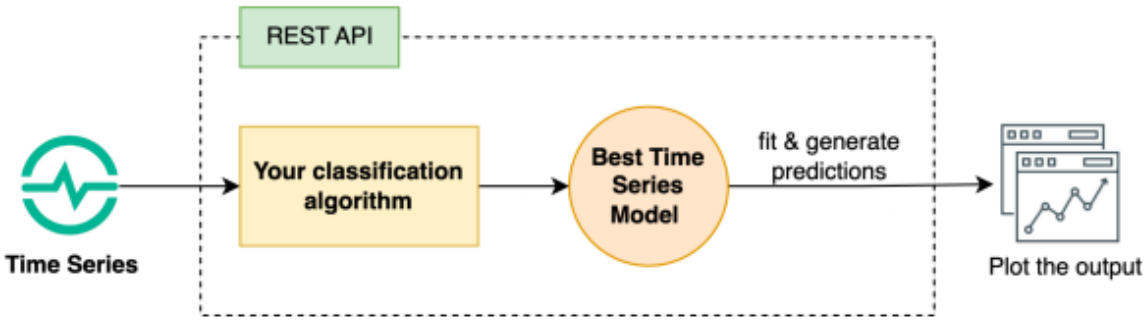
DATAGENIE HACKATHON
Time Series Classification
Mahitej K 20PD14

Problem Statement:

The goal is to create an efficient time series model selection algorithm



We need to create an efficient time series classification algorithm so that if we give a new time series data it needs to classify which model is better for the given time series model and we need to generate predictions with help of the best model.

Initial Thoughts about Problem:

Initially when I read the problem statement I understood that we need to build a classifier which chooses the best time series model for given time series data. So it will be a multiway classification problem. And I have to build different time series models for training data to check which model is better by using MAPE values.

Data Used for Training:

I have used the sample csv files which are given in the problem statement. The sample data contains different datasets which are daily, weekly, hourly, monthly. By using all these datasets we have to build different time series models.

What is the Time Series Data?

Time series data refers to a sequence of data points that are collected at regular intervals over time. The data points can be measurements, observations, or any other type of information that varies over time.

In univariate time series data contains one independent feature which is time and one dependent feature which depends on time and in multivariate there will be more than one independent variable.

Time series analysis involves using statistical and mathematical techniques to analyze and understand patterns and trends in time series data. This analysis can be used to make forecasts and identify underlying patterns and relationships.

Components of Time series:

- Trend
- Seasonal
- Cyclical
- Irregularity

By understanding these four components, time series analysts can develop models that help to explain the behavior of the data and make accurate forecasts.

Preprocessing Steps:

In the Preprocessing stage we need to check whether we have null values and we need to fill null values. As it is time series data we cannot directly fill with mean or mode so we need to use interpolation for filling null data. After interpolation if there is any null data left then we can drop them

One more thing is to change the index. We need to set the index as a timestamp and remove unnecessary columns. And also we need to change the datatype of index from object to datetime using pandas.

Model:

We need to build a time series classification model for which we need to create a dataset which contains different independent variables and one dependent variable which is output variable which has different classes.

For creating a dataset we need to extract some features from the time series data so that these can be used as features and the output variable is the best time series model for the given features.

Feature extraction:

For creating a dataset we need to extract features from time series data.

For each dataset in a sample time series we need to extract features and the features will be stored in a dataframe. For each sample we have one entry in a new dataset which is used for classification.

Features extracted:

- **Stationarity Check: 1-> data is stationary, 0-> data is not stationary**
- **Type of Data: 0->Daily, 1-> Monthly, 2-> Weekly, 3-> Hourly**
- **ACF Value:** found by using **lag = 2**
- **PACF Value:** found by using **lag = 2**
- **Mean and Standard deviation of dependent variable**
- **Minimum and maximum of the dependent variable**
- **Cosine mean and Sine mean**
- **Median**
- **Quantiles and RBF value**
- **Number of Holidays**

I have created a function called feature engineering where I will get values of all these features by calling this function for every dataset in sample Time series data.

Algorithms used for building different Time series models:

- ## ARIMA Model

  ARIMA (Autoregressive Integrated Moving Average) is a popular time series analysis technique used to model and forecast time series data. It combines three components: autoregression (AR), differencing (I), and moving average (MA).

  Autoregression refers to the use of past values of a variable to predict future values. The order of autoregression, denoted by p, represents the number of lagged values used in the model.Differencing is a method used to make a time series stationary by taking the difference between consecutive observations.Moving average refers to the use of past forecast errors to predict future values. The order of moving average, denoted by q.

  In this model we have built ARIMA Time series model for all sample time series data and calculator MAPE value to compare with other models.

  Hyperparameters in ARIMA are its parameters(p,d,q). We need to tune them to get the best order possible for a given data.Normally we find the order of AR and MA using PACF and ACF plot respectively.But as it is selected without analyzing graphs I have used the GridSearch method. In this method first we take a random range for p,d,q and then get all possible combinations of (p,d,q). Then for every combination we need to run ARIMA and find the error. The combination which has less error has the best_parameters.

- ## SARIMAX Model

SARIMAX (Seasonal Autoregressive Integrated Moving Average with Exogenous Variables) is an extension of the ARIMA model that incorporates seasonal factors and exogenous variables

SARIMAX models are often written as SARIMAX(p, d, q)(P, D, Q, s), where the lowercase parameters (p, d, q) represent the non-seasonal ARIMA terms, and the uppercase parameters (P, D, Q) represent the seasonal ARIMA terms. The s parameter represents the number of time periods in a seasonal cycle.

This model is used because ARIMA cannot be built if we have seasonal terms in data so we go for this advanced model.

- ## XGBOOST Model

XGBoost (Extreme Gradient Boosting) is a popular machine learning algorithm that can be used for time series forecasting. It is a powerful ensemble model that uses decision trees to model the relationship between features and the target variable.

When using XGBoost for time series forecasting is to handle the temporal nature of the data. This can be done by including lagged values of the target variable and any relevant features as inputs to the model. But in our dataset there are no features so we need to create new features like day of year, day of week, week of year etc.

I have written a function which gives following features:

- Day of week
- Quarter
- Month
- Year
- Day of month
- Week of year etc

In the XGBoost model Hyperparameters are n_estimators, learning rate which need to be tuned.

Evaluation Metrics:

For evaluating the model we used the Mean Absolute Percentage Error (MAPE) which can be calculated by using sklearn.metrics.

MAPE is calculated by taking the average of the absolute percentage errors between the forecasted values and the actual values, expressed as a percentage.

MAPE = (1/n) * Σ(|(actual - forecast)/actual|) * 100

MAPE values range from 0% to 100%, with lower values indicating better accuracy. A MAPE of 0% indicates a perfect forecast, while a MAPE of 100% indicates that the forecast is completely off.

Till now we have just built these 3 time series models for each of the training dataset and found the MAPE values. Now for each dataset we need to find out which is the best time series model by choosing the model which has the least MAPE value. This will be the output variable for the new data we are creating to build a classification model.

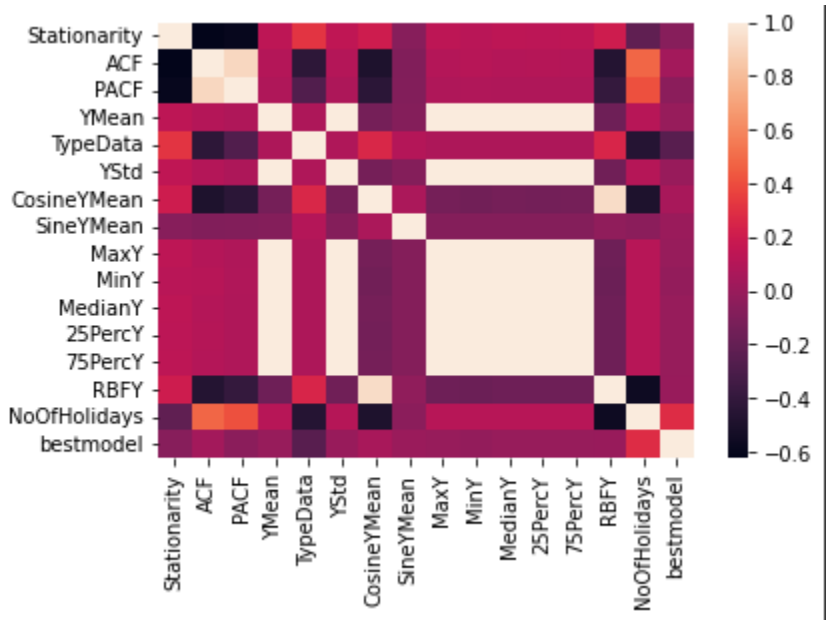
Now the dataset will look like:

```
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Stationarity   36 non-null     int64
 1   ACF            36 non-null     float64
 2   PACF           36 non-null     float64
 3   YMean          36 non-null     float64
 4   TypeData       36 non-null     int64
 5   YStd           36 non-null     float64
 6   CosineYMean    36 non-null     float64
 7   SineYMean      36 non-null     float64
 8   MaxY           36 non-null     float64
 9   MinY           36 non-null     float64
 10  MedianY        36 non-null     float64
 11  25PercY        36 non-null     float64
 12  75PercY        36 non-null     float64
 13  RBFY           36 non-null     float64
 14  NoOfHolidays   36 non-null     int64
 15  bestmodel      36 non-null     object
dtypes: float64(12), int64(3), object(1)
```

Where **#0 to #14** are features and the best model is the time series model which has least MAPE for that data.

In the new data we have **36 data points** because we trained only 36 datasets. For each data there will be one entry in the new dataset.Now we need to build a multi way classification model using these 36 data points.

Insights about new dataset:



This is the correlation between variables in newly created data

Classification Model:

Now after getting the data we need to build a multi way classifier so that if we want to predict which model is better for a time series data first we create features of those data and then give those features as input to this model so that it classifies which model is best for that time series data.

For classification I have used Random Forest regressor.Random forest regressor is a machine learning algorithm that can be used for regression tasks, such as predicting a continuous output variable. It is a type of ensemble model that combines multiple decision trees to make a prediction.

The basic idea behind a random forest regressor is to build multiple decision trees on randomly selected subsets of the data and features. Each tree is trained on a different subset of the data and features, and then the outputs of the individual trees are averaged to make a final prediction. This approach helps to reduce overfitting and improve the accuracy of the model.

Now Model building is completed, I have to create a simple REST API which allows the user to upload any time series data and get predictions for it along with the details of which time series model was used and MAPE value. For creating this REST API I have used FastAPI.

In FastAPI the User can make a request like asking forecast for a particular date then It gives best model, MAPE value and prediction as output

Backend:

FastAPI:

To build an API that accepts CSV input and returns a result, you can use FastAPI's File and ‘UploadFile’ classes to handle the file upload.Once the file is uploaded, we read it using pandas and perform some processing on the dataframe.

When a user makes a POST request to this endpoint and uploads a file, the function returns a JSON object with the name of the uploaded file as the value of the filename key.

After Reading the data we are converting it into result.csv to use for calculations and the data is stored in local storage.

UI:

After Creating the backend I have created a basic UI so that users can give input and can get results as output.

I created a UI using basic html which is functional.

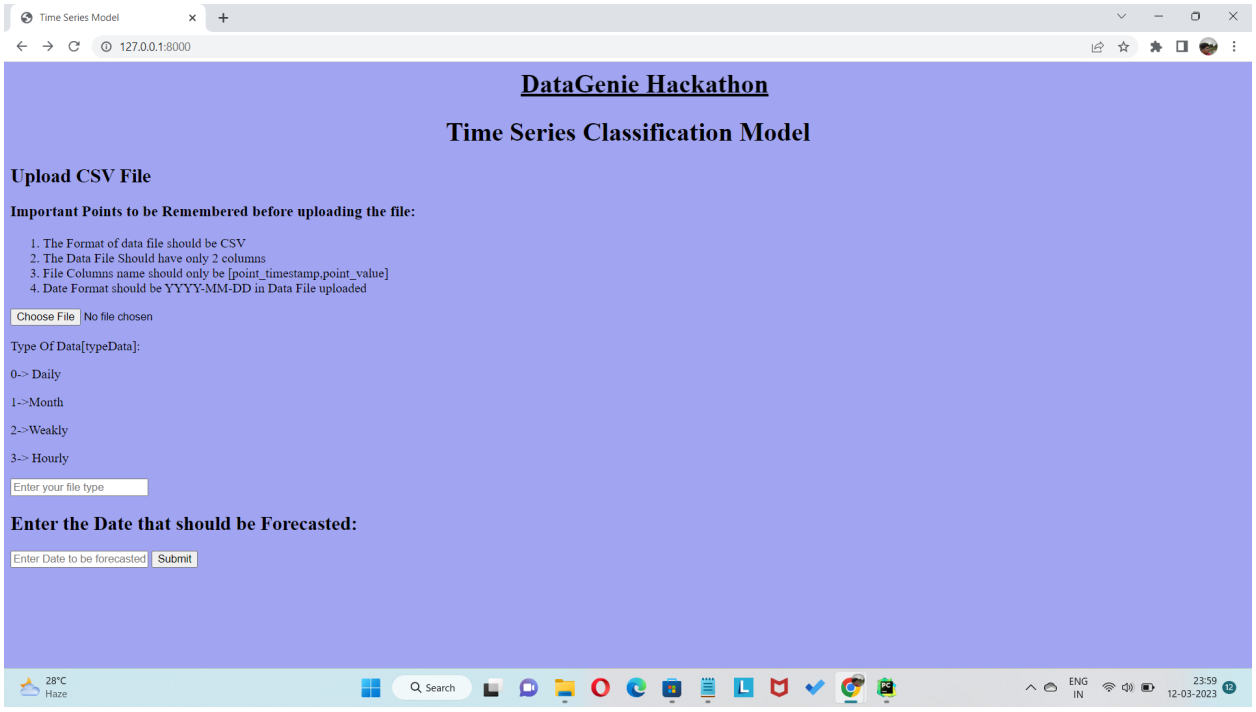
The UI contains two pages. On the first page users should upload a csv file which follows the points that are mentioned and also need to say which type of data it is. Like whether the interval of data is daily, weekly, monthly or hourly. Csv file we need to choose from local storage. And when you submit the model takes the input.

Whereas on the second page of website it shows the results that model predicted for the given input.

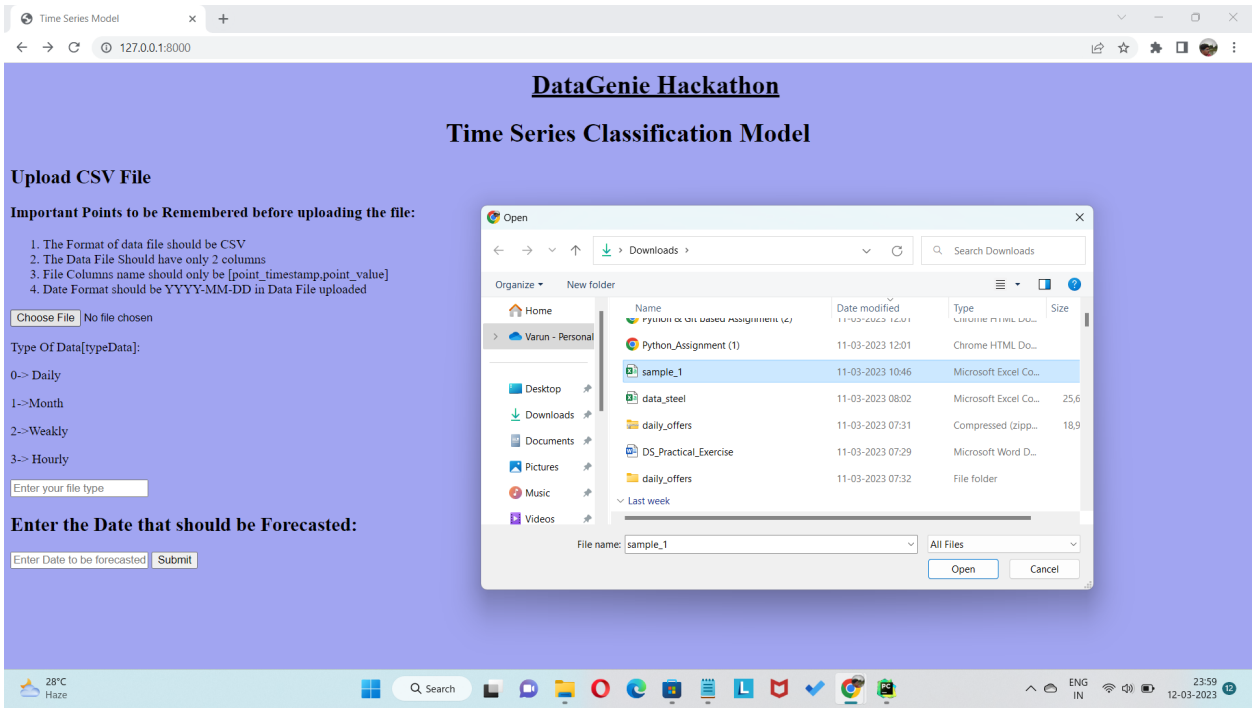
WORKING OF UI:

Step 1: We need to run the FastAPI using **uvicorn main:app –reload**

Step 2: After running successfully we will get a link to localhost. When we paste that link in web browser we can see the Start page

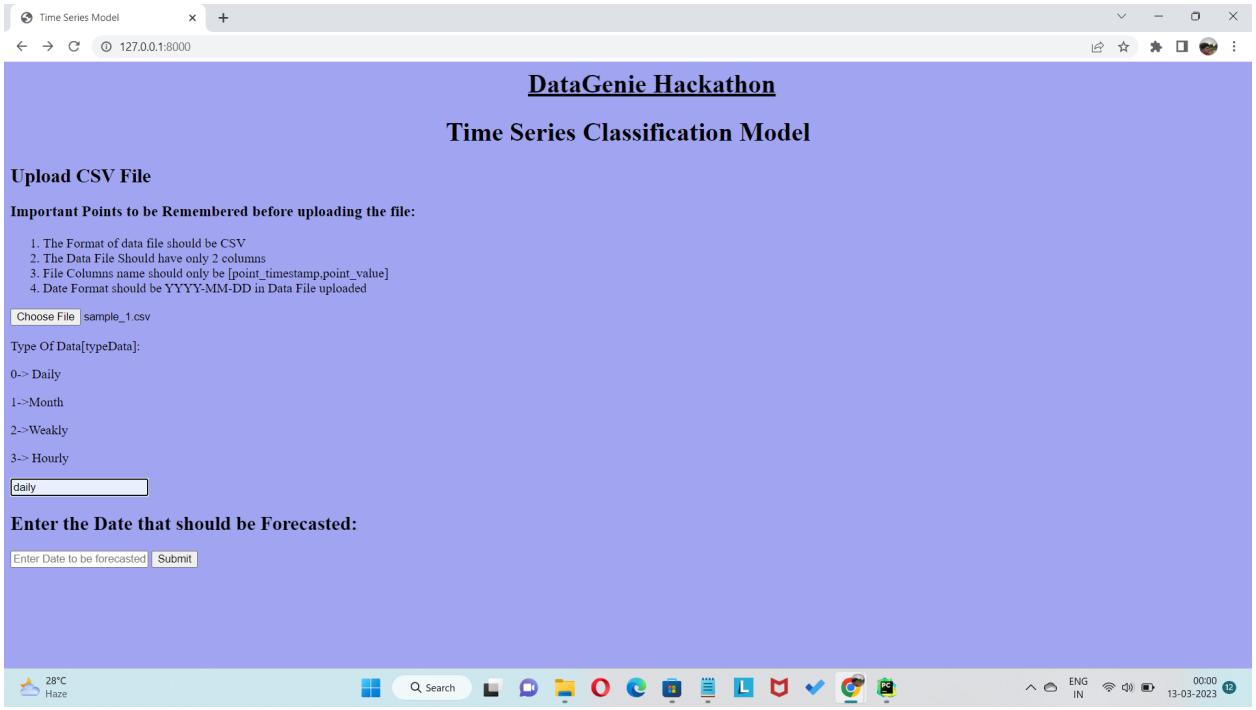


Step 3:The simple UI looks like this. This is page 1 where users need to upload csv file with the rules written over there.



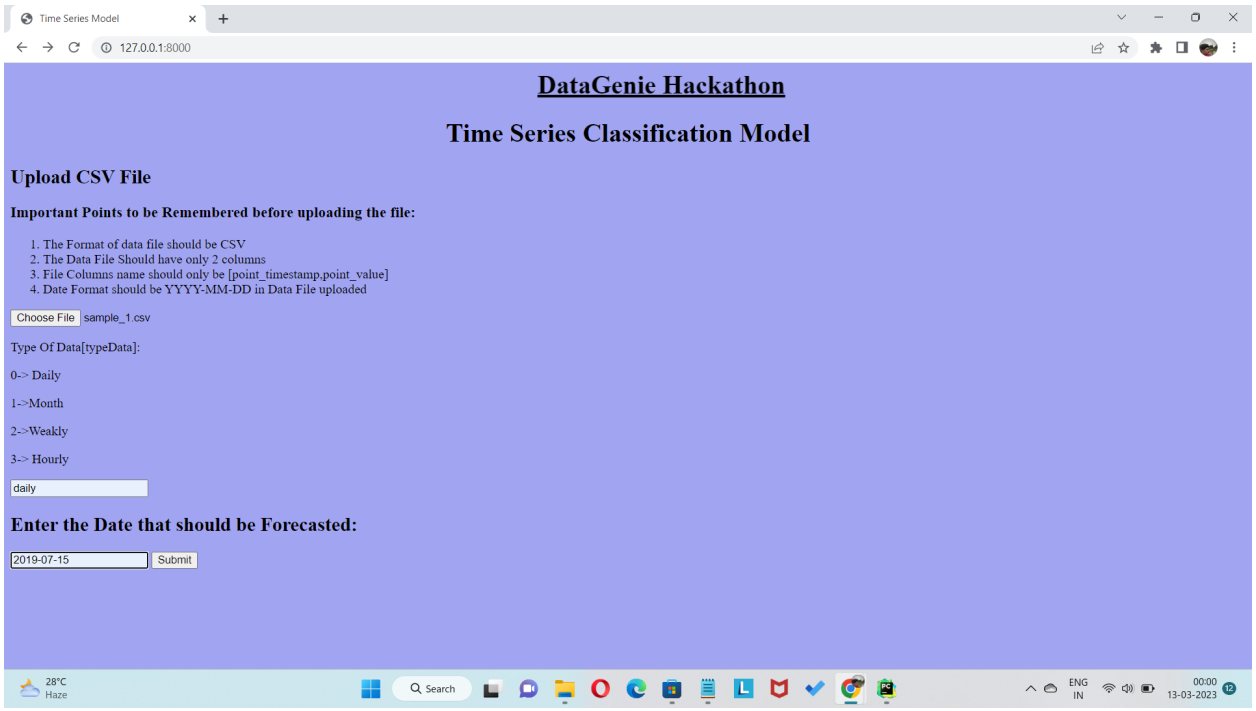
As you can see here when you click on choose file it opens the local drive. From here we need to upload csv file

Step 4: We need to enter the type of data i.e, whether the data is weekly, hourly, monthly or daily



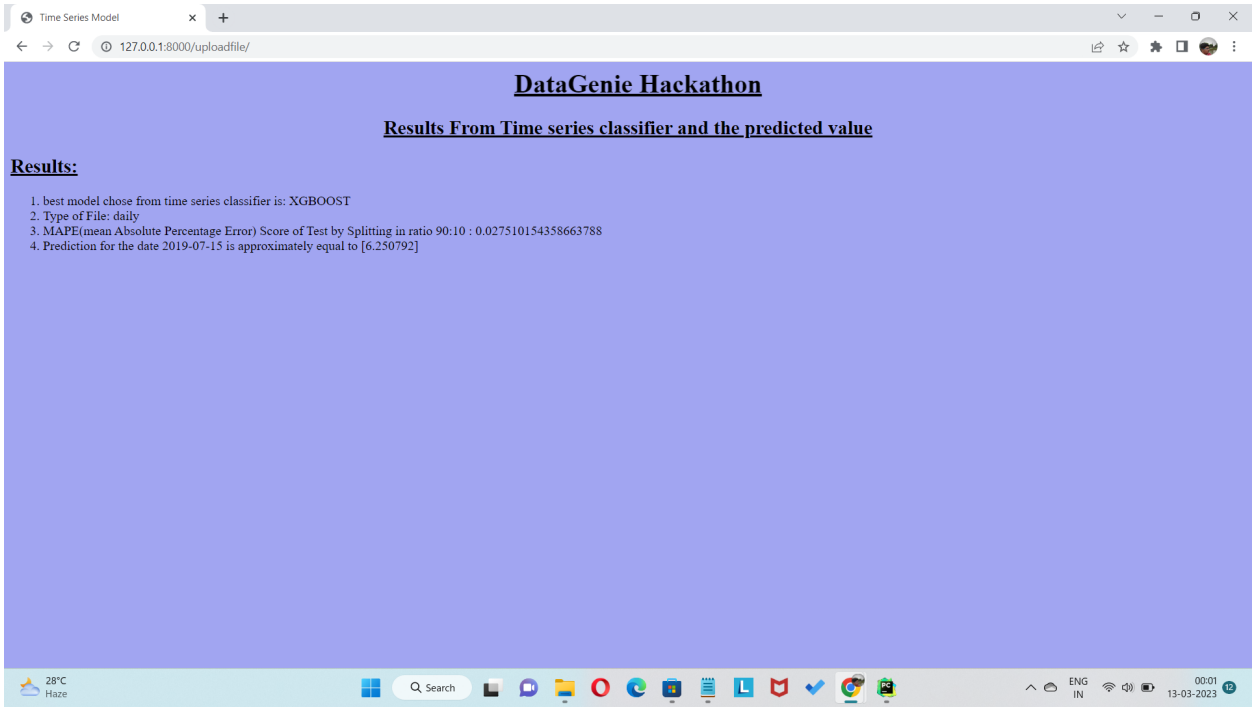
As we can see here we need to enter the type of data in the box given here.

Step 5: users need to enter the date that should be predicted in YYYY-MM–DD format



As we can see in the above picture we need to enter the date in the given format.

Step 6: Click on submit button then it will show results and predictions



Challenges faced while doing hackathon:

- Getting the idea about how to start was bit difficult because we need to build a classifier which classifies the best time series model for the given data
- As it is time series data it is hard to understand concepts of time series like stationarity, ACF PACF graphs, hyperparameter tuning for orders in models. So first I started learning about time series to be familiar with all the concepts.
- I also don't know FastAPI which I found difficult to implement the backend part so I have done backend by browsing google to know about FastAPI
- After completing the project I wanted to deploy the model but couldn't deploy it because of some error. As I am using FastAPI for first time I am not familiar with how to deploy it.
- As the number of datapoints are less in the new dataset I wanted to do oversampling but due to time constraint I was not able to complete.

Checkpoints:

Checkpoint 1: **Completed**

Created a time series classifier using sample data

Checkpoint 2: Completed

Generated predictions for a new dataset using Pre Trained classifier and calculate MAPE value and also if the user gives input for forecasting it predicts the output.

Checkpoint 3: Completed

Using FastAPI I have built a basic backend.

Checkpoint 4: Partially completed

Created UI using HTML but unable to deploy the project.

Summary About Project(Hackathon):

Initially I got an idea about how to solve this problem then I started exploring time series and its concepts. I have 36 sample datasets.From that I have taken one sample data from the sample time series and built ARIMA, SARIMAX, XGboost models and found out the MAPE values for each model. For XGBoost as features are not available I have created some new features. After that for time series classification we need to create a new dataset which contains features and output variables. To create features we need to call the FeatureEngineering function for every sample so that it will create new features. Now for the Output variable we need to find MAPE values of 3 models for all 36 datasets.

Then we need to find the minimum MAPE value in each sample and append them in the list. Now with these features and output variables we need to create a dataset. I did multiway classification for this dataset to find which is the best time series model.

Now when a new sample is uploaded by user first basic preprocessing will be done and features will be extracted with help of FeatureEngineering then we need to predict the best model by using the classifier model and after getting the best model we need to model this sample with the best model and predict the future data.

After completing the model I started doing backend using FastAPI. It was hard at the beginning to build the backend but eventually by browsing I have created a simple backend which takes a csv file as input and stores it in a local drive. Then use that csv file for further processes. Then I have created a simple UI using HTML. It just takes csv file and other inputs and prints the results in second page.