

DATA SCIENCE

EXAMPLE OF AMAZON FORECAST USING DEEPAR

Amazon Forecast: Train and Deploy build-in DeepAR+ using golf dataset with additional weather information

Yiran Jing

July 12, 2019

Contents

1	Row Data set description	1
2	Data Cleaning	2
2.1	Input dataset for Amazon forecast	2
2.1.1	Target time series data	2
2.1.2	Related time series data	2
2.2	Golf Data Cleaning	2
2.3	Weather Data Cleaning	3
3	EDA and Feature Engineering	4
3.1	Plot log data	4
4	Stationary transformation	5
4.1	Testing Stationary: augmented Dickey fuller test	5
4.2	Differencing golf dataset	5
5	Modeling, Deploying, and Forecasting using Amazon forecast console	7
5.1	Step 1: Create a Dataset Group and Dataset	7
5.1.1	Create dataset group	7
5.1.2	Import target time series data	8
5.1.3	Import related time series data	9
5.2	Step 2: Train a Predictor	12
5.3	Step 3: Deploy the Predictor	14
5.4	Step 4: Generate Forecasts	17
6	Model performance validation and Comparison	20

1 Row Data set description

We use two datasets here:

1. **OneStreamGolfData.csv** Object URL (Target time series data)
2. **weatherdata.csv** Object URL (Related time series data)
3. **Target of Forecast:** ***Amount*** in *OneStreamGolfData* dataset.
4. **Item_id:** *Region*

2 Data Cleaning

The following details in `EDA_FeatureEngineer-key_region.ipynb` and `Split_weather_data.ipynb`.

2.1 Input dataset for Amazon forecast

2.1.1 Target time series data

Target time series input data Input requirment:

1. **one-dimentional** time series data set with **three column**
2. **Item_id:** *string* *region*
3. **Target_value:** *float* *amount*
4. **timestamp:** *timestamp* *month*

2.1.2 Related time series data

Related time series input data Input requirment:

1. **multi-dimentional** time series data set
2. **Item_id:** *string* *region* (must be same with targe data)
3. **timestamp:** *timestamp* *month* (must be same with targe data)
4. **related_featuer:** *float* *related_featuer* can be more than one.

2.2 Golf Data Cleaning

We have 36 regions, Some regions have 2 years data (2010 and 2011), but more regions have 1 year data only (2011). also a few region has 1.5 years data. Steps for Golf Data Cleaning:

1. convert column name to all lower cases
2. strip the leading space for columns
3. rename 'ud3' to 'region', same as the column name in weather data.
4. rename 'time' to 'month', same as the column name in weather data.
5. rename 'ud2' to 'product'
6. rename 'ud4' to 'retailer'
7. remove space in the column entry
8. delete the columns if nan values only
9. delete the columns if no missing value and just one value within that column, since this kind of column give us no useful info for modelling.
10. Convert month column to **timestamp**: require for Amazon forecast

11. Select relevant columns to build Target time series data.

After these steps, Golf dataset contains the following columns. See figure 1,

	region	amount
month		
2010-01-01	Australia	1165742.039
2010-01-01	Southwest	2478456.728
2010-01-01	Southeast	5827990.362
2010-01-01	South Korea	55775.918
2010-01-01	SA	497842.220

Figure 1: Cleaned Golf data set

2.3 Weather Data Cleaning

Similar with the data cleaning process of golf data, we use Weather data to create **related time series dataset**. details in [Split_weather_data.ipynb](#) *click me to open this notebook*. The **multi-dimentional** time series data is: See figure 2.

	month	high	low	rainfall	snowfall	daylight_hours	region
1							
I	2010-01-01	2	-11	11	178	10	Midwest
I	2010-01-01	0	-6	49	192	10	Illinois
I	2010-01-01	0	-7	57	290	10	Canada
I	2010-01-01	4	-3	97	211	10	Northeast
I	2010-01-01	14	2	99	6	10	South Carolina

Figure 2: Cleaned Weather data set

Note that we can also put *one-dimensional* related time series data to build model, and in this case, it gives much better forecast result than *multi-dimentional* related time series data. The relevant forecast result and discuss in *section 5: Model performance validation and Comparison*.

3 EDA and Feature Engineering

We tried stationary time series data in this part, but note that I still use **non-stationary** time series data for modelling. There are three reasons:

1. We will lose one observation for each difference, *click me to see what is differencing time series data*, but the time series data is *monthly-based*, and we donot have many observations for each region. (*some regions have 2 years data (2010 and 2011), but more regions have 1 year data only (2011). also a few region has 1.5 years data.*).
2. We cannot find the uniform way to stationary time series data for all regions. Since regions contain different length of time (1 year, 1.5 year, or 2 years).
3. The **DeepAR+** algo in AWS forecast, combine ML models with classical time series models. And **ML time series models donot need stationary time series data**.

3.1 Plot log data

Since original data show large variance, I tried **log transformation** to stable variance, but based on the figure 3 and figure 4 log still cannot stable variance, so just keep orgin price in modelling.

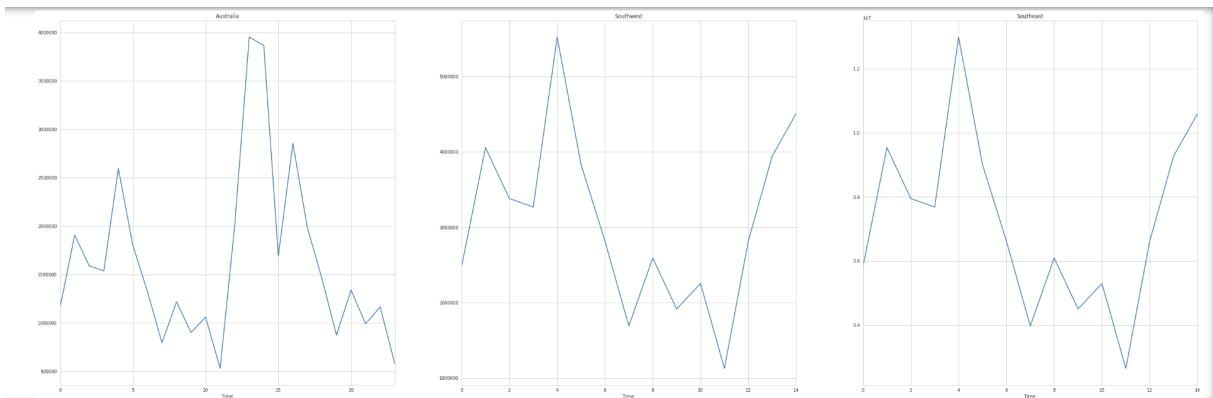


Figure 3: Original time series data for Australia, Southwest and Southeast

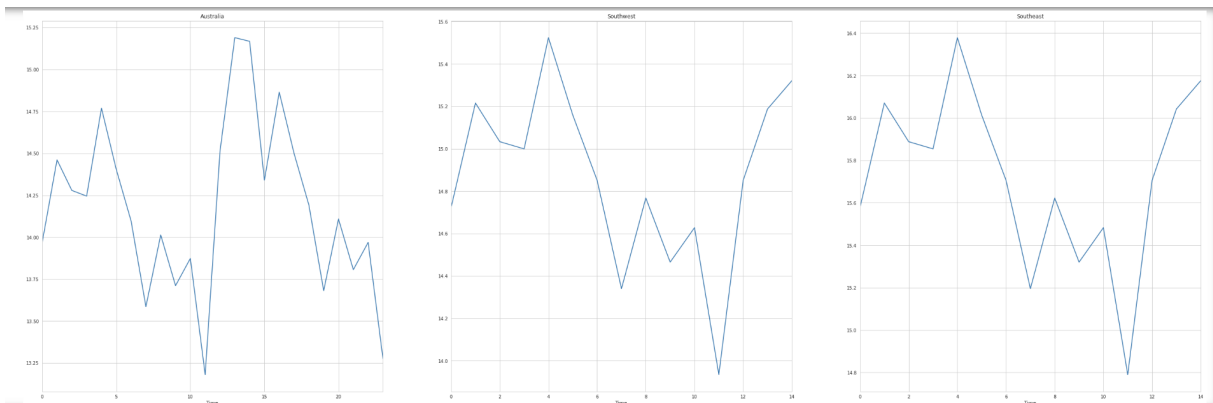


Figure 4: log time series data for Australia, Southwest and Southeast

4 Stationary transformation

4.1 Testing Stationary: augmented Dickey fuller test

First part of classical time series model is that we require stationarity of the time series. We use the augmented Dickey fuller test.

Recall that the null hypothesis is that we don't have a stationary time series, so a rejection suggests stationary of the data. In other word, **if p_value less than 0.01, we can say we have stationary data**. From the hypothesis test result, all regions have non-stationary data. Thus we need to stationary data before train classical time series model. See figure 5, the p value actually is much larger than 0.01, thus we can ensure that they do not stationary even at 10 percentage confident level.

```
from statsmodels.tsa.stattools import adfuller

products = final_trans.region.unique()
products_stationary = {} # collect stationary data: key is the name or product, value is p_value
products_nonstationary = {} # collect non-stationary data

"""
Run ADF test on each product and check test.
"""
for prod in products:
    prod_df = final_trans[final_trans.region == prod]
    p_value = adfuller(prod_df['amount'])[1]
    if p_value < 0.01: # we do at the 1% comcident level.
        products_stationary[prod] = p_value
    else:
        products_nonstationary[prod] = p_value

print("There are {} regions with stationary data".format(len(products_stationary)))
print("There are {} regions with non-stationary data".format(len(products_nonstationary)))

There are 0 regions with stationary data
There are 36 regions with non-stationary data

products_nonstationary
{'Australia': 0.1255014346686854,
 'Southwest': 0.958532086060056,
 'Southeast': 0.958532086060056,
 'South Korea': 0.1335311940014372,
 'SA': 0.13479323891475536,
 'Other NA': 0.12980698873382973,
 'Other Asia': 0.13627939882895396,
 'Northeast': 0.958532086060056,
 'West': 0.958532086060056,
 'Midwest': 0.958532086060056,
 'Middle East': 0.12344011656120618,
 'Mexico': 0.13179784247011272,
 'Tanan': 0.12861832844019655}
```

Figure 5: Dickey fuller test for Stationary

4.2 Differencing golf dataset

Recall that $difference(t) = observation(t) - observation(t - 1)$. After first difference for all regions, second difference for the regions if they have more than one year data, and 12-th difference for the regions if they have 2 year data. we still have 4 regions with non-stationary data. Function details in figure 6,

```

def difference(final_trans, interval1,interval2):
    """
    Differencing data and test stationarity.
    """
    products = final_trans.region.unique()
    products_stationary = {} # collect stationary data: key is the name or product, value is p_value
    products_nonstationary = {} # collect non-stationary data
    for prod in products:
        prod_df = pd.Series(final_trans[final_trans.region == prod]['amount'])
        prod_df = prod_df.diff(1).dropna()
        if len(prod_df) > 10: # if we have more than 1 year data
            prod_df = prod_df.diff(interval1).dropna()
        if len(prod_df) > 20: # if we have 2 year2 data
            prod_df = prod_df.diff(interval2).dropna()
        """
        Dickey fuller test to check stationary
        """
        try:
            p_value = adfuller(prod_df)[1]
        except ValueError:
            print(prod_df)
        if p_value < 0.05: # we do at the 1% comcident level.
            products_stationary[prod] = p_value
        else:
            products_nonstationary[prod] = p_value
        """
        Check by ACF and PACF after make stationary
        """
        print(prod)
        fig, ax = plt.subplots(1,2, figsize=(12,3))
        sm.graphics.tsa.plot_acf(prod_df, lags=len(prod_df) -1, ax=ax[0])
        sm.graphics.tsa.plot_pacf(prod_df, lags=len(prod_df) -1, ax=ax[1])
    return products_stationary, products_nonstationary

```

```

diff_first_stationary, diff_first_nonstationary = difference(final_trans, 2,12)
print("{} left non stationary regions ".format(len(diff_first_nonstationary)))

```

```

Southwest
Other NA
Middle East
South Carolina
4 left non stationarv regions

```

Figure 6: The best strategy to maximum the number of stationary data for 36 regions

5 Modeling, Deploying, and Forecasting using Amazon forecast console

Another detailed example using Amazon Forecast console: *click me to open webpage*

5.1 Step 1: Create a Dataset Group and Dataset

On the Amazon Forecast home page, if you haven't created any dataset groups, choose Create dataset group.

5.1.1 Create dataset group

On the **Create dataset group** page, for Dataset group details, provide the following information and then choose Next. **IAM role** is `arn:aws:iam::855212401545:role/ForecastServiceR`. The target dataset location is `s3://forecast-datasources/GolfDataforecast/nonstationary_amount_region`. Your screen should look similar to figure 7:

Create dataset group [Info](#)

Dataset groups are containers for all your datasets.

Dataset group details

Dataset group name
The name that you enter here can help you distinguish this dataset group from other dataset groups on the Dataset groups dashboard.

The dataset group name must have 1 to 32 characters. Valid characters: a-z, A-Z, 0-9, and . : + = @ _ %

Forecasting domain [Info](#)
A forecasting domain defines a forecasting use case. You can choose a predefined domain, or you can create your own domain.

Choose this domain if none of the other domains are applicable to your forecas...

IAM Role [Info](#)
Dataset groups require permissions from IAM to read your dataset files on S3. Choose or create a role using this control.

Custom IAM role ARN

[Cancel](#) [Next](#)

Figure 7: Create dataset group page Example

5.1.2 Import target time series data

On the **Import target time series data** page, Your screen should look similar to figure 8 and figure 9:

Import target time series data [Info](#)

Target time series data import configuration

Dataset name
The name that you enter here can help you distinguish this dataset from other datasets on your Datasets dashboard.

The dataset name must have 1 to 32 characters. Valid characters: a-z, A-Z, 0-9, and .:+=@_%

Automatic dataset import jobs [Info](#)
Automatically run update your dataset by running dataset import jobs. They will do so by replacing your existing dataset's data with the latest data in your S3 bucket.

☐ **On**
Dataset import jobs will be created on a fixed cadence.

☒ **Off**
You'll have to manually update your dataset by creating new dataset import jobs.

Target time series data formatting

Frequency of your data
This is the frequency at which entries are registered into your data file.

Your data entries have a time interval of

Timestamp format [Info](#)
This is the format of the timestamp in your dataset. The format that you enter here must match the format in your data file.

Figure 8: Import target time series data page Example part 1

Delimiter [Info](#)

The delimiter is the character that separates the entries in your dataset file.

CSV file headers [Info](#)

- ☐ Your .csv file has the required headers
Choose this option if the file in your dataset already has these required headers: date (timestamp), demand (float), item (string).
- ☒ Your .csv file doesn't have the required headers
Choose this option to define the schema with the required headers.

Data schema

To help Amazon Forecast understand the fields in your data, you must define the schema. Specify the headers in the same order as they appear in your .csv file.

```
1  {
2      "Attributes": [
3          {
4              "AttributeName": "timestamp",
5              "AttributeType": "timestamp"
6          },
7          {
8              "AttributeName": "target_value",
9              "AttributeType": "float"
10         },
11         {
12             "AttributeName": "item_id",
13             "AttributeType": "string"
14         }
15     ]
16 }
```

Data location [Info](#)

The location is the path to your S3 bucket or a folder in your bucket that contains your data.


Your files must be in CSV format.

Figure 9: Import target time series data page Example part 2

5.1.3 Import related time series data

To import related time series data, you need to click import related time series data on the dashboard page. Your screen should look similar to figure 10, figure 11 and figure 12:

Overview



Import your data
Datasets are required to train predictors, which are then used to generate forecasts.

Target time series data	<div><div>✓ Active</div><div>View Edit</div></div>
Item metadata data	<div>Import</div>
Related time series data	<div>Import</div>

Figure 10: Import related time series data page Example part 1

Import related time series data [Info](#)

Related time series data import configuration

Dataset name

The name that you enter here can help you distinguish this dataset from other datasets on your Datasets dashboard.

The dataset name must have 1 to 32 characters. Valid characters: a-z, A-Z, 0-9, and .:+=@_%

Automatic dataset import jobs [Info](#)

Automatically run update your dataset by running dataset import jobs. They will do so by replacing your existing dataset's data with the latest data in your S3 bucket.

☐ On

Dataset import jobs will be created on a fixed cadence.

☒ Off

You'll have to manually update your dataset by creating new dataset import jobs.

Related time series data formatting

Frequency of your data

This is the frequency at which entries are registered into your data file.

Your data entries have a time interval of

Timestamp format [Info](#)

This is the format of the timestamp in your dataset. The format that you enter here must match the format in your data file.

Figure 11: Import related time series data page Example part 2

Delimiter [Info](#)

The delimiter is the character that separates the entries in your dataset file.

CSV file headers [Info](#)

- ☐ Your .csv file has the required headers
Choose this option if the file in your dataset already has these required headers: date (timestamp), demand (float), item (string).
- ☒ Your .csv file doesn't have the required headers
Choose this option to define the schema with the required headers.

Data schema

To help Amazon Forecast understand the fields in your data, you must define the schema. Specify the headers in the same order as they appear in your .csv file.

```
1  {
2      "Attributes": [
3          {
4              "AttributeName": "timestamp",
5              "AttributeType": "timestamp"
6          },
7          {
8              "AttributeName": "snowfall",
9              "AttributeType": "float"
10         },
11         {
12             "AttributeName": "item_id",
13             "AttributeType": "string"
14         }
15     ]
16 }
```

Data location [Info](#)

The location is the path to your S3 bucket or a folder in your bucket that contains your data.

Your files must be in CSV format.

Figure 12: Import related time series data page Example part 3

5.2 Step 2: Train a Predictor

To train a predictor, you use a recipe on the imported target time-series data. You can manually choose a particular recipe, or choose AutoML to have Amazon Forecast process your data and choose a recipe to best suit your dataset group. In this example, I use **DeepAR+**. See figure 13,

- On your dataset group's **Dashboard**, under **Train a predictor**, choose **Start**. The **Train predictor** page is displayed.

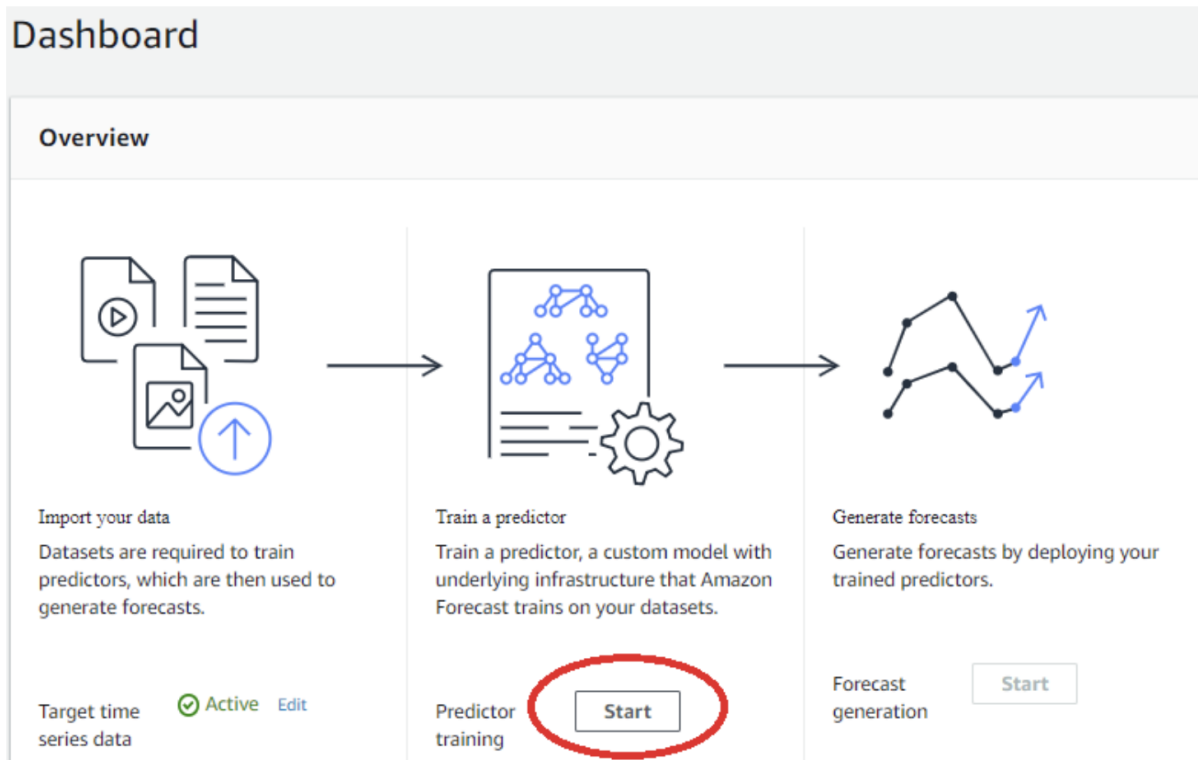


Figure 13: Train a Predictor Part 1

On the **Train predictor** page, provide the following information for Predictor details. Your screen should look similar to figure 14:

Predictor details

Predictor name

The name that you enter here can help you distinguish this predictor from your other predictors.

DeepAR_Predictor

The predictor name must have 1 to 32 characters. Valid characters: a-z, A-Z, 0-9, and . : + = @ _ %

Forecast horizon [Info](#)

The range tells Amazon Forecast how far into the future to forecast your data. The number you enter here will be multiplied by the data update interval of your target time-series dataset.

4

Recipe selection [Info](#)

A recipe includes forecasting algorithms that train your predictor.

☐ Automatic (AutoML)

Let Amazon Forecast choose the right recipe for your dataset.

☒ Manual

Explore the recipes and choose one.

Recipe

The recipe that you want Amazon Forecast to use to train your predictor.

forecast_deep_ar_plus

Automatic retraining

This option automatically retrains your predictor on a regular schedule so that you won't have to manually do so.

☐ On

Your predictor will be retrained on a fixed cadence.

☒ Off

You'll have to manually retrain your predictor.

Figure 14: Train a Predictor Part 2

Then wait until training is finished. The process can take several minutes or longer to complete. On the dataset group's Dashboard, while the training is in progress, the status of Predictor training is *Creating*. When the training has finished, the status changes to **Active**. After your predictor has finished training, you can deploy it.

5.3 Step 3: Deploy the Predictor

Your screen should look similar to figure 15, figure 16 and figure 17:

After the predictor has finished training, you can deploy it and generate forecasts.

To deploy the predictor and generate forecasts

1. On your dataset group's **Dashboard**, for **Generate forecasts**, choose **Start**. The **Deploy predictor** page is displayed.

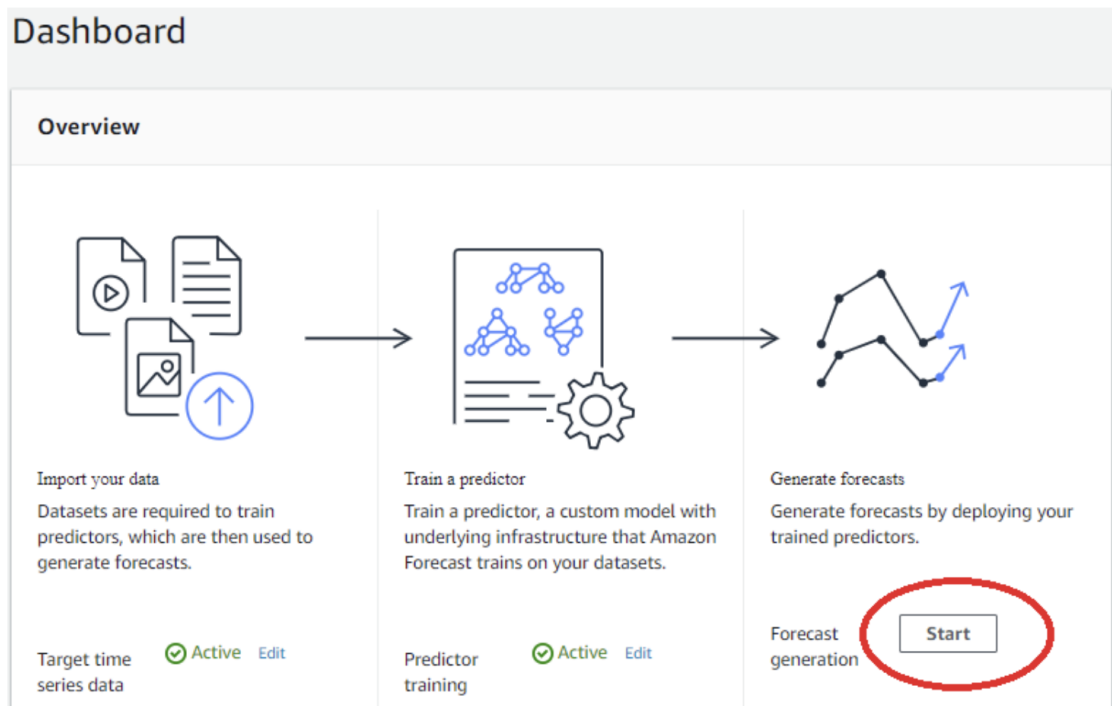


Figure 15: Deploy the Predictor Part 1

2. On the **Deploy predictor** page, for **Predictor details**, provide the following information:

- **Predictor** – Choose the predictor you trained in the previous step from the drop-down menu.
- **Predictor version** – No need to specify as there is only one version.
- **Automatic redeployment** – For this exercise, choose **Off**. If you want to automatically redeploy this predictor on a regular schedule, choose **On**.

Your screen should look similar to the following:

Deploy predictor [Info](#)

When you deploy your predictor, Amazon Forecast generates your forecasts.

Predictor details

Predictor [Info](#)
Tell Amazon Forecast which predictor to deploy.

Predictor version – optional
Choose which version of your predictor should be deployed.

By default, the latest version of your predictor will be chosen.

Automatic redeployment
This option automatically redeploys your predictor on a regular schedule so that you won't have to manually do so.

☐ **On**
Your latest trained predictor will be redeployment on a fixed cadence.

☒ **Off**
You'll have to manually redeploy your predictor.

Figure 16: Deploy the Predictor Part 2

Choose **Deploy predictor** and then wait for the deployment to finish. The process can take several minutes or longer to complete. While the deployment is in progress, the status of **Forecast generation** is **Creating**. When the deployment has finished, the status changes to **Active**. Don't proceed to the next step until the status is **Active**.

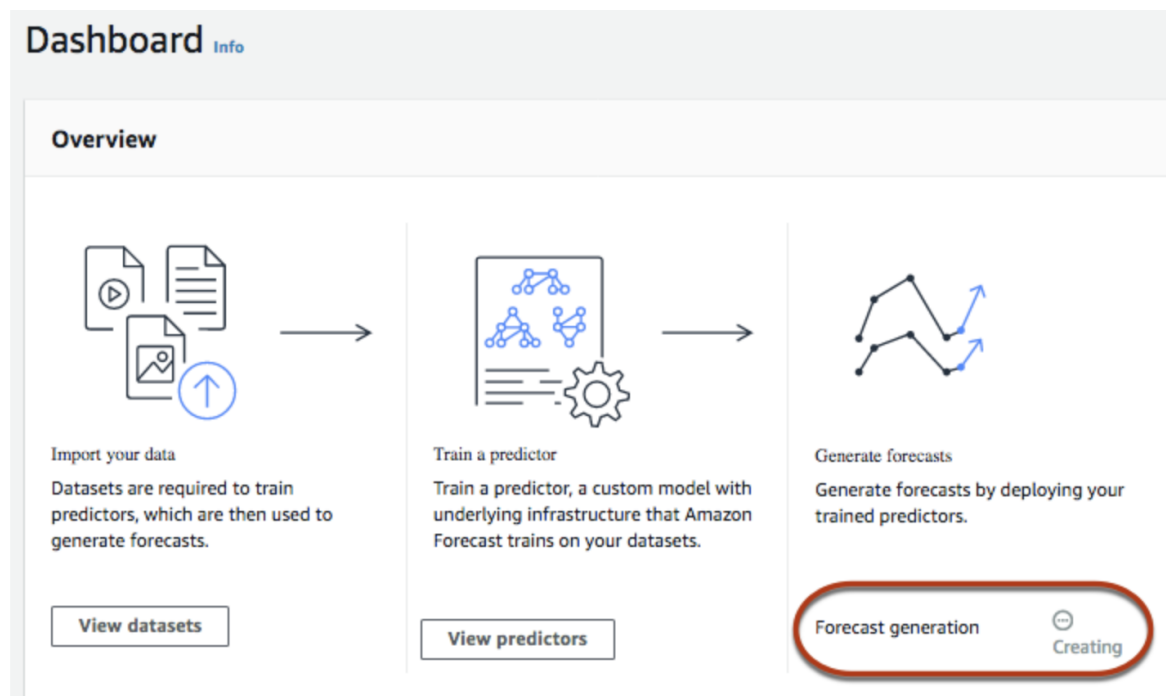


Figure 17: Deploy the Predictor Part 3

5.4 Step 4: Generate Forecasts

After the predictor has been deployed, you can generate and view forecasts for the items in your target time-series dataset. To **get and view your forecast**, On the Dashboard page, choose **Lookup forecast** under Generate forecasts. The Forecast lookup page is displayed. See figure 18,

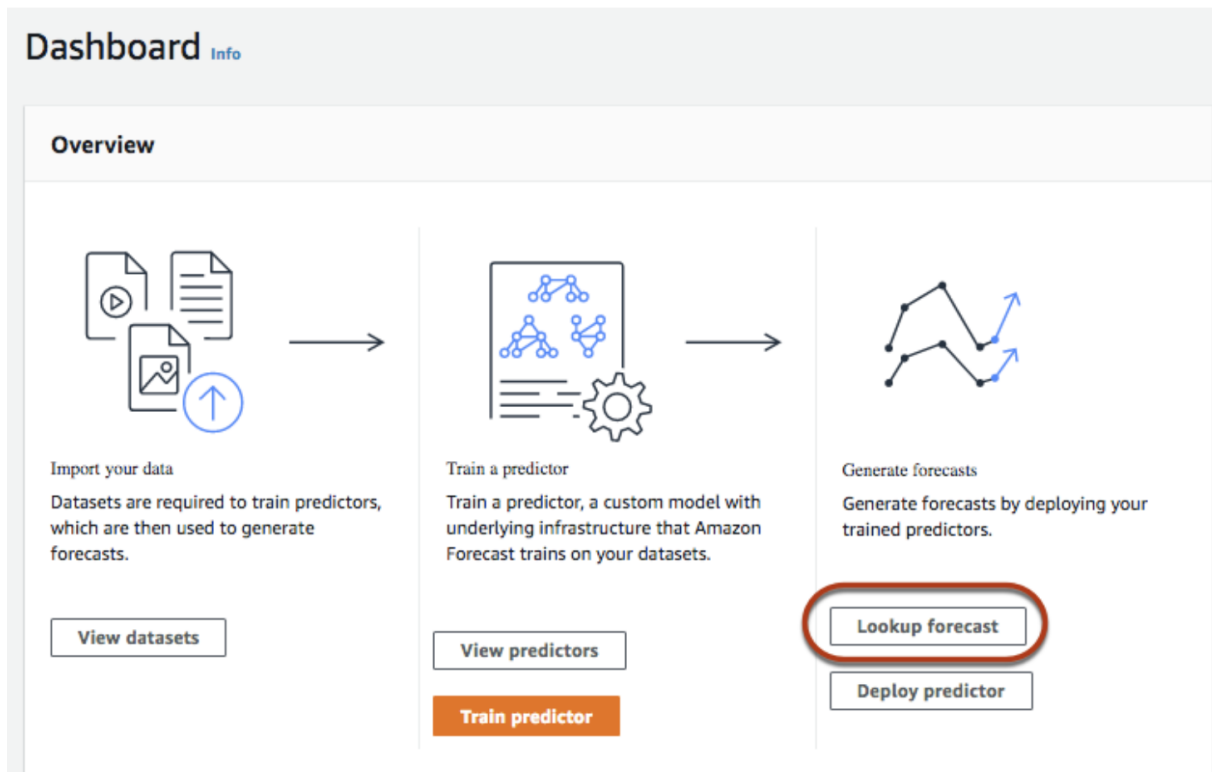


Figure 18: Lookup forecast on DashBoard page

On the **Forecast lookup** page, for Forecast details, provide the following information:

1. **Item id:** Australia.
2. **Predictor** Choose the predictor you previously deployed from the drop-down menu.
3. **Predictor version** No need to specify as there is only one version.
4. **Start date:**2012-01-01
5. **End date:**2012-03-01
6. **Time interval:** month

Your forecast should look similar to the following See figure 19,

Item_id: Australia

Product Demand

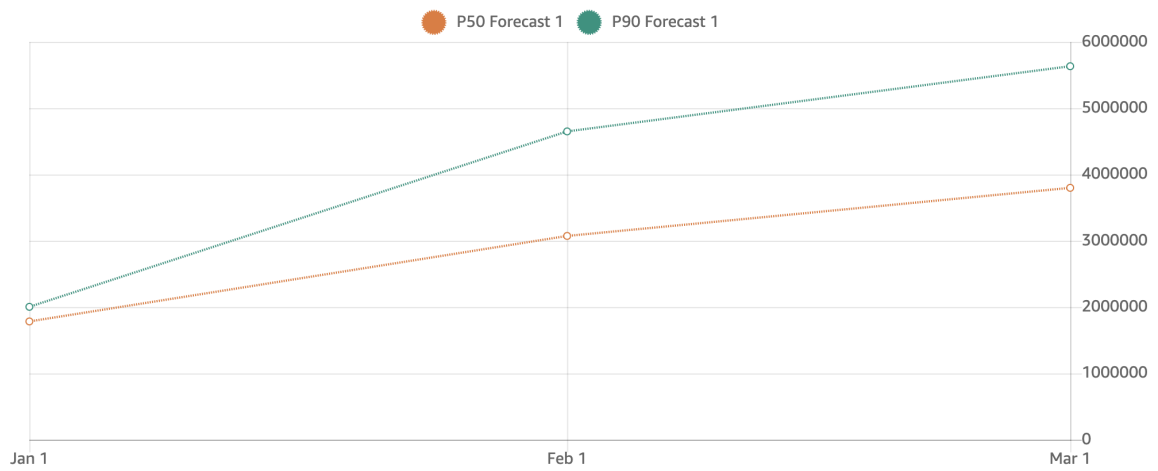


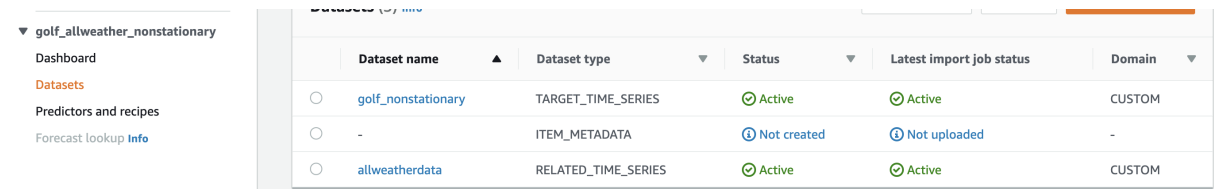
Figure 19: Forecast visulaization for Australia

6 Model performance validation and Comparison

We compare the model performance based on **P10, P50, P90**. The lower error is better. Comparison result:

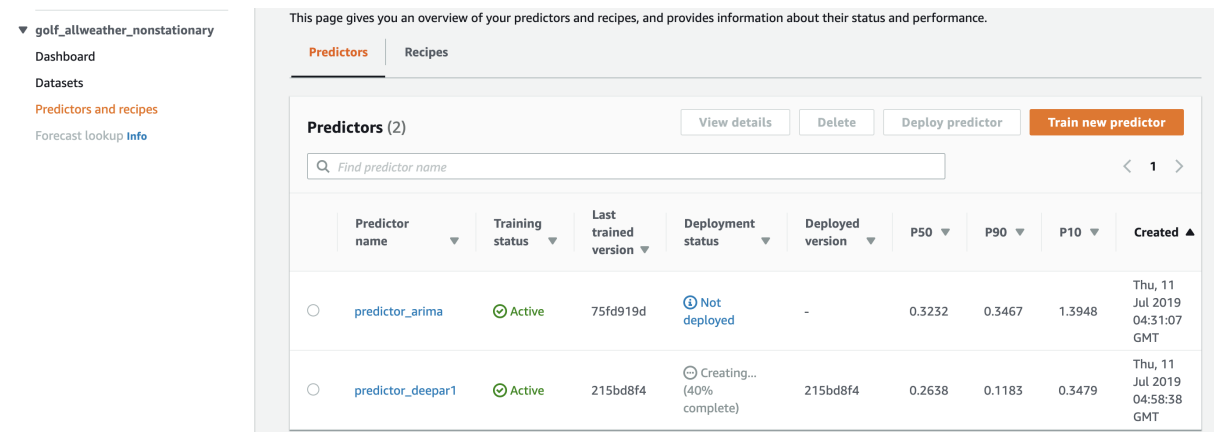
1. **DeepAR+** gets much better result using any choice of *related time series*. For example, see figure 23 and figure ???. Clearly because ML models performs better than classical models and donot need stationary data.
2. Using *high* information only gives better result than using other weather infomation. For example, See two examples below.
3. The best forecasting model is **DeepAR with High information**. See example 2 below.

Example 1: Using golf data and all weather data (5 columns):



Dataset name	Dataset type	Status	Latest import job status	Domain
golf_nonstationary	TARGET_TIME_SERIES	Active	Active	CUSTOM
-	ITEM_METADATA	Not created	Not uploaded	-
allweatherdata	RELATED_TIME_SERIES	Active	Active	CUSTOM

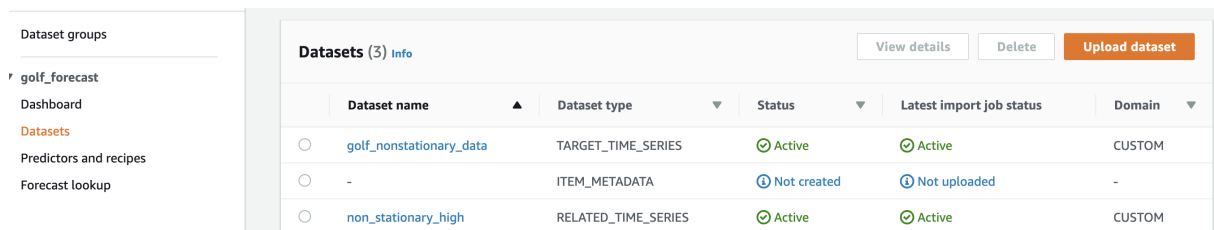
Figure 20: Forecast using golf data and all weather columns part 1



Predictor name	Training status	Last trained version	Deployment status	Deployed version	P50	P90	P10	Created
predictor_arima	Active	75fd919d	Not deployed	-	0.3232	0.3467	1.3948	Thu, 11 Jul 2019 04:31:07 GMT
predictor_deepar1	Active	215bd8f4	Creating... (40% complete)	215bd8f4	0.2638	0.1183	0.3479	Thu, 11 Jul 2019 04:58:38 GMT

Figure 21: Forecast using golf data and all weather columns part 2

Example 2: Using golf data and high weather data (1 column):



Dataset name	Dataset type	Status	Latest import job status	Domain
golf_nonstationary_data	TARGET_TIME_SERIES	Active	Active	CUSTOM
-	ITEM_METADATA	Not created	Not uploaded	-
non_stationary_high	RELATED_TIME_SERIES	Active	Active	CUSTOM

Figure 22: Forecast using golf data and high weather column part 1

golf_forecast
Dashboard
Datasets
Predictors and recipes
Forecast lookup

This page gives you an overview of your predictors and recipes, and provides information about their status and performance.

Predictors

Recipes

Predictors (3)

View details

Delete

Deploy predictor

Train new predictor

Find predictor name

< 1 >

	Predictor name	Training status	Last trained version	Deployment status	Deployed version	P50	P90	P10	Created
<input type="radio"/>	nonstationary_high	Active	d6321683	Active	d6321683	0.4085	0.3435	1.3306	Thu, 11 Jul 2019 02:29:42 GMT
<input type="radio"/>	predictor_deepar	Active	adcce967	Not deployed	-	0.0536	0.0290	0.0572	Thu, 11 Jul 2019 04:19:32 GMT
<input type="radio"/>	predictor_eps	Active	c413714b	Not deployed	-	0.4951	0.3561	1.2617	Thu, 11 Jul 2019 04:20:18 GMT

Figure 23: Forecast using golf data and high weather column part 2

21