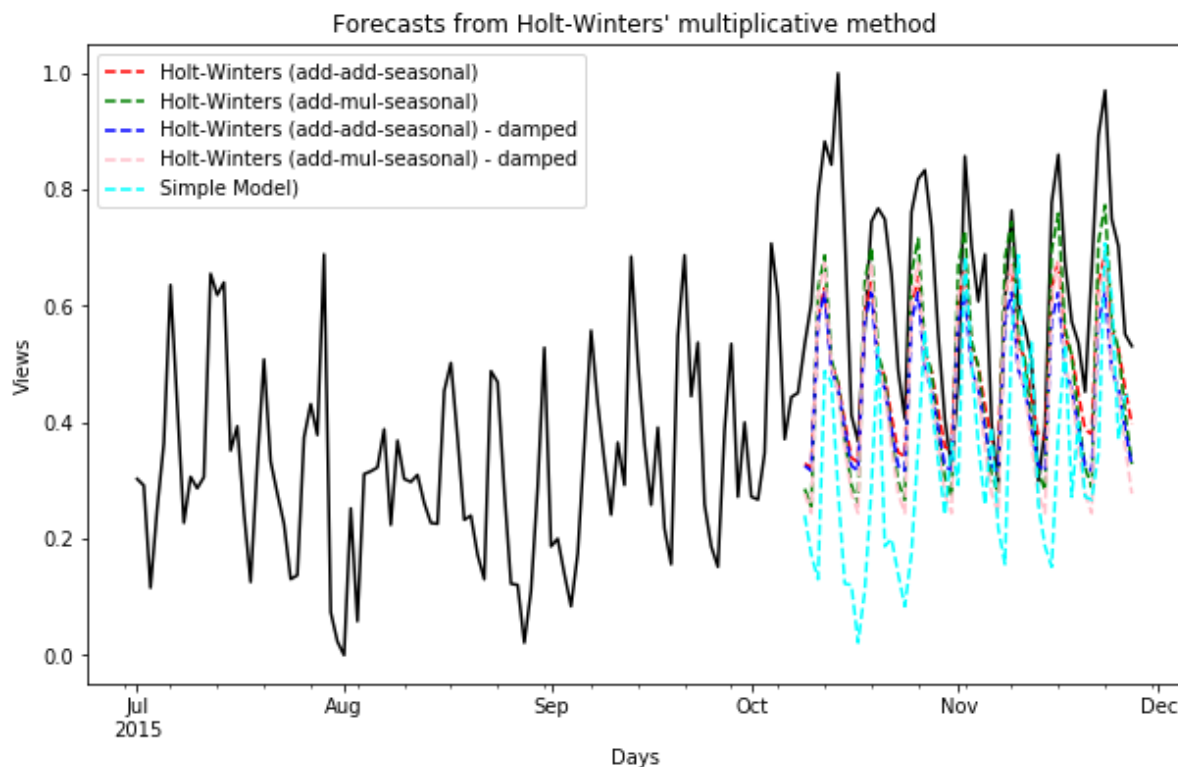


Fig 10. Forecasting total daily views of all English wiki pages

Fig 10 shows the prediction of daily views for English wiki pages. The solid black line shows the observed values while broken line shows the predictions using different options in Holt-Winters method. Each rise and drop is about a week.

Just visually, the prediction seem to follow the rise and fall in views very well. The figure shows that the predictions are poor in the beginning as the model is not able to predict the sudden jump in traffic in the second week of October. However, the prediction seem to improve as the traffic becomes slightly stable.

Here we calculate the RMSE of the predictions on a weekly basis and plot it.

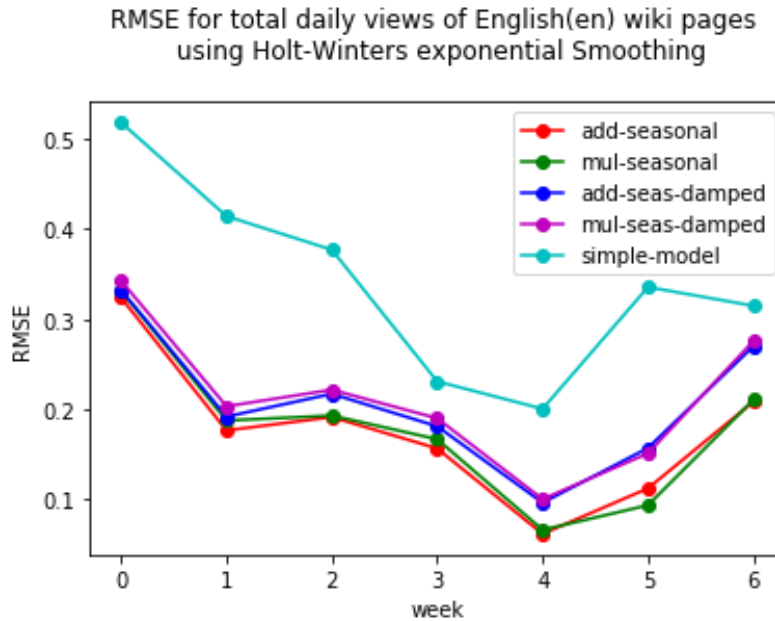


Fig 11. RMSE for total daily views of all English wiki pages

The RMSE error from the benchmark "simple" model (which predicts that next n values will be same as the last n values) is much higher than the Holt-Winter techniques. It is almost twice the error from other Holt-Winters' method which shows that our predictive models are providing a huge value.

There is significant improvement in RMSE week 2 onwards with RMSE dropping from ~ 0.3 to ~ 0.2 in second week and upto ~ 0.06 by fourth week

If we look at the average RMSE from different techniques, additive seasonal and multiplicative season give the lowest errors.

Now we will repeat this exercise with Spanish and Japanese wiki pages.

b. Spanish

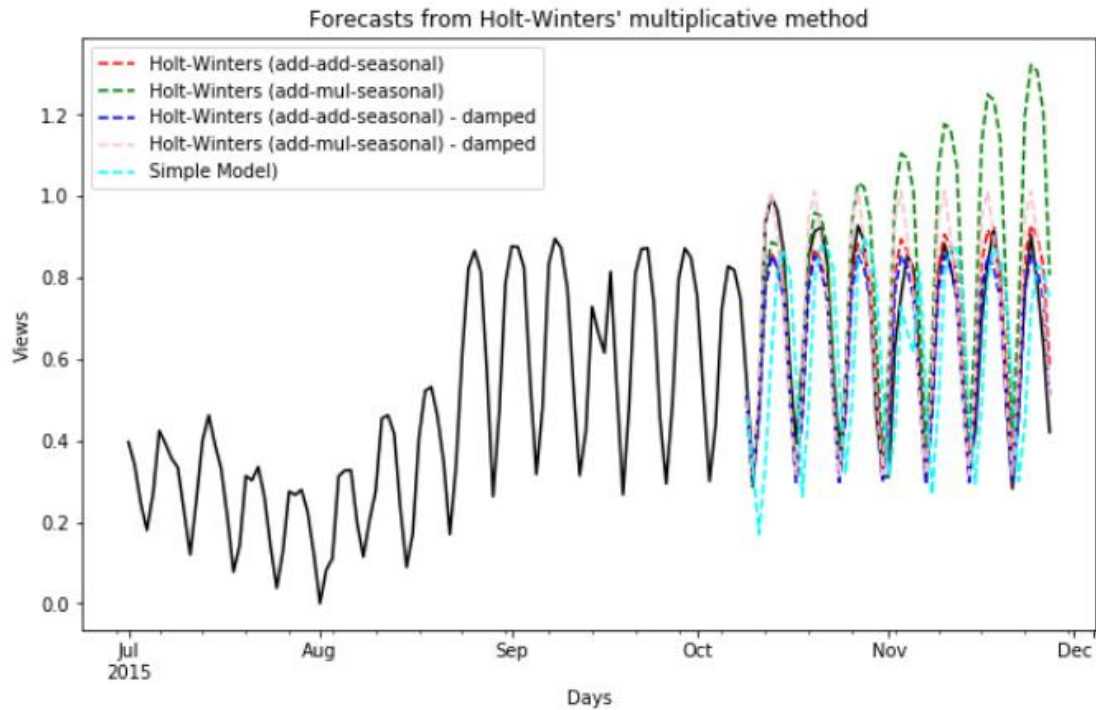


Fig 12. Forecasting total daily views of all Spanish wiki pages

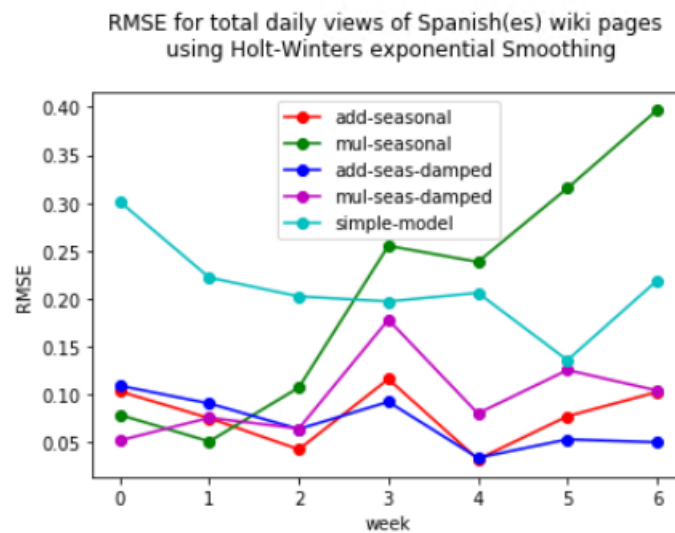


Fig 13. RMSE for total daily views of all Spanish wiki pages

The errors from the "simple mode" and multiplicative seasonal are much higher than the other Holt-Winter techniques. The errors from multiplicative seasonality seems to increase over week. However, the errors in all the weeks are pretty low at around ~0.1.

If we look at the average RMSE from different techniques, additive-seasonal and additive-seasonal-damped give the lowest errors.

c. Japanese

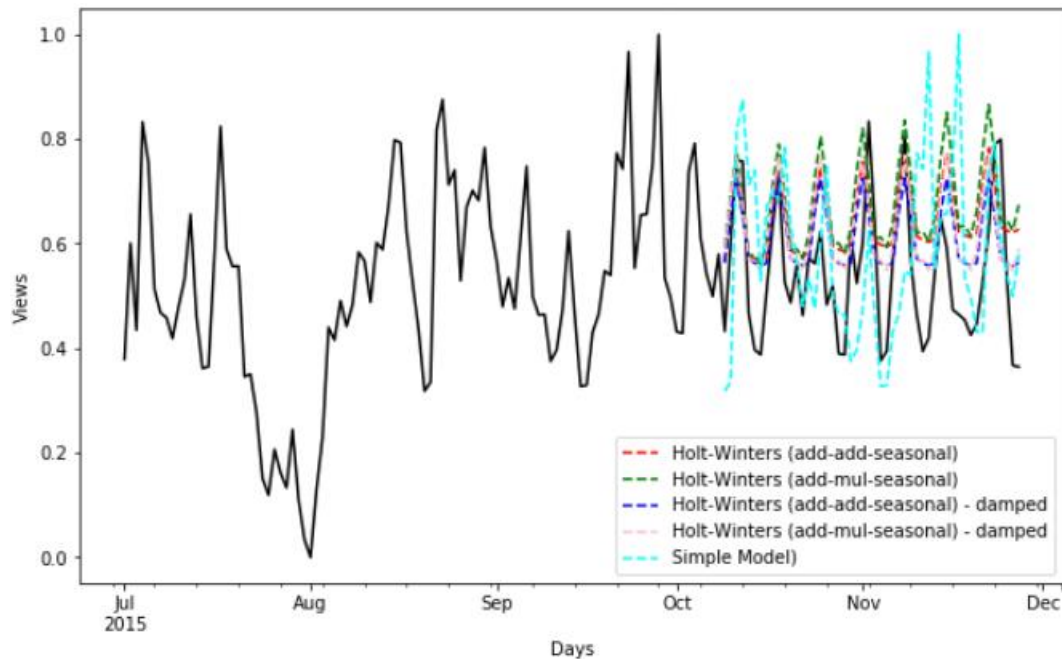


Fig 14. Forecasting total daily views of all Japanese wiki pages

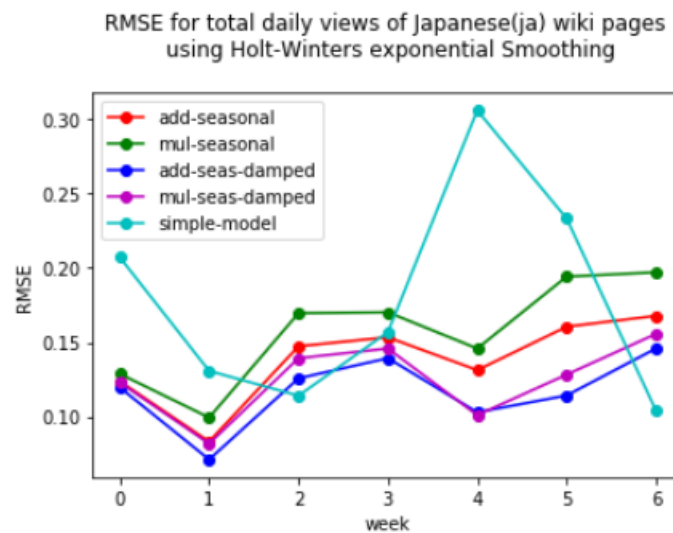


Fig 15. RMSE for total daily views of all Japanese wiki pages

The errors from the "simple mode" and multiplicative seasonal continue to be much higher than other Holt-Winter techniques. The errors in all the weeks are acceptable at

around ~ 0.12 . If we look at the average RMSE from different techniques, additive-seasonal-damped gives the lowest errors once again.

In conclusion, additive seasonal damping (trend='add', seasonal='add', damped=True) method works better than other techniques for Holt-Winters method based on the predictions for english, spanish and japanese wiki pages.

Predictions for daily views of individual pages

We randomly sample 100 wiki pages and organize them in order of the mean views. The idea is to make prediction for pages which have more number of views. Looking at the table, the english language wiki pages with highest average views are: *Inside Out* and *Pretty Little Liars*. The following shows results from two pages: *Inside Out* and *Pretty Little Liars*.

a. *Inside Out*

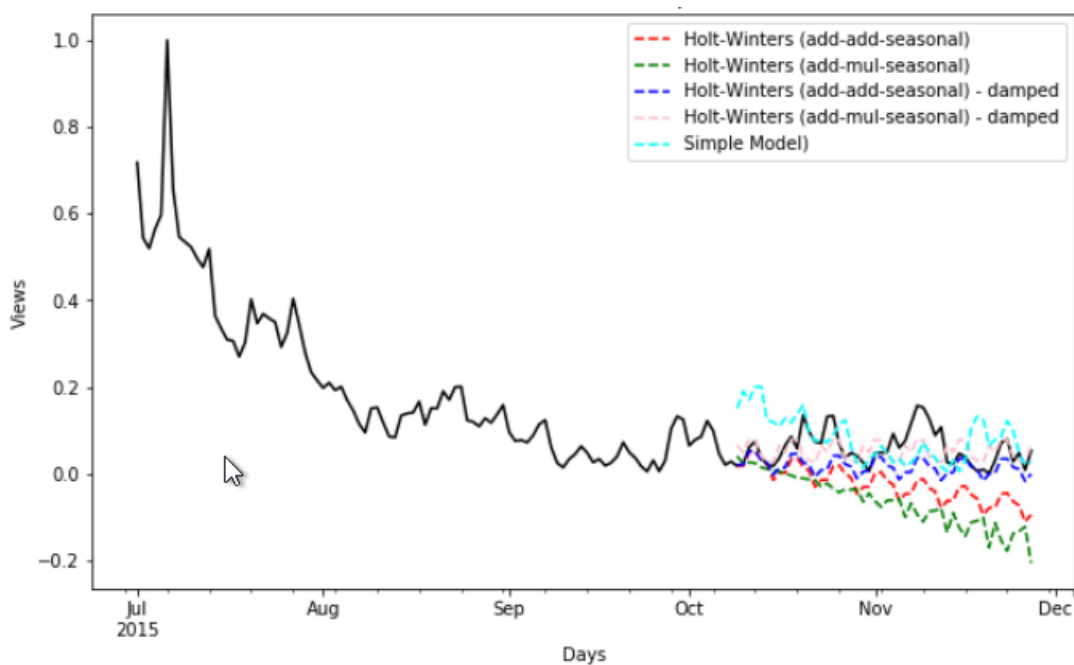


Fig 16. Forecasting total daily views for “Inside Out” wiki page

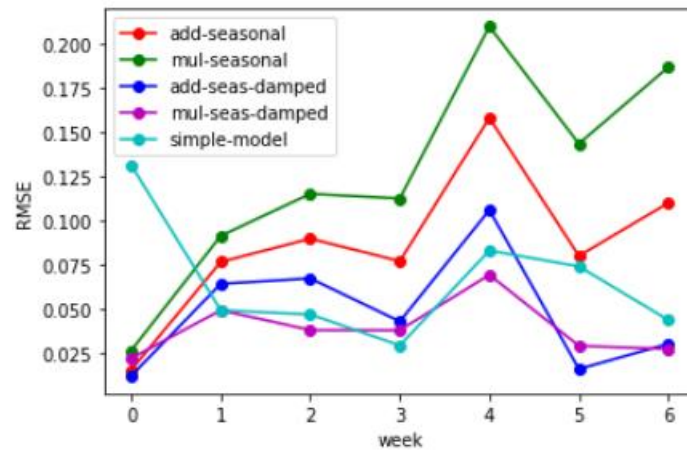


Fig 17. RMSE for total daily views of “Inside Out” wiki page

The average errors from the different techniques are as follows:

add-seas	0.086983
mul-seas	0.126704
add-seas-dam	0.048756
mul-seas-dam	0.039391
simple-model	0.065701

The minimum errors are from additive-seasonal-damped and multiplicative-seasonal-damped. The benchmark model performs reasonably well in this case.

b. Pretty Little Liars

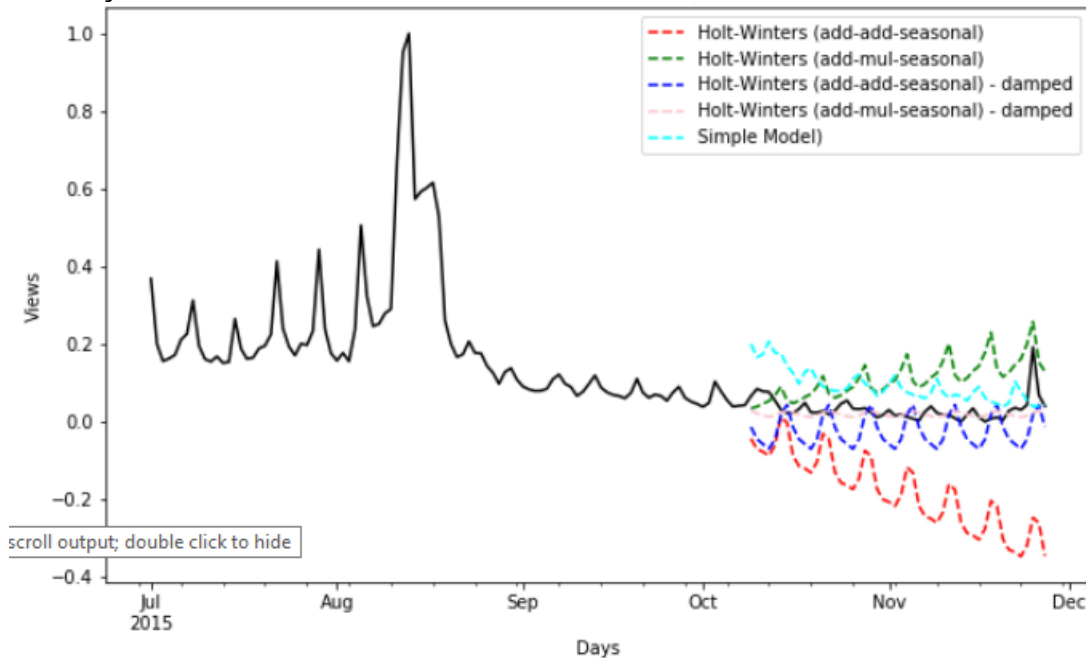


Fig 18. Forecasting total daily views for “Pretty Little Liars” wiki page

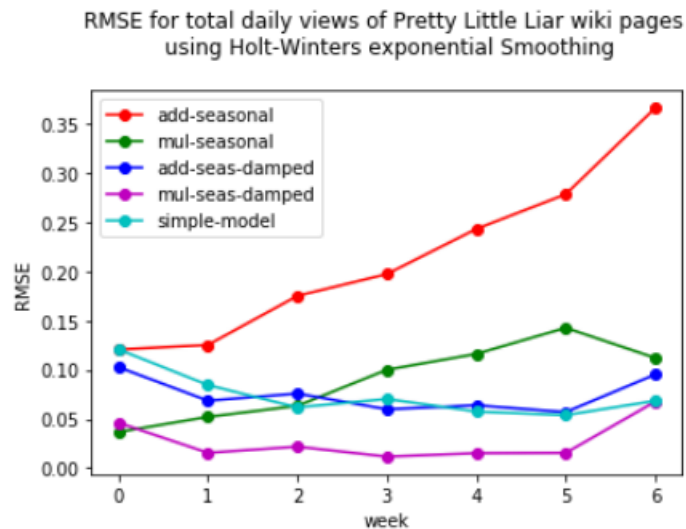


Fig 19. RMSE for total daily views of “Pretty Little Liars” wiki page

The average errors from the different techniques are as follows:

add-seas	0.214901
mul-seas	0.088909
add-seas-dam	0.074631
mul-seas-dam	0.027634
simple-model	0.073802

Additive-Seasonal-damped continues to give very low errors (<0.1) for the three wiki pages that we explored. We may conclude that it is technique of choice for most forecasting. It should be noted that in these examples, the time series did not across any sudden jump in values.

Our model with damping reduces the prediction by a huge margin especially in the case of overall daily views when compared to the benchmark model. The final solution using additive-seasonal-damping also gives errors much less than 0.25 which was our target. Therefore, the solution is significant enough to have solved the problem.

V. Conclusion

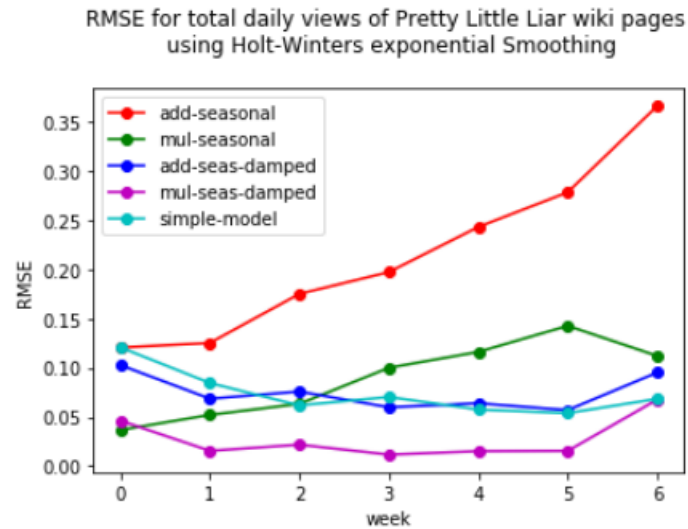


Fig 20. RMSE for total daily views of “Pretty Little Liars” wiki page

The above plot showing the RMSE for predictions for different methods over weeks is a very significant results because it shows how well different models are performing over time. It is important to know that the error sometime increases significantly over the weeks. So a model which may be performing well when forecasting for a week may not perform as well when the forecasting is extended to several weeks. A sudden jump in error may also be due to an anomaly in the data itself rather than bad model behavior. Therefore, it is important to look at the error on a granular level to rather than just look at the average to determine the best model.

To summarize, a model has been created to predict web traffic to Wikipedia pages over weeks which forecasts web traffic with root mean square error less than 20%. The predictions are for overall traffic to wiki pages in different languages and also for individual pages. The original data needed some clean up as it has some anomalous data point. Once the outliers were removed from the time series, it was broken down into training and testing and Holt-Winters’ exponential method was used to make predictions. Different combination on trend, seasonality and damping was used make prediction and we found that the combination with additive trend, additive seasonal and damping performed the best. It was also compared to a benchmark predictive model to show that the results were indeed significant.

Deciding which method to use for solving this problem was a difficult and critical one because there are many techniques available which are popular for e.g. ARIMA, SARIMA, XGBoost, LightGBM, LSTM and libraries such as Facebook’s prophet. In the end, it was a

good decision to go with Holt Winters' Exponential smoothing as it provides a good groundwork for understanding other time series analysis techniques and is easier to implement than some other techniques such as ARIMA.

The other challenge was to be able to visualize these results in concise manner to get a good overview how the models are performing. The plots for actual predictions and the errors painted a good picture of how the models perform.

One shortcoming of this implementation was that the models never came across an anomaly during prediction. For example, the passing away of a public figure would create a spike in web traffic to their wiki pages but with no way for the model to account for that its predictions will highly erroneous. In future, I would like to explore more of that and experiment with other available techniques to see how they may handle such a scenario. The Kaggle completion from which this project is inspired has the top winner announced. So looking at their solution may be a good way to get started with exploring some of the best performing techniques.