Rainfall Data Fitting and Forecasting

ME789 Course Project

By

1. Krushna Bhagat (Roll no: 213104022)
2. Shivprasad Kathane (Roll no: 180110076)

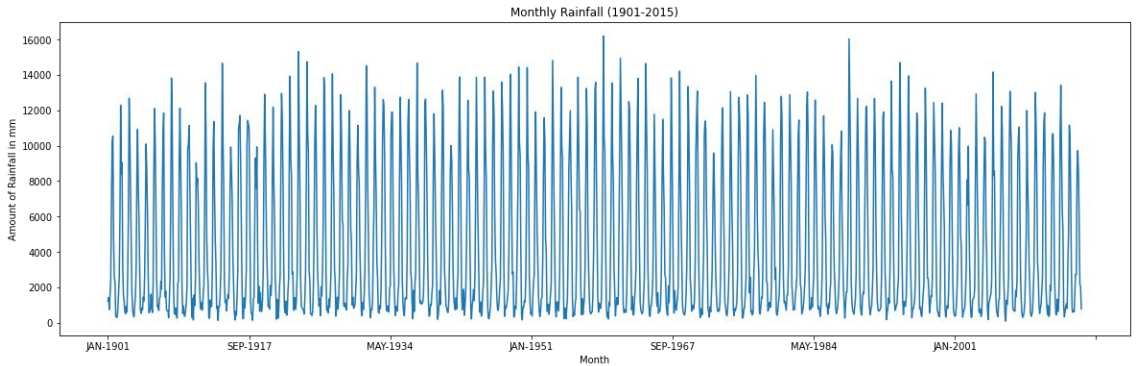
Problem Statement

- Rainfall is a very complex and seasonal phenomenon.
- Amount of rainfall varies with time without any obvious trend or underlying simple model.
- It is important to find ways to predict/forecast rainfall in order to plan things which are dependent on it such as agriculture.
- The objective of this project is to forecast rainfall for India using historical monthly data.
- Different models like regression, ARIMA and LSTM are to be implemented and results analysed.

Raw Data followed by Time Series Plot

	SUBDIVISION	YEAR	Parameter	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC	ANNUAL	JF	MAM	JJAS	OND
0	ANDAMAN & NICOBAR ISLANDS	1901-2015	Mean	49.2	27.6	30.0	72.2	355.6	471.4	397.5	400.5	431.3	289.5	233.0	153.3	2911.0	76.8	457.8	1700.7	675.8
1	ANDAMAN & NICOBAR ISLANDS	1901-2015	Standard deviation	71.3	38.8	43.6	66.8	151.2	147.0	151.9	142.6	146.7	99.4	119.4	129.5	395.9	81.1	176.8	286.3	199.0
2	ANDAMAN & NICOBAR ISLANDS	1901-2015	Coefficient of variation	144.9	140.4	145.5	92.5	42.5	31.2	38.2	35.6	34.0	34.3	51.2	84.5	13.6	105.6	38.6	16.8	29.4
3	ANDAMAN & NICOBAR ISLANDS	1901	Actual	49.2	87.1	29.2	2.3	528.8	517.5	365.1	481.1	332.6	388.5	558.2	33.6	3373.2	136.3	560.3	1696.3	980.3
4	ANDAMAN & NICOBAR ISLANDS	1901	Percentage departure	-0.1	215.9	-2.6	-96.8	48.7	9.8	-8.2	20.1	-22.9	34.2	139.5	-78.1	15.9	77.5	22.4	-0.3	45.1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
12451	WEST UTTAR PRADESH	2014	Percentage departure	177.6	66.4	103.9	-14.0	-11.2	-71.4	-39.1	-68.2	-43.5	-50.3	-100.0	127.5	-41.8	121.6	31.2	-53.5	-24.0
12452	WEST UTTAR PRADESH	2014	No. of districts	30.0	30.0	30.0	30.0	30.0	30.0	30.0	30.0	30.0	30.0	30.0	30.0	NaN	NaN	NaN	NaN	NaN
12453	WEST UTTAR PRADESH	2015	Actual	31.6	7.2	66.8	21.0	8.1	72.0	194.2	143.5	26.5	6.9	2.0	3.0	582.7	38.8	95.9	436.1	11.9
12454	WEST UTTAR PRADESH	2015	Percentage departure	81.6	-59.4	501.7	238.5	-34.9	-6.4	-22.1	-43.7	-82.3	-76.3	-52.0	-57.7	-30.3	10.6	222.7	-40.3	-70.6
12455	WEST UTTAR PRADESH	2015	No. of districts	30.0	30.0	30.0	30.0	30.0	30.0	30.0	30.0	30.0	30.0	30.0	30.0	NaN	NaN	NaN	NaN	NaN

12456 rows × 20 columns



Approach – Computational Details

Raw data consisted of monthly rainfall data from 1901 to 2015 for different subdivisions of India.
It was transformed into a time series data (1380 rows) for the whole country.
Then, it was split into training (1901-1984) & testing (1985-2015) data.

Models: Regression (Linear & Polynomial), ARIMA, LSTM

Error Metric: Mean Absolute Percentage Error (MAPE)

Regression: This just helps identify an average trend over the years via fitting a linear graph or a polynomial curve.

```
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
```

ARIMA: This helps capture the seasonality and recent history. As per its name, ‘Autoregressive Integrated Moving Average’ combines ‘regressing on its own lagged/prior values’, ‘differencing of raw observations for stationary time series’ and ‘capturing dependency between observations and residual errors via moving averages’.

```
from statsmodels.tsa.statespace.sarimax import SARIMAX
from pmdarima import auto_arima
from statsmodels.tsa.seasonal import seasonal_decompose
```

LSTM: With the help of 3 gates in its memory block, this neural network is able to keep track of a lot of old information (thus long-term memory) and hence can intelligently use it to predict any future values it affects.

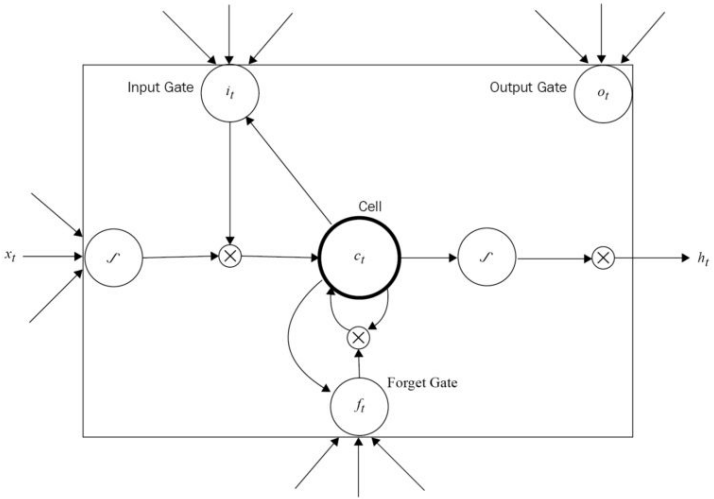
Rainfall values were scaled to be between 0 and 1 and were input to the LSTM in the form of a time series. A sequential keras model was chosen having a LSTM matrix layer with two variables: no of input time step values (consecutive months) and no of hidden units, followed by a general neural network dense layer.

```
from sklearn.preprocessing import MinMaxScaler
from keras.preprocessing.sequence import TimeseriesGenerator
from keras.models import Sequential
from keras.layers import Dense, LSTM
```

Algorithm

LSTM:

```
def LSTM_Prediction(months, units):
  generator = TimeseriesGenerator(train_data, train_data, length = months, batch_size = 1)
  lstm_model = Sequential()
  lstm_model.add(LSTM(units, activation='relu', input_shape=(months, 1)))
  lstm_model.add(Dense(1))
  lstm_model.compile(optimizer='adam', loss='mse')
  lstm_model.fit_generator(generator,epochs=10,verbose=0)
  output_generator = TimeseriesGenerator(test_data, test_data, length = months, batch_size = 1)
predictions = lstm_model.predict_generator(output_generator)
  actual_data = test_data[months:]
  return mape(actual_data,predictions)
```



Polynomial Regression:

function polyregression(x,y,m)
Inputs x : independent variable
y : dependent variable
m : degree of the fit
Output coeff : coefficients
Initialise coeff list
[x x^2 ... x^m] = [x1 x2 ... xm]
Iterate:
- Multiply coeff and x elements
- Subtract from y list to get errors
- Calculate ss = sum of squared errors
- Change coeff values to reduce ss
- Stop when negligible reduction in ss

Minimsed ss yields optimum coeff values
Predict: multiply optimum coeff and new x

ARIMA:

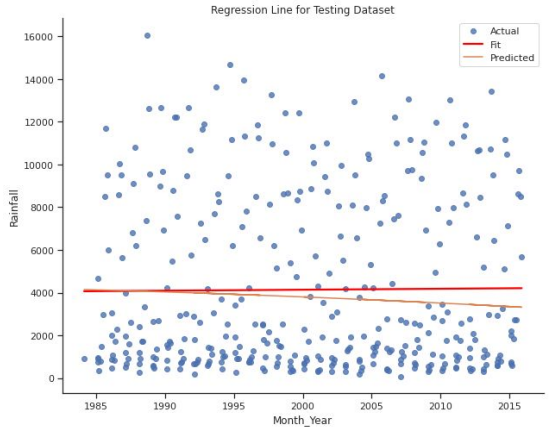
If the autocorrelation function dies off smoothly at a geometric rate, and the partial autocorrelations were zero after one lag, then “a first order autoregressive model”
If the autocorrelations=0 after one lag and the partial autocorrelations declined geometrically, “a first order moving average process”
Combination of three tools:
1. autoregressive (AR) = use of a lagged value of the residual
2. integration order term
3. MA, or moving average term.

The autoregressive and moving average specifications can be combined to form an ARMA (p, q) specification
$d(X, n)$ would specify the nth order difference of the series X with lag parameter L
auto_arima and SARIMAX functions help combine them to build a special kind of time-series regression model which then can be fit and used for predictions

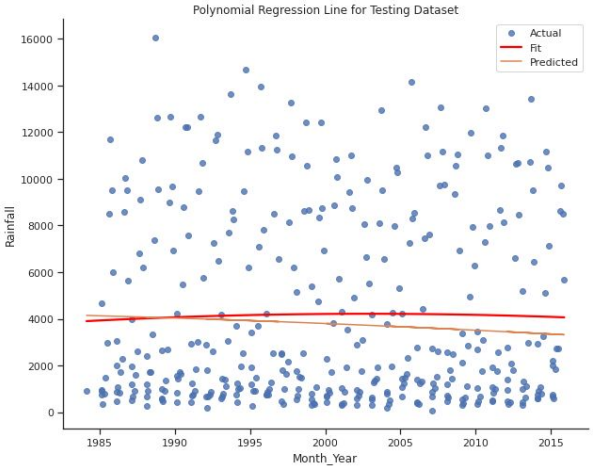
$$u_t = \rho_1 u_{t-1} + \rho_2 u_{t-2} + \cdots + \rho_p u_{t-p} + \varepsilon_t$$
$$u_t = \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q}$$
$$u_t = \rho_1 u_{t-1} + \rho_2 u_{t-2} + \cdots + \rho_p u_{t-p} + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q}$$
$$d(X, n) = (1 - L)^n x$$

Results: Regression and ARIMA

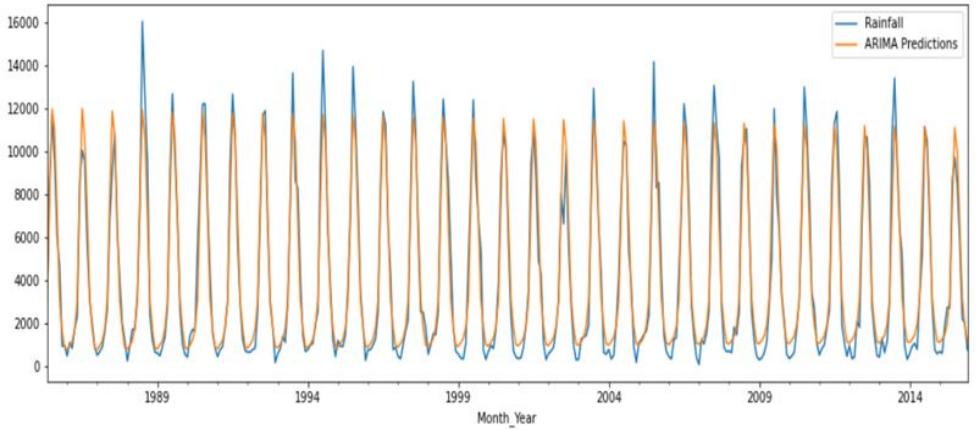
Linear Regression



Polynomial Regression

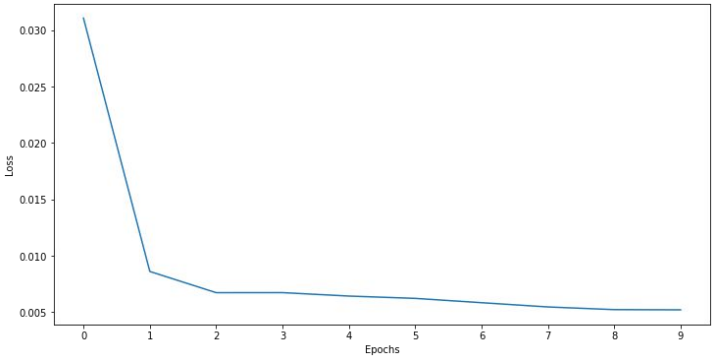


ARIMA



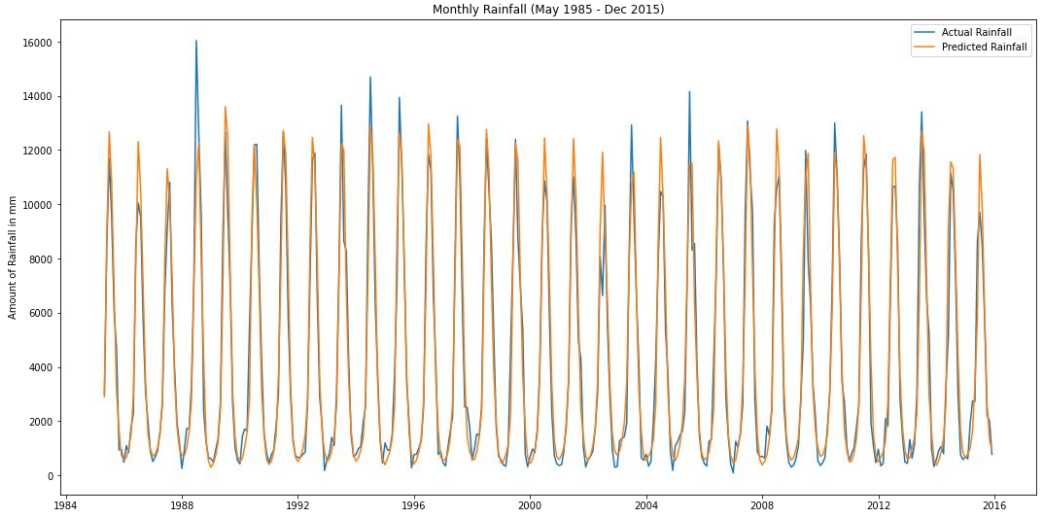
Sr. #	Model Type	MAPE
1	ARIMA	46.712
2	Linear Regression	301.29
3	Polynomial Regression	245.52

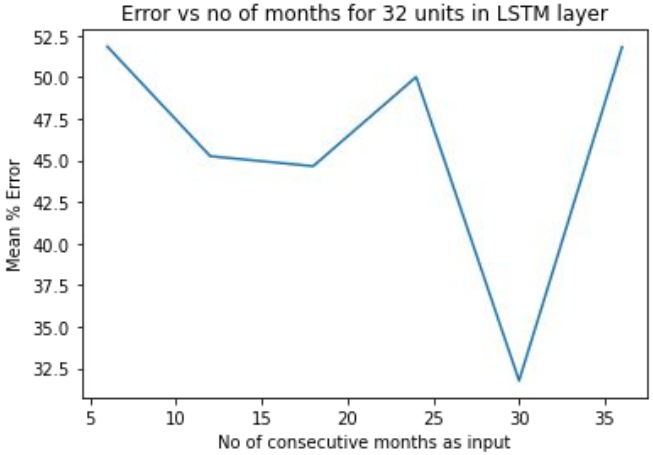
Results: LSTM Neural Network



MAPE for 64 units in LSTM layer and 12 months input data = 31.65%

units = [8,16,32,64,128,256]





months = [6,12,18,24,30,36]

Thank you