ASSIGNMENT 5 – SPM (CS 587)

Pavitra Sai Vegiraju
A20525304

COMPARATIVE ANALYSIS REPORT

Forecasting tasks in GitHub repositories, like commits, pull requests, branches, and issues, were the main source of information. We reviewed data from five designated repositories covering the previous two months in order to create these estimates.

1. Openai-cookbook
2. ElasticSearch
3. Openai-python
4. Pymilvus
5. Angular-Google-Maps

We used three different approaches for the forecasting method:

- TensorFlow/Keras LSTM
- Facebook/Prophet
- StatsModel

TensorFlow/Keras LSTM:

For applications involving sequential data dependencies, including time-series forecasting, Long Short-Term Memory (LSTM) Networks, a type of Recurrent Neural Networks (RNNs), are widely utilized. Using the TensorFlow/Keras library has several benefits, the primary one being that it allows practitioners to use long short-term memory (LSTM) networks to model and predict complicated temporal patterns found in a variety of areas, including as natural language processing, finance, and weather forecasting. These models are able to accurately forecast future sequences because they are trained on historical data, which captures the dependencies over several time steps.

Facebook/Prophet:

Facebook's Core Data Science team has made their forecasting tool, Facebook Prophet, available to the public. It is intended to produce precise time-series forecasts, particularly in commercial contexts where data may exhibit numerous seasonality patterns or missing values. Prophet makes time-series data usable for consumers without extensive experience in time-series modeling by automatically managing its intricacies, such as holidays, patterns, and seasonality. Prophet has grown in popularity due to its ease of use, adaptability, and capacity to generate accurate forecasts with little effort.

Statsmodel:

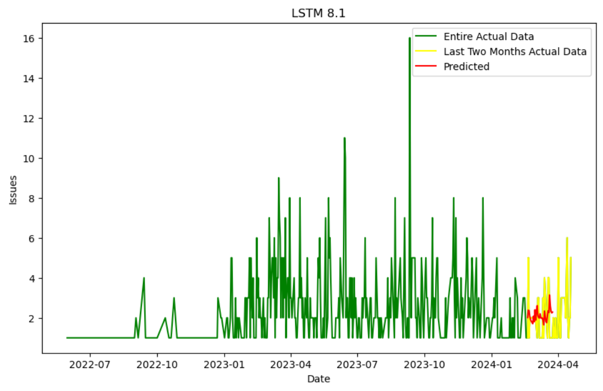
A Python package called Statsmodels is used to estimate and analyze different statistical models. It provides a large number of statistical methods, such as time-series analysis, generalized linear models, and linear regression. Performing hypothesis tests, examining data correlations, and producing statistical summaries are three areas in which Statsmodels excels.

LSTM, StatsModel and Prophet Forecasting

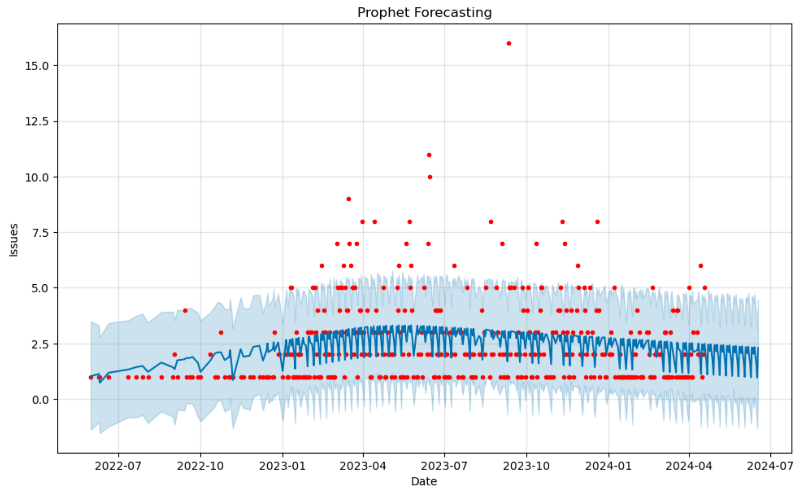
Requirement 8.1 & 9.1 & 10.1

8.1 & 9.1 & 10.1 - Repository: openai/openai-cookbook

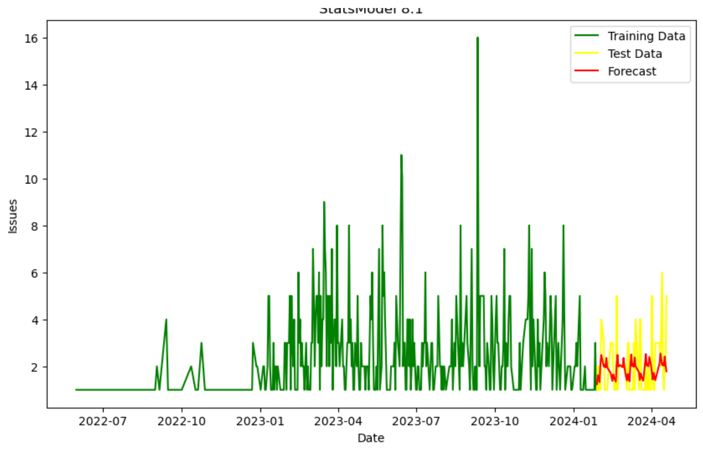
8.1 LSTM Forecasting



9.1 Prophet Forecasting



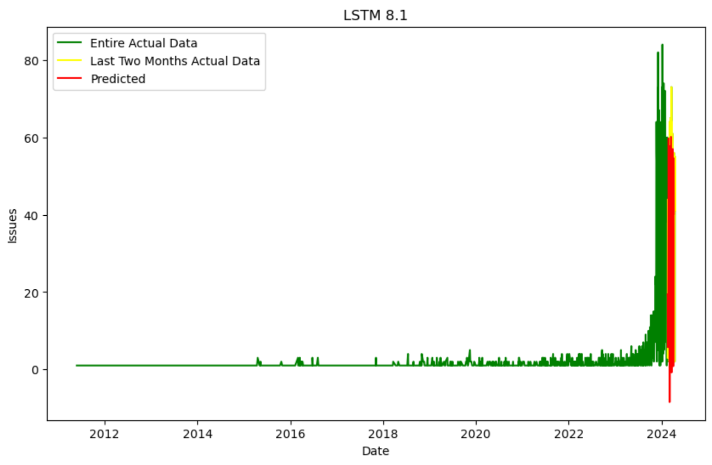
10.1 StatsModel Forecasting



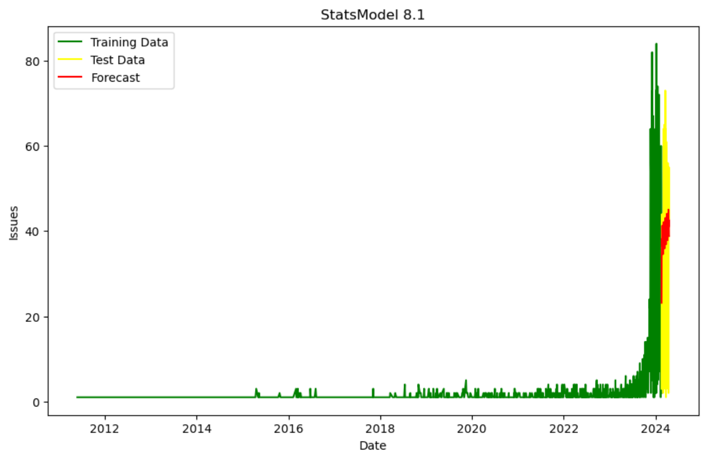
The above is the implementation of the all three models for forecasting the day of the weekmaximum number of issues created in the repo openAI- Cookbook.

8.1 & 9.1 & 10.1 - Repository: openai/openai-python

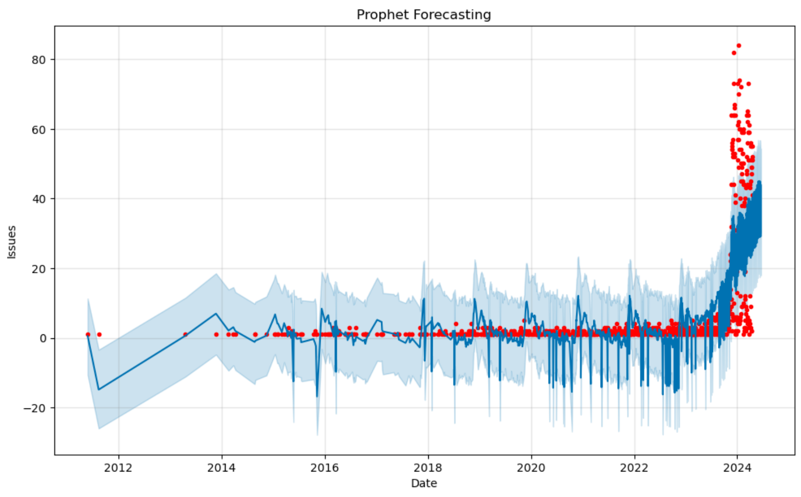
8.1 LSTM Forecasting



10.1 StatsModel Forecasting



9.1 Prophet Forecasting



The application of all three models for predicting the day of the week with the highest number of issues resolved for the openai/openai-python repository is shown above.

Nearly we have 30 graphs I have included all gcp cloud and URL link is below .

Analysis/Conclusion:

When we compared the three techniques' ability to anticipate future events, the Prophet library performed the best. This was quantified by using a metric known as "validation loss," which is essentially the degree to which our forecasts agreed with the actual results.

The statsmodels approach, which used ARIMA, performed poorly, on the other hand. When we took into account the validation loss we suffered along with the trends we were attempting to predict, it performed the worst.

To get even better results, we could tweak a few "hyperparameters" for each technique, which could help reduce prediction loss and increase accuracy.

Though there is a catch, we could improve and speed up these procedures. The data source, GitHub, has implemented policies known as API Rate Limits that restrict the frequency of information requests. This has an impact on how soon we can make predictions. In order to avoid asking GitHub for data too frequently and encountering issues, we had to incorporate timers into our code. It's similar to having to wait in line for your turn to receive the necessary information.

Hosted URL:

React-app: https://spmreactapp-5kzxwvbiya-uc.a.run.app

Flask App: https://flaskappspm-hij6xhfi2q-uc.a.run.app

Forecasting Flask App: https://forecastflaskapp-lwfb57gjeq-uc.a.run.app

Live demo - Panopto Video Recording:
https://iit.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=90ada299-b5a9-4ef6-a5ed-b15a0003c61d