PROJECT ABSTRACT

Website Traffic Forecasting using ARIMA

A complete ARIMA time-series forecasting pipeline on Wikipedia pageview data: ADF stationarity testing, AIC order selection, OLS fitting, forecasts with 95% confidence intervals, and holdout evaluation against naive baselines — plus a single-file browser demo that runs the whole pipeline live. Delivered built-to-order with code, notebook, report, PPT and viva kit.

01. Title

Website Traffic Forecasting using ARIMA

02. Introduction / Background

Every website operator faces one recurring question: how much traffic is coming next week? The answer drives server provisioning, advertising rates and content planning. Web traffic is a time series with trend, weekly seasonality and noise tangled together, and naive heuristics — yesterday's number, last week's average — throw away the autocorrelation structure that actually carries the signal. ARIMA (Autoregressive Integrated Moving Average) is the classical statistical answer: it learns how the series correlates with its own past, differences away trend to reach stationarity, and projects forward with principled uncertainty bands. This project builds that pipeline end to end on the real Kaggle Web Traffic Time Series Forecasting dataset — about 145,000 daily Wikipedia pageview series from July 2015 to September 2017 — and demonstrates every step in a demo app that performs the actual computations in the browser.

03. Problem Statement

Capacity planning, ad-revenue forecasting and content scheduling all depend on knowing future traffic, yet small publishers plan from gut feel or from naive moving averages that ignore the series' own dynamics. Trend plus weekly seasonality plus irregular fluctuations make web traffic hard to eyeball: a dip might be a dying page or just a weekend. The project addresses this by building a statistically grounded forecaster — stationarity testing to handle trend, AIC-based order selection to pick model complexity, and confidence intervals that quantify uncertainty — and by evaluating it honestly against naive and seasonal-naive baselines on a held-out period, so the report shows where the model wins and where it does not.

04. Objectives

- Build a data pipeline that loads daily Wikipedia pageview series from the Kaggle Web Traffic dataset and handles missing values per a documented protocol.

- Implement the Augmented Dickey-Fuller stationarity test to determine the differencing order d from the data, reporting the test statistic, p-value and critical values.
- Grid-search ARIMA(p,d,q) orders scored by the Akaike Information Criterion and select the lowest-AIC model automatically.
- Fit AR and MA coefficients with ordinary least squares on the differenced series, implemented from scratch rather than treated as a black box.
- Generate multi-step-ahead forecasts with analytical 95% confidence intervals.
- Evaluate on a trailing holdout window with RMSE, MAE and MAPE against naive and seasonal-naive baselines, and deliver a single-file browser demo that runs the whole pipeline live.

05. Proposed Methodology

The pipeline follows the Box-Jenkins methodology. (1) Data preparation: daily pageview series are loaded with pandas; missing values are treated per the documented protocol. (2) Identification: the ADF test checks for a unit root; the series is differenced until the test rejects non-stationarity, fixing d. Autocorrelation and partial-autocorrelation plots guide the (p,q) search space. (3) Estimation: each candidate ARIMA(p,d,q) is fitted by ordinary least squares on the differenced series and scored by AIC; the minimum-AIC model is selected. (4) Diagnostics: residuals are checked for remaining autocorrelation. (5) Forecasting and evaluation: the fitted model forecasts a trailing holdout window, forecast-error variance yields 95% intervals, and RMSE/MAE/MAPE are compared with naive and seasonal-naive baselines. A parallel JavaScript implementation reproduces the same steps in the browser demo.

06. System Architecture / Working

In the Python pipeline, a series flows from CSV loading through missing-value handling, the ADF test, differencing, the AIC grid search, OLS fitting and finally the forecaster, which emits the point forecast plus the confidence band; the evaluator then scores the holdout window against the two baselines and renders the forecast chart, the AIC heatmap and residual diagnostics. In the browser demo, the same stages run in JavaScript on a built-in realistic traffic series: the user picks a page, the app runs ADF, grids orders by AIC, fits, forecasts and plots the series with the forecast horizon and confidence band, plus a holdout-region overlay so the evaluation can be inspected visually. Both paths keep training (model selection/fitting) and evaluation (holdout scoring) strictly separated.

07. Hardware / Software Requirements

- Python 3.10 with statsmodels, pandas, NumPy, Matplotlib (pipeline and notebook)
- Jupyter notebook for the step-by-step pipeline walkthrough

- Kaggle Web Traffic Time Series Forecasting dataset (public download)
- Any modern desktop browser for the single-file HTML/CSS/JS demo app — no server, no installs, works offline
- No special hardware; the pipeline runs on an ordinary student laptop

08. Expected Outcomes

- Working pipeline: ADF test, AIC order selection, OLS ARIMA fitting, forecasting with 95% intervals.
- Holdout evaluation reporting RMSE, MAE and MAPE against naive and seasonal-naive baselines — numbers produced by the buyer's own build.
- Interactive charts: traffic with forecast band, AIC heatmap, residual diagnostics.
- Single-file browser demo running the entire pipeline live on realistic data.
- Complete documentation: report, PPT, methodology notes and viva Q&A.

09. Applications

- Capacity and ad-revenue planning demos for small publishers and blogs.
- Teaching platform for time-series statistics: stationarity, AIC, autocorrelation.
- Baseline forecaster against which neural time-series models can be compared.
- Reference build for other univariate forecasting problems (sales, footfall, demand).

10. Future Scope

- Seasonal ARIMA (SARIMA) to model weekly seasonality explicitly.
- Exogenous regressors (ARIMAX) such as holidays or marketing events.
- Probabilistic and multi-horizon evaluation with proper scoring rules.
- Comparison study with deep forecasters (LSTM, Temporal Fusion Transformer).

11. References

- Box, G. E. P., Jenkins, G. M., Reinsel, G. C. and Ljung, G. M. — Time Series Analysis: Forecasting and Control (5th ed., Wiley).
- Dickey, D. A. and Fuller, W. A. — Distribution of the Estimators for Autoregressive Time Series with a Unit Root (JASA, 1979).
- Kaggle — Web Traffic Time Series Forecasting dataset (Google, 2017).
- statsmodels documentation — ARIMA and stationarity testing.